Induction and Recursion Principles in a Higher-Order Quantitative Logic for Probability

Giorgio Bacci
Department of Computer Science, Aalborg University, Denmark

Rasmus Ejlers Møgelberg
IT University of Copenhagen, Denmark

Abstract

Quantitative logic reasons about the degree to which formulas are satisfied. This paper studies the fundamental reasoning principles of higher-order quantitative logic and their application to reasoning about probabilistic programs and processes.

We construct an affine calculus for 1-bounded complete metric spaces and the monad for probability measures equipped with the Kantorovich distance. The calculus includes a form of guarded recursion interpreted via Banach's fixed point theorem, useful, e.g., for recursive programming with processes. We then define an affine higher-order quantitative logic for reasoning about terms of our calculus. The logic includes novel principles for guarded recursion, and induction over probability measures and natural numbers.

We illustrate the expressivity of the logic by a sequence of case studies: Proving upper limits on bisimilarity distances of Markov processes, showing convergence of a temporal learning algorithm and of a random walk using a coupling argument.

2012 ACM Subject Classification Theory of computation → Logic and verification; Theory of computation → Random walks and Markov chains; Theory of computation → Higher order logic

Keywords and phrases Quantitative Logic, Probabilistic Processes, Affine Logic, Guarded Recursion, Metric Spaces.

Digital Object Identifier 10.4230/LIPIcs.LICS.2026.34

Funding *Giorgio Bacci*: This work was supported by Digital Research Centre Denmark (DIREC), under the Robust NIDS project.
Rasmus Ejlers Møgelberg: This work was supported by the Independent Research Fund Denmark, grant number 2032-00134B.

1 Introduction

In computer science, logics are traditionally designed for proving precise qualitative properties of programs, such as program equality. However, in many modern applications, especially those that involve probabilistic programming, one is often interested in proving quantitative properties of programs, such as upper limits on program distances, sensitivity of program outputs to program inputs, or convergence of sequences of programs. Such properties are important in diverse application areas such as differential privacy [14, 34], security [3, 5] and machine learning [33]. Similarly, in process algebra, it has long been known that for probabilistic processes, the notion of bisimilarity should be stated quantitatively, in the form of a metric, to be robust to small perturbations that may otherwise compromise the exact comparison of behaviours [20, 16]. Also program logics for probabilistic programming languages are often defined quantitatively, using random variables valued in $[0, \infty]$ as a quantitative notion of predicates of states [21, 31, 30, 2, 4].

What has been lacking so far is a general notion of quantitative logic in which all these applications can be expressed as special cases. This paper develops such a logic, and in particular, the fundamental principles governing equality, those for reasoning about

41st Annual Symposium on Logic in Computer Science (LICS 2026).
Editors: Claudia Faggian and Joost-Pieter Katoen; Article No. 34; pp. 34:1–34:27
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

distributions, and the notion of guarded recursion. What we found is that these simple basic principles are extremely powerful, and that by reading standard definitions of concepts such as a coupling and bisimulation in our logic we can reason about quantitative versions of these concepts in a very natural way.

A sensitivity calculus for complete metric spaces. Quantitative logic is the logic of metric spaces. A metric space can be thought of as a set with a quantitative notion of equality: The smaller the distance between two elements, the more equal they are. From the logical perspective, the appropriate notion of morphism between metric spaces is that of a non-expansive map: a function f satisfying $d(f(x), f(y)) \leq d(x, y)$, for all x, y. These are precisely the maps that preserve equality in the quantitative sense, sending quantitatively equal inputs to quantitatively equal outputs. We will restrict our attention to those metric spaces that are non-empty, complete, and 1-bounded, and denote the resulting category by **CMet**. We found this category to be the most suitable for our purposes, although alternative models are also possible, as we discuss further in Section 12.

The category **CMet** is symmetric monoidal closed, and moreover has a strong monad $\mathcal{D}$ on it, mapping a metric space to the set of Radon distributions equipped with the Kantorovich metric. The Kantorovich metric is a natural notion of metric on distributions, and is used in most applications, including program logics and bisimilarity distance. Another important operation is that of scaling a metric space by a factor $r \geq 0$. The resulting space rX has the same underlying set as X, but all distances are scaled by r (but bounded by 1). Scaling is used to express the sensitivity of functions to changes in input data: A map has type $rX \to Y$ iff it is r-Lipschitz continuous or, using the terminology of the programming languages Fuzz [34] and DFuzz [17], is r-*sensitive* in X. An important example is the operation that maps two distributions μ and ν to their convex combination $\mu \oplus_p \nu$, corresponding to choosing μ with probability p and ν with probability $1-p$, for $p \in (0, 1)$. This is an operation of type

$$\oplus_p \colon p\mathcal{D}X \otimes (1-p)\mathcal{D}X \to \mathcal{D}X \tag{1}$$

expressing that $\oplus_p$ is p-sensitive in the first argument and $(1-p)$-sensitive in the second, meaning $d(\mu \oplus_p \nu, \mu' \oplus_p \nu) \leq p \cdot d(\mu, \mu')$ and $d(\mu \oplus_p \nu, \mu \oplus_p \nu') \leq (1-p) \cdot d(\nu, \nu')$ hold simultaneously, where both distances are measured in $\mathcal{D}X$.

Inspired by Fuzz [34], we define a λ-calculus for programming in **CMet**. The calculus is affine, because the $\otimes$ of the monoidal structure has projections, but not diagonals. The calculus is simply-typed, with typing judgements using contexts with sensitivity annotations on all variables. For example, the sensitivity of $\oplus_p$ is expressed by the typing judgement $\mu :^p \mathcal{D}X, \nu :^{1-p} \mathcal{D}X \vdash \mu \oplus_p \nu : \mathcal{D}X$.

Metric spaces do not model general recursion, but they do model a form of guarded recursion via the Banach fixed point theorem: Any non-expansive map $f \colon pX \to X$ has a unique fixed point if $p < 1$ and X is complete and non-empty. Guarded recursion on this form has previously been studied for ultra-metric spaces [15], but that setting is simpler, because it models simply typed lambda calculus and intuitionistic logic. Generalising to the affine setting of general metric spaces has surprising consequences. For example, in the ultra-metric setting, the fixed point operator defines a non-expansive map $(pX \to X) \to X$ whereas in the general metric setting, this must be weakened to $(pX \multimap X) \to (1-p)X$ where $\multimap$ is the closed monoidal structure. In our calculus the *guarded fixed point combinator* can be given type $\Gamma \vdash \mathsf{fix}\, x.t : A$ if $(1-p)\Gamma, x :^p A \vdash t : A$, for $p < 1$. Here, the multiplication refers to the operation of scaling all sensitivity annotations in a context.

The type (1) of the $\oplus_p$ combinator means that many distributions and processes can be defined as fixed points. For example, the geometric distribution is recursively defined as

$\mathsf{geo}_p = \delta(0) \oplus_p \mathcal{D}(\mathsf{succ})(\mathsf{geo}_p)$, where $\delta(0)$ is the Dirac distribution, and $\mathsf{succ}\colon \mathbb{N} \to \mathbb{N}$ is the successor function. This works because the defining equation for geo_p is productive: it only calls itself recursively with probability $1 - p < 1$.

A higher-order quantitative logic. Our main contribution is a quantitative logic for reasoning about terms in our calculus.

Logical reasoning is via inference of judgements $\Psi \vdash \varphi$ which express that the predicate φ logically follows from the sequence of predicates $\Psi = \psi_1, \ldots, \psi_n$. Predicates are valued in the unit interval $[0, 1]$ with 0 being true and 1 being false. For example, the equality predicate is interpreted as distance between elements of a metric space. This leads naturally to an affine logic, because transitivity $x = y, y = z \vdash x = z$ can be interpreted as the triangle inequality $d(x, y) + d(y, z) \geq d(x, z)$ (one of the axioms of metric spaces) if the comma is interpreted as sum.

The first step in defining a logic is to state what the well-formed predicates are. Since the unit interval is itself an element in **CMet**, we use our calculus to program with predicates using a special type Prop interpreted as $[0, 1]$. The interval $[0, 1]$ carries a closed monoid structure given by (truncated) sum and subtraction, and universal and existential quantification can be modelled using infimum and supremum, respectively. Since the logic allows quantification over all objects of **CMet**, including exponents of Prop, it is a higher-order logic.

Like metric spaces, propositions can be rescaled by factors $r \in [0, \infty]$ and this can be used to express a logical principle of guarded recursion. In a closed context this states that if we can prove ϕ from $p \cdot \phi$ for some $p < 1$, then ϕ holds. The general rule reflects that of the fixed point combinator: if $(1 - p)\Psi, p\phi \vdash \phi$ holds, then $\Psi \vdash \phi$.

We also study induction principles, both for natural numbers and for the Radon distribution monad $\mathcal{D}$. In the latter case, the principle states that $\Psi \vdash \phi[\mu/x]$ holds for all $\mu \in \mathcal{D}X$ if it does on Dirac distributions and $p(\phi[\mu/x]), (1 - p)\phi[\nu/x] \vdash \phi[\mu \oplus_p \nu/x]$, for all $p \in (0, 1)$. At first sight, this may appear to only prove it for all finitely supported distributions, and $\mathcal{D}X$ also includes continuous distributions. The principle is nevertheless sound, because the finite distributions are dense in $\mathcal{D}X$, and because the principle has the side condition that x has finite sensitivity in ϕ, so that ϕ is continuous in x. This principle reflects the observation by Mardare et al. [27, 28] that $\mathcal{D}$ is generated freely by Dirac distributions and $\oplus_p$.

When stating the elimination rule for the equality predicate, one must take sensitivity into account. More precisely, if x has sensitivity p in ϕ then $\phi[t/x], p(t = s) \vdash \phi[s/x]$. This was previously observed by Dagnino and Pasquali [11] in a propositional logic, and we adapt one of their rules as an elimination principle for equality. We show that the rule has wide-ranging consequences, including that equality can be proved symmetric and transitive. Equality can also be proved a congruence when this is formulated in a way that takes sensitivities into account. For example, using the type (1) of $\oplus_p$, one can prove that

$$p(x = y), (1 - p)(z = w) \vdash x \oplus_p z = y \oplus_p w\,. \tag{2}$$

Case studies. To show the expressiveness of our logic we develop a sequence of case studies. The first concerns bisimilarity distance of Markov processes. We model these in **CMet** as the terminal solution $\mathbb{P}_c$ to $\mathbb{P}_c \cong A \otimes c\mathcal{D}(\mathbb{P}_c)$. Here $c \in (0, 1]$ is a discount factor. The smaller it is, the more the distance emphasizes short-term behaviour. In $\mathbb{P}_c$ the metric is exactly the bisimilarity distance. We show how to define recursive processes using the fixed point combinator, and prove upper bounds on distances $d(t, u) \leq c$ by proving $c \cdot \mathsf{ff} \vdash t = u$ in the logic, using the guarded recursion principle. Some examples require $c < 1$, but in many cases the productivity requirement follows from (2). We also show how to define the notion of bisimilarity inside the logic and prove it equivalent to equality when $c < 1$.

We also show how to prove convergence of a temporal learning algorithm and of a random walk on a hypercube. The latter example is particularly interesting, as it requires a coupling proof. Coupling arguments [25] are a technique for showing equivalence or closeness of probabilistic programs by relating the probabilistic choices made by one program to those by another. We show how this technique can be used in our logic by internalising the definition of the Kantorovich distance: We prove that two distributions are equal if and only if there exists an equality-coupling between them. Note that since this biimplication is a statement in quantitative logic, its semantic meaning is an equality of numbers in the unit interval. The proof of this illustrates the power of the principles of our logic, in particular the elimination rule for equality and the induction principle for distributions.

Contributions. In summary, our contributions are:

- We present an affine lambda calculus with sensitivity annotations for programming in the category **CMet** of complete 1-bounded metric spaces;
- We formulate a first (to our knowledge) higher-order logic for quantitative reasoning in which equality is interpreted as distance in metric spaces. This includes new rules for recursion over natural numbers and probability measures, as well as a guarded recursion principle;
- We show by example how to use the logic to reason about Markov processes, convergence in temporal learning and for coupling arguments, illustrating the power of the combination of the above mentioned principles;

Overview. The paper is organised as follows. We first discuss preliminaries on metric spaces in Section 2. Sections 3 and 4 discuss the Banach fixed point operator and the probability measure monad. Sections 5 and 6 discuss the syntax and semantics of the calculus and the logic, respectively. Section 7 shows how to prove basic properties of the logic, including that equality is a congruence which is equivalent to an internalisation of the usual definition of the Kantorovich measure. Section 8 shows applications to Markov processes, Section 9 shows an application to temporal learning, and Section 10 illustrates how to use coupling arguments in the logic. Finally, Section 11 discusses related work, and Section 12 concludes.

2 Preliminaries on metric spaces

A (1-bounded) *metric space* is a set X equipped with a distance function $d_X \colon X \times X \to [0, 1]$ satisfying (*reflexivity*) $d_X(x, y) = 0$ iff $x = y$; (*symmetry*) $d_X(x, y) = d_X(y, x)$; and (*triangular inequality*) $d_X(x, z) \leq d_X(x, y) + d_X(y, z)$.

A function $f \colon X \to Y$ between metric spaces is r-*Lipschitz continuous*, for $r \geq 0$, if $r \cdot d_X(x, x') \geq d_Y(f(x), f(x'))$; *non-expansive* when $r = 1$; and a *contraction* when $r < 1$ and $X = Y$. A metric space is complete if all Cauchy sequences converge.

In this paper, we work with non-empty 1-bounded complete metric spaces. These form a category **CMet** with non-expansive maps as morphisms. Restricting to 1-bounded spaces allows us to include sets as *discrete* metric spaces by setting all distances between distinct elements to 1. This defines a left adjoint to the forgetful functor from **CMet** to **Set**: if X is discrete, then all maps $f \colon X \to Y$ are non-expansive. As a consequence, we can regard **Set** as a full subcategory of **CMet**. An important example for this paper is $\mathbb{N}$, the set of natural numbers, which is an object in **CMet**.

The category **CMet** has both binary products and coproducts: $X \times Y$ combines spaces by equipping the Cartesian product with the point-wise maximum distance; $X + Y$ combines them into a disjoint union by keeping elements from different components as far as possible from each other (*i.e.*, at distance 1). The singleton metric space $\mathbf{1}$ is the final object.

There is another natural structure on the Cartesian product, the *tensor* $X \otimes Y$, that combines distances by 1-bounded truncated sum:

$$d_{X \otimes Y}((x, y), (x', y')) = \min\{d_X(x, x') + d_Y(y, y'), 1\}\,.$$

Also the tensor product has non-expansive projections $X \otimes Y \to X$ and $X \otimes Y \to Y$, but generally no diagonal $X \to X \otimes X$, unless X is discrete. Note the following universal property of $\otimes$: A map from $X \otimes Y$ to Z is non-expansive if and only if it is non-expansive in each variable.

Unlike the categorical product, $\otimes$ allows currying and function application. More precisely, $(\mathbf{CMet}, \otimes, \mathbf{1})$ is a symmetric monoidal category, with unit $\mathbf{1}$ and adjunction $(-\otimes X) \dashv (X \multimap -)$ making this structure closed. Here, $X \multimap Y$ denotes the set of non-expansive functions from X to Y endowed with point-wise supremum metric $d_{X \multimap Y}(f, g) = \sup_{x \in X} d_Y(f(x), g(x))$. The counit of the adjunction is function evaluation $\mathsf{ev}\colon (X \multimap Y) \otimes X \to Y$.

Nonexpansive morphisms in **CMet** subsume the notion of Lipschitz continuity through the rescaling functor rX, which scales distances by a factor $r > 0$ as

$$d_{rX}(x, x') = \min\{r \cdot d_X(x, x'), 1\}\,.$$

Indeed, by unpacking the definition, $f\colon rX \to Y$ is a morphism in **CMet** iff f considered as a map $f\colon X \to Y$ is r-Lipschitz continuous. For convenience we allow rescaling also for $r = 0$ and $r = \infty$, and define $0X$ as $\mathbf{1}$, the one-point metric space, and ∞X as the discrete metric space on X. Scaling preserves products and, under suitable restrictions, coproducts:

$$sA \times sB \cong s(A \times B) \quad (\text{for } s \in [0, \infty])\,, \qquad rA + rB \cong r(A + B) \quad (\text{for } r \in [1, \infty])\,.$$

The next theorem states the properties of the interaction between $\otimes$ and scaling. First recall that $[0, \infty]$ is an ordered semiring with $\leq$, addition and multiplication defined as usual in most cases, and by $\infty \cdot 0 = 0$ and $0 \cdot \infty = 0$. The properties together imply that the scaling operation is a $[0, \infty]$-graded comonad on **CMet** in the sense of [10, Definition 13] and [18, Section 5.2].

▶ **Theorem 1.** *There are natural transformations of types*

$$\begin{aligned} &m_{r,A,B}\colon rA \otimes rB \to r(A \otimes B) && n_r\colon \mathbf{1} \to r\mathbf{1} \\ &c_{r,s,A}\colon (r+s)A \to rA \otimes sA && w_A\colon 0A \to \mathbf{1} \\ &\beta_{r,s,A}\colon (rs)A \to r(sA) && \epsilon_A\colon 1A \to A \\ &\kappa_{r,s,A}\colon sA \to rA \quad (\textit{for } r \leq s) \end{aligned}$$

Moreover, if $s \leq 1$ or $r \geq 1$ then $\beta_{r,s,A}$ is an isomorphism, and if $r \geq 1$ then $m_{r,A,B}$ is an isomorphism.

Note that without the conditions above, $\beta_{r,s,A}$ and $m_{r,A,B}$ need not be isomorphisms. For example, if A is discrete then $\frac{1}{2}(2A) = \frac{1}{2}A$.

3 Fixed points

The Banach fixed point theorem [8] states that any contractive function on a non-empty complete metric space has a unique fixed point. If $f\colon X \otimes pY \to Y$ is a morphism in **CMet** and $p < 1$ then, for any $x \in X$, the map $f(x, -)\colon pY \to Y$ is a contraction on Y and so has a unique fixed point $\mathsf{fp}(f)(x)$.

▶ **Proposition 2.** *Let $p < 1$. If $f\colon (1-p)X \otimes pY \to Y$ then $\mathsf{fp}(f)\colon X \to Y$ is non-expansive. There is a non-expansive map $\mathsf{fix}\colon (pY \multimap Y) \to (1-p)Y$ mapping functions to fixed points.*

Proof. For the proof of the second statement, define fix as the function mapping a contraction $f\colon pY \to Y$ to its unique fixed point. Let $f, f'\colon pY \to Y$, then

$$\begin{aligned} d(\mathsf{fix}(f), \mathsf{fix}(f')) &\le d(\mathsf{fix}(f), f(\mathsf{fix}(f')) + d(f(\mathsf{fix}(f')), \mathsf{fix}(f')) \\ &= d(f(\mathsf{fix}(f)), f(\mathsf{fix}(f')) + d(f(\mathsf{fix}(f')), f'(\mathsf{fix}(f'))) \\ &\le pd(\mathsf{fix}(f), \mathsf{fix}(f')) + d(f, f') \end{aligned}$$

so that $(1-p)d(\mathsf{fix}(f), \mathsf{fix}(f')) \le d(f, f')$, as desired. For the first statement, by currying f and composing with fix we see that

$$\mathsf{fix} \circ \mathsf{curry}(f)\colon (1-p)X \to (1-p)Y\,.$$

Thus, we can define $\mathsf{fp}(f)\colon X \to Y$ as the composition $\beta_Y^{-1} \circ \frac{1}{1-p}(\mathsf{fix} \circ \mathsf{curry}(f)) \circ \beta_X$, where β is natural isomorphims from Theorem 1. ◀

4 Probability measures

We introduce the *Radon probability monad* on **CMet** and recall its presentation as the free complete interpolative barycentric algebra [28].

A (Borel) probability measure μ on a metric space X is *Radon* if for any Borel set $E \subseteq X$, $\mu(E)$ is the supremum of $\mu(K)$ over all compact subsets K of E. Examples of Radon probability measures are Dirac measures, (the Borel restriction of) the Lebesgue measure over the unit interval, any probability measure with finite support or over a Polish space (*i.e.*, a complete metric space with a countable dense subset).

A *coupling* between two probability measures μ and ν on X is a probability measure ω on $X \times X$ whose left and right marginals are, respectively, μ and ν (*i.e.*, $\omega(E \times X) = \mu(E)$ and $\omega(X \times E) = \nu(E)$, for all Borel sets E). The product measure $\mu \times \nu$ is always a coupling between μ and ν. Moreover, if μ and ν are Radon, so is any coupling ω between them.

For X a metric space, denote by $\mathcal{D}X$ the space of Radon probability measures over X equipped with the *Kantorovich distance*, defined by

$$d_{\mathcal{D}X}(\mu, \nu) = \min_{\omega} \int d_X(x, x')\, \omega(\mathrm{d}x, \mathrm{d}x') \tag{3}$$

where ω runs over the couplings between μ and ν. If X is a complete metric space, so is $\mathcal{D}X$.

The definition above extends to a monad $\mathcal{D}$ on **CMet**, called *Radon probability monad*, with underlying functor acting on morphisms $f\colon X \to Y$ as $\mu \in \mathcal{D}X \mapsto \mu \circ f^{-1} \in \mathcal{D}Y$ (a.k.a., the pushforward measure along f). The unit of $\mathcal{D}$ is the Dirac measure $\delta_X\colon X \to \mathcal{D}X$, but rather than describing the multiplication, we recall that this monad has an algebraic presentation as the free complete *interpolative barycentric algebra* [27, 28].

▶ **Definition 3** (IB Algebra). *An* interpolative barycentric algebra *is a complete metric space X with non-expansive operations $\oplus_p \colon pX \otimes (1-p)X \to X$, for all $p \in (0,1)$, such that*

$$x \oplus_p x = x \tag{IDEM}$$

$$x \oplus_p y = y \oplus_{1-p} x \tag{COMM}$$

$$(x \oplus_p y) \oplus_q z = x \oplus_{pq} (y \oplus_{\frac{q-pq}{1-pq}} z) \tag{ASSOC}$$

A homomorphism $f \colon X \to Y$ of IB algebras is a non-expansive map such that $f(x \oplus_p y) = f(x) \oplus_p f(y)$ for all $p \in (0,1)$.

The axioms are those of barycentric algebras (a.k.a., convex algebras), axiomatizing probabilistic choice by means of binary convex combinations $x \oplus_p y$. The definition above is equivalent to that in [28], as the type imposed on the operations $\oplus_p$ is equivalent to requiring

$$d_X(x \oplus_p y, x' \oplus_p y') \leq p d_X(x, x') + (1-p) d_X(y, y') \,.$$

The formulation proposed in Definition 3 is preferable in our context because it incorporates the Lipschitz constants directly into the type of the operation. This not only enables the remaining conditions to be expressed purely as equations but also ensures that many probabilistic processes can be defined using the Banach fixed point combinator as we shall see below.

It is not difficult to show that, for any $X \in \mathbf{CMet}$, $\mathcal{D}X$ is a complete interpolative barycentric algebra, by interpreting the operations as $\mu \oplus_p \nu = p\mu + (1-p)\nu$, for all $\mu, \nu \in \mathcal{D}X$. The next result states that $\mathcal{D}X$ is the free algebra with respect to all Lipschitz maps, which follows as a corollary of [28, Theorem 3.8]. Before we state it, note that if A and B are IB-algebras, the equational definition of IB-homomorphism extends to Lipschitz maps $f \colon rA \to B$. In terms of diagrams, this can be stated as the commutativity of

$$\begin{array}{ccccc}
(pr)A \otimes (\bar{p}r)A & \xrightarrow{m \circ (\beta \otimes \beta)} & r(pA \otimes \bar{p}A) & \longrightarrow & rA \\
{\scriptstyle \beta \otimes \beta}\big\downarrow & & & & \big\downarrow{\scriptstyle f} \\
p(rA) \otimes \bar{p}(rA) & \xrightarrow{pf \otimes \bar{p}f} & pB \otimes \bar{p}B & \longrightarrow & B
\end{array} \qquad (\text{where } \bar{p} = 1-p)$$

▶ **Proposition 4.** *If $f \colon \Gamma \otimes rX \to Y$ (with $r < \infty$) and Y is an IB algebra, there exists a unique $\overline{f} \colon \Gamma \otimes r\mathcal{D}X \to Y$ which is a homomorphism in its second argument, satisfying $f = \overline{f} \circ (\Gamma \otimes r\delta_X)$.*

Observe that the special case of $r = 1$ and $\Gamma = \mathbf{1}$ in Proposition 4 can be rephrased as the existence of a left adjoint to the forgetful functor from the category of IB algebras to $\mathbf{CMet}$. Thus, as anticipated, $\mathcal{D}$ forms a monad on $\mathbf{CMet}$. This monad is moreover strong. The restriction on r being finite means that $\overline{f}$ is continuous. This is necessary because the other requirements otherwise only determine the value of $\overline{f}$ on finite distributions, which form a dense subset of $\mathcal{D}X$.

▶ **Lemma 5.** *If X, Y are IB algebras, so are, $X \otimes Y$, $Z \multimap X$, and qX for any Z and $q \leq 1$.*

5 A calculus for CMet

We now define a calculus for programming in the category $\mathbf{CMet}$.

Syntax. The syntax is based on a simply-typed λ-calculus with products and sums, extended with primitives for probabilistic distributions, recursion, and fixed points.

$$\begin{aligned} t, u, v ::= {} & x \mid \lambda x.t \mid tu \mid () \mid \langle t, u\rangle \mid \pi_1 t \mid \pi_2 t \mid inj_1 t \mid inj_2 t \mid \mathsf{case}\ t\ \mathsf{of}\ [inj_1 x.u \mid inj_2 y.v] \\ & \mid (t, u) \mid \mathsf{let}\ (x, y) = u\ \mathsf{in}\ t \mid \delta t \mid t \oplus_p u \mid \mathsf{let}\ x = u\ \mathsf{in}\ t \\ & \mid \mathsf{zero} \mid \mathsf{succ}(t) \mid \mathsf{rec}(u, (x, y).t, v) \mid \mathsf{fix}\ x.t \end{aligned}$$

There are two pairs constructors, $\langle t, u\rangle$ and (t, u), corresponding to the Cartesian and monoidal products, respectively. The first one is eliminated using the projections $\pi_i t$, whereas the second one is eliminated using ($\mathsf{let}\ (x, y) = u\ \mathsf{in}\ t$). The term $()$ is the unit value. The injections $inj_i t$ form expressions of sum type, which are eliminated by case analysis ($\mathsf{case}\ t\ \mathsf{of}\ [inj_1 x.u \mid inj_2 y.v]$). The term $\delta(t)$ denotes a Dirac distribution, and $t \oplus_p u$ the convex sum of t and u. Probability distributions are sampled using ($\mathsf{let}\ x = u\ \mathsf{in}\ t$). The terms zero and $\mathsf{succ}(t)$ are constructors for natural numbers. $\mathsf{rec}(u, (x, y).t, v)$ denotes a term obtained by primitive recursion on natural numbers. Finally, $\mathsf{fix}\ x.t$ is the "Banach" fixed point combinator.

The *types* of the calculus are defined by the grammar

$$A, B ::= \mathbb{N} \mid 1 \mid A \times B \mid A + B \mid A\ {}_r\!\otimes_s B \mid A \multimap_r B \mid \mathcal{D}A \qquad (r, s \geq 0)$$

essentially corresponding to the categorical operations in **CMet** from Section 2. Although rescaling of metric spaces played a central role in Section 2, it is not a primitive type former in the calculus. Instead, it is part of the tensor type $A\ {}_r\!\otimes_s B$ and function type $A \multimap_r B$ constructors. This choice was made to minimize the book keeping necessary for scalars in terms. Finally, $\mathcal{D}A$ is the type of Radon probability measures on A.

Terms are typed with judgements of the form $\Gamma \vdash t : A$, where A is a type and Γ a typing context. The complete list of typing rules is given in Figure 1. The typing system is inspired by Fuzz [34], where the sensitivity of each variable used in a term is tracked by annotations of the form $x :^r A$ in typing contexts. More precisely, a binding $x :^r A$ in a context Γ means that x has type A under Γ and that terms typed under Γ are r-sensitive with respect to x.

Formally, typing contexts are constructed according to the following formation rules

$$\frac{}{\langle\rangle :: \mathsf{ctx}} \qquad\qquad \frac{\Gamma :: \mathsf{ctx} \quad x \notin \Gamma \quad r \in [0, \infty]}{\Gamma, x :^r A :: \mathsf{ctx}}$$

As usual, Γ, Γ' denotes concatenation of contexts with disjoint variable bindings. Most rules use the operations of sum $\Gamma + \Gamma'$ and scaling $r\Gamma$ of contexts, to keep track of the sensitivities. These are defined as the point-wise sum and scaling of the sensitivity in each variable binding:

$$\begin{aligned} \langle\rangle + \langle\rangle &= \langle\rangle & (\Gamma, x :^r A) + (\Gamma', x :^s A) &= (\Gamma + \Gamma'), x :^{r+s} A\,, \\ r\langle\rangle &= \langle\rangle & r(\Gamma, x :^s A) &= r\Gamma, x :^{r \cdot s} A\,. \end{aligned}$$

Observe that $\Gamma + \Gamma'$ is defined only when Γ and Γ' are compatible, *i.e.*, they agree on the order and the types of all variable bindings —for example $(x :^r A, y :^s B) + (y :^s B, x :^r A)$ is not defined. In the rest of the paper, whenever we take a sum $\Gamma + \Gamma'$, compatibility of the contexts is assumed implicitly.

The rule (VAR) for variable introduction reflects that projection can be given any Lipschitz factor $r \geq 1$. We allow all such r, and not just $r = 1$ to incorporate weakening directly into the typing rules. Note that, in the rule (CASE) for sum elimination, the sensitivities of the bound variables x and y are required to match and be greater or equal than 1, as scaling by $r < 1$ does not commute with coproducts. In the rule ($\otimes$) for tensor introduction, Γ'' is

$$\frac{r \geq 1}{\Gamma, x :^r A, \Gamma' \vdash x : A}\ (\textsc{var}) \qquad \frac{}{\Gamma \vdash () : 1}\ (\textsc{unit}) \qquad \frac{\Gamma, x :^r A \vdash t : B}{\Gamma \vdash \lambda x.t : A \multimap_r B}\ (\textsc{abs})$$

$$\frac{\Gamma \vdash t : A \multimap_r B \quad \Gamma' \vdash u : A}{\Gamma + r\Gamma' \vdash t\,u : B}\ (\textsc{app}) \qquad \frac{\Gamma \vdash t : A \quad \Gamma \vdash u : B}{\Gamma \vdash \langle t, u\rangle : A \times B}\ (\textsc{pair}) \qquad \frac{\Gamma \vdash t : A_1 \times A_2}{\Gamma \vdash \pi_i t : A_i}\ (\pi_i)$$

$$\frac{\Gamma \vdash t : A_i}{\Gamma \vdash inj_i t : A_1 + A_2}\ (inj_i) \qquad \frac{\Gamma, x :^r A \vdash u : C \quad \Gamma, y :^r B \vdash v : C \quad \Gamma' \vdash t : A + B \quad r \geq 1}{\Gamma + r\Gamma' \vdash \mathsf{case}\ t\ \mathsf{of}\ [inj_1 x.u \mid inj_2 y.v] : C}\ (\textsc{case})$$

$$\frac{\Gamma \vdash t : A \quad \Gamma' \vdash u : B}{r\Gamma + s\Gamma' + \Gamma'' \vdash (t, u) : A\,{}_r\!\otimes_s B}\ (\otimes) \qquad \frac{\Gamma, x :^r A, y :^s B \vdash t : C \quad \Gamma' \vdash u : A\,{}_r\!\otimes_s B}{\Gamma + \Gamma' \vdash \mathsf{let}\ (x, y) = u\ \mathsf{in}\ t : C}\ (\textsc{let-}\otimes)$$

$$\frac{\Gamma \vdash t : A}{\Gamma \vdash \delta t : \mathcal{D}A}\ (\delta) \qquad \frac{\Gamma \vdash t : \mathcal{D}A \quad \Gamma' \vdash u : \mathcal{D}A \quad p \in (0,1)}{p\Gamma + (1-p)\Gamma' \vdash t \oplus_p u : \mathcal{D}A}\ (\oplus_p)$$

$$\frac{\Gamma, x :^r A \vdash t : E \quad \Gamma' \vdash u : \mathcal{D}A \quad E \text{ IB algebra} \quad r < \infty}{\Gamma + r\Gamma' \vdash \mathsf{let}\ x = u\ \mathsf{in}\ t : E}\ (\textsc{let}) \qquad \frac{}{\Gamma \vdash \mathsf{zero} : \mathbb{N}}\ (\textsc{zero})$$

$$\frac{\Gamma \vdash t : \mathbb{N}}{\Gamma \vdash \mathsf{succ}(t) : \mathbb{N}}\ (\textsc{succ}) \qquad \frac{\Gamma \vdash z : A \quad \Gamma', x :^1 A, y :^1 \mathbb{N} \vdash s : A \quad \Gamma'' \vdash n : \mathbb{N}}{\Gamma + \infty\Gamma' + \Gamma'' \vdash \mathsf{rec}(z, (x, y).s, n) : A}\ (\textsc{rec})$$

$$\frac{(1-p)\Gamma, x :^p A \vdash t : A \quad p < 1}{\Gamma \vdash \mathsf{fix}\ x.t : A}\ (\textsc{fix})$$

Figure 1 Typing rules.

used merely to incorporate weakening in the rule[1]. In the rule (REC) for natural number recursion, the successor case term s can have additional variables to x and y. However, s must be applied n times, and therefore the sensitivity of Γ' must be scaled by n. Since n is not known statically, the only possible upper limit is ∞. The type of the fixed point operator reflects Proposition 2. The elimination rule for $\mathcal{D}$ uses the judgment of a type being an IB algebra. These are defined by the grammar

$$E, F ::= \mathcal{D}A \mid E\,{}_p\!\otimes_q F \mid A \multimap_r E \qquad (p, q \leq 1 \text{ and } r \geq 0)$$

as justified by Lemma 5.

▶ **Example 6** (The geometric distribution). Let $p \in (0,1)$. The geometric distribution $\mathsf{geo}_p : \mathcal{D}(\mathbb{N})$ is uniquely characterized by the recurrence $\mathsf{geo}_p \equiv \delta(0) \oplus_p \mathcal{D}(\mathsf{succ})(\mathsf{geo}_p)$. It is therefore natural to define it as the following fixed point:

$$\mathsf{geo}_p \triangleq \mathsf{fix}\ x.\delta(0) \oplus_p \mathcal{D}(\mathsf{succ})(x)\,,$$

where, the functorial action of $\mathcal{D}$ on a function $f : A \multimap_r B$ (with $r < \infty$) is given by the term $\mathcal{D}(f) \triangleq \lambda x.\mathsf{let}\ a = x\ \mathsf{in}\ \delta(f(a)) : \mathcal{D}(A) \multimap_r \mathcal{D}(B)$. In particular, $\mathcal{D}(\mathsf{succ}) : \mathcal{D}(\mathbb{N}) \multimap_1 \mathcal{D}(\mathbb{N})$. Hence, geo_p is well typed by (FIX), since x has sensitivity $1 - p$ in the body of the fixed point.

As usual, terms are considered equal up to α-equivalence. We denote by $t[u/x]$ the capture-avoiding substitution of the term u for the free variable x in t.

The following two forms of context weakening for typing judgements hold.

[1] The use of a free context, is necessary for build weakening in the rule in the case both r and s are 0. This detail was overlooked in [14] in their rule for introduction of scaled type as their weakening lemma fails when scaling by 0.

$$
\begin{array}{rl}
(\lambda x.t)u \equiv t[u/x] & \pi_i(\langle t_1, t_2\rangle) \equiv t_i \\
t \equiv (\lambda x.t\,x) & \langle \pi_1(t), \pi_2(t)\rangle \equiv t \\
t \equiv () & u_i[t/x_i] \equiv \mathsf{case}\ (inj_i t)\ \mathsf{of}\ [inj_1 x_1.u_1 \mid inj_2 x_2.u_2] \\
\mathsf{let}\ (x,y) = (s,t)\ \mathsf{in}\ u \equiv u[s/x, t/y] & u[t/z] \equiv \mathsf{case}\ t\ \mathsf{of}\ [inj_1 x.u[inj_1 x/z] \mid inj_2 y.u[inj_2 y/z]] \\
\mathsf{let}\ (x,y) = t\ \mathsf{in}\ u[(x,y)/z] \equiv u[t/z] & \mathsf{let}\ x = \delta(t)\ \mathsf{in}\ u \equiv u[t/x] \\
 & \mathsf{let}\ x = (\mathsf{let}\ y = s\ \mathsf{in}\ t)\ \mathsf{in}\ u \equiv \mathsf{let}\ y = s\ \mathsf{in}\ (\mathsf{let}\ x = t\ \mathsf{in}\ u) \\
\mathsf{fix}\ x.t \equiv t[\mathsf{fix}\ x.t/x] & \mathsf{let}\ x = s \oplus_p t\ \mathsf{in}\ u \equiv (\mathsf{let}\ x = s\ \mathsf{in}\ u) \oplus_p (\mathsf{let}\ x = t\ \mathsf{in}\ u) \\
\mathsf{rec}(z, (x,y).s, \mathsf{zero}) \equiv z & \mathsf{rec}(z, (x,y).s, \mathsf{succ}(n)) \equiv s[\mathsf{rec}(z, (x,y).s, n)/x, n/y]
\end{array}
$$

Figure 2 Judgemental equality (to these should be added the axioms of IB algebras of Definition 3).

▶ **Lemma 7** (Weakening).
1. *If* $\Gamma, \Gamma' \vdash t : A$*, then* $\Gamma, \Delta, \Gamma' \vdash t : A$*;*
2. *If* $\Gamma \vdash t : A$*, then* $\Gamma + \Delta \vdash t : A$*.*

Moreover, we have the substitution lemma.

▶ **Lemma 8** (Substitution). *If* $\Gamma, x{:}^r A, \Gamma' \vdash t : B$ *and* $\Delta \vdash u : A$*, then* $(\Gamma, \Gamma') + r\Delta \vdash t[u/x] : B$*.*

The judgemental equality relation on terms is the least congruence relation generated by the rules in Figure 2. We use the symbol $\equiv$ for judgemental equality to distinguish it from the propositional equality $t = u$, which is a predicate in the logic to be defined in Section 6. Formally, judgemental equality is a relation on terms of the same type, and we will sometimes underline that by writing $\Gamma \vdash t \equiv u : A$. Similarly, the rules of Figure 2 are to be understood as equalities in a typing context in which both sides have the same type. For example, in the case of the η-rule for function types, t is assumed to have function type.

Semantics. Types and contexts are interpreted as objects in **CMet**:

$$
\begin{array}{ccc}
[\![\mathbb{N}]\!] \triangleq \mathbb{N} & [\![1]\!] \triangleq \mathbf{1} & [\![\mathcal{D}A]\!] \triangleq \mathcal{D}[\![A]\!] \\
[\![A \times B]\!] \triangleq [\![A]\!] \times [\![B]\!] & [\![A + B]\!] \triangleq [\![A]\!] + [\![B]\!] & [\![\langle\rangle]\!] \triangleq \mathbf{1} \\
[\![A\ {}_r\!\otimes_s B]\!] \triangleq r[\![A]\!] \otimes s[\![B]\!] & [\![A \multimap_r B]\!] \triangleq r[\![A]\!] \multimap [\![B]\!] & [\![\Gamma, x :^r A]\!] \triangleq [\![\Gamma]\!] \otimes r[\![A]\!]
\end{array}
$$

Judgements are interpreted as morphisms

$$[\![\Gamma \vdash t : A]\!] \colon [\![\Gamma]\!] \to [\![A]\!]$$

in **CMet**. The interpretation of terms is for most parts the usual set-theoretic interpretation. For example, function abstraction and application are precisely the usual set-theoretic abstractions and applications. The exercise when defining the interpretation is ensuring the Lipschitz conditions associated with types and typing judgements. This can be done entirely on the abstract category theoretic level, using maps such as those of Theorem 1. One key point in doing so is that there exist morphisms

$$
\begin{array}{ll}
\mathsf{split}\colon [\![\Gamma + \Gamma']\!] \to [\![\Gamma]\!] \otimes [\![\Gamma']\!]\,, & \mathsf{proj}\colon [\![\Gamma, \Delta, \Gamma']\!] \to [\![\Gamma, \Gamma']\!]\,, \\
\mathsf{dist}\colon [\![p\Gamma]\!] \to p[\![\Gamma]\!]\,, & \mathsf{weak}\colon [\![\Gamma + \Gamma']\!] \to [\![\Gamma]\!]\,,
\end{array}
$$

easily defined by induction on Γ and Γ' using the morphisms c, m, κ of Theorem 1 and projections. Note that neither of these is generally an isomorphism, but dist is when $r \geq 1$. Using these, one can interpret function application $[\![t\,u]\!]$ as the composition

$$[\![\Gamma + r\Gamma']\!] \xrightarrow{([\![\Gamma]\!]\otimes\mathsf{dist})\circ\mathsf{split}} [\![\Gamma]\!] \otimes r[\![\Gamma']\!] \xrightarrow{[\![t]\!]\otimes r[\![u]\!]} [\![A \multimap_r B]\!] \otimes r[\![A]\!] \xrightarrow{\mathsf{ev}} [\![B]\!]\,.$$

A similar argument can be used for the interpretation of most other constructions in the language, including $t \oplus_p u$ which is interpreted using the IB algebra structure on $\mathcal{D}[\![A]\!]$. Let binding is interpreted using Proposition 4, and for the interpretation of natural number recursion, in the inductive case we use the fact that ∞X is discrete for any X, and so can be copied $\infty X \to \infty X \otimes \infty X$.

▶ Remark 9 (Independence on the choice of derivation). Formally, the above recipe gives an interpretation of a *derivation* of a typing judgement. Ideally, one would like that $[\![\Gamma \vdash t : A]\!]$ is defined only in terms of Γ, t and A, but since a typing judgement can have many derivations, this is not *a priori* clear. The main reason that judgements can have many derivations is that a given context Γ can be split as a sum $\Gamma' + \Gamma''$ in many different ways. We prove that the interpretation of judgements is independent of the derivation by induction on terms. In order to do that, however, we must annotate terms with enough information to infer the types of all subterms from Γ, t and A in judgements $\Gamma \vdash t : A$. This is not possible for the syntax given above. For example, for $\Gamma \vdash \pi_1(t) : A$ to have a derivation t must have type $A \times B$ for some B, but different derivations could use different B, which prevents the application of the induction hypothesis. The following meta-theoretic statements about the calculus should be read as statements about this 'official' annotated syntax. However, when writing terms, we will use the informal syntax presented above; it will always be the case that the annotations can be inferred.

▶ **Theorem 10.** *The interpretation of typing judgements $[\![\Gamma \vdash t : A]\!]$ is well-defined and independent of the choice of derivation.*

The interpretation is sound in the following sense.

▶ **Theorem 11** (Soundness). *If $\Gamma \vdash t \equiv u : A$ then $[\![t]\!] = [\![u]\!]$.*

The proof of soundness relies on the following lemmas.

▶ **Lemma 12** (Semantic Weakening). *For the derivations of Lemma 7, the following hold*
1. $[\![\Gamma, \Delta, \Gamma' \vdash t : A]\!] = [\![\Gamma, \Gamma' \vdash t : A]\!] \circ \mathsf{proj}$*;*
2. $[\![\Gamma + \Delta \vdash t : A]\!] = [\![\Gamma \vdash t : A]\!] \circ \mathsf{weak}$*.*

▶ **Lemma 13** (Semantic Substitution). *If $\Gamma, x :^r A, \Gamma' \vdash t : B$ and $\Delta \vdash u : A$, for the derivation of Lemma 8, the following holds*

$$[\![(\Gamma, \Gamma') + r\Delta \vdash t[u/x] : B]\!] = [\![t]\!] \circ ([\![\Gamma]\!] \otimes (r[\![u]\!]\circ\mathsf{dist}) \otimes [\![\Gamma']\!]) \circ \mathsf{split}\,.$$

6 Logic

In this section, we introduce a higher-order logic to reason about the terms of the calculus. Compared to standard logics, which have a Boolean semantics, our logic is interpreted over the commutative unital quantale

$$\mathsf{Prop} = ([0,1], \geq, \oplus, \multimap, 0)$$

$$\frac{}{\Gamma \vdash \mathsf{tt} : \mathsf{Prop}} \qquad \frac{}{\Gamma \vdash \mathsf{ff} : \mathsf{Prop}} \qquad \frac{\Gamma \vdash t : A \quad \Gamma' \vdash s : A}{\Gamma + \Gamma' \vdash t =_A u : \mathsf{Prop}}$$

$$\frac{\Gamma \vdash \varphi : \mathsf{Prop} \quad \Gamma' \vdash \psi : \mathsf{Prop}}{\Gamma + \Gamma' \vdash \varphi \bullet \psi : \mathsf{Prop}} \qquad \frac{\Gamma \vdash \varphi : \mathsf{Prop} \quad \Gamma' \vdash \psi : \mathsf{Prop}}{\Gamma + \Gamma' \vdash \varphi \multimap\!\!\bullet\, \psi : \mathsf{Prop}} \qquad \frac{\Gamma \vdash \varphi : \mathsf{Prop}}{r\Gamma + \Gamma' \vdash r\varphi : \mathsf{Prop}}$$

$$\frac{\Gamma \vdash \varphi : \mathsf{Prop}}{\Gamma \vdash \neg\varphi : \mathsf{Prop}} \qquad \frac{\Gamma \vdash \varphi : \mathsf{Prop} \quad \Gamma \vdash \psi : \mathsf{Prop}}{\Gamma \vdash \varphi \wedge \psi : \mathsf{Prop}} \qquad \frac{\Gamma \vdash \varphi : \mathsf{Prop} \quad \Gamma \vdash \psi : \mathsf{Prop}}{\Gamma \vdash \varphi \vee \psi : \mathsf{Prop}}$$

$$\frac{\Gamma, x :^{\infty} A \vdash \varphi : \mathsf{Prop}}{\Gamma \vdash \exists x : A.\varphi : \mathsf{Prop}} \qquad \frac{\Gamma, x :^{\infty} A \vdash \varphi : \mathsf{Prop}}{\Gamma \vdash \forall x : A.\varphi : \mathsf{Prop}}$$

Figure 3 Typing rules for logical predicates.

with truncated sum $x \oplus y = \min\{x+y, 1\}$ as tensor, unit 0, and adjoint $x \multimap y = \max\{y-x, 0\}$ defined as truncated reversed subtraction. Observe that the order in Prop corresponds to the reverse order in $[0,1]$: the bottom element is 1, the top element is 0, meet is sup and join is inf. This fits with the idea of interpreting equality as distance in metric spaces: The logical statement $t = u$ is true in the model if its interpretation as the distance between t and u is 0.

The formulas of the logic are well-typed *predicates* in our calculus. Formally, we extend the calculus with Prop as base type with usual Euclidean distance on $[0,1]$ and add

$$\varphi, \psi \ ::= \mathsf{tt} \mid \mathsf{ff} \mid t =_A u \mid \varphi \bullet \psi \mid \varphi \multimap\!\!\bullet\, \psi \mid r\psi \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \exists x : A.\varphi \mid \forall x : A.\varphi$$

to the syntax of terms, with typing rules as in Figure 3. The formulas are those of a higher-order logic with equality, but extended with connectives specific to the quantale Prop: $r\psi$ is interpreted as (1-bounded) rescaling by a factor $r \geq 0$, $\varphi \bullet \psi$ as the tensor, and $\varphi \multimap\!\!\bullet\, \psi$ as its adjoint. Note that the latter are not new to our logic, but common connectives of fuzzy logics (e.g., Łukasiewicz logic [26, 35] or the logic of Riesz MV algebras [32]).

For readability, we will often omit type annotations from $=_A$ and quantifiers when they can be inferred from the context.

We also add Prop to the grammar of IB algebra types, with operation defined as

$$\phi \oplus_p \psi \triangleq p\phi \bullet (1-p)\psi$$

and add the axioms of IB algebras (Definition 3) to the judgemental equality theory also for this derived operator on predicates.

The interpretation of predicates is defined in Figure 4. These are well-defined morphisms in **CMet** because of the following.

▶ **Lemma 14.** *The following are non-expansive maps (where $r > 0$)*

$$\oplus, \multimap : \mathsf{Prop} \otimes \mathsf{Prop} \to \mathsf{Prop} \qquad \min\{r \cdot -, 1\} : r\mathsf{Prop} \to \mathsf{Prop}$$
$$\sup, \inf : (\infty X \multimap \mathsf{Prop}) \to \mathsf{Prop} \qquad d_X : X \otimes X \to \mathsf{Prop}$$
$$\max, \min : \mathsf{Prop} \times \mathsf{Prop} \to \mathsf{Prop}$$

Observe that, after the extension of the calculus, independence of the choice of the derivation (Theorem 10) and soundness (Theorem 11) are still valid, as well as weakening and substitution lemmas for typing judgments (Lemmas 7 and 8) and their semantic variants (Lemmas 12 and 13).

$$
\begin{aligned}
[\![\mathsf{tt}]\!] &\triangleq 0 \\
[\![\mathsf{ff}]\!] &\triangleq 1 \\
[\![t =_A u]\!] &\triangleq d_{[\![A]\!]} \circ ([\![t]\!] \otimes [\![s]\!]) \circ \mathsf{split} \\
[\![\varphi \bullet \psi]\!] &\triangleq \oplus \circ ([\![\varphi]\!] \otimes [\![\psi]\!]) \circ \mathsf{split} \\
[\![\varphi \multimap\!\!\bullet\, \psi]\!] &\triangleq \multimap \circ ([\![\varphi]\!] \otimes [\![\psi]\!]) \circ \mathsf{split} \\
[\![r\varphi]\!] &\triangleq \begin{cases} 0 & (r = 0) \\ \min\{r \cdot -, 1\} \circ r[\![\varphi]\!] \circ \mathsf{dist} & (r > 0) \end{cases}
\end{aligned}
\qquad
\begin{aligned}
[\![\neg\varphi]\!] &\triangleq 1 - [\![\varphi]\!] \\
[\![\varphi \wedge \psi]\!] &\triangleq \max \circ \langle [\![\varphi]\!], [\![\psi]\!] \rangle \\
[\![\varphi \vee \psi]\!] &\triangleq \min \circ \langle [\![\varphi]\!], [\![\psi]\!] \rangle \\
[\![\exists x : A.\varphi]\!] &\triangleq \inf \circ \mathsf{curry}([\![\varphi]\!]) \\
[\![\forall x : A.\varphi]\!] &\triangleq \sup \circ \mathsf{curry}([\![\varphi]\!])
\end{aligned}
$$

Figure 4 Interpretation of logical predicates.

A *predicate in context* Γ is a term ϕ such that $\Gamma \vdash \phi : \mathsf{Prop}$. Observe that predicates can be constructed not just using logical connectives but also via the other constructions in our calculus. For example, since Prop is an IB algebra, if ϕ is a predicate in context $\Gamma, x :^r A$ and $\mu : \mathcal{D}A$, then $(\mathsf{let}\ x = \mu\ \mathsf{in}\ \phi)$ is also a predicate.

Logical reasoning on the terms of the calculus is done via the inference of logical judgments. The judgments of the logic are of the form

$$\Delta \mid \Psi \vdash \varphi \,,$$

where Δ is a typing context, $\Psi = \psi_1, \ldots, \psi_n$ is a list of predicates (*logical context*), and φ a predicate (*conclusion*).

Hereafter, we always assume to work with well-formed logical judgments:

▶ **Definition 15** (Well-formed judgments)**.** *A logical judgment* $\Delta \mid \Psi \vdash \varphi$ *is* well-formed *if*

- Δ *is a discrete context, i.e., all variables in* Δ *have sensitivity annotation* ∞*.*
- *all occurring predicates are well-typed in context* Δ*, i.e.,* $\Delta \vdash \varphi : \mathsf{Prop}$ *and* $\Delta \vdash \psi : \mathsf{Prop}$*, for all* $\psi \in \Psi$*.*

The reason for the first condition is that sensitivity factors for term variables in logical predicates are irrelevant for logical judgments, and keeping track of these adds unnecessary complications to the logic. For example, allowing more general Δ, for many rules it would not be the case that well-formedness of the assumptions implies well-formedness of the conclusion, nor vice-versa. This would also raise meta-theoretic questions about the logic that we prefer to avoid. Note that by Lemma 7, ∞ is the most general sensitivity annotation possible: If $\Delta \vdash t : A$ then also $\Delta' \vdash t : A$ where Δ' is obtained from Δ by setting all sensitivity annotations to ∞. In logical judgements we use the notation $\Delta, x : A$ as shorthand for the rigorous $\Delta, x :^\infty A$.

The inference rules for the logic are given in Figure 5. The notation $p\Psi$ means to multiply each proposition in Ψ by p. Rules given by double line are double rules, so can be used in both directions.

The logic is sound with respect to the semantic interpretation of logical judgments in the following sense, where the notation $[\![\psi_1, \ldots, \psi_n]\!]$ means $[\![\psi_1]\!] \oplus \cdots \oplus [\![\psi_n]\!]$.

▶ **Theorem 16** (Soundness)**.** *If* $\Delta \mid \Psi \vdash \varphi$ *is derivable, then* $[\![\Psi]\!](\delta) \geq [\![\varphi]\!](\delta)$ *for all* $\delta \in [\![\Delta]\!]$*.*

Note the similarity between the rule (G-REC) for guarded recursion and the typing rule for fixed points. The logic includes the classical rule ($\neg$-E), because it is verified by the model,

$$\frac{}{\Delta \mid \Psi \vdash \mathsf{tt}}\ (\text{TRUE}) \qquad \frac{}{\Delta \mid \Psi, \mathsf{ff} \vdash \varphi}\ (\text{FALSE}) \qquad \frac{}{\Delta \mid \Psi, \varphi \vdash \varphi}\ (\text{ASS})$$

$$\frac{\Delta \mid \Psi, \varphi, \psi, \Psi' \vdash \rho}{\Delta \mid \Psi, \psi, \varphi, \Psi' \vdash \rho}\ (\text{EX}) \qquad \frac{\Delta \mid \Psi \vdash \varphi}{\Delta \mid r\Psi \vdash r\varphi}\ (\text{PR}) \qquad \frac{\Delta \mid \Psi, (r+s)\varphi \vdash \psi}{\Delta \mid \Psi, r\varphi, s\varphi \vdash \psi}\ (\text{DUP})$$

$$\frac{\Delta \mid \Psi, \psi \vdash \varphi}{\Delta \mid \Psi, 1\psi \vdash \varphi}\ (\text{DER}) \qquad \frac{\Delta \mid \Psi, 0\psi \vdash \varphi}{\Delta \mid \Psi \vdash \varphi}\ (\text{ZCON}) \qquad \frac{\Delta \mid \Psi, r\psi \vdash \varphi \quad r \leq s}{\Delta \mid \Psi, s\psi \vdash \varphi}\ (\text{INC})$$

$$\frac{\Delta \mid \Psi, r(s\psi) \vdash \varphi}{\Delta \mid \Psi, (rs)\psi \vdash \varphi}\ (\text{ASSOC}_1) \qquad \frac{\Delta \mid \Psi, (rp)\psi \vdash \varphi \quad p \leq 1 \text{ or } r \geq 1}{\Delta \mid \Psi, r(p\psi) \vdash \varphi}\ (\text{ASSOC}_2)$$

$$\frac{\Delta \mid (1-p)\Psi, p\varphi \vdash \varphi \quad p < 1}{\Delta \mid \Psi \vdash \varphi}\ (\text{G-REC}) \qquad \frac{\Delta \mid \Psi \vdash \varphi \quad \Delta \mid \Psi' \vdash \varphi'}{\Delta \mid \Psi, \Psi' \vdash \varphi \bullet \varphi'}\ (\bullet\text{-I})$$

$$\frac{\Delta \mid \Psi, \varphi, \psi \vdash \rho}{\Delta \mid \Psi, \varphi \bullet \psi \vdash \rho}\ (\bullet\text{-E}) \qquad \frac{\Delta \mid \Psi, \varphi \vdash \psi}{\Delta \mid \Psi \vdash \varphi \multimap\!\bullet\, \psi}\ (\multimap\!\bullet\text{-I}) \qquad \frac{\Delta \mid \Psi \vdash \varphi \multimap\!\bullet\, \psi \quad \Delta \mid \Psi' \vdash \varphi}{\Delta \mid \Psi, \Psi' \vdash \psi}\ (\multimap\!\bullet\text{-E})$$

$$\frac{\Delta \mid \Psi, \varphi \vdash \mathsf{ff}}{\Delta \mid \Psi \vdash \neg\varphi}\ (\neg\text{-I}) \qquad \frac{\Delta \mid \Psi, \neg\varphi \vdash \mathsf{ff}}{\Delta \mid \Psi \vdash \varphi}\ (\neg\text{-E}) \qquad \frac{\Delta \mid \Psi \vdash r\varphi \quad \Delta \mid \Psi \vdash r\psi}{\Delta \mid \Psi \vdash r(\varphi \wedge \psi)}\ (\wedge\text{-I})$$

$$\frac{\Delta \mid \Psi \vdash \varphi \wedge \psi}{\Delta \mid \Psi \vdash \varphi}\ (\wedge\text{-EL}) \qquad \frac{\Delta \mid \Psi \vdash \varphi \wedge \psi}{\Delta \mid \Psi \vdash \psi}\ (\wedge\text{-ER}) \qquad \frac{\Delta \mid \Psi \vdash \varphi}{\Delta \mid \Psi \vdash \varphi \vee \psi}\ (\vee\text{-IL})$$

$$\frac{\Delta \mid \Psi \vdash \psi}{\Delta \mid \Psi \vdash \varphi \vee \psi}\ (\vee\text{-IR}) \qquad \frac{\Delta \mid \Psi, r\varphi \vdash \rho \quad \Delta \mid \Psi, r\psi \vdash \rho}{\Delta \mid \Psi, r(\varphi \vee \psi) \vdash \rho}\ (\vee\text{-E})$$

$$\frac{\Delta \vdash t : A \quad \Delta \mid \Psi \vdash \varphi[t/x]}{\Delta \mid \Psi \vdash \exists x : A.\varphi}\ (\exists\text{-I}) \qquad \frac{\Delta, x : A \mid \Psi, r\varphi \vdash \psi \quad r < \infty}{\Delta \mid \Psi, r(\exists x : A.\varphi) \vdash \psi}\ (\exists\text{-E})$$

$$\frac{\Delta, x : A \mid \Psi \vdash r\varphi}{\Delta \mid \Psi \vdash r(\forall x : A.\varphi)}\ (\forall\text{-I}) \qquad \frac{\Delta \mid \Psi \vdash \forall x : A.\varphi \quad \Delta \vdash t : A}{\Delta \mid \Psi \vdash \varphi[t/x]}\ (\forall\text{-E}) \qquad \frac{\Delta \vdash t \equiv s : A}{\Delta \mid \Psi \vdash t =_A s}\ (\text{EQ-I})$$

$$\frac{\Delta, x :^r A \vdash \varphi : \mathsf{Prop} \quad \Delta \vdash t : A \quad \Delta \vdash u : A \quad \Delta \mid \Psi \vdash \varphi[t/x] \quad \Delta \mid \Psi' \vdash r(t =_A u)}{\Delta \mid \Psi, \Psi' \vdash \varphi[u/x]}\ (\text{EQ-E})$$

$$\frac{\Delta, x : A, y : B \mid \Psi \vdash \varphi[(x,y)/z] \quad \Delta \vdash t : A_r{\otimes}_s B}{\Delta \mid \Psi \vdash \varphi[t/z]}\ (\text{IND}_\otimes)$$

$$\frac{\Delta \mid \Psi \vdash \varphi[\mathsf{zero}/n] \quad \Delta, n : \mathbb{N} \mid \varphi \vdash \varphi[\mathsf{succ}(n)/n] \quad \Delta \vdash t : \mathbb{N}}{\Delta \mid \Psi \vdash \varphi[t/n]}\ (\text{IND}_\mathbb{N})$$

$$\frac{\Delta, x : A \mid \Psi \vdash \varphi[inj_1(x)/z] \quad \Delta, y : B \mid \Psi \vdash \varphi[inj_2(y)/z] \quad \Delta \vdash t : A + B}{\Delta \mid \Psi \vdash \varphi[t/z]}\ (\text{IND}_+)$$

$$\frac{\begin{array}{cc} r < \infty & \\ \Delta \vdash t : \mathcal{D}A & \Delta, y : A \mid \Psi \vdash \varphi[\delta y/x] \\ \Delta, x :^r \mathcal{D}A \vdash \varphi : \mathsf{Prop} & \forall p.\left(\Delta, \mu : \mathcal{D}A, \nu : \mathcal{D}A \mid p\varphi[\mu/x], (1-p)\varphi[\nu/x] \vdash \varphi[\mu \oplus_p \nu/x]\right) \end{array}}{\Delta \mid \Psi \vdash \varphi[t/x]}\ (\text{IND}_\mathcal{D})$$

Figure 5 Logic. (The logical judgments appearing above are assumed to be well-formed) but our examples below do not use it. The rule (EQ-E) for elimination of equality is perhaps most easily understood via a sketch proof of soundness: The sensitivity of φ in x ensures that $\varphi[s/x]$ is at most at distance $r(t = s)$ from $\varphi[t/x]$. Therefore, since $[\![\Psi]\!] \geq [\![\varphi[t/x]]\!]$ and $[\![\Psi']\!] \geq [\![r(t = s)]\!]$, also $[\![\Psi]\!] \oplus [\![\Psi']\!] \geq [\![\varphi[s/x]]\!]$.

One consequence of the induction principles (IND$_\otimes$), (IND$_+$), (IND$_\mathbb{N}$), and (IND$_\mathcal{D}$) is that judgements involving predicates defined using recursion and let-bindings can be proved. For example, by (IND$_\otimes$), to prove

$$\Delta \mid \Psi, \mathsf{let}\ (x, y) = t\ \mathsf{in}\ \phi \vdash \mathsf{let}\ (x, y) = t\ \mathsf{in}\ \psi,$$

it suffices to show that $\Delta, x : A, y : B \mid \Psi, \phi \vdash \psi$. The induction principle for $\mathbb{N}$ requires the induction case to be proven not using Ψ, so that Ψ is only used once, at the base case. The use of convex combinations in the hypothesis of the induction principle for $\mathcal{D}$ ensures that any probability measure constructed inductively from Dirac distributions and convex combinations satisfies the conclusion. A priori, these only give the finite distributions and, semantically, $\mathcal{D}$ also contains continuous distributions. However, the finitely supported distributions are dense, and the requirement that r be finite means that φ is continuous in x, which suffices to verify soundness.

▶ Remark 17. The universal quantification over p in (IND$_\mathcal{D}$) makes it an infinitary rule. If one wants a finitary proof system, the induction case can be replaced by $\frac{1}{2}\varphi[\mu/x], \frac{1}{2}\varphi[\nu/x] \vdash \varphi[\mu \oplus_{\frac{1}{2}} \nu/x]$. This principle is still sound as the finitely supported dyadic distributions are also dense in $\mathcal{D}X$ (see e.g., proof of [39, Theorem 21]). We prefer to keep the rule as stated in Figure 5, because it follows the standard induction principle for inductive types: One proof obligation for each constructor.

The rules ($\wedge$-I), ($\vee$-E), ($\exists$-E), and ($\forall$-I) include scaling factors in the conclusions. This is to recover distributivity of scalar multiplication over the logical connectives, which cannot be otherwise derived due to the necessary restrictions in (ASSOC$_2$).

▶ **Lemma 18.** *For all $r \geq 0$, the predicates $r(\varphi \wedge \psi)$, $r(\varphi \vee \psi)$, and $r(\forall x : A.\varphi)$ are respectively equivalent to $r\varphi \wedge r\psi$, $r\varphi \vee r\psi$, and $\forall x : A.r\varphi$. If $r < \infty$, then $r(\exists x : A.\varphi)$ is equivalent to $\exists x : A.r\varphi$.*

The condition that $r < \infty$ in the last statement is necessary: $\infty(\exists x : (0, 1].x = 0)$ is interpreted as 0 in the model, but $\exists x : (0, 1].\infty(x = 0)$ is 1.

The logic is affine but not relevant, that is, weakening is derivable but contraction is not.

▶ **Lemma 19.** *If $\Delta \mid \Psi \vdash \varphi$ is derivable, so is $\Delta \mid \Psi, \psi \vdash \varphi$.*

Moreover, derivability is closed under weakening of the typing context and term substitution in predicates.

▶ **Lemma 20.**
1. *If $\Delta \mid \Psi \vdash \varphi$ is derivable, then so is $\Delta, x : A \mid \Psi \vdash \varphi$;*
2. *If $\Delta \vdash t : A$ and $\Delta, x : A \mid \Psi \vdash \varphi$ is derivable, then so is $\Delta \mid \Psi[t/x] \vdash \varphi[t/x]$.*

7 Proving basic properties

We show some basic consequences of the rules of the logic, focusing on equality.

We first show that the rule for equality elimination implies that equality is symmetric and transitive. The derivations mirror those of the corresponding rules for identity types in type theory (see e.g. [36]), although the setting of affine logic used here is different.

▶ **Proposition 21.** *Propositional equality is symmetric and transitive in the sense that*

$$x : A, y : A \mid r(x = y) \vdash r(y = x)$$
$$x : A, y : A, z : A \mid r(x = y), r(y = z) \vdash r(x = z)$$

Proof. We just prove transitivity. Let $\Delta \triangleq x : A, y : A, z : A$ and apply (EQ-E) to $\Delta, w :^r A \vdash r(x = w) : \mathsf{Prop}$. Because we have $\Delta \mid r(x = y) \vdash r(x = y)$ we obtain $\Delta \mid r(x{=}y), r(y{=}z) \vdash r(x{=}z)$. ◀

Note that since the comma is interpreted as 1-bounded addition, in the case where $r = 1$, transitivity corresponds to the triangle inequality. Equality is also a congruence relation.

▶ **Proposition 22** (Congruence)**.** *Let* $\Delta \vdash u : A$ *and* $\Delta \vdash v : A$*. If* $\Delta, x :^r A \vdash t : B$*, then* $\Delta \mid r(u = v) \vdash t[u/x] = t[v/x]$.

Proof. Apply (EQ-E) to $\Delta, y :^r A \vdash t[u/x] = t[y/x] : \mathsf{Prop}$. ◀

As a special case, for $\Delta = x : \mathcal{D}A, y : \mathcal{D}A, z : \mathcal{D}A, w : \mathcal{D}A$, we have

$$\Delta \mid p(x = y), (1 - p)(z = w) \vdash x \oplus_p z = y \oplus_p w\,. \tag{4}$$

Using (IND$_\otimes$) and (IND$_+$), we can prove the following extensionality principles.

▶ **Lemma 23.** *If* $x, y : A\ {}_r\!\otimes_s B$ *then* $x = y$ *is equivalent to*

$$\mathsf{let}\ (a, b) = x, (a', b') = y\ \mathsf{in}\ r(a = a') \bullet s(b = b').$$

▶ **Lemma 24.** *If* $x, y : A + B$ *then* $x = y$ *is equivalent to*

$$\big(\exists a, a'.(x = inj_1 a) \bullet (y = inj_1 a') \bullet (a = a')\big) \vee \big(\exists b, b'.(x = inj_2 b) \bullet (y = inj_2 b') \bullet (b = b')\big)$$

Since Prop is a type in our calculus, the logic is higher order. We can define types of predicates and relations

$$\mathsf{Pred}_r(A) \triangleq A \multimap_r \mathsf{Prop} \qquad\qquad \mathsf{Rel}_{r,s}(A, B) \triangleq A\ {}_r\!\otimes_s B \multimap_1 \mathsf{Prop}\,. \tag{5}$$

We simply write $\mathsf{Pred}(A)$ and $\mathsf{Rel}(A, B)$ as shorthands for $\mathsf{Pred}_1(A)$ and $\mathsf{Rel}_{1,1}(A, B)$, respectively, when all the sensitivities are 1.

One can prove that equality is equivalent to Leibniz equality.

▶ **Proposition 25.** *The predicates* $\forall \phi : \mathsf{Pred}_r(A).\phi(x) \multimap\!\!\bullet\ \phi(y)$ *and* $r(x =_A y)$ *are equivalent.*

7.1 Internalising the Kantorovich distance

We show that the Kantorovich distance (3) can be internalised in the logic. This, in turn, will enable reasoning via couplings to establish equalities between probability distributions. The integral in the definition of the Kantorovich distance computes the mean of the distance $d_X(x, y)$ when (x, y) is distributed according to the given coupling ω. We start by defining, more generally, the mean $E_{x\sim\mu}[\phi]$ of a predicate $\phi : \mathsf{Pred}(A)$ over a distribution $\mu : \mathcal{D}A$ as

$$E_{x\sim\mu}[\phi] \triangleq \mathsf{let}\ x = \mu\ \mathsf{in}\ \phi(x) \tag{6}$$

This defines a mean because it satisfies the equations

$$E_{x\sim\delta(y)}[\phi] \equiv \phi(y) \tag{7}$$
$$E_{x\sim\mu\oplus_p\mu'}[\phi] \equiv p(E_{x\sim\mu}[\phi]) \bullet (1 - p)(E_{x\sim\mu'}[\phi])\,. \tag{8}$$

For $R : \mathsf{Rel}(A, B)$ and $\omega : \mathcal{D}(A\,{}_1{\otimes}_1 B)$ we write $E_{(a,b)\sim\omega}[R(a, b)]$ rather than $E_{z\sim\omega}[R]$ to emphasize that R is a relation. When R is the equality relation we write $E_{(x,y)\sim\omega}[x = y]$. The notion of R-coupling can be expressed quantitatively in the logic as

$$\omega \in \mathsf{Cpl}(\mu, \nu) \triangleq (\mathcal{D}(\pi_1)\omega = \mu) \bullet (\mathcal{D}(\pi_2)\omega = \nu)\,, \tag{9}$$

$$\omega \in \mathsf{Cpl}_R(\mu, \nu) \triangleq E_{(x,y)\sim\omega}[R(x, y)] \bullet \omega \in \mathsf{Cpl}(\mu, \nu)\,. \tag{10}$$

Notably, $\omega \in \mathsf{Cpl}(\mu, \nu)$ evaluates to 0 (the distinguished truth value in Prop) if and only if ω is a coupling between μ and ν. However, when it evaluates to a value different from 0, ω is not necessarily a coupling. Thus, (9) captures a notion that is strictly weaker and more permissive than that of an actual coupling.

Finally, using that existential quantification is interpreted as infimum, we can internalise the Kantorovich distance as follows

$$K(\mu, \nu) \triangleq \exists\omega : \mathcal{D}(A\,{}_1{\otimes}_1 A).\, \omega \in \mathsf{Cpl}_{\mathsf{eq}}(\mu, \nu)\,,$$

where $\mathsf{eq} \triangleq \lambda z.\mathsf{let}\ (x, y) = z\ \mathsf{in}\ x = y : \mathsf{Rel}(A, A)$ is the equality relation.

▶ **Theorem 26.** *The predicates $K(\mu, \nu)$ and $\mu = \nu$ are equivalent.*

Proof. To prove that $\mu = \nu$ implies $K(\mu, \nu)$, by (EQ-E) it suffices to show

$$\mu : \mathcal{D}A \mid \cdot \vdash K(\mu, \mu)$$

Define $\mu :^2 \mathcal{D}A \vdash \omega(\mu) : \mathcal{D}(A\,{}_1{\otimes}_1 A)$ as $\mathsf{let}\ a = \mu\ \mathsf{in}\ \delta(a, a)$. Then $\mathcal{D}\pi_i(\omega(\mu)) = \mu$ for $i = 1, 2$. Moreover, $\cdot \vdash E_{(a,a')\sim\omega(\mu)}[a = a']$ can be proved by induction on μ as follows. If $\mu = \delta(a)$, by (7) and $\omega(\mu) \equiv \delta(a, a)$, $E_{(a,a')\sim\omega(\mu)}[a = a']$ reduces to $a = a$, which is true. If $\mu = \mu_1 \oplus_p \mu_2$, we must show $E_{(a,a')\sim\omega(\mu_1\oplus_p\mu_2)}[a = a']$ in context

$$p(E_{(a,a')\sim\omega(\mu_1)}[a = a']), (1 - p)(E_{(a,a')\sim\omega(\mu_2)}[a = a'])$$

which holds by (8) because $\omega(\mu_1 \oplus_p \mu_2) \equiv \omega(\mu_1) \oplus_p \omega(\mu_2)$. For the other direction, it suffices to show that $E_{(x,y)\sim\omega}[x{=}y]$ implies $\mathcal{D}(\pi_1)(\omega) = \mathcal{D}(\pi_2)(\omega)$, for any $\omega : \mathcal{D}(A\,{}_1{\otimes}_1 A)$. This can be done by induction on ω. ◀

7.2 Uniqueness of fixed points

One might hope that the uniqueness of fixed points from the Banach fixed point theorem can be internalised in our logic as the statement

$$x = f(x), y = f(y) \vdash x = y$$

whenever $f : X \multimap_p X$ for $p < 1$. However, this is not true. Take for example $f : \mathsf{Prop} \multimap_{\frac{1}{2}} \mathsf{Prop}$ to be multiplication by $\frac{1}{2}$, x to be 0 and y to be 1. Then the semantics of the above statement evaluates to the false statement $\frac{1}{2} \geq 1$. However, if we assume that x is a fixed point for f in the global sense, then it equals the unique fixed point.

▶ **Lemma 27.** *Let $f :^1 X \multimap_p X$ for $p < 1$. Then,* $\mathsf{tt} \vdash x = f(x)$ *implies* $\mathsf{tt} \vdash x = \mathsf{fix}\ y.f(y)$.

Proof. By (G-REC), it suffices to prove that $p(x = \mathsf{fix}\ y.f(y)) \vdash x = \mathsf{fix}\ y.f(y)$. This follows from $\mathsf{tt} \vdash x = f(x)$ and transitivity of propositional equality (Proposition 21) after observing that $p(x = \mathsf{fix}\ y.f(y)) \vdash f(x) = f(\mathsf{fix}\ y.f(y))$, which is true by Proposition 22. ◀

8 Case study: Markov processes

Markov processes describe systems with memoryless transitions between states, governed by probabilities. Formally, they consist of a set of states $\mathcal{S}$, a transition function $\mathcal{S} \to \mathcal{D}(\mathcal{S})$ that specifies the probabilities of moving to the next state, and a labeling function $\mathcal{S} \to A$ that assigns labels to states. Following [37], we treat A as a metric space and analyze the behavior of Markov processes quantitatively using distances that discount future differences in observations by means of a discount factor $c \in (0, 1]$. The smaller c is, the more the focus shifts toward short-term differences. Categorically, this corresponds to interpreting Markov processes as coalgebras $S \to A \otimes c\mathcal{D}(S)$ in **CMet**. The behaviour of a state is abstractly characterised as an element of the final coalgebra[2], corresponding to the coinductive solution $\mathbb{P}_c$ to the functorial equation $\mathbb{P}_c \cong A \otimes c\mathcal{D}(\mathbb{P}_c)$. The behavioral distance is just the distance in $\mathbb{P}_c$ between behaviours [38].

In order to program with $\mathbb{P}_c$ we extend the calculus with types $\mathbb{P}_c$ and A as well as terms

$$\mathsf{ufld} : \mathbb{P}_c \multimap_1 A\ {}_1{\otimes}_c \mathcal{D}(\mathbb{P}_c) \qquad\qquad \mathsf{fld} : A\ {}_1{\otimes}_c \mathcal{D}(\mathbb{P}_c) \multimap_1 \mathbb{P}_c$$

We will write $a; m$ for $\mathsf{fld}\,(a, m)$. Finally we add judgemental equalities stating that fld and ufld are inverses of each other.

As a first example, consider a process m satisfying the recursive definition $m \equiv a; (\delta(m) \oplus_{\frac{1}{3}} \delta(z))$ where z is some other given process. This recursive definition is productive in the sense that it only calls itself with probability $\frac{1}{3}$. Therefore it can be defined as a term of type $\mathbb{P}_1$ similarly to the definition of the geometric distribution of Example 6. Precisely, because

$$z :^{\frac{2}{3}} \mathbb{P}_1, m :^{\frac{1}{3}} \mathbb{P}_1 \vdash a; (\delta(m) \oplus_{\frac{1}{3}} \delta(z)) : \mathbb{P}_1$$

we can define $z :^1 \mathbb{P}_1 \vdash m : \mathbb{P}_1$ as

$$m \triangleq \mathsf{fix}\ m.a; (\delta(m) \oplus_{\frac{1}{3}} \delta(z))$$

which then by the equality for fixed point unfolding satisfies the desired equality.

Now, let n satisfying $n \equiv a; (\delta(n) \oplus_{\frac{1}{2}} \delta(z))$ be defined similarly. Using the logic we will now show that the distance between m and n is at most $\frac{1}{4}$, which in the logic corresponds to showing that $\frac{1}{4}\mathsf{ff} \vdash m = n$. In the following, for $r > 0$, we simply write r for $r\,\mathsf{ff}$. By guarded recursion (G-REC), it suffices to show that

$$\frac{2}{3} \cdot \frac{1}{4}, \frac{1}{3}(m = n) \vdash m = n$$

Since

$$n = a; (\delta(n) \oplus_{\frac{1}{3}} (\delta(n) \oplus_{\frac{1}{4}} \delta(z))) \qquad\qquad m = a; (\delta(m) \oplus_{\frac{1}{3}} (\delta(z) \oplus_{\frac{1}{4}} \delta(z)))$$

by (4) and Proposition 22 it suffices to show

$$\frac{2}{3} \cdot \frac{1}{4}, \frac{1}{3}(m = n) \vdash \frac{1}{3}(m = n) \bullet \frac{2}{3}\left(\frac{1}{4}(n = z) \bullet \frac{3}{4}(z = z)\right)$$

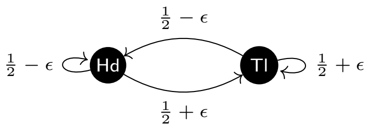
which in turn reduces to the following three judgements, all of which are true:

$$m = n \vdash m = n \qquad\qquad \mathsf{ff} \vdash n = z \qquad\qquad \mathsf{tt} \vdash z = z$$

[2] The final coalgebra always exists as $A \otimes c\mathcal{D}(-)$ is an accessible functor. See [38, 37] for details.

8.1 A biased coin tossing process

The next example describes a probabilistic process generated by a coin toss with a biased coin, where the current state remembers the result of the last coin toss. The label space is the discrete set $A = \{\mathsf{Hd}, \mathsf{Tl}\}$ and the two states should satisfy the mutually recursive equations



$$\begin{aligned}\mathsf{hd}_\epsilon &\equiv \mathsf{Hd}; (\delta(\mathsf{hd}_\epsilon) \oplus_{\frac{1}{2}-\epsilon} \delta(\mathsf{tl}_\epsilon)) \\ \mathsf{tl}_\epsilon &\equiv \mathsf{Tl}; (\delta(\mathsf{hd}_\epsilon) \oplus_{\frac{1}{2}-\epsilon} \delta(\mathsf{tl}_\epsilon))\,.\end{aligned}$$

Unlike the previous example, this definition is not productive, so $\mathsf{hd}_\epsilon, \mathsf{tl}_\epsilon$ cannot be defined by guarded recursion as elements of $\mathbb{P}_1$. However, they can be defined as elements of $\mathbb{P}_c$ for any $c \in (0, 1)$. We define them by mutual recursion using a fixed point of a contraction on $\mathbb{P}_c \times \mathbb{P}_c$, as $\mathsf{hd}_\epsilon \triangleq \pi_1(\mathsf{hdtl}_\epsilon)$ and $\mathsf{tl}_\epsilon \triangleq \pi_2(\mathsf{hdtl}_\epsilon)$ for

$$\mathsf{hdtl}_\epsilon \triangleq \mathsf{fix}\ x.\langle \mathsf{Hd}; \mathsf{flip}_\epsilon(x), \mathsf{Tl}; \mathsf{flip}_\epsilon(x)\rangle$$

where $\mathsf{flip}_\epsilon(x) \triangleq \delta(\pi_1(x)) \oplus_{\frac{1}{2}-\epsilon} \delta(\pi_2(x))$. Observe that hdtl_ϵ is well-defined as

$$x :^c \mathbb{P}_c \times \mathbb{P}_c \vdash \langle \mathsf{Hd}; \mathsf{flip}_\epsilon(x), \mathsf{Tl}; \mathsf{flip}_\epsilon(x)\rangle : \mathbb{P}_c \times \mathbb{P}_c\,.$$

Consider the special case of a fair coin $\mathsf{hd} \triangleq \mathsf{hd}_0$, $\mathsf{tl} \triangleq \mathsf{tl}_0$. We now show that the distance between hd and hd_ϵ is at most $\frac{c\epsilon}{1-c+c\epsilon}$. Logically, this corresponds to the statement

$$\frac{c\epsilon}{1-c+c\epsilon} \vdash \mathsf{hd} = \mathsf{hd}_\epsilon \tag{11}$$

The statement (11) must be proved simultaneously with a similar statement for the two tail states, and by guarded recursion it suffices to prove that, for $\mathsf{d}_{c,\epsilon} \triangleq \frac{c\epsilon}{1-c+c\epsilon}$, we have

$$c\,(\mathsf{d}_{c,\epsilon} \multimap\!\bullet (\mathsf{hd}{=}\mathsf{hd}_\epsilon \wedge \mathsf{tl}{=}\mathsf{tl}_\epsilon)) \vdash (\mathsf{d}_{c,\epsilon} \multimap\!\bullet (\mathsf{hd}{=}\mathsf{hd}_\epsilon \wedge \mathsf{tl}{=}\mathsf{tl}_\epsilon))\ .$$

We show that

$$c\,(\mathsf{d}_{c,\epsilon} \multimap\!\bullet (\mathsf{hd} = \mathsf{hd}_\epsilon \wedge \mathsf{tl} = \mathsf{tl}_\epsilon))\,, \mathsf{d}_{c,\epsilon} \vdash \mathsf{hd} = \mathsf{hd}_\epsilon \tag{12}$$

The case for $\mathsf{tl} = \mathsf{tl}_\epsilon$ is similar. By (4) to show $\mathsf{hd} = \mathsf{hd}_\epsilon$ it suffices to show c times the formula

$$\left(\frac{1}{2} - \epsilon\right)(\mathsf{hd} = \mathsf{hd}_\epsilon) \bullet \epsilon(\mathsf{hd} = \mathsf{tl}_\epsilon) \bullet \frac{1}{2}(\mathsf{tl} = \mathsf{tl}_\epsilon) \tag{13}$$

and so (12) reduces by rule (PR) to showing (13) in context

$$\mathsf{d}_{c,\epsilon} \multimap\!\bullet (\mathsf{hd} = \mathsf{hd}_\epsilon \wedge \mathsf{tl} = \mathsf{tl}_\epsilon)\,, \frac{\epsilon}{1-c+c\epsilon} \tag{14}$$

Since

$$\frac{\epsilon}{1-c+c\epsilon} = \left(\frac{1}{2} - \epsilon\right)\mathsf{d}_{c,\epsilon} + \epsilon + \frac{1}{2}\mathsf{d}_{c,\epsilon}$$

Using (DUP) and (INC) it suffices to prove (13) in context

$$\left(\frac{1}{2} - \epsilon\right)(\mathsf{d}_{c,\epsilon} \multimap\!\bullet (\mathsf{hd} = \mathsf{hd}_\epsilon \wedge \mathsf{tl} = \mathsf{tl}_\epsilon))\,, \frac{1}{2}\,(\mathsf{d}_{c,\epsilon} \multimap\!\bullet (\mathsf{hd} = \mathsf{hd}_\epsilon \wedge \mathsf{tl} = \mathsf{tl}_\epsilon))\,, \left(\frac{1}{2} - \epsilon\right)\mathsf{d}_{c,\epsilon}, \epsilon, \frac{1}{2}\mathsf{d}_{c,\epsilon}$$

which can be done using (•-I).

We remark that the upper bound shown above is tight, in the sense that it is the actual behavioual distance between the two states. This can be checked by direct calculations because, as observed in [40], it corresponds to the c-discounted bisimilarity distance of Desharnais et al. [16].

8.2 Bisimulation

Let $\xi\colon X \to A \otimes c\mathcal{D}(X)$ be a Markov process, and decompose ξ into two maps: $\xi_1\colon X \to A$ for labels and $\xi_2\colon X \to c\mathcal{D}(X)$ for probabilistic transitions. A probabilistic bisimulation [24] for ξ is a binary relation on X such that $R(x, y)$ implies (i) $\xi_1(x) = \xi_1(y)$, states have the same label; and (ii) that there exists an R-coupling ρ (*i.e.*, a coupling ρ for $\xi_2(x)$ and $\xi_2(y)$ whose support is in R) ensuring the probabilistic behaviors of x and y remain related under the bisimulation.

While exact equivalence is often too rigid, metrics offer a more flexible alternative. Next, we internalise the definition of bisimilarity in our logic. For $R : \mathsf{Rel}(X, X)$, define

$$\mathsf{Bisim(R)} \triangleq \forall x, y : X.R(x, y) \multimap\!\bullet\ \mathsf{let}\ (l, \mu) = \xi(x),\ (l', \nu) = \xi(y)$$
$$\mathsf{in}\ l = l' \bullet c(\exists \rho.\rho \in \mathsf{Cpl}_R(\mu, \nu))$$

using the quantitative notion of R-coupling defined in (10).

In the case where $c \in (0, 1)$, we can define the bisimilarity relation $\sim : \mathsf{Rel}(X, X)$ by guarded recursion as

$$\sim \triangleq \mathsf{fix}\ R.\lambda x.\lambda y.\mathsf{let}\ (l, \mu) = \xi(x),\ (l', \nu) = \xi(y)\ \mathsf{in}\ l = l' \bullet c(\exists \rho.\rho \in \mathsf{Cpl}_R(\mu, \nu))$$

using that R occurs with sensitivity c in the body of the fixed point. The following can be proved using guarded recursion.

▶ **Proposition 28.** *Bisimilarity is equivalent to equality in $\mathbb{P}_c$.*

Proof (sketch). We just sketch the proof of bisimilarity implying equality. The proof is by guarded recursion, reducing to $c(\forall x, y.x \sim y \multimap\!\bullet\ x = y), x \sim y \vdash x = y$. The assumption $x \sim y$ unfolds to

$$\mathsf{let}\ (l, \mu) = \mathsf{ufld}(x),\ (l', \nu) = \mathsf{ufld}(y)\ \mathsf{in}\ l = l' \bullet c(\exists \rho.\rho \in \mathsf{Cpl}_\sim(\mu, \nu))$$

and using the guarded recursion hypothesis, the last part can be rewritten to $c(\exists \rho.\rho \in \mathsf{Cpl}_{\mathsf{eq}}(\mu, \nu))$, which by Theorem 26 is equivalent to $c(\mu = \nu)$. We conclude by Lemma 23. ◀

Since the distance in $\mathbb{P}_c$ coincides with probabilistic bisimilarity distance, Proposition 28 shows that $\sim : \mathsf{Rel}(\mathbb{P}_c, \mathbb{P}_c)$ is interpreted as bisimilarity distance.

9 Case study: temporal learning

The next example, adapted from Aguirre et al [2], showcases the expressivity of our calculus and a use of natural number induction.

A *Markov decision process* comprises a set of states $\mathcal{S}$, a set of actions $\mathcal{A}$, a transition function $\mathcal{P}\colon \mathcal{S} \times \mathcal{A} \to \mathcal{D}(\mathcal{S})$ and a reward function $\mathcal{R}\colon \mathcal{S} \times \mathcal{A} \to \mathcal{D}([0, r])$. We will consider $\mathcal{A}$ a discrete set, and assume that states $\mathcal{S}$ is a finite discrete set. Moreover, for simplicity we will assume that $\mathcal{S}$ is simply the set $\mathcal{S} = \{0, \ldots, N-1\}$ for some N, so that $\mathcal{S} \multimap X$ can be expressed as an N-fold product X^N. Finally, we will assume that $r = 1$, so that we can give the type of $\mathcal{R}$ internally in the calculus as $\mathcal{S}\ {}_1\!\otimes_1\!\mathcal{A} \multimap \mathcal{D}(\mathsf{Prop})$. The latter is not a restriction in practice, as the actual values of rewards is inessential, and so the reward function can be appropriately scaled. However, it is necessary, as we need the reward space to be an IB-algebra and a 1-bounded metric space.

When doing reinforcement learning, one must estimate a value function $V : \mathsf{Prop}^N$ mapping states to rewards, for a given policy $\pi : \mathcal{A}^N$. *Temporal difference* is one approach to doing this, which works by iteratively refining the value function V as follows: For each state i, compute an action a from the policy distribution $\pi(i)$, sample a reward r from the reward distribution $\mathcal{R}(a, i)$, and sample a transition j from the transition function $\mathcal{P}(a, i)$. From this the updated value function V' at i can be defined as the convex combination

$$V'(i) = (1-\alpha)V(i) + \alpha(r + \gamma V(j))$$

of the previous value $V(i)$ and the reward associated with the next state j, for fixed values $\alpha, \gamma \in (0, 1)$. Of course, $V'(i)$ should be a distribution, since the definition above involves sampling.

We will show that this refinement function can be defined and proved convergent in our logic. We first define the function $\mathsf{TDstep}\, V : \mathcal{D}(\mathsf{Prop}^N)$ taking one step of the refinement in context

$$\mathcal{P} :^\infty \mathcal{A} \multimap \mathcal{D}(N)^N, \mathcal{R} :^\infty \mathcal{A} \multimap \mathcal{D}(\mathsf{Prop})^N, \pi :^\infty \mathcal{A}^N, V :^k \mathsf{Prop}^N$$

where $k \triangleq 1 - \alpha + \gamma\alpha$. Note that all the parameters of the reinforcement learning setup are given sensitivity ∞, because these are assumed to be closed terms that will be called repeatedly. We define $\mathsf{TDstep}\, V$ as $\mathsf{st}(\mathsf{TDstep}'\, V)$ where

$$\mathsf{st} : \mathcal{D}(\mathsf{Prop})^N \multimap \mathcal{D}(\mathsf{Prop}^N)$$

is obtained in the standard way by induction on N, and

$$\mathsf{TDstep}'\, V \triangleq \langle \mathsf{let}\ r = \mathcal{R}(\pi(i))(i),\ j = \mathcal{P}(\pi(i))(i)\ \mathsf{in}\ \delta((1-\alpha)V(i) \bullet \alpha(r \bullet \gamma V(j)))\rangle_{i \leq N}$$

Since $k < 1$, one can define the refinement function as the fixed point of TDstep. In practice, however, refinement is only iterated a finite number n of times, defining $\mathsf{TD} : \mathsf{Prop}^N \multimap \mathbb{N} \multimap \mathcal{D}(\mathsf{Prop}^N)$ by recursion on the second argument as

$$\mathsf{TD}\, V\, 0 \triangleq V \qquad \mathsf{TD}\, V\, (n+1) \triangleq \mathsf{let}\ V' = (\mathsf{TD}\, V\, n)\ \mathsf{in}\ \mathsf{TDstep}\, V'$$

Then, since $k(V = W) \vdash \mathsf{TDstep}\, V = \mathsf{TDstep}\, W$, one can prove by induction on n that

$$k^n(V = W) \vdash \mathsf{TD}\, V\, n = \mathsf{TD}\, W\, n$$

10 Case study: hypercube walk

This section shows how the internalisation of the Kantorovich distance (Theorem 26) can be used for coupling proofs in our logic. The example is a random walk on a hypercube adapted from Aguirre et al. [2]. Much of the example is done by reasoning in the model, as is most natural. Our logic is then used as an internal language of the model to apply Theorem 26 in the last step.

A position on an N-dimensional hypercube is an element of Bool^N, and we consider this a metric space with the normalised Hamming distance: The distance between p and q is

$\frac{1}{N}$ times the number of positions where p and q differ. In other words, the metric space of positions can be defined as

$$\mathsf{Pos} \triangleq \bigotimes_{i=1}^{N} \frac{1}{N}\mathsf{Bool}$$

where $\mathsf{Bool} \triangleq \mathbf{1}+\mathbf{1}$. Let $\mathsf{unif}_{0,N} : \mathcal{D}(\mathbb{N})$ be the uniform distribution on $\{0,\dots,N\}$, and $\mathsf{flip}_i : \mathsf{Pos} \to \mathsf{Pos}$ be the operation that flips the i-th coordinate of a position if $i = 1,\dots,N$, and acts as the identity if $i = 0$. Define the one-step hypercube random walk as

$$\mathsf{hwalk} \triangleq \lambda p.\mathsf{let}\ i = \mathsf{unif}_{0,N}\ \mathsf{in}\ \mathsf{flip}_i(p) : \mathsf{Pos} \multimap_1 \mathcal{D}(\mathsf{Pos})$$

We show that

$$\frac{N-1}{N+1}(p = q) \vdash \mathsf{hwalk}\,p = \mathsf{hwalk}\,q \tag{15}$$

from which one can show that repeated iteration of hwalk converges. To prove this, first construct, for each pair of positions p and q, a bijection $\sigma_{p,q}$ of $\{0,\dots,N\}$ to itself by cases:

- If p and q are equal take $\sigma_{p,q}$ to be the identity
- If p and q differ in exactly one position i, let $\sigma_{p,q}$ be the permutation that swaps i and 0
- If p and q differ in positions $i_1,\dots,i_n$ for $n > 1$, let $\sigma_{p,q}$ be the permutation that cycles $i_1,\dots,i_n$.

Below we just write σ for $\sigma_{p,q}$. One can then show that

$$\frac{N-1}{N+1}(p = q) \vdash \sum_{i=0}^{N} \frac{1}{N+1}(\mathsf{flip}_i\,p = \mathsf{flip}_{\sigma(i)}\,q) \tag{16}$$

holds in the model. This is done by analysing cases of p and q. For example, if p and q differ in exactly one position j, then $\mathsf{flip}_i\,p = \mathsf{flip}_{\sigma(i)}\,q$ is 0 for $i = j$ and $i = 0$, and at all other values it equals $p = q$, from which the judgement follows.

Let $\rho : \mathcal{D}(\mathbb{N}\,{}_1\!\otimes_1\mathbb{N})$ be the uniform distribution on the finite set $\{(0,\sigma(0)),\dots,(N,\sigma(N))\}$, and define

$$\rho' \triangleq \mathsf{let}\ z = \rho\ \mathsf{in}\ \big(\mathsf{let}\ (i,j) = z\ \mathsf{in}\ (\mathsf{flip}_i\,p, \mathsf{flip}_j\,q)\big) : \mathcal{D}(\mathsf{Pos}_1\!\otimes_1\mathsf{Pos})$$

Then

$$E_{(x,y)\sim\rho'}[x = y] = \sum_{i=0}^{N} \frac{1}{N+1}(\mathsf{flip}_i\,p = \mathsf{flip}_{\sigma(i)}\,q)$$

An easy argument shows that the equalities $\mathcal{D}(\pi_1)(\rho') = \mathsf{hwalk}\,p$ and $\mathcal{D}(\pi_2)(\rho') = \mathsf{hwalk}\,q$ can be proved in the empty context, so that we have shown

$$\frac{N-1}{N+1}(p = q) \vdash K(\mathsf{hwalk}\,p, \mathsf{hwalk}\,q)$$

which, by Theorem 26 is equivalent to (15).

► Remark 29. We remark that while the example above can be done for any N, the quantification over N must be external, as internalising it would require dependent types. Similarly, the uniform distribution $\mathsf{unif}_{0,N} : \mathcal{D}(\mathbb{N})$ on $\{0,\dots,N\}$ used in the example, can be constructed in our language for each N as externally quantified only. It is not possible to write a function mapping N to $\mathsf{unif}_{0,N}$ in our language, because the sensitivity annotations of variables must be given externally.

11 Related work

Our calculus is closely related to Fuzz [34]. One difference is that Fuzz has recursive types, and we have guarded recursion. Fuzz is not a calculus of metric spaces, since these do not model general recursion. Indeed, de Amorim et al. [14] use metric CPOs to model Fuzz. Interestingly, de Amorim et al. [14] note that a fixed point operation on terms encoded using the recursive types of Fuzz can be given a typing rule similar to our rule for fixed points. Fuzz also has a probability distributions monad, but with a different metric designed for reasoning about differential privacy. Unlike the Kantorovich metric, it is not clear whether such a metric allows for a principle of induction. Fuzz also has an operational semantics. We believe operational semantics could be given for our calculus as well, but leave this to future work. Our calculus is also related to graded lambda calculus [10, 18].

Dagnino and Pasquali [11] were the first to notice that the sensitivity of predicates must be taken into account when expressing elimination principles for equality in quantitative logic. They present an affine propositional logic for quantitative reasoning about terms written in a first-order language where the operations carry sensitivity annotations, like e.g., $\oplus_p$ does in this paper. One of the rules they present for equality is our rule (EQ-E), but transitivity is a separate axiom, and they do not study applications like the ones studied here. Instead, Dagnino and Pasquali [11] study general categorical notions of models; we just study one model.

The idea of using metric spaces and guarded recursion using scaling factors $r < 1$ for programming and reasoning about probabilistic processes goes back at least to the late 1990s [15, 7]. To our knowledge, all previous work has used ultra-metric spaces, which means that one can use simply typed lambda calculus and a simpler type for fixed points, as explained in the introduction. The category of complete bisected ultrametric spaces forms a subcategory of the topos of trees [9], so this line of work is connected to the recent developments on reasoning about processes using guarded recursion [1, 22]. In these works, guarded recursion is formulated for a modal operator $\triangleright$, which is not related to probabilities. Equality, therefore, is not interpreted in terms of Kantorovich distance, as it is here.

The work on quantitative equational logic [27, 28, 29] is also related. However, their equational approach fundamentally differs from ours by using a Boolean-valued logical relation $=_\epsilon$ to reason about distances. Scaling of propositions and guarded recursion as used here would not work for a Boolean-valued logic. Indeed, the upper bound shown in Section 8.1 can be proven in quantitative equational logic only by using the infinitary rule (Arch) [6]. Mardare et al. [27] show how a range of monads on metric spaces can be considered generated by quantitative algebraic theories. These include the p-Wasserstein metrics on distributions and the Hausdorff metric on compact subsets of a metric space. It is unclear how many of these can be captured as quantitative algebras for theories with operations equipped with sensitivity factors, in the same way we algebraically encode the Kantorovich metric in this paper. While this approach also works for the Hausdorff metric, it is unlikely that the p-Wasserstein metrics for $p \neq 1$ can be encoded the same way. [29] use quantitative equational logic to reason about terms of a first-order language with fixed points modelled using the Banach fixed point theorem. Their syntax uses ‘Banach patterns’ as a form of sensitivity annotations on terms. Interestingly, the Banach pattern for fixed point terms is similar to our typing rule (FIX). We are unaware of any extensions of quantitative equational logic to higher-order. Dal Lago et al. [23] extends quantitative equational logic to weak λ-theories, and Dahlqvist and Neves [12, 13] to linear and affine λ-calulus. Both extensions do not deal with recursion.

Quantitative program logics for imperative languages with probabilistic choice operators have a long history going back to Kozen [21]. For example, Avanzini et al. [4] define a relational Hoare logic where relations on stores $s : \mathbb{S}$ are random variables $\mathbb{S} \times \mathbb{S} \to [0, \infty]$, so assertions are quantitative, but the interpretation of Hoare triples is qualitative. Recent research has extended this idea to give quantitative interpretations also to Hoare triples themselves. For example, Approxis [19] uses error credits to define an approximate separation logic. In future work we will explore how our logic offers a different viewpoint on these results: By reading the qualitative definition of Hoare quadruple in our logic, one obtains a quantitative interpretation of Hoare logic. Approximate statement about Hoare quadruples should then be provable in our logic by showing that $\epsilon \cdot \mathsf{ff}$ implies a Hoare quadruple similarly to how we reason about distances between Markov processes in Section 8.

12 Conclusions

We defined an affine calculus for sensitivity and a higher-order logic for reasoning about it. The calculus includes a form of guarded recursion, where the guards are sensitivities in the open interval $(0, 1)$, and we saw how this could be used for programming recursive processes. The logic likewise includes guarded recursion, which can be used, e.g., for proving upper bounds on distances between processes. We also saw how the principles of induction in the logic, in particular the one for induction over $\mathcal{D}$ are powerful principles. For example, we saw how they lead to proofs by coupling.

One might ask to what extent the semantics of our logic generalises to other settings. Our goal has been to reason about metric spaces, and **CMet** is essentially the largest possible category in which the entire logic can be interpreted soundly. For example, we need to include discrete sets into the category to model natural numbers. This requires either a finite upper limit on distances as we do, or using extended metric spaces, *i.e.*, spaces where the distance function maps into $[0, \infty]$, and correspondingly use $[0, \infty]$ also as the set of truth values. The latter choice, however, invalidates both the Banach fixed point theorem (fixed points might not exist or be unique) and the guarded recursion principle.

On the other hand, it should be possible to adapt our language and logic to extended metric spaces by introducing a new form of judgements for finiteness of truth values and for extended metric spaces being metric spaces (so all distances are finite). The logical guarded recursion principle (G-REC) should then be restricted to only prove finite ϕ, and similarly, fixed points should only be applicable to metric spaces. This model would have the benefit that scalar multiplication distributes over $\otimes$ and $\bullet$, and also rules like (ASSOC$_1$) would be invertible. We leave exploring such systems to future work.

In future work, it would also be interesting to address the limitations of our language mentioned in Remark 29. It might be possible to allow types and sensitivity annotations to depend on sets such as $\mathbb{N}$, but allowing them to depend on metric spaces seems more difficult. One possible starting point could be DFuzz [17], a version of Fuzz with lightweight dependent types.

References

1 Alejandro Aguirre, Gilles Barthe, Lars Birkedal, Ales Bizjak, Marco Gaboardi, and Deepak Garg. Relational reasoning for markov chains in a probabilistic guarded lambda calculus. In Amal Ahmed, editor, *Programming Languages and Systems - 27th European Symposium on Programming, ESOP 2018, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2018, Thessaloniki, Greece, April 14-20, 2018, Proceedings*,

volume 10801 of *Lecture Notes in Computer Science*, pages 214–241. Springer, 2018. doi:10.1007/978-3-319-89884-1_8.

2 Alejandro Aguirre, Gilles Barthe, Justin Hsu, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. A pre-expectation calculus for probabilistic sensitivity. *Proc. ACM Program. Lang.*, 5(POPL):1–28, 2021. doi:10.1145/3434333.

3 José Bacelar Almeida, Manuel Barbosa, Gilles Barthe, Benjamin Grégoire, Vincent Laporte, Jean-Christophe Léchenet, Tiago Oliveira, Hugo Pacheco, Miguel Quaresma, Peter Schwabe, Antoine Séré, and Pierre-Yves Strub. Formally verifying kyber part I: implementation correctness. *IACR Cryptol. ePrint Arch.*, page 215, 2023. URL: https://eprint.iacr.org/2023/215.

4 Martin Avanzini, Gilles Barthe, Davide Davoli, and Benjamin Grégoire. A quantitative probabilistic relational hoare logic. *Proc. ACM Program. Lang.*, 9(POPL):1167–1195, 2025. doi:10.1145/3704876.

5 Martin Avanzini, Gilles Barthe, Benjamin Grégoire, Georg Moser, and Gabriele Vanoni. Hopping proofs of expectation-based properties: Applications to skiplists and security proofs. *Proc. ACM Program. Lang.*, 8(OOPSLA1):784–809, 2024.

6 Giorgio Bacci, Giovanni Bacci, Kim G. Larsen, and Radu Mardare. A complete quantitative deduction system for the bisimilarity distance on markov chains. *Log. Methods Comput. Sci.*, 14(4), 2018.

7 Christel Baier and Marta Z. Kwiatkowska. Domain equations for probabilistic processes. *Math. Struct. Comput. Sci.*, 10(6):665–717, 2000. URL: http://journals.cambridge.org/action/displayAbstract?aid=71051.

8 Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae*, 3(1):133–181, 1922.

9 Lars Birkedal, Rasmus Ejlers Møgelberg, Jan Schwinghammer, and Kristian Støvring. First steps in synthetic guarded domain theory: step-indexing in the topos of trees. *Log. Methods Comput. Sci.*, 8(4), 2012. doi:10.2168/LMCS-8(4:1)2012.

10 Aloïs Brunel, Marco Gaboardi, Damiano Mazza, and Steve Zdancewic. A core quantitative coeffect calculus. In Zhong Shao, editor, *Programming Languages and Systems - 23rd European Symposium on Programming, ESOP 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8410 of *Lecture Notes in Computer Science*, pages 351–370. Springer, 2014. doi:10.1007/978-3-642-54833-8_19.

11 Francesco Dagnino and Fabio Pasquali. Logical foundations of quantitative equality. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 16:1–16:13. ACM, 2022. doi:10.1145/3531130.3533337.

12 Fredrik Dahlqvist and Renato Neves. An internal language for categories enriched over generalised metric spaces. In Florin Manea and Alex Simpson, editors, *30th EACSL Annual Conference on Computer Science Logic, CSL 2022, February 14-19, 2022, Göttingen, Germany (Virtual Conference)*, volume 216 of *LIPIcs*, pages 16:1–16:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. URL: https://doi.org/10.4230/LIPIcs.CSL.2022.16, doi:10.4230/LIPICS.CSL.2022.16.

13 Fredrik Dahlqvist and Renato Neves. A complete v-equational system for graded lambda-calculus. In Marie Kerjean and Paul Blain Levy, editors, *Proceedings of the 39th Conference on the Mathematical Foundations of Programming Semantics, MFPS XXXIX, Indiana University, Bloomington, IN, USA, June 21-23, 2023*, volume 3 of *EPTICS*. EpiSciences, 2023. URL: https://doi.org/10.46298/entics.12299, doi:10.46298/ENTICS.12299.

14 Arthur Azevedo de Amorim, Marco Gaboardi, Justin Hsu, Shin-ya Katsumata, and Ikram Cherigui. A semantic account of metric preservation. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 545–556. ACM, 2017. doi:10.1145/3009837.3009890.

15 Erik P. de Vink and Jan J. M. M. Rutten. Bisimulation for probabilistic transition systems: A coalgebraic approach. *Theor. Comput. Sci.*, 221(1-2):271–293, 1999. doi:10.1016/S0304-3975(99)00035-3.

16 J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Approximation of labeled Markov processes. In *Proceedings of the Fifteenth Annual IEEE Symposium On Logic In Computer Science*, pages 95–106. IEEE Computer Society Press, June 2000.

17 Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C. Pierce. Linear dependent types for differential privacy. In *POPL*, pages 357–370. ACM, 2013.

18 Marco Gaboardi, Shin-ya Katsumata, Dominic A. Orchard, Flavien Breuvart, and Tarmo Uustalu. Combining effects and coeffects via grading. In Jacques Garrigue, Gabriele Keller, and Eijiro Sumii, editors, *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming, ICFP 2016, Nara, Japan, September 18-22, 2016*, pages 476–489. ACM, 2016. doi:10.1145/2951913.2951939.

19 Simon Oddershede Gregersen, Alejandro Aguirre, Philipp G. Haselwarter, Joseph Tassarotti, and Lars Birkedal. Asynchronous probabilistic couplings in higher-order separation logic. *Proc. ACM Program. Lang.*, 8(POPL):753–784, 2024. doi:10.1145/3632868.

20 Alessandro Giacalone Chi-Chang Jou and Scott A. Smolka. Algebraic reasoning for probabilistic concurrent systems. In *PROCOMET*, pages 443–458. North Holland, 1994.

21 Dexter Kozen. A probabilistic PDL. *J. Comput. Syst. Sci.*, 30(2):162–178, 1985. doi:10.1016/0022-0000(85)90012-1.

22 Magnus Baunsgaard Kristensen, Rasmus Ejlers Møgelberg, and Andrea Vezzosi. Greatest hits: Higher inductive types in coinductive definitions via induction under clocks. In Christel Baier and Dana Fisman, editors, *LICS '22: 37th Annual ACM/IEEE Symposium on Logic in Computer Science, Haifa, Israel, August 2 - 5, 2022*, pages 42:1–42:13. ACM, 2022. doi:10.1145/3531130.3533359.

23 Ugo Dal Lago, Furio Honsell, Marina Lenisa, and Paolo Pistone. On quantitative algebraic higher-order theories. In Amy P. Felty, editor, *7th International Conference on Formal Structures for Computation and Deduction, FSCD 2022, Haifa, Israel, August 2-5, 2022*, volume 228 of *LIPIcs*, pages 4:1–4:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022. URL: https://doi.org/10.4230/LIPIcs.FSCD.2022.4, doi:10.4230/LIPICS.FSCD.2022.4.

24 Kim Guldstrand Larsen and Arne Skou. Bisimulation through probabilistic testing. *Information and Computation*, 94(1):1–28, 1991.

25 Torgny Lindvall. *Lectures on the Coupling Method*. Wiley Series in Probability and Mathematical Statistics. John Wiley, New York, 1992.

26 Jan Lukasiewicz and Alfred Tarski. Untersuchungen über den aussagenkalkül. In *Comptes Rendus des Séances de la Sociélé des Scierices et des Lettres des Varsovie Classe III*, volume 23, pages 30–50, 1930.

27 Radu Mardare, Prakash Panangaden, and Gordon D. Plotkin. Quantitative algebraic reasoning. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 700–709. ACM, 2016. doi:10.1145/2933575.2934518.

28 Radu Mardare, Prakash Panangaden, and Gordon D. Plotkin. Free complete wasserstein algebras. *Log. Methods Comput. Sci.*, 14(3), 2018. doi:10.23638/LMCS-14(3:19)2018.

29 Radu Mardare, Prakash Panangaden, and Gordon D. Plotkin. Fixed-points for quantitative equational logics. In *LICS*, pages 1–13. IEEE, 2021.

30 Annabelle McIver and Carroll Morgan. *Abstraction, Refinement and Proof for Probabilistic Systems*. Monographs in Computer Science. Springer, 2005. URL: https://doi.org/10.1007/b138392, doi:10.1007/B138392.

31 Carroll Morgan, Annabelle McIver, and Karen Seidel. Probabilistic predicate transformers. *ACM Trans. Program. Lang. Syst.*, 18(3):325–353, 1996. doi:10.1145/229542.229547.

32 Antonio Di Nola and Ioana Leustean. Riesz mv-algebras and their logic. In *EUSFLAT Conf*, pages 140–145. Atlantis Press, 2011.

33 Federico Olmedo, Benjamin Lucien Kaminski, Joost-Pieter Katoen, and Christoph Matheja. Reasoning about recursive probabilistic programs. In Martin Grohe, Eric Koskinen, and Natarajan Shankar, editors, *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, New York, NY, USA, July 5-8, 2016*, pages 672–681. ACM, 2016. doi:10.1145/2933575.2935317.

34 Jason Reed and Benjamin C. Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In Paul Hudak and Stephanie Weirich, editors, *Proceeding of the 15th ACM SIGPLAN international conference on Functional programming, ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, pages 157–168. ACM, 2010. doi:10.1145/1863543.1863568.

35 Alfred Tarski. *Logic, semantics, metamathematics: papers from 1923 to 1938*. Hackett Publishing, 1983.

36 The Univalent Foundations Program. *Homotopy Type Theory: Univalent Foundations of Mathematics*. https://homotopytypetheory.org/book, Institute for Advanced Study, 2013.

37 Franck van Breugel. A behavioural pseudometric for metric labelled transition systems. In *CONCUR*, volume 3653 of *Lecture Notes in Computer Science*, pages 141–155. Springer, 2005.

38 Franck van Breugel, Claudio Hermida, Michael Makkai, and James Worrell. An accessible approach to behavioural pseudometrics. In *ICALP*, volume 3580 of *Lecture Notes in Computer Science*, pages 1018–1030. Springer, 2005.

39 Franck van Breugel, Claudio Hermida, Michael Makkai, and James Worrell. Recursively defined metric spaces without contraction. *Theor. Comput. Sci.*, 380(1-2):143–163, 2007.

40 Franck van Breugel and James Worrell. A behavioural pseudometric for probabilistic transition systems. *Theor. Comput. Sci.*, 331(1):115–142, 2005.