On Feller Continuity and Full Abstraction

GILLES BARTHE, MPI Security and Privacy, Germany
RAPHAËLLE CRUBILLÉ, CNRS, France
UGO DAL LAGO, University of Bologna, Italy and INRIA, France
FRANCESCO GAVAZZO, University of Bologna, Italy and INRIA, France

We study the nature of applicative bisimilarity in λ-calculi endowed with operators for sampling from *continuous* distributions. On the one hand, we show that bisimilarity, logical equivalence, and testing equivalence all coincide with contextual equivalence when real numbers can be manipulated through continuous functions only. The key ingredient towards this result is a notion of Feller-continuity for labelled Markov processes, which we believe of independent interest, giving rise a broad class of LMPs for which coinductive and logically inspired equivalences coincide. On the other hand, we show that if no constraint is put on the way real numbers are manipulated, characterizing contextual equivalence turns out to be hard, and most of the aforementioned notions of equivalence are even *unsound*.

CCS Concepts: • **Theory of computation → Semantics and reasoning**; **Probabilistic computation**.

Additional Key Words and Phrases: Applicative Bisimilarity, Labelled Markov Processes, Lambda Calculus, Event Bisimilarity

ACM Reference Format:
Gilles Barthe, Raphaëlle Crubillé, Ugo Dal Lago, and Francesco Gavazzo. 2022. On Feller Continuity and Full Abstraction. *Proc. ACM Program. Lang.* 6, ICFP, Article 120 (August 2022), 29 pages. https://doi.org/10.1145/3547651

1 INTRODUCTION

There exist many different definitions of program equivalence for higher-order programs. These definitions come in different flavours, including contextual equivalence [Morris 1969], coalgebraic and bisimulation-based definitions [Abramsky 1990; Lassen 1999; Sangiorgi 1994], logical definitions based on modal logics [Abramsky 1990; Ong 1988], and operational definitions based on tests [Ong 1993]. In the baseline setting, where program evaluation does not produce computational effects, all these definitions coincide, and provide a rich set of methods for proving program equivalence.

In contrast, these different notions are not known to coincide in effectful programming languages; and even for some specific effects, such as sampling from distributions, it can be difficult to relate, or even define, the underlying relations. On the positive side, it is known that the aforementioned notions of equivalence coincide for *discrete* probabilistic effects, i.e. when values are sampled from discrete distributions [Crubillé and Dal Lago 2014; Crubillé et al. 2015]. However, little is known about the correspondence between different notions of equivalence for higher-order languages with sampling from *continuous* distributions, which are crucial in many application domains such as Bayesian programming. One main challenge with these languages is that their

Authors' addresses: Gilles Barthe, MPI Security and Privacy, Germany, gilles.barthe@mpi-sp.org; Raphaëlle Crubillé, CNRS, France, raphaelle.crubille@lis-lab.fr; Ugo Dal Lago, University of Bologna, Italy and INRIA, France, ugo.dallago@unibo.it; Francesco Gavazzo, University of Bologna, Italy and INRIA, France, francesco.gavazzo2@unibo.it.

2475-1421/2022/8-ART120
https://doi.org/10.1145/3547651

underlying operational semantics goes beyond well-understood classes of probabilistic processes. When sampling from continuous distributions is available, programs no longer form a labelled Markov chain, but form instead a *labelled Markov process* [Panangaden 2009] (LMP in the following). Unfortunately, even if these LMPs have an *analytic* state space, they come with an *uncountable* set of labels. As a consequence, they fall outside of the class of LMPs with analytic state spaces *and* countable sets of labels, for which correspondence between the different definitions of equivalence still holds [Panangaden 2009].

The question addressed in this paper is therefore: what is (*if any*) a natural condition on LMPs (with *analytic* state space and an *uncountable* set of labels), such that the different notions of equivalence coincide? The idea explored in this paper is to ask that programs are only built from *continuous* functions. And indeed, asking basic functions to be continuous has the consequence of making contextual equivalence, both forms of bisimilarity from the literature (i.e. state bisimilarity [Desharnais et al. 2002] and event bisimilarity [Danos et al. 2006, 2005]), logical equivalence and testing equivalence to coincide. The path to this result is not trivial, and goes through the definition of a new class of LMPs, called *Feller-continuous* LMPs, whose transition functions satisfy a continuity constraint.

Now equipped with Feller continuity as a sufficient condition for proving correspondence between the different notions, we ask for the *necessity* of Feller continuity. We do not provide a definitive answer to this question. However, our partial exploration provides several results of independent interest. On the positive side, we prove that several key technical results used to establish equivalence between the different notions extend to the general setting. On the negative side, we show that *event* bisimilarity, as well as *testing* and *logical* equivalence, are not *sound* in the general case. We also point to some undesirable behavior of *state* bisimilarity. These negative results do not rule out the existence of alternative better behaved definitions. However, we also show that contextual equivalence is not measurable. This, in itself, points to the broader challenge of proving program equivalence in absence of Feller continuity.

At a higher level, our results show a conceptual gap between discrete and continuous distributions. When working with calculi with *discrete* probabilistic choice, operational reasoning is mostly set-theoretic: programs form an ordinary set, and program execution is modeled as a set-theoretic function; accordingly, operational reasoning is mostly performed relying on the classic notion of a (program) relation. When we enter the realm of continuous probability, operational reasoning drastically changes, moving from a set-theoretic to a *measure-theoretic* base: programs now form not just a set, but a *measurable* space, and their evaluation becomes a *measurable function*. What about relations? One would expect that relations should become, morally, measurable too: and indeed, mathematically speaking, things work smoothly if one restricts to measurable relations, with beautiful, classic theorems — such as the well-known Monge-Kantorovich duality [Villani 2008] — extending to measurable spaces, functions, and relations. That suggests that to have well-behaved program equivalences, one should require such equivalences to be measurable. However, such a requirement is simply too strong: in absence of continuity constraints, contextual equivalence is *not* measurable. Consequently, even if in principle we could define *measurable* notions of program equivalence, the latter are doomed not to capture the full power of contextual equivalence: the very expressive power of the calculus simply goes beyond the world of measurable relations. Looking back at the notion of Feller-continuity from this perspective, we can see the latter as a somehow minimal condition ensuring contextual equivalence (and, *a fortiori*, all the aforementioned notions of equivalence) to be measurable, this way establishing a new, deep connection between the syntax of a calculus and the semantics of its associated operational reasoning.

Organization of the Paper and Summary of Contributions. The paper is organized as follows:

- In Section 2, we introduce $\Lambda_{\mathbf{P}}$, a simply-typed probabilistic λ-calculus with recursive types and real numbers, which we use as a vehicle language for our study. We show that for some choice of primitive functions, contextual equivalence for $\Lambda_{\mathbf{P}}$ is not measurable. We also introduce technical notions, such as that of a preterm, which are central to our development. Finally, we introduce a fragment of $\Lambda_{\mathbf{P}}$, called $\Lambda^{c}_{\mathbf{P}}$, in which all function symbols are continuous.
- We then present, in Section 3, a unified overview of different notions of equivalence and of their correspondence, in the case of LMPs with analytic state spaces and countable set of labels. We also single out correspondences that extend to the case of arbitrary LMPs. We also define the LMP describing the interactive behaviour of $\Lambda_{\mathbf{P}}$ programs, and observe that its set of labels is uncountable; as a consequence, it falls out of the class of LMPs for which different notions of correspondence are related;
- We introduce the notion of a Feller continuous LMP and prove that state and event bisimilarity coincide for Feller continuous LMPs (Theorem 4.11). It follows that the correspondence between different notions of equivalence for Feller continuous LMPs holds, similar to LMPs with analytical state space and discrete set of labels. All this is in Section 4.
- We prove that the LMP resulting from $\Lambda^{c}_{\mathbf{P}}$ is indeed Feller continuous. As a corollary, we get that full abstraction holds for $\Lambda^{c}_{\mathbf{P}}$, and that all the aforementioned notions of equivalence coincide (Theorem 5.9). This is in Section 5.
- We present in Section 6 a preliminary analysis of the general case. Our main result is that event bisimilarity, as defined in Section 3, is unsound. Additionally, we illustrate by way of examples how measurability issues pose a challenge to the role of state bisimilarity in the context of $\Lambda_{\mathbf{P}}$.

An extended version of this paper with more details is available [Barthe et al. 2022].

2 THE STOCHASTIC λ-CALCULUS

The target language of this work is a (call-by-value) λ-calculus with finite sums and recursive types endowed with specific operators for sampling from continuous distributions, and for first-order functions on the real numbers. As such, the calculus is very expressive, both at the level of terms and at the level of types. We separate values from terms [Levy et al. 2003], this way requiring terms to be explicitly sequenced via the let-in construct.

2.1 Syntax and Static Semantics

The syntax and static semantics of $\Lambda_{\mathbf{P}}$ are defined in Figure 1. Types of $\Lambda_{\mathbf{P}}$ are built starting from a countable set of type variables (denoted by the letter α in Figure 1) and the basic type **real** for real numbers, using finite sums, arrows, and recursion. In particular, in a type of the form $\sum_{i\in I}\sigma_i$ we assume the letter I to stand for a finite set whose elements are denoted by $\hat{\imath}, \hat{\jmath}, \ldots$. We write **void** for the empty sum type, **unit** for **void** $\to$ **void**, and **bool** for **unit** + **unit**. Concerning the latter, we also use standard syntactic sugar, viz. **true**, **false**, and **if** V **then** M **else** N. As usual, we write VM for **let** $x = M$ **in** Vx and MN for **let** $x = M$ **in** xN.

We use *term* judgments (resp. *value* judgments) of the form $\Gamma \vdash^{\mathscr{T}} M : \sigma$ (resp. $\Gamma \vdash^{\mathscr{V}} V : \sigma$) to state that M (resp. V) is a term (resp. value) of *closed* type σ in the environment Γ, whereas an environment is a finite sequence $x_1 : \sigma_1, \ldots, x_n : \sigma_n$ of distinct variables with associated *closed* types (we denote by $\cdot$ the empty environment). Notice that we work with closed types only. We use letters $M, N, \ldots$ and $V, W, \ldots$ to denote terms and values, respectively. We write $\mathscr{T}_\sigma$, $\mathscr{V}_\sigma$ for the set of closed terms and values, respectively, of type σ.

The syntax of $\Lambda_{\mathbf{P}}$ consists of three distinct parts: the first one is a standard λ-calculus with recursive types and coproducts; the second one allows one to perform real-valued computations; finally, the third one is the actual probabilistic core of $\Lambda_{\mathbf{P}}$. Concerning real-valued computations,

$$\sigma ::= \alpha \mid \mathbf{real} \mid \sum_{i\in I} \sigma_i \mid \sigma \to \sigma \mid \mu\alpha.\sigma$$
$$V ::= x \mid \boldsymbol{r} \mid \langle \imath, V\rangle \mid \lambda x.M \mid \mathbf{fold}\ V$$
$$M ::= V \mid \mathbf{sample} \mid \boldsymbol{f}(V,\dots,V) \mid \boldsymbol{b}(V,\dots,V) \mid \mathbf{pm}\ V\ \mathbf{as}\ \langle i, x\rangle.M \mid VV \mid \mathbf{let}\ x = M\ \mathbf{in}\ M \mid \mathbf{unfold}\ V$$

$$\frac{}{\Gamma, x:\sigma \vdash^{\mathscr{V}} x:\sigma} \quad \frac{r \in \mathbb{R}}{\Gamma \vdash^{\mathscr{V}} \boldsymbol{r} : \mathbf{real}} \quad \frac{}{\Gamma \vdash^{\mathscr{T}} \mathbf{sample} : \mathbf{real}}$$

$$\frac{\Gamma \vdash^{\mathscr{V}} V_1 : \mathbf{real}, \dots, \Gamma \vdash^{\mathscr{V}} V_n : \mathbf{real} \quad f \in C_n}{\Gamma \vdash^{\mathscr{T}} \boldsymbol{f}(V_1,\dots,V_n) : \mathbf{real}} \quad \frac{\Gamma \vdash^{\mathscr{V}} V_1 : \mathbf{real}, \dots, \Gamma \vdash^{\mathscr{V}} V_n : \mathbf{real} \quad b \in \mathcal{B}_n}{\Gamma \vdash^{\mathscr{T}} \boldsymbol{b}(V_1,\dots,V_n) : \mathbf{bool}}$$

$$\frac{\Gamma \vdash^{\mathscr{V}} V : \sigma}{\Gamma \vdash^{\mathscr{T}} V : \sigma} \quad \frac{\Gamma \vdash^{\mathscr{T}} M : \sigma \quad \Gamma, x:\sigma \vdash^{\mathscr{T}} N : \tau}{\Gamma \vdash^{\mathscr{T}} \mathbf{let}\ x = M\ \mathbf{in}\ N : \tau} \quad \frac{\Gamma, x:\sigma \vdash^{\mathscr{T}} M : \tau}{\Gamma \vdash^{\mathscr{V}} \lambda x.M : \sigma \to \tau} \quad \frac{\Gamma \vdash^{\mathscr{V}} V : \sigma \to \tau \quad \Gamma \vdash^{\mathscr{V}} W : \sigma}{\Gamma \vdash^{\mathscr{T}} VW : \tau}$$

$$\frac{\Gamma \vdash^{\mathscr{V}} V : \sigma_{\hat{\imath}}}{\Gamma \vdash^{\mathscr{V}} \langle \hat{\imath}, V\rangle : \sum_{i\in I}\sigma_i} \quad \frac{\Gamma \vdash^{\mathscr{V}} V : \sum_{i\in I}\sigma_i \quad \Gamma, x:\sigma_i \vdash^{\mathscr{T}} M_i : \sigma\ (\forall i \in I)}{\Gamma \vdash^{\mathscr{T}} \mathbf{pm}\ V\ \mathbf{as}\ \langle i, x\rangle.M_i : \sigma}$$

$$\frac{\Gamma \vdash^{\mathscr{V}} V : \sigma[\mu\alpha.\sigma/\alpha]}{\Gamma \vdash^{\mathscr{V}} \mathbf{fold}\ V : \mu\alpha.\sigma} \quad \frac{\Gamma \vdash^{\mathscr{V}} V : \mu\alpha.\sigma}{\Gamma \vdash^{\mathscr{T}} \mathbf{unfold}\ V : \sigma[\mu\alpha.\sigma/\alpha]}$$

Fig. 1. Syntax and Static semantics of $\Lambda_{\mathbf{P}}$.

we indicate real number in boldface with metavariables like $\boldsymbol{r}$, while we take $\Lambda_{\mathbf{P}}$ parametric with respect to a $\mathbb{N}$-indexed family C of *countable* sets C_n, each of which contains measurable functions from $\mathbb{R}^n$ to $\mathbb{R}$. In particular, we assume standard real-valued arithmetic operations to be in C. We also assume the existence of a comparison operator $op_{\leq} \in C_2$ such that $op_{\leq} : \mathbb{R}\times\mathbb{R} \to [0,1]$, and $op_{\leq}(x,y) = 1$ if and only if $x \leq y$. We use letters $f, g, \dots$ to denote elements in C_n, writing their $\Lambda_{\mathbf{P}}$ syntactic representation in boldface. To relate numerical and general computations, we assume to have collections $\mathcal{B}_n$ of functions $b : \mathbb{R}^n \to \{0,1\}$ to perform Boolean tests on real numbers. For simplicity, we just take $\mathcal{B} = \{=, <\}$. Further functions can be encoded relying on programs on **bool** representing Boolean operators.

Probabilistic behaviours are triggered by the term **sample**. The latter samples from the uniform distribution over the unit interval, which is nothing but the Lebesgue measure λ on $[0,1]$. It is well-known [Ehrhard et al. 2018] that starting from **sample** several probabilistic measures (e.g. binomial, geometric, and exponential distribution) can be defined through functions in C.

Example 2.1. Given closed terms M_μ, M_σ of type **real** (encoding mean μ and standard deviation σ), we represent the normal distribution with mean μ and standard deviation σ as the expression $\mathbf{normal}\ M_\mu\ M_\sigma$, where the term $\mathbf{normal}_{\mathbf{std}}$ encodes the normal distribution with mean 0 and standard deviation 1.

$$\mathbf{normal}_{\mathbf{std}} \triangleq \mathbf{let}\ x = \mathbf{sample}\ \mathbf{in}\ (\mathbf{let}\ y = \mathbf{sample}\ \mathbf{in}\ (\sqrt{-2\log(x)}\cos(2\pi y)))$$
$$\mathbf{normal}\ M_\mu\ M_\sigma \triangleq \mathbf{let}\ x_\mu = M_\mu,\ x_\sigma = M_\sigma, y = \mathbf{normal}_{\mathbf{std}}\ \mathbf{in}\ ((x_\sigma * y) + x_\mu).$$

For any type σ, we have the Bernoulli choice term $\mathbf{bernoulli} : \sigma \to \sigma \to \mathbf{real} \to \sigma$ acting as a probabilistic choice construct. Such a term is defined as $\lambda m, n, p.\mathbf{let}\ x = \mathbf{sample}\ \mathbf{in}\ (\mathbf{if}\ x > p\ \mathbf{then}\ n\ \mathbf{else}\ m)$ In particular, the term $\mathbf{bernoulli}\ M\ N\ \mathbf{0.5}$ is the fair probabilistic choice between M and N. We employ the notation $M \oplus N$ in place of $\mathbf{bernoulli}\ M\ N\ \mathbf{0.5}$.

Finally, we adopt standard notational conventions [Barendregt 1984]. In particular, we denote by $FV(M)$ the collection of free variables of M (resp. V) and we refer to closed expressions as *programs*. We denote by $M[V/x]$ the capture-avoiding substitution of the value V for all free occurrences of x in M. We extend the aforementioned conventions to types. For instance, we denote by $\sigma[\tau/\alpha]$ the result of capture-avoiding substitution of the type τ for the type variable α in σ.

2.2 Dynamic Semantics

The dynamic semantics of $\Lambda_{\mathbf{P}}$ is given by a type-indexed family of evaluation functions $[\![-]\!]_\sigma$ mapping programs of type σ to *sub-probability measures* over values of type σ. As such, the semantics is measure-theoretic and defining it requires some preliminary definitions that we are going to give. Due to lack of space, we assume that the reader is familiar with basic definitions from measure theory and refer to classic textbooks for additional background [Billingsley 1986].

Measurable Spaces. We denote by **Meas** the category of measurable spaces and measurable functions. When X is a measurable space, we write Σ_X for the underlying σ-algebra. Moreover, given a measurable space (X, Σ_X) with a measure μ on it and a function $f : X \to [0, \infty]$, we denote by $\int_X f d\mu$ or $\int_X f(x)\mu(dx)$ the number in $[0, \infty]$ obtained by Lebesgue integrating f over X with respect to μ. Oftentimes, we work with measurable spaces associated to a Polish space. More precisely, we shall work with standard Borel spaces, i.e. those measurable spaces generated by complete, separable metric spaces equipped with their Borel σ-algebra. We denote by **Pol** the full subcategory of **Meas** whose objects are standard Borel spaces (notice that arrows in **Pol** remain measurable functions). Finally, we remark that both **Meas** and **Pol** have countable products and coproducts.

Probability Measures. We denote by $\Delta(X)$ the collection of sub-probability measures on a measurable space (X, Σ_X), using the notation $\emptyset$ and $\{x^1\}$ for the empty-subdistribution (on X) and the Dirac distribution on x, respectively. We endow $\Delta(X)$ with the σ-algebra generated by the set of evaluation maps $\{ev_A : \Delta(X) \to [0, 1] \mid A \in \Sigma_X\}$, where $ev_A(\mu) = \mu(A)$. Recall that, given a measurable function $f : X \to Y$, the push forward of f is the (measurable) function $f_\sharp : \Delta(X) \to \Delta(Y)$ defined by $f_\sharp(\mu)(B) = \mu(f^{-1}(B))$. These data induce a functor $\mathcal{G}$ on **Meas**, which is the functor part of the *Giry monad* [Giry 1982]. In fact, $\mathcal{G}$ has a unit map η associating to each element x the Dirac distribution $\{x^1\}$, and for a measurable function $f : X \to \mathcal{G}(Y)$, the map $f^\dagger$ defined by $f^\dagger(\mu)(B) = \int f(x)(B)\mu(dx)$ gives the Kleisli extension of f. When (X, Σ_X) is a standard Borel space, then $\Delta(X)$ is a standard Borel space too — so that $\mathcal{G}$ is also a monad on **Pol** — and the σ-algebra of $\Delta(X)$ coincides with the Borel algebra of the weakest topology making the integration map $\mu \mapsto \int f d\mu$, continuous, for any bounded continuous real-valued function f (see [Kechris 2012], Theorem 17.23 and 17.24).

Probability Kernels. Given measurable spaces X, Y, we recall that a *(sub)probability kernel* [Panangaden 1999] is a map $f : X \times \Sigma_Y$ such that:

- For every $A \in \Sigma_Y$, the map $x \in X \mapsto f(x, A)$ is measurable;
- For every $x \in X$, the map $A \in \Sigma_Y \mapsto f(x, A)$ is a (sub)probability measure.

We write $f : X \nrightarrow Y$ when f is a sub-probability kernel, and $f : X \nrightarrow^{=} Y$ when f is a probability kernel (also called Markov kernel). For any space X, the Dirac measure on X induces a kernel $X \nrightarrow X$. Moreover, given $f : X \nrightarrow Y$ and $g : Y \nrightarrow Z$, we define their composition $g \circ f : X \nrightarrow Z$ by $(g \circ f)(x, C) = \int_Y g(y, C) \cdot f(x, dy)$. By the monotone convergence theorem, the composition operation $\circ$ is associative and has Dirac measures as unit, meaning that kernels form a category. Such a category is nothing but the Kleisli category of $\mathcal{G}$. Panangaden [1999] showed that the category of sub-probability kernels is partially additive [Manes and Arbib 1986]. As a consequence,

for all spaces X, Y, the set of sub-probability kernels from X to Y forms an ω-cppo when ordered pointwise. In particular, the bottom element associates to any element x the empty distribution $\emptyset$, whereas the least upper bound of an ω-chain $\{f_n\}_{n\geq 0}$ of kernels is defined as $\sup_n f_n$. Additionally, kernel composition is monotone and continuous with respect to such ω-cppo structure, meaning that the category of sub-probability kernels is ω-**cppo**-enriched [Kelly 2005], where ω-**cppo** is the category of ω-cppos and continuous functions. All of this works both on **Meas** and **Pol** (with the notion of a probability kernel restricted to standard Borel spaces).

Dynamic Semantics. We now turn to the dynamic semantics of $\Lambda_{\mathbf{P}}$. Such a semantics is given as a monadic evaluation semantics [Dal Lago et al. 2017] for the Giry monad in the category **Pol**. More concretely, the semantics is given by type-indexed (sub-probability, due to the presence of full recursion) kernels $\mathscr{T}_\sigma \nrightarrow \mathscr{V}_\sigma$. Recall that for any type σ, the sets $\mathscr{T}_\sigma$ and $\mathscr{V}_\sigma$ carry a a standard Borel space structure [Borgström et al. 2016; Ehrhard et al. 2018; Staton et al. 2016; Vákár et al. 2019]. We recall this structure here, since we will use it explicitly in the technical developments.

Definition 2.2. We call *pre-terms* and *pre-values* the syntactic objects obtained from terms and values respectively, by removing all occurrences of real numbers and replacing them by numbered holes $[\cdot]^n$. Moreover, we add the constraint that whenever P is a pre-term (or a pre-value) with n holes, it means that each hole $[\cdot]^i$ — where $1 \leq i \leq n$ — must occur exactly once in P. Whenever P is a pre-term (or a pre-value) with n holes, and $\vec{r} \in \mathbb{R}^n$, we write $P[\vec{r}]$ for the term obtained by replacing each hole $[\cdot]^i$ by the corresponding real value $\boldsymbol{r}_i$. We write $\mathscr{T}_P$ (resp. $\mathscr{V}_V$) for the set of all terms (resp. values) of the form $P[\vec{r}]$ (resp. $V[\vec{r}]$), where $\vec{r}$ ranges over $\mathbb{R}^n$.

Observe that, for any pre-term P with n holes, the set $\mathscr{T}_P$ is in bijection with $\mathbb{R}^n$, so that we can naturally endow $\mathscr{T}_P$ with a standard Borel space structure. Since $\mathscr{T}$ consists of the disjoint union of the sets $\mathscr{T}_P$, it is in fact the countable coproduct of standard Borel spaces, thus a standard Borel space itself. A standard Borel space structure can be similarly given to $\mathscr{V}$.

Definition 2.3. Define the type-indexed family of ($\mathbb{N}$-indexed) kernels $[\![-]\!]^{(n)}_\sigma : \mathscr{T}_\sigma \nrightarrow \mathscr{V}_\sigma$ as follows (for readability, we omit type annotations):

$$
\begin{aligned}
[\![M]\!]^{(0)}(A) &= 0\\
[\![V]\!]^{(n+1)}(A) &= \{V^1\}(A)\\
[\![\boldsymbol{f}(\boldsymbol{r}_1,\dots,\boldsymbol{r}_{\boldsymbol{n}})]\!]^{(n+1)}(A) &= [\![\boldsymbol{f}(\boldsymbol{r}_1,\dots,\boldsymbol{r}_{\boldsymbol{n}})]\!]^{(n)}(A)\\
[\![\boldsymbol{b}(\boldsymbol{r}_1,\dots,\boldsymbol{r}_{\boldsymbol{n}})]\!]^{(n+1)}(A) &= [\![\boldsymbol{b}(\boldsymbol{r}_1,\dots,\boldsymbol{r}_{\boldsymbol{n}})]\!]^{(n)}(A)\\
[\![\mathbf{sample}]\!]^{(n+1)}(A) &= \lambda\{r \mid \boldsymbol{r} \in A\}\\
[\![(\lambda x.M)V]\!]^{(n+1)}(A) &= [\![M[V/x]]\!]^{(n)}(A)\\
[\![\mathbf{let}\ x = M\ \mathbf{in}\ N]\!]^{(n+1)}(A) &= \int [\![N[V/x]]\!]^{(n)}(A)[\![M]\!]^{(n)}(dV)\\
[\![\mathbf{pm}\ \langle \hat{\imath}, V\rangle\ \mathbf{as}\ \langle i, x\rangle.M_i]\!]^{(n+1)}(A) &= [\![M_i[V/x]]\!]^{(n)}(A)\\
[\![\mathbf{unfold}\ (\mathbf{fold}\ V)]\!]^{(n+1)}(A) &= [\![V]\!]^{(n)}(A).
\end{aligned}
$$

By observing that for any term $\Gamma, x : \sigma \vdash^{\mathscr{T}} M : \tau$, the map $M[-/x] : \mathscr{V}_{\Gamma\vdash\sigma} \to \mathscr{T}_{\Gamma\vdash\tau}$ is measurable [Borgström et al. 2016; Ehrhard et al. 2018; Staton et al. 2016], we immediately notice that each $[\![-]\!]^{(n)}$ is indeed a kernel and that $\{[\![-]\!]^{(n)}_\sigma\}_{n\geq 0}$ forms an ω-chain, so that we can define $[\![M]\!]_\sigma$ as $\sup_n [\![M]\!]^{(n)}_\sigma$. By ω-**cppo**-enrichment, $[\![M]\!]$ is the least solution to the system of equations induced by Definition 2.3.

2.3 Contextual Equivalence

Before moving to the study of coinductively-based notions of equivalence, we recall the notion of *probabilistic contextual equivalence*.

Definition 2.4 (Context Equivalence). For every type σ, an *observable context for* σ is a program $\mathfrak{C}$ with a hole $[\cdot]$ such that $[\cdot] : \sigma \vdash \mathfrak{C} : \mathbf{real}$. The set of all observable contexts for σ is indicated as $\mathfrak{C}_\sigma$. Contextual equivalence of terms and values is defined as usual, by quantifying over all possible observable contexts:

$$\forall M, N \in \mathscr{T}_\sigma,\ M \equiv^{ctx} N \text{ when } \forall \mathfrak{C} \in \mathfrak{C}_\sigma, [\![\mathfrak{C}[M]]\!] = [\![\mathfrak{C}[N]]\!]$$
$$\forall V, W \in \mathscr{V}_\sigma,\ V \equiv^{ctx} W \text{ when } \forall \mathfrak{C} \in \mathfrak{C}_\sigma, [\![\mathfrak{C}[V]]\!] = [\![\mathfrak{C}[W]]\!]$$

By Definition 2.4, the underlying notion of observation is the whole distribution produced in output by any observable context. This notion can, as usual, be proved robust to small changes in the language and in the notion of observation, see [Barthe et al. 2022] for further discussion. It is now instructive to observe that contextual equivalence is *not* a *measurable relation*, meaning that the graph of $\equiv^{ctx}$ is not a measurable (i.e. Borel, in our setting) set.[1]

Proposition 2.5. *There exists a choice of measurable functions in $\mathbf{C}$ such that the graph of $\equiv^{ctx}$ is not a Borel set.*

Proof. We are going to use the existence of a non-Borel co-analytical set $X \subseteq \mathbb{R}$ to build measurable primitive functions that lead to a non-Borel equivalence class for $\equiv^{ctx}$ (and thus to a non-Borel graph for $\equiv^{ctx}$). We know that such a set exists from descriptive set theory (see [Kechris 2012], 14.2). Moreover, we also know (see [Kechris 2012], 32.B) that there exists some open set O of $\mathbb{R} \times \mathcal{N}$, such that $X = \{y \mid \forall x, (x, y) \in O\}$: here $\mathcal{N}$ is the Baire space, that is known ([Kechris 2012], 3.4) to be homomorphic to the set of all irrational numbers. It means that we can embedd O in $\mathbb{R} \times \mathbb{R}$, and we again obtain a Borel set (observe that O is not an open set anymore, but we can show it is an open set minus a set of rational numbers, and since $\mathbb{Q}$ is countable this removal operation preserves Borel sets). From O, we define a function $f_O : \mathbb{R}^2 \to \mathbb{R}$ defined as $f_O(x, y) = 1$ whenever $(x, y) \in O$, and 0 otherwise. Observe that since $O \subseteq \mathbb{R} \times \mathbb{R}$ is a Borel set, it holds that f_O is measurable. We are now going to consider the (very simple) program $M \triangleq \lambda x.0 : \mathbf{real} \to \mathbf{real}$, and show that its equivalence class for $\equiv^{ctx}$ is not a Borel set. We are able to caracterise exactly the equivalence class of M: it consists of all the $N : \mathbf{real} \to \mathbf{real}$ such that for every $r \in \mathbf{real}$, $[\![Nr]\!] = \{0^1\}$. (In order to show that formally, we can use our results of soundness and completeness of event bisimulation and state bisimulation respectively, or we can for instance use denotational arguments in the quasi-Borel space model). Now, looking at the way we define the standard Borel space of terms as a countable co-product, we see that it is enough to show that there exists one pre-term P, such that $\{\vec{r} \mid P[\vec{r}] \equiv^{ctx} M\}$ is not measurable. We consider the pre-term P with one hole defined by $P[r] \triangleq \lambda x.$ if $f_O(x, r) = 1$ then 0 else 1. Using the caracterisation of M's equivalence class we have previously established, we see that $\{r \mid P[r] \equiv^{ctx} M\} = \{r \mid \forall r' \in \mathbb{R}, f(r, r') \in O\} = X$, and we can conclude from there. □

In other words, even if we insist, like we do, in constraining the basic building blocks of our calculus in such a way that everything stays measurable, the central concept of contextual equivalence somehow breaks the mold. There is however a way to recover measurability of contextual equivalence, which we will now describe.

[1] Even more, we can rely on descriptive set theory [Kechris 2012] to bound the non-measurability of contextual equivalence (and its equivalence classes), showing, e.g., that the graph of $\equiv^{ctx}$ is a co-analytic set.

2.4 Recovering Measurability

We consider a natural restriction of $\Lambda_{\mathbf{P}}$, called $\Lambda^c_{\mathbf{P}}$, in which we require primitive functions to be continuous.

Definition 2.6. The language $\Lambda^c_{\mathbf{P}}$ is the fragment of $\Lambda_{\mathbf{P}}$ obtained by requiring $\mathcal{C}$ to only contain symbols denoting *continuous* functions, and $\mathcal{B}$ to be empty.

Noticeably, what we obtained is not a *different* calculus, but rather an instance of $\Lambda_{\mathbf{P}}$ in which $\mathcal{C}$ and $\mathcal{B}$ satisfy certain conditions. Even if functions in $\mathcal{C}$ are required to be continuous, the presence in $\mathcal{C}_2$ of a (now, continuous) comparison operator $op_{\leq}$ is still required. Observe that sampling from discrete distributions is not available anymore from within $\Lambda^c_{\mathbf{P}}$, because standard tests on real numbers are not continuous. Discrete distributions could however be added natively to $\Lambda^c_{\mathbf{P}}$, e.g. by endowing the language with a standard binary choice operator $\oplus_p$ complementing **sample**. For the sake of simplicity, we do not pursue this further, and work with a rather minimal language. Our proof techniques, however, are robust enough to ensure that our results would not be affected by the presence of $\oplus_p$.

We will extensively study the nature of program equivalence for $\Lambda^c_{\mathbf{P}}$ in Section 4 and Section 5. For the moment, we simply observe that moving from $\Lambda_{\mathbf{P}}$ to $\Lambda^c_{\mathbf{P}}$ indeed ensures contextual equivalence to be measurable.

Proposition 2.7. *The graph of contextual equivalence on $\Lambda^c_{\mathbf{P}}$ is a Borel set.*

Proof. First, since Borel sets are stable by countable union, we see that it is sufficient to show that for every type σ, $\equiv^{ctx}$ restricted to programs of type σ is Borel. By definition of context equivalence, we can write:

$$\equiv^{ctx}_{\sigma} = \bigcap_{\substack{C \text{ a pre-context} \\ \text{with } k \text{ holes}}} \{(M,N) \mid \forall \vec{r} \in \mathbb{R}^k,\ [\![C[\vec{r}][M]]\!] = [\![C[\vec{r}][N]]\!]\}$$

$$= \bigcap_{\substack{C \text{ a pre-context} \\ \text{with } k \text{ holes}}} \bigcap_{q \in \mathbb{Q} \cup +\infty} \{(M,N) \mid \forall \vec{r} \in \mathbb{R}^k,\ [\![C[\vec{r}][M]]\!](]-\infty, q]) = [\![C[\vec{r}][N]]\!](]-\infty, q)\}$$

where a formal proof for the last equality can be found in [Barthe et al. 2022]. Borel sets are also closed by countable intersections, thus it is enough to show that for every pre-context C, and every $q \in \mathbb{Q} \cup \{+\infty\}$, the set $S_{C,q} \triangleq \{(M,N) \mid \forall \vec{r} \in \mathbb{R}^k,\ [\![C[\vec{r}][M]]\!](]-\infty, q]) = [\![C[\vec{r}][N]]\!](]-\infty, q)\}$ is Borel. As a first step, observe that we can rewrite $S_{C,q}$ as the complementary of:

$$\overline{S_{C,q}} \triangleq \{(M,N) \mid \exists r \in \mathbb{R}^k, (\vec{r}, M, N) \in A_{C,q}\}, \tag{1}$$

where $A_{C,q} = \{(r,M,N) \mid [\![C[\vec{r}][M]]\!](]-\infty, q]) \neq [\![C[\vec{r}][N]]\!](]-\infty, q)\} \subseteq \mathbb{R}^k \times \mathscr{T}_\sigma \times \mathscr{T}_\sigma$. It means that it is enough to show that the projection of $A_{C,q}$ is a Borel set. Observe that it is not true that the projection of a Borel set is a Borel set; however it is true that the projection of an open set is an open set, and thus a Borel set. Using the fact that now all our primitive functions are continuous, we are now able to show that $A_{C,q}$ is an open set. We write $A_{C,q}$ as:

$$A_{C,q} = (h_C, q)^{-1}(\{(a,b) \in \mathbb{R}^2 \mid a \neq b\}),$$

where $h_C, q : \mathbb{R}^k \times \mathscr{T}_\sigma \times \mathscr{T}_\sigma \to \mathbb{R}^2$ is the sequential composition $h_{C,q} := f_C; (g_q \times g_q)$, where f_C and g_q are defined thus:

$$f_C : (\vec{r}, M, N) \in \mathbb{R}^k \times \mathscr{T}_\sigma \times \mathscr{T}_\sigma \mapsto (C[\vec{r}][M], C[\vec{r}][N]) \in \mathscr{T}_{\mathbf{real}} \times \mathscr{T}_{\mathbf{real}}$$
$$g_q : P \in \mathscr{T}_{\mathbf{real}} \mapsto [\![P]\!](]-\infty, q]) \in \mathbb{R}.$$

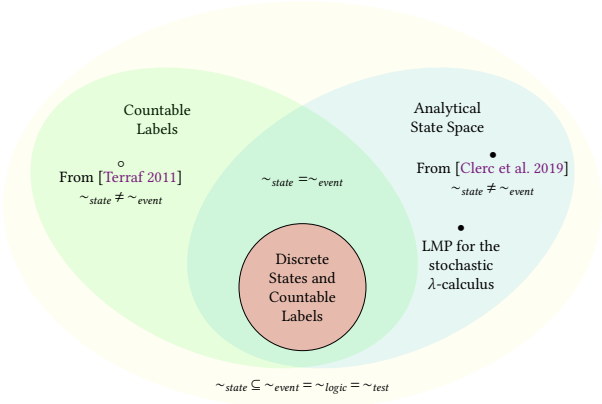

Fig. 2. Comparison of LMP-equivalences

Looking at the definition of the standard Borel space $\mathscr{T}$, we can see that elementary syntactic operations are not only measurable, but also continuous: from there, we obtain that f_C is continuous. Moreover, enforcing continuity of primitive functions allows us to show that the function g_q is also continuous: we will show formally this result in Section 5 (it is a direct corollary of Proposition 5.4). Since continuous functions are stable by composition and cartesian product, $h_{C,q}$ is continuous too. Consequently, $A_{C,q}$ is the inverse image of an open $\mathbb{R}^2$-set by a continuous function, and thus it is also open. This concludes the proof. □

Having defined the target calculi of this work and their associated notion of contextual equivalence, we can now proceed to the introduction of bisimulation-based equivalences and their logical and testing-based characterizations.

3 ON LABELLED MARKOV PROCESSES AND EQUIVALENCES ON THEM

In this section, we briefly recap the definition of *state*-bisimilarity, and we give the alternative notion of *event* bisimilarity, introduced by Danos et al. [2006, 2005]. A summary of all results presented in this section can be found in Figure 2.

3.1 Relational Reasoning

Before introducing bisimulation-based equivalences, it is convenient to introduce some basic definitions on measurable spaces, relational reasoning on them, and the crucial notion of a relational extension on probability measures. We denote binary relations by $\mathcal{R}, \mathcal{S}, \ldots$, writing $\mathcal{I}$ for the identity relation (with subscripts, when necessary), $\mathcal{R};\mathcal{S}$ for relation composition (in diagrammatic order), and $\mathcal{R}^-$ for the converse (or transpose) of $\mathcal{R}$. Even if most of the results presented in this section hold for arbitrary (endo)relations, it is convenient to work right from the beginning with equivalence relations: given a set X, we denote by $\mathscr{R}(X)$ the complete lattice of equivalence relations on X. Finally, when reasoning relationally, we implicitly view any function f as a relation (via its graph).

The crucial ingredient to define bisimulation-like equivalences on Markov processes (as introduced later in the paper) is the notion of an extension of relations over a space X to relations over $\Delta(X)$. Relational extensions of that form are known in the literature as *relators* or *lax extensions* [Backhouse and Hoogendijk 1993; Barr 1970]. In this paper, we are mostly concerned with the notion of relational extension used by Dal Lago and Gavazzo [2019] to define applicative bisimilarity on stochastic λ-calculi. At a high level, this extension is obtained as the codensity lifting of the

Giry monad [Katsumata et al. 2018]. To define such an extension formally, let us first recall that given a space (X, Σ_X) and a relation $\mathcal{R} \in \mathscr{R}(X)$, a set $A \subseteq X$ is $\mathcal{R}$-closed if $\mathcal{R}[A] = A$. Moreover, we define $\Sigma(\mathcal{R})$ as the sub-σ-algebra of Σ defined as: $\Sigma(\mathcal{R}) = \{A \in \Sigma \mid A \text{ is } \mathcal{R}\text{-closed}\}$.

Definition 3.1 (Probabilistic Relation Lifting). Let (X, Σ) be a measurable space. Define $\Gamma : \mathscr{R}(X) \to \mathscr{R}(\Delta(X))$ by:[2]

$$\mu \; \Gamma\mathcal{R} \; \nu \iff \forall A \in \Sigma(\mathcal{R}).\, \mu(A) = \nu(A).$$

When working with standard Borel spaces, there is another well-known relational extensions corresponding to the so-called Wasserstein-Kantorovich distance [Villani 2008]. Such an extension crucially relies on the notion of a probabilistic coupling.

Definition 3.2. Let (X, Σ_X), (Y, Σ_Y) be standard Borel spaces and $\mu \in \Delta(X)$, $\nu \in \Delta(Y)$ be probability measures. A *coupling* of μ, ν is a probability measure $\omega \in \Delta(X \times Y)$ such that for all $A \in \Sigma_X$, $B \in \Sigma_Y$, we have: $\omega(A \times Y) = \mu(A)$ and $\omega(X \times B) = \nu(B)$. We denote by $\Omega(\mu, \nu)$ the set of couplings of μ and ν. Observe that such a set is non-empty (e.g., product distributions form a coupling).

We now extend any relation $\mathcal{R}$ to probability measures by asking the existence of a coupling compatible with $\mathcal{R}$ (cf. the existential character of the following extension with the universal one of Definition 3.1).

Definition 3.3. Let (X, Σ_X) be a standard Borel space. We define $\Theta : \mathscr{R}(X) \to \mathscr{R}(\Delta(X))$ by:

$$\mu \; \Theta\mathcal{R} \; \nu \iff \exists \omega \in \Omega(\mu, \nu). \exists \mathcal{S} \in \Sigma_{X \times X}.\, \mathcal{S} \subseteq \mathcal{R} \;\&\; \omega(\mathcal{S}) = \omega(X \times X).$$

It is easy to see that $\Theta\mathcal{R} \subseteq \Gamma\mathcal{R}$, although Γ and Θ define different relational extensions, in general.[3] This is due to the fact that we are working with arbitrary equivalence relations, which do not reflect the (standard) Borel structure of the spaces considered. Moving to relations reflecting such a structure — known as *Borel* relations — we indeed obtain the desired equality.

Definition 3.4. Given a standard Borel space (X, Σ_X) and $\mathcal{R} \in \mathscr{R}(X)$, we say that $\mathcal{R}$ is a *Borel* relation when the set $\{(x, y) \mid x \mathcal{R} y\}$ is a Borel set in $X \times X$.

The advantage of working with Borel relations is that they are extremely well-behaved. In particular, Borel relations allow us to generalize Monge-Kantorovich duality to the measurable case, a result that we will need in Section 5 .

Theorem 3.5. *Let (X, Σ_X) be a standard Borel space and $\mathcal{R} \in \mathscr{R}(X)$ be a Borel relation. Then $\Gamma\mathcal{R} = \Theta\mathcal{R}$.*

Proof. It is a result from [Katsumata and Sato 2015] that for any $\mu, \nu \in \Delta(X)$ and $\mathcal{R} \in \mathscr{R}(X)$, one has $\mu \; \Gamma\mathcal{R} \; \nu$ if and only if $\int_X f d\mu \leq \int_X f d\nu$, for any measurable function $f : X \to [0, 1]$ such that $x \; \mathcal{R} \; y \implies f(x) \leq f(y)$. From there, we can conclude by applying the Monge-Kantorowitch duality result for Borel cost functions from e.g. [Kellerer 1984] [Beiglböck et al. 2009]. □

3.2 State Bisimilarity

The first notion of equivalence we consider is the natural generalization of bisimilarity to transition systems on continuous state spaces. The natural extension of a (probabilistic) transition system to continuous states is the notion of a Labelled Markov Process (LMP, for short).

[2] Notice that if $\mathcal{R}$ is an equivalence, then so is $\Gamma\mathcal{R}$.

[3] In fact, whereas Γ gives a lax relational extension of $\mathcal{G}$, the map Θ does not. In particular, the map Θ corresponds to the so-called Barr lifting [Barr 1970] of $\mathcal{G}$. Good properties (viz. being a lax extension) of such a lifting are deeply connected with weak-pullback preservation of $\mathcal{G}$ which, in general, does not hold [Viglizzo 2005].

Definition 3.6 (Labelled Markov Processes). A *Labelled Markov Process* (LMP) $\mathcal{M}$ is a triple $(\mathcal{S}, \mathcal{A}, \{h_a \mid a \in \mathcal{A}\})$, where $\mathcal{S}$ (the set of states) is measurable, $\mathcal{A}$ (the set of labels) is an arbitrary set, and for every $a \in \mathcal{A}$ the map $h_a : \mathcal{S} \times \Sigma_{\mathcal{S}} \to [0,1]$ is a sub-probability kernel. If moreover $\mathcal{A}$ is a measureable set, and for every $s \in \mathcal{S}$ the map $a \in \mathcal{A} \times A \in \Sigma_{\mathcal{S}} \mapsto h_a(s, A)$ is a sub-probability kernel, then we say that $\mathcal{M}$ is a *Measurably Labelled Markov Process*.

In the literature, the set $\mathcal{A}$ is very often taken to be countable. However, this is not the case for the LMPs which model parameter passing in calculi whose set of terms is itself not countable, since labels *include* terms. Notice that a LMP $(\mathcal{S}, \mathcal{A}, \{h_a \mid a \in \mathcal{A}\})$ gives a $\mathcal{A}$-indexed family of measurable maps $h_a : \mathcal{S} \to \mathcal{G}(\mathcal{S})$. Exploiting this observation, a state bisimulation is defined in terms of relational extensions for the Giry monad.

Definition 3.7 (State Bisimulations). Given a LMP $\mathcal{M}$, a *state bisimulation relation* is a relation $\mathcal{R} \in \mathscr{R}(\mathcal{S})$ such that, for all states s, t, if $s \mathcal{R} t$ holds, then so does $h_a(s) \; \Gamma\mathcal{R} \; h_a(t)$, for any label a.

We say that s and t are *state bisimilar* — and write $s \sim^{\mathcal{M}}_{state} t$ — if there exists a state bisimulation $\mathcal{R}$ such that $s \mathcal{R} t$. When the LMP $\mathcal{M}$ is clear from the context, we write $\sim_{state}$ in place of $\sim^{\mathcal{M}}_{state}$ (and adopt the same convention for all the equivalences defined in this paper).

Notice the crucial presence of the map Γ in Definition 3.7. In fact, one can obtain another definition of state bisimilarity simply by replacing Γ with Θ in Definition 3.7. Since $\Theta\mathcal{R} \subseteq \Gamma\mathcal{R}$ holds for any relation $\mathcal{R}$, bisimilarity based on Θ is included in bisimilarity based on Γ, and the two coincide when they are Borel relations (Theorem 3.5). Unless otherwise specified, state bisimilarity will always refer to Definition 3.7, and thus to the map Γ. The reason behind such a choice is twofold: on the one hand, Γ-based bisimilarity is the one usually finds in the literature; on the other hand, Γ has in general nicer properties than Θ ensuring, e.g., Γ-based bisimilarity to be an equivalence relation.

3.3 Logical Equivalence

Another classic approach to define program equivalence is via logic [Desharnais et al. 1998; Larsen and Skou 1991]. Informally, formulas of the logic are used as discriminators between programs, and two programs are equivalent if they yield the same interpretation for all formulas. In our setting, such a logic takes the form of a probabilistic modal logic.

Definition 3.8 (Probabilistic Modal Logic). Given a set $\mathcal{A}$ of actions, we define a class of logical formulas by way of the following grammar, where $a \in \mathcal{A}$ and $q \in \mathbb{Q} \cap [0,1]$:

$$\phi \in \mathcal{L} \;::=\; \top \;\mid\; \phi_1 \wedge \phi_2 \;\mid\; \langle a \rangle_q \phi.$$

The semantics of formulas is given parametrically on a LMP $(\mathcal{S}, \mathcal{A}, \{h_a \mid a \in \mathcal{A}\})$ by associating to each formula in $\mathcal{L}$ an element of Σ as follows, where $\langle a \rangle_r A = \{s \mid h_a(s, A) > r\}$.

$$[\![\top]\!] = \mathcal{S}; \qquad [\![\phi_1 \wedge \phi_2]\!] = [\![\phi_1]\!] \cap [\![\phi_2]\!]; \qquad [\![\langle a \rangle_q \phi]\!] = \langle a \rangle_q [\![\phi]\!].$$

We say that two states $s, t \in \mathcal{S}$ are *equivalent* with respect to $\mathcal{L}$ — and write $s \sim_{logic} t$ — when for every $\phi \in \mathcal{L}$, it holds that $(s \in [\![\phi]\!] \Leftrightarrow t \in [\![\phi]\!])$.

3.4 Testing Equivalence

The idea behind testing equivalence [Larsen and Skou 1991; van Breugel et al. 2005] is close to the one behind logical equivalence: in both cases, syntactic entities play the role of state discriminators. However, in a testing scenario one is allowed to use tests whose outcome is not necessarily a *truth value*, but rather the probability of passing the underlying test. The shift from a boolean to a

quantitative setting allows one to define equivalence between states of a LMP using a restricted grammar of tests.

Definition 3.9. Let $\mathcal{A}$ be a set of labels. We consider the following test language:

$$\mathcal{T} := \omega \;\mid\; \alpha \wedge \alpha \;\mid\; a \cdot \alpha \qquad \text{with } a \in \mathcal{A}.$$

For a given $\mathcal{M} = (\mathcal{S}, \Sigma, \{h_a \mid a \in \mathcal{A}\})$, we define by induction on tests the *success probability of the test α starting from a state s* as the measurable 1-bounded function $\mathrm{Prob}(\alpha, \cdot) : \mathcal{S} \to [0,1]$:

$$\mathrm{Prob}(\omega, s) = 1; \quad \mathrm{Prob}(a \cdot \alpha, s) = \int \mathrm{Prob}(\cdot, \alpha) dh_a(s, \cdot); \quad \mathrm{Prob}(\alpha \wedge \beta, s) = \mathrm{Prob}(\alpha, s) \cdot \mathrm{Prob}(\beta, s).$$

Notice that since tests have numerical outcomes, we do not need modalities in the test language.

Definition 3.10. Let $\mathcal{M}$ be a LMP. We write $s \sim_{test} t$ if $\mathrm{Prob}(s, \alpha) = \mathrm{Prob}(t, \alpha)$ for every $\alpha \in \mathcal{T}$.

3.5 Event Bisimilarity

The last notion of equivalence on LMPs we consider is *event bisimilarity*; it has been introduced in [Danos et al. 2006, 2005] as a coinductively defined notion of equivalence larger than state bisimilarity and coinciding with testing and logical equivalence for a wide class of LMPs. From a categorical viewpoint, it can be seen as going from a span-based notion of bisimulation to a co-span-based one. From an operational point of view, it is obtained by shifting the focus from equivalent states to visible *events* — represented as sub σ-algebras. In order to introduce event bisimilarity, it useful to recall how state bisimilarity can be characterized measure-theoretically:

Lemma 3.11. *Given an LMP $(\mathcal{S}, \Sigma, \{h_a :\mid a \in \mathcal{A}\})$, a binary relation $\mathcal{R}$ is a bisimulation if and only if $(\mathcal{S}, \Sigma(\mathcal{R}), \{h_a \mid a \in \mathcal{A}\})$ is a LMP.*

Event bisimilarity arises as a variation on the characterization in Lemma 3.11 obtained by removing the constraint about the underlying σ-algebra:

Definition 3.12. An event bisimulation on a LMP $(\mathcal{S}, \Sigma, \{h_a :\mid a \in \mathcal{A}\})$ is a sub-σ-algebra Λ of Σ, such that $(\mathcal{S}, \Lambda, \{h_a \mid a \in \mathcal{A}\})$ is a LMP. Let Λ be a σ-algebra on $\mathcal{S}$. We note $\mathcal{R}_\Lambda$ for the binary relation on states defined by $\mathcal{R}_\Lambda = \{(s,t) \mid \forall A \in \Lambda, s \in A \Leftrightarrow t \in A\}$. We say that two states $s, t \in \mathcal{S}$ are event bisimilar — and write $s \sim_{event} t$ — when there exists an an event bisimulation Λ such that $(s,t) \in \mathcal{R}_\Lambda$.

3.6 Comparing the Equivalences

Now that we have introduced four notions of equivalence on the states of any LMP — viz. *state* and *event* bisimilarity, and *logical* and *testing* equivalence — it is natural to consider the correspondence between them. As long as one focus on systems with discrete states (and countable actions) all the equivalences mentioned so far coincide [Danos et al. 2006, 2005; Fijalkow et al. 2017; van Breugel et al. 2005]. The correspondence still holds if LMPs have a *countable* number of labels and their underlying states space is *analytic*:

Theorem 3.13. *Let $\mathcal{M}$ be a LMP with analytic state space and countably many labels. Then:*

$$\sim^{\mathcal{M}}_{state} = \sim^{\mathcal{M}}_{logic} = \sim^{\mathcal{M}}_{event} = \sim^{\mathcal{M}}_{test}.$$

Proof. Fijalkow et al. [2017] have proved $\sim^{\mathcal{M}}_{state} = \sim^{\mathcal{M}}_{logic}$, whereas Danos et al. [2005] have shown $\sim^{\mathcal{M}}_{state} = \sim^{\mathcal{M}}_{event}$. The remaining identities actually hold on arbitrary LMPs, as we are going to see. □

Moving from LMPs with an analytic state space and countable actions to arbitrary LMPs, the logical and testing characterizations of event bisimilarity remains valid [Danos et al. 2006; van Breugel et al. 2005], whereas state bisimilarity has been proved to be finer then event bisimilarity [Danos et al. 2006].

Theorem 3.14 ([Danos et al. 2006; van Breugel et al. 2005]). *Let $\mathcal{M}$ be any LMP. Then*

$$\sim^{\mathcal{M}}_{state} \subseteq \sim^{\mathcal{M}}_{event} = \sim^{\mathcal{M}}_{logic} = \sim^{\mathcal{M}}_{test}.$$

Proof. Danos et al. [2006] have proved $\sim^{\mathcal{M}}_{state} \subseteq \sim^{\mathcal{M}}_{event}$, whereas van Breugel et al. [2005] and Danos et al. [2006] have shown $\sim^{\mathcal{M}}_{event} = \sim^{\mathcal{M}}_{logic} = \sim^{\mathcal{M}}_{test}$. □

As corollaries of Theorem 3.14, we obtain an approximation scheme for event bisimulation on LMPs with *uncountably many* labels and that event bisimilarity on LMPs with countably many actions is a Borel relation, a result we will use in Section 5 below.

Lemma 3.15. *(1) Let $\mathcal{A}$ be a (possibly uncountable) set and let $(\mathcal{S}, \Sigma, \{h_a \mid a \in \mathcal{A}\})$ be a LMP. Then two states s, t are event bisimilar in this LMP if and only if they are event-bisimilar in every LMP of the form $(\mathcal{S}, \Sigma, \{h_a \mid a \in \mathcal{B}\})$ where $\mathcal{B}$ a* countable *subset of $\mathcal{A}$.*

(2) Let $\mathcal{M}$ be a LMP with a countable set of actions, then $\sim^{\mathcal{M}}_{event}$ is Borel.

Proof sketch. (1) The proof relies on the equality $\sim^{\mathcal{M}}_{event} = \sim^{\mathcal{M}}_{logic}$ and the fact that to check whether a given formula ϕ holds for some state s, it is sufficient to consider a finite numbers of labels. More details are given in the long version of the paper.

(2) We notice that if $\mathcal{M}$ has countably many actions, then the set $\mathcal{L}$ of logical formulas for $\mathcal{M}$ is countable. Since for any such formula ϕ the set $[\![\phi]\!]$ is Borel, we have that $\sim^{\mathcal{M}}_{logic}$, and thus $\sim^{\mathcal{M}}_{event}$, is Borel.

□

3.7 A LMP for Λ_{P}.

The dynamic semantics of the language Λ_{P} we introduced in Section 2 can be presented as a LMP, this way enabling the various forms of equational reasoning principles presented above. In this section, we define the aforementioned LMP, then discuss why this LMP is problematic, deferring a more thorough discussion about it to Section 6 below. Actually, there is a standard way of turning (possibly effectful) typed λ-calculi into labelled transition systems: states are partitioned into two classes, namely computations and values, and the environment interacts with the former through an evaluation action and with the latter by inspecting the value (e.g. if it has a coproduct type) or by passing it an argument (if it is of an arrow type). We are going to follow this pattern, but an additional difficulty is present here, namely the one of dealing with real numbers as values. How should the environment inspect such a value? Ground types are generally managed through actions which fully reveal the underlying value. Remarkably, this works quite well in presence of discrete ground types and discrete probabilities. However, when we go from discrete to continuous probabilities — and thus from natural numbers to real numbers as base type — things become more complicated. In particular, a real number value can be inspected through the comparison operator $op_{\leq}$, which we ask to always be part of C.

Definition 3.16. We define $\mathcal{M}_{\Lambda_{\mathrm{P}}}$ as the LMP $(\mathcal{S}_{\Lambda_{\mathrm{P}}}, \mathcal{A}_{\Lambda_{\mathrm{P}}}, \{h_a \mid a \in \mathcal{A}_{\Lambda_{\mathrm{P}}}\})$ where:

- The set of states is $\mathcal{S}_{\Lambda_{\mathrm{P}}} = \bigcup_{\sigma \in T}\{(M, \sigma) \mid M \in \mathscr{T}_\sigma\} \cup \{(V, \sigma) \mid V \in \mathscr{V}_\sigma\}$; we take as topology on $\mathcal{S}_{\Lambda_{\mathrm{P}}}$ the countable disjoint union topology, see [Borgström et al. 2016; Ehrhard et al. 2018] (recall that **Pol** has countable coproducts).

- The set of actions is $\mathcal{A}_{\Lambda_P} = T \cup (\bigcup_{\sigma \in T} \mathscr{V}_\sigma) \cup \{\text{eval}\} \cup \{\overset{?}{\leq} q \mid q \in \mathbb{Q}\} \cup \{\text{case}(\hat{\imath}) \mid i \in I\} \cup \{\text{unbox}\}$. Again, we take the disjoint union topology on $\mathcal{A}_{\Lambda_P}$.
- The transition functions are defined as follows, where $h_a(s) = \emptyset$ in all cases not listed below.

$$
\begin{aligned}
h_\sigma(s) &= \{s^1\} && \text{if } s = (M, \sigma) \text{ or } s = (V, \sigma);\\
h_V(s) &= \{(WV, \tau)^1\} && \text{if } s = (W, \sigma \to \tau);\\
h_{\text{eval}}(s) &= (\llbracket M \rrbracket, \sigma) && \text{if } s = (M, \sigma);\\
h_{\overset{?}{\leq} q}(s) &= op_{\leq}(r, q) \cdot \{s^1\} && \text{if } \exists r \in \mathbb{R}, s = (\boldsymbol{r}, \textbf{real})\\
h_{\text{case}(\hat{\imath})}(s) &= \{(V, \sigma_i)^1\} && \text{if } s = ((\hat{\imath}, V), \sum_{j \in I} \sigma_j)\\
h_{\text{unbox}}(s) &= \{(V, \sigma[\mu\alpha.\sigma/\alpha])^1\} && \text{if } s = (\textbf{fold}\ V, \mu\alpha.\sigma)
\end{aligned}
$$

The LMP $\mathcal{M}_{\Lambda_P}$ we have just introduced naturally captures the interaction between a term and the environment, in the spirit of Abramsky's applicative bisimilarity. In particular, *state* bisimilarity on $\mathcal{M}_{\Lambda_P}$ captures applicative bisimilarity on Λ_P [Dal Lago and Gavazzo 2019], meaning that $M, N \in \mathscr{T}_\sigma$ are applicative bisimilar precisely when $(M, \sigma) \sim_{state}^{\mathcal{M}_{\Lambda_P}} (N, \sigma)$ (oftentimes, we use the notation $M \sim_{state}^{\mathcal{M}_{\Lambda_P}} N$) and similarly for values. Unfortunately, $\mathcal{M}_{\Lambda_P}$ is not among those LMPs for which nice correspondence results exist between the various notions of equivalences. Although the *state* space is analytic, indeed, the *action* space is inherently uncountable, due to the presence of values among the labels. This is unavoidable, given the nature of applicative bisimilarity, and has some deep consequences, as we are going to see in Section 6 below. But how about Λ_P^c? Actually, one can naturally define $\mathcal{M}_{\Lambda_P^c}$ exactly as $\mathcal{M}_{\Lambda_P}$, with the additional proviso that terms occurring in states and actions are taken from Λ_P^c rather than from Λ_P. In the next two sections we are going to show that going from $\mathcal{M}_{\Lambda_P}$ to $\mathcal{M}_{\Lambda_P^c}$ indeed makes a difference as far as the nature of relational reasoning is concerned.

4 CONTINUITY TO THE RESCUE

As already mentioned, most of the literature on relational reasoning on LMPs focuses on cases where labels are countable. A notable exception is [Fijalkow et al. 2017], in which it is shown that adding a *continuity* condition on the LMP transition function allows, in the case of *uncountably* many labels, to recover the coincidence between some notions of equivalence. We now briefly present the setting from [Fijalkow et al. 2017], showing that, unfortunately, it cannot be applied to LMPs like $\mathcal{M}_{\Lambda_P}$ or $\mathcal{M}_{\Lambda_P^c}$.

Definition 4.1. Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \{h_a \mid a \in \mathcal{A}\})$ be a LMP, such that $\mathcal{S}$ is a standard Borel space. We say it has a *continuous transition function* if, whenever $a \in \mathcal{A}$, $X \in \Sigma_{\mathcal{S}}$, the function $s \in \mathcal{S} \mapsto h_a(s, X) \in [0, 1]$ is continuous.

Unfortunately, the continuity requirement on the underlying transition function is not suitable for probabilistic transition systems built out of the operational semantics of higher-order programming languages like Λ_P. Intuitively—and as illustrated in Remark 4.2 below — this comes from the fact that Definition 4.1 forces any transition to either have a fuzzy probabilistic behaviour, or to go into a constant state *not* depending on the reals in the state of departure. As a consequence, it excludes deterministic (labelled) transitions in which a value is passed as an argument to a function. Since such steps exist independently on any continuity requirements, the continuous transition function constraint does not hold in (the LMPs underlying) any meaningful fragment of Λ_P.

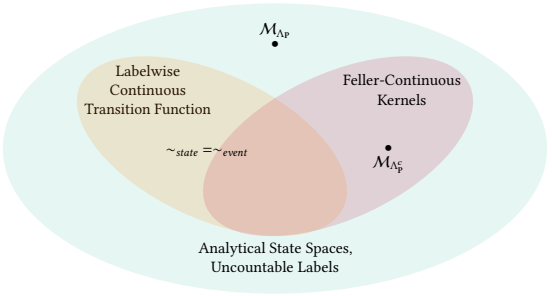

Fig. 3. LMP-equivalences for Analytical States Spaces and Uncountable Labels

Remark 4.2. *We illustrate on an example why neither $\mathcal{M}_{\Lambda_P^c}$ nor $\mathcal{M}_{\Lambda_P}$ have continuous transition functions. We start from the action eval, and the measureable set $X = \{\mathbf{0}\}$. We consider the family of terms $M_n = \boldsymbol{f}(\boldsymbol{r_n})$, where $(r_n)_{n\in\mathbb{N}}$ is a sequence of non-zero numbers converging to 0, and f is the identity, thus a continuous function on $\mathbb{R}$. In the standard Borel space $\mathcal{S}_{\Lambda_P^c}$ (or $\mathcal{S}_{\Lambda_P}$), we define a sequence $(u_n)_{n\in\mathbb{N}}$ by $u_n \triangleq (M_n, \mathbf{real})$. Observe that $u_n \xrightarrow[n\to\infty]{} (\boldsymbol{f}(\mathbf{0}), \mathbf{real})$ in the $\mathcal{S}_{\Lambda_P^c}$ (or $\mathcal{S}_{\Lambda_P}$) topology. By contrast, we see that $h_{eval}(M_n, \mathbf{real})(X) = 0$ for every n, while $h_{eval}(\boldsymbol{f}(\mathbf{0}), \mathbf{real})(X) = 1$; thus h_{eval} is not a continuous transition function.*

We thus need to go towards a different notion of continuity, which is what we are going to do in the next subsection.

4.1 Feller-Continuous LMPs

Our goal now is to introduce a different continuity requirement on LMPs, that both holds for $\mathcal{M}_{\Lambda_P^c}$ and enforces coincidence of event and state bisimilarities. We first introduce a notion of convergence for measures on a standard Borel space, which is in fact standard [Parthasarathy 2005; Swart and Winter 2013]. Recall that for any program M, $[\![M]\!]$ is a measure on the standard Borel space $\mathscr{V}$.

Definition 4.3. Let X be a standard Borel space, equipped with its Borel σ-algebra. Let $(\mu_n)_{n\in\mathbb{N}}$ be a sequence of measures over X. We say that $(\mu_n)_{n\in\mathbb{N}}$ *converges weakly* towards μ when for every bounded and continuous $f : X \to \mathbb{R}$ it holds that

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

The notion of weak convergence corresponds to a *topology* (actually, the space of sub-probability measures of a standard Borel space equipped with the weak convergence forms a Polish space, and thus a standard Borel space). Since we have equipped the space of measures with a topology, all this gives us a notion of continuity for kernels, called *Feller continuity* in the literature (see, e.g., [Swart and Winter 2013]).

Definition 4.4 (Feller-Continuous Kernels). Let X, Y be two standard Borel spaces. We say that a sub-probability kernel $k : X \nrightarrow Y$ is *Feller continuous* when the function $f_k : X \to \Delta(Y)$ is continous when equipping $\Delta(Y)$ with the weak topology.

It can be shown that Feller continuity is preserved by composition (see [Barthe et al. 2022]). We now use this notion of Feller kernel to enforce additional continuity requirements on LMPs.

Definition 4.5. Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \{h_a \mid a \in \mathcal{A}\})$ be a measureably labelled Markov process. We say that $\mathcal{M}$ is *Feller continuous* whenever $\mathcal{S}$ and $\mathcal{A}$ are two standard Borel spaces, and:

- for every $a \in \mathcal{A}$, the kernel $h_a : \mathcal{S} \times \Sigma_{\mathcal{S}} \to [0,1]$ is Feller continuous;
- for every $s \in \mathcal{S}$ the kernel $(a \in \mathcal{A}) \times (A \in \Sigma_{\mathcal{S}}) \mapsto h_a(s, A)$ is Feller continuous.

Please notice that it is enough for a Feller LMP to be continuous on $\mathcal{A}$ and $\mathcal{S}$ *separately*, and that we do not ask it to be jointly continuous on $\mathcal{A} \times \mathcal{S}$, which is in general a stronger condition [Piotrowski 1985]. Being a Feller continuous LMP is a well-behaved notion: in particular, we can observe that whenever a LMP $\mathcal{M}$ is actually a (non-probabilistic) LTS, its transition kernel is Feller. The same holds when actions and states are both countable. Our aim is to show that this notion of Feller continuity can indeed be applied to the applicative LMP for $\Lambda^c_{\mathbf{P}}$, but not the one for $\Lambda_{\mathbf{P}}$.

Remark 4.6. *Let us now discuss why $\mathcal{M}_{\Lambda_{\mathbf{P}}}$ is* not *Feller continuous: we consider the action eval, and we show that the kernel $h_{eval} : \mathcal{S}_{\Lambda_{\mathbf{P}}} \times \Sigma_{\mathcal{S}_{\Lambda_{\mathbf{P}}}} \to [0,1]$ is not Feller continuous. We consider the sequence of terms defined as: $M_n = \boldsymbol{f}(\boldsymbol{r_n})$, where $f : \mathbb{R} \to \mathbb{R}$ is the measurable function defined as $f(r) = 1$ if $r < 0$, and 0 otherwise, and $(r_n)_{n \in \mathbb{N}}$ is a sequence of strictly negative real numbers tending towards 0. We see that $h_{eval}(M_n, \mathbf{real}) = \{\mathbf{1}^1\}$ for every $n \in \mathbb{N}$, but $h_{eval}(\boldsymbol{f}(\mathbf{0}), \mathbf{real}) = \{\mathbf{0}^1\}$. Since it is not the case that the stationary sequence $(\{\mathbf{1}^1\})_{n \in \mathbb{N}}$ converges weakly towards $\{\mathbf{0}^1\}$ — with respect to the Borel topology on $\mathcal{S}_{\Lambda_{\mathbf{P}}}$ — we can conclude that $\mathcal{M}_{\Lambda_{\mathbf{P}}}$ is not Feller continuous.*

Remark 4.6 indeed shows that $\mathcal{M}_{\Lambda_{\mathbf{P}}}$ is not Feller continuous. Note that our counterexample is based on the existence of *non-continuous* primitive real functions in $\Lambda_{\mathbf{P}}$. We will deal with the Feller continuity of $\mathcal{M}_{\Lambda^c_{\mathbf{P}}}$ in Section 5 below.

4.2 State and Event Bisimilarity Coincide for Feller-Continuous LMPs

Notation 4.7. *In this section, we fix a Feller continuous LMP $\mathcal{M}$ with $\mathcal{A}$ as set of labels, and we fix a countably dense subset $\mathcal{A}^{\mathbb{Q}}$ of $\mathcal{A}$ (such a dense subset always exists, since Polish spaces are separable). We write $\mathcal{M}^{\mathbb{Q}}$ for the LMP extracted from $\mathcal{M}$ by retaining only the actions in $\mathcal{A}^{\mathbb{Q}}$. Observe that in the particular case of the LMP $\mathcal{M}_{\Lambda^c_{\mathbf{P}}}$ — i.e. the applicative LMP for the language $\Lambda^c_{\mathbf{P}}$ where all primitives are continuous — we can for instance define $\mathcal{M}^{\mathbb{Q}}_{\Lambda^c_{\mathbf{P}}}$ as in Definition 3.16, but restricting to values built from rationals.*

4.2.1 A Key Result from Optimal Transport. The proofs in this section will use in a crucial way a result coming from the literature on *optimal transport* (see [Villani 2008] for an introduction). A coupling $\omega \in \Omega(\mu, \nu)$ is said to be *optimal* with respect to some measurable cost function $c : X \times Y \to \mathbb{R}$ whenever it maximizes the quantity $\int c \cdot d\omega$. The field of optimal transport consists broadly in the study of such optimal couplings. For a Borel relation $\mathcal{R}$, we can express the relator Θ from Definition 3.3, by associating an appropriate cost function $c_{\mathcal{R}}$ to the relation $\mathcal{R}$: then it holds that $\mu \Theta \mathcal{R} \nu$ if and only if the cost of the optimal transport from μ to ν with respect to this cost function is 0. We will make the construction of $c_{\mathcal{R}}$ more precise in the proof of Corollary 4.9. This caracterisation of Θ in terms of optimal transport gives us a way to import results from this field. In this section, the key result we use comes from [Villani 2008], and is stated in Theorem 4.8 below. It can be read as the stability of the operator Θ from Definition 3.3 under weak convergence of measures.

Theorem 4.8 (from [Villani 2008]). *Let X and Y be standard Borel spaces, and let $c : X \times Y \to \mathbb{R}$ be a real-valued bounded continuous cost function. Let $(\mu_k)_{k \in \mathbb{N}}$, and $(\nu_k)_{k \in \mathbb{N}}$ be sequences of probability measures on X and Y respectively. Assume that μ_k converges to μ (resp. ν_k converges to ν) weakly. For each k, let $\pi_k \in \Omega(\mu_k, \nu_k)$ be an optimal coupling with respect to c. Then, there exists a coupling*

$\pi \in \Omega(\mu, \nu)$ such that, up to extraction of a subsequence, the sequence $(\pi_k)_{k\in\mathbb{N}}$ converges weakly to π, and moreover π is an optimal coupling.

Corollary 4.9. *Let R be a relation on standard Borel spaces X, Y such that R is a closed set. Let $(\mu_n)_{n\in\mathbb{N}}$, and $(\nu_n)_{n\in\mathbb{N}}$ be two sequences of sub-distributions such that $(\mu_n)_{n\in\mathbb{N}}$ and $(\nu_n)_{n\in\mathbb{N}}$ converge weakly towards μ and ν respectively, and moreover for every n, it holds that $\mu_n(\Theta R)\nu_n$. Then $\mu(\Theta R)\nu$.*

Proof. We can suppose without loss of generality—see [Barthe et al. 2022]—that the $\mu_n, \nu_n \dots$ are proper distributions. We define a bounded continuous function $c_R : X \times Y \to \mathbb{R}_{\geq 0}$ such that c_R is zero on R, and strictly positive everywhere else. We take $c_R(z) = \inf\{1\} \cup \{\|z - w\| \mid w \in R\}_{X\times Y}$. Observe that this function is continuous *because of* the closeness of R. From there—see again [Barthe et al. 2022]—we are able to apply Theorem 4.8 to c_R, and conclude using the fact that the couplings of weight 0 for c_R are exactly the couplings compatible with $\mathcal{R}$. □

4.2.2 Application to Feller-Continuous LMPs. We start with studying the bisimilarity landscape for the LMP $\mathcal{M}^{\mathbb{Q}}$. First, since $\mathcal{A}^{\mathbb{Q}}$ is countable, we see that event and state bisimilarity coincide on $\mathcal{M}^{\mathbb{Q}}$ (by Theorem 3.13). From now on, we will denote this relation $\sim^{\mathcal{M}^{\mathbb{Q}}}$, in order to emphasize that there is no difference here between state and event bisimilarities. We now look at how we can relate $\sim^{\mathcal{M}^{\mathbb{Q}}}$ to the state and event bisimilarities on $\mathcal{M}$: our end goal is to use Feller continuity constraints to show that all these relations coincide. The first step is to show, using Corollary (4.9) and Feller continuity of $\mathcal{M}$, that the topological closure of $\sim^{\mathcal{M}^{\mathbb{Q}}}$ is a state bisimulation for $\mathcal{M}$, thus is contained into $\sim^{\mathcal{M}}_{state}$.

Proposition 4.10. *The topological closure of $\sim^{\mathcal{M}^{\mathbb{Q}}}$ is a state bisimulation for $\mathcal{M}$.*

Proof. We write $\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}}$ for the topological closure of $\sim^{\mathcal{M}^{\mathbb{Q}}}$, i.e., $\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}}$ is the set of all pairs $(x, y) \in \mathcal{S}\times\mathcal{S}$ such that there exists a $(\sim^{\mathcal{M}^{\mathbb{Q}}})$-sequences $(x_n, y_n)_{n\in\mathbb{N}}$ with $x = \lim_{n\to\infty} x_n$, and $y = \lim_{n\to\infty} y_n$. Let $s, t \in \mathcal{S}$ be such that $s \sim^{\mathcal{M}^{\mathbb{Q}}} t$, and $a \in \mathcal{A}$. Our goal from here is to show that for every $a \in \mathcal{A}$, $h_a(s)\Gamma\mathcal{R}h_a(t)$. Let us first unfold our hypothesis:

1. Since $\mathcal{A}^{\mathbb{Q}}$ is dense in $\mathcal{A}$, there exists a sequence $(a_m)_{m\in\mathbb{N}}$ of $\mathcal{A}^{\mathbb{Q}}$ elements such that $a_m \xrightarrow[n\to\infty]{} a$;
2. Since $\mathcal{M}$ is a Feller continuous LMP, $x_n \xrightarrow[n\to\infty]{} x$, $y_n \xrightarrow[n\to\infty]{} y$, $a_m \xrightarrow[m\to\infty]{} a$ implies that: $\forall m, h_{a_m}(x_n) \xrightarrow[n\to\infty]{} h_{a_m}(x)$, $h_{a_m}(y_n) \xrightarrow[n\to\infty]{} h_{a_m}(y)$, and $\forall z \in \{x, y\}, h_{a_m}(z) \xrightarrow[m\to\infty]{} h_a(z)$, thanks to weak convergence;
3. For every $n \in \mathbb{N}$, since $x_n \sim^{\mathcal{M}^{\mathbb{Q}}} y_n$, it holds that for every $m \in \mathbb{N}$, $h_{a_m}(x_n)\,\Gamma(\sim^{\mathcal{M}^{\mathbb{Q}}})\,h_{a_m}(y_n)$, thus since $(\sim^{\mathcal{M}^{\mathbb{Q}}})$ is contained into its topological closure–and by monotonicity of the Γ operator, we have that $h_{a_m}(x_n)\,\Gamma(\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}})\,h_{a_m}(y_n)$.

Recall that since $(\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}})$ is closed, it is in particular Borel, thus we obtain by applying Theorem 3.5 that $\Gamma(\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}}) = \Theta(\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}})$. From there (and again since $(\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}})$ is closed), we can apply Corollary 4.9: first we fix a m, and we make n tends towards infinity, obtaining that $h_{a_m}(x)\,\Gamma(\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}})\,h_{a_m}(y)$. Now, since this is true for every m, we can apply once again Corollary 4.9 by making m tends toward infinity, and we obtain that $h_a(x)\,\Gamma(\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}})\,h_a(y)$, which ends the proof. □

We are now able to combine Proposition(4.10) with the logical caracterisation of state bisimilarity for LMPs with countable labels (Proposition (3.13)) to show a logical caracterisation of state bisimilarity for Feller LMPs. Moreover, we obtain the aditional result that bisimilarity on Feller LMPs is a Borel relation, which is not necessarily the case for bisimilarities on arbitrary LMPs, as we will discuss in more details later in Section 6.

Theorem 4.11. *Let $\mathcal{M}$ is a Feller continuous LMP. Then it holds that $\sim^{\mathcal{M}}_{state} = \sim^{\mathcal{M}}_{logic} = \sim^{\mathcal{M}}_{event} = \sim^{\mathcal{M}}_{test}$, and moreover the graph of these relations is a topological closed set, thus a Borel set.*

Proof. Recall that we know already from the literature (see Theorem 3.14) that $\sim^{\mathcal{M}}_{state} \subseteq \sim^{\mathcal{M}}_{event} = \sim^{\mathcal{M}}_{logic} = \sim^{\mathcal{M}}_{test}$, thus it is enough to show that $\sim^{\mathcal{M}}_{state} = \sim^{\mathcal{M}}_{event}$. The main tool here is Proposition 4.10: it tells us that the topological closure of $\sim^{\mathcal{M}^{\mathbb{Q}}}$ is a state bisimulation on $\mathcal{M}$. Recall that by definition, $\sim^{\mathcal{M}}_{state}$ is the largest state bisimulation on $\mathcal{M}$; it means that:

$$\overline{\sim^{\mathcal{M}^{\mathbb{Q}}}} \subseteq \sim^{\mathcal{M}}_{state} \subseteq \sim^{\mathcal{M}}_{event}. \tag{2}$$

We are now going to build an inclusion chain in the reverse direction. Since $\mathcal{M}^{\mathbb{Q}}$ is an approximation of $\mathcal{M}$, we know that $\sim^{\mathcal{M}}_{event} \subseteq \sim^{\mathcal{M}^{\mathbb{Q}}}$ (it can be deduced for instance from Lemma 3.15), and since $\sim^{\mathcal{M}^{\mathbb{Q}}}$ is contained in its topological closure, it holds that $\sim^{\mathcal{M}^{\mathbb{Q}}} \subseteq \overline{\sim^{\mathcal{M}^{\mathbb{Q}}}}$. Summing up, we have:

$$\sim^{\mathcal{M}}_{event} \subseteq \sim^{\mathcal{M}^{\mathbb{Q}}} \subseteq \overline{\sim^{\mathcal{M}^{\mathbb{Q}}}}. \tag{3}$$

Combining (2) and (3), we obtain that $\sim^{\mathcal{M}}_{state} = \sim^{\mathcal{M}}_{event} = \sim^{\mathcal{M}^{\mathbb{Q}}} = \overline{\sim^{\mathcal{M}^{\mathbb{Q}}}}$, which ends the proof. □

5 FULL ABSTRACTION FOR $\Lambda^c_{\mathbf{p}}$

In this section, we show that the operational semantics of $\Lambda^c_{\mathbf{p}}$ is Feller continuous and prove, as a main consequence of that, a powerful full abstraction theorem stating that all the equivalences on $\Lambda^c_{\mathbf{p}}$ considered so far coincide. To achieve such a goal, we rely on the characterization of terms as pre-terms (Section 2) and show that the evaluation of any pair of programs $M[\vec{r}]$ and $M[\vec{s}]$, (where $\vec{r}$ and $\vec{s}$ are vectors of real numbers) is essentially the same, and only the reals *inside* the output value change. Moreover, the reals produced in output by $M[\vec{r}]$ depend continuously from $\vec{r}$. To formally express this idea, we define a *refinement* of the evaluation semantics of Section 2 that keeps track separately of the evolution of the program structure — which is discrete — and of the evolution of the continuous data. As a consequence, such a refinement needs to be a semantics on *pre-programs*; we call this semantics the *modular semantics* for pre-programs.

Notation 5.1. *To define the modular semantics, we use the following notation:*

- *We define $\mathcal{E}_k = \{(P, f) \mid P$ a pre-value with p holes, $f : \mathbb{R}^k \to^{=} \mathbb{R}^p$ a Feller continuous kernel$\}$, for $k \in \mathbb{N}$.*
- *If P is a pre-term with l_P holes and a free variable x that occurs l_x times in P, and V is a pre-value with l_V holes, we write $P[V/x]$ for the pre-term with $l_{P[V/x]} = l_P + l_x \cdot l_V$ holes, obtained by doing the substitution and re-indexing the holes canonically to obtain an increasing enumeration while reading the terms. We write $subst_{[\cdot]}$ for the associated function $subst^{[\cdot]}_{P,V} : \mathbb{R}^{l_P + l_V} \to \mathbb{R}^{l_{P[V/x]}}$;*
- *Finally, recall that we write λ for the uniform distribution on $[0, 1]$, and that given kernels f and g and a function $\phi : \mathbb{R}^n \to \mathbb{R}^m$, we write $g \circ f$ for the composition of f and g (in the category of kernels) and $\overline{\phi} : \mathbb{R}^n \to^{=} \mathbb{R}^m$ for the deterministic kernel associated to ϕ.*

Definition 5.2. We define *modular approximation semantics* as a function mapping a pre-term M with k holes to a non-decreasing family $(\llbracket M \rrbracket^{(n)}_{\star})_{n \in \mathbb{N}}$ of discrete sub-distributions over $\mathcal{E}_k$. The definition, by induction on n, is in Figure 4. Since $(\llbracket M \rrbracket^{(n)}_{\star})_{n \in \mathbb{N}}$ is a non-decreasing family on $\Delta(\mathcal{E}_k)$, it has a supremum, which is also a discrete distribution over $\mathcal{E}_k$, that we note $\llbracket M \rrbracket_{\star}$.

Before proceeding any further, let us observe that we can characterize the semantics of programs using our modular semantics on pre-terms.

$$[\![M]\!]_\star^{(0)} = \emptyset; \qquad [\![V]\!]_\star^{(n+1)} = \{(V, \overline{id})^1\} \text{ for } V \text{ pre-value}; \qquad [\![\boldsymbol{f}([]^1, \ldots, []^n)]\!]_\star^{(n+1)} = \{([], \overline{\boldsymbol{f}})^1\};$$

$$[\![\mathbf{sample}]\!]_\star^{(n+1)} = \{([], \overline{\lambda})^1\}; \qquad [\![\mathbf{unfold\ fold}\ V]\!]_\star^{n+1} = [\![V]\!]_\star^n;$$

$$[\![(\lambda x.P)V]\!]_\star^{(n+1)} = \sum_{(W,f)} [\![P[x/V]]\!]_\star^{(n)}(W, f) \cdot \{(W, f \circ \overline{subst^{[\cdot]\cdot}_{P,V}})^1\};$$

$$[\![\mathbf{pm}\ \langle \hat{\imath}, V\rangle\ \mathbf{as}\ \langle i, x\rangle.M_i]\!]_\star^{n+1} = \sum_{(W,f)} [\![M_i[V/x]]\!]_\star^n(W, f) \cdot \{(W, f \circ \overline{subst^{[\cdot]\cdot}_{M_i,V}})^1\};$$

$$[\![\mathbf{let}\ x = N_1\ \mathbf{in}\ N_2]\!]_\star^{(n+1)} =$$

$$\sum_{(W_1,f_1)} [\![N_1]\!]_\star^{(n)}(W_1, f_1) \cdot \sum_{(W_2,f_2)} [\![N_2[W_1/x]]\!]_\star^{(n)}(W_2, f_2) \cdot \{(W_2, f_2 \circ \overline{subst^{[\cdot]}_{N_2,W_1}} \circ (f_1 \times id))^1\}.$$

Fig. 4. Modular Semantics for Λ_{P}^c

Lemma 5.3. *For any pre-term M with p holes and $\vec{r} \in \mathbb{R}^p$, we have:*

$$\forall n \in \mathbb{N}. [\![M[\vec{\boldsymbol{r}}]]\!]^{(n)} = \sum [\![M]\!]_\star^{(n)}(V, f) \cdot V[\boldsymbol{f}(\vec{\boldsymbol{r}})] \tag{4}$$

$$[\![M[\vec{\boldsymbol{r}}]]\!] = \sum [\![M]\!]_\star(V, f) \cdot V[\boldsymbol{f}(\vec{\boldsymbol{r}})] \tag{5}$$

Proof. The proof, by induction on n, can be found in the long version [Barthe et al. 2022]. □

The notion of modular approximation semantics from Definition 5.2 has been designed as a mean to express that the semantics of a program $M[\vec{\boldsymbol{r}}] \in \Lambda_{\mathrm{P}}^c$ depends continuously on $\vec{r}$. We are now able to show this formally.

Proposition 5.4. *Let $(M_n)_{n\in\mathbb{N}}$ be a sequence of terms that converges towards M. Then the sequence of distributions $([\![M_n]\!])_{n\in\mathbb{N}}$ converges weakly towards $[\![M]\!]$.*

Proof. Observe that the fact that $(M_n)_{n\in\mathbb{N}}$ is a sequence of terms that converges towards M actually means that there exists a pre-term N (with k holes), a real-valued vector $\vec{r} \in \mathbb{R}^k$, and a sequence $\vec{r_n} \in \mathbb{R}^k$ such that $\vec{r_n} \to \vec{r}$ and $M = N[\vec{\boldsymbol{r}}]$, $M_n = N[\vec{\boldsymbol{r}}_n]$. Consequently, to prove that $([\![M_n]\!])_{n\in\mathbb{N}}$ converges weakly towards $[\![M]\!]$, it is enough to prove the following: let $g : \mathscr{V} \to \mathbb{R}$ be a bounded and continuous function and $\epsilon > 0$. Then, we need to show: $\exists N \in \mathbb{N}, \forall n \geq N, |\int_{\mathscr{V}} g.d[\![N[\vec{\boldsymbol{r}}_n]]\!] - \int_{\mathscr{V}} g.d[\![N[\vec{\boldsymbol{r}}]]\!]| \leq \epsilon$. Using the equivalence between semantics on programs and modular semantics on pre-programs given by Lemma 5.3, we see that:

$$\left|\int_{\mathscr{V}} g.d[\![N[\vec{\boldsymbol{r}}_n]]\!] - \int_{\mathscr{V}} g.d[\![N]\!][\vec{\boldsymbol{r}}]\right|$$

$$= \left|\sum_{(V,f)\in \mathrm{supp}([\![N]\!]_\star)} ([\![N]\!]_\star(V, f) \cdot (\int_{\mathscr{V}} g.dV[\boldsymbol{f}(\vec{\boldsymbol{r_n}})] - \int_{\mathscr{V}} g.dV[\boldsymbol{f}(\vec{\boldsymbol{r}})])\right|.$$

From there, we are able to conclude using the fact that all the $f : \mathbb{R}^k \to^= \mathbb{R}^p$ are Feller continuous kernels. More details can be found in the long version [Barthe et al. 2022]. □

Our goal now is to use Proposition 5.4 to show that $\mathcal{M}_{\Lambda_{\mathrm{P}}^c}$ is a Feller LMP, in the sense of Definition 4.4. That amounts to prove Feller continuity of two families of kernels: $h_a : \mathcal{S}_{\Lambda_{\mathrm{P}}^c} \to \mathcal{S}_{\Lambda_{\mathrm{P}}^c}$, for $a \in \mathcal{A}_{\Lambda_{\mathrm{P}}^c}$; and $\widehat{h^s} = (a \in \mathcal{A}_{\Lambda_{\mathrm{P}}^c}) \times (A \in \Sigma_{\mathcal{S}_{\Lambda_{\mathrm{P}}^c}}) \mapsto h_a(s, A)$, for $s \in \mathcal{S}_{\Lambda_{\mathrm{P}}^c}$.

Before doing these proofs, however, we need a technical lemma on how to combine Feller kernels on a measurable space build as a countable coproduct of standard Borel spaces. In fact, recall from Definition 2.2 that we endowed the set of terms and values with a (countable) coproduct standard Borel space structure.

Lemma 5.5. *Let $(X_n)_{n\in\mathbb{N}}$ be a countable family of (disjoint) standard Borel spaces, and Z be a standard Borel space. Let $\{f_n : X_n \to\!\!\!\!\bullet\ Z\}_{n\in\mathbb{N}}$ be a family of kernels, and let us write $\bigcup_{n\in\mathbb{N}} X$ for the standard Borel space obtained by endowing the disjoint union of the X_ns with the coproduct topology. We define $\bigsqcup_{n\in\mathbb{N}} f_n : \bigcup_{n\in\mathbb{N}} X \to\!\!\!\!\bullet\ Z$ as:*

$$\bigsqcup_{n\in\mathbb{N}} f_n(s) = f_{n_0}(s) \text{ with } n_0 \in \mathbb{N} \text{ unique such that } s \in X_{n_0}.$$

Then, if all the f_n are Feller continuous, also $\bigsqcup_{n\in\mathbb{N}} f_n$ is Feller continuous.

We are now ready to show that all relevant kernels for the LMP $\mathcal{M}_{\Lambda_{\mathrm{p}}^c}$ are Feller continuous. We do this below in Lemma 5.6 and Lemma 5.7.

Lemma 5.6. *For every $a \in \mathcal{A}$, the kernel $h_a : \mathcal{S}_{\Lambda_{\mathrm{p}}^c} \to\!\!\!\!\bullet\ \mathcal{S}_{\Lambda_{\mathrm{p}}^c}$ is Feller continuous.*

Proof. We present only the most interesting cases, the others can be found in the long version of this paper, and are obtained by small variations on the proofs below.

- If a = eval, then the thesis directly follows from Proposition 5.4.
- If $a = V \in \mathscr{V}_\sigma$, then we observe that $h_V = \bigsqcup_\tau h_V^\sigma \sqcup 0$, where $h_V^\sigma : \mathscr{V}_{\sigma\to\tau} \to\!\!\!\!\bullet\ \mathcal{S}_{\Lambda_{\mathrm{p}}^c}$. From Lemma 5.5, we see that it is enough to show that for every τ, and every $V \in \mathscr{V}_\sigma$, the kernel $h_V^\sigma : \mathscr{V}_{\sigma\to\tau} \to\!\!\!\!\bullet\ \mathcal{S}_{\Lambda_{\mathrm{p}}^c}$ is Feller. Moreover, we see that h_V^σ is deterministically generated by the function $\widehat{h}_V^{\sigma\to\tau} = W \in \mathscr{V}_{\sigma\to\tau} \mapsto WV \in \mathcal{S}_{\Lambda_{\mathrm{p}}^c}$. Looking at the topologies on $\mathscr{V}_\sigma$ and $\mathcal{S}_{\Lambda_{\mathrm{p}}^c}$, we see immediately that $\widehat{h}_V^\sigma$ is a continuous function.
- If $a \overset{?}{=\!\leq} q$, then we first recall that we have a comparison operator $op_\leq \in C_2$ such that $op_\leq : \mathbb{R}\times\mathbb{R} \to [0,1]$, and $op_\leq(x,y) = 1$ if and only if $x \leq y$. Since all primitive functions are continuous, $op_\leq$ is continuous too. Let us split $\mathcal{S}_{\Lambda_{\mathrm{p}}^c}$ as $\mathcal{S}_{\Lambda_{\mathrm{p}}^c} = \mathscr{V}_{\mathbf{real}} \sqcup Z$, and observe that $h_a = h_a^{\mathscr{V}_{\mathbf{real}}} \sqcup 0$. As before, we apply Lemma 5.5, and we see that it is enough to show that $h_a^{\mathscr{V}_{\mathbf{real}}}$ is Feller continuous. Recall that $h_a^{\mathscr{V}_{\mathbf{real}}}(\boldsymbol{r}) = p_\leq(r,q)\cdot\{r^1\}$ Let $(\mu_n)_{n\in\mathbb{N}}$ be a sequence of measures over $\mathbb{R}$ that converges weakly towards μ and let $f : \mathbb{R} \to \mathbb{R}$ be a bounded continuous function. Notice that for every distribution ν over reals, we have:

$$\int_{\mathbb{R}} f d(h_a^{\mathscr{V}_{\mathbf{real}}}(\nu)) = \int_{\mathbb{R}} f \times op_\leq(\cdot,q) d\nu. \tag{6}$$

  Since $op_\leq$ is bounded and continuous, we deduce from weak convergence of $(\mu_n)_{n\in\mathbb{N}}$ that $\int_{\mathbb{R}} f \times op_\leq(\cdot,q) d\mu_n \to_{n\to\infty} \int_{\mathbb{R}} f \times op_\leq(\cdot,q) d\mu$. By combining this with (6), we conclude that $\int_{\mathbb{R}} f d(h_a^{\mathscr{V}_{\mathbf{real}}}(\mu_n)) \to_{n\to\infty} \int_{\mathbb{R}} f d(h_a^{\mathscr{V}_{\mathbf{real}}}(\mu))$. Since it is true for every bounded continuous function f, it means that $h_a^{\mathscr{V}_{\mathbf{real}}}(\mu_n)$ converges weakly towards $h_a^{\mathscr{V}_{\mathbf{real}}}(\mu)$, and we can conclude from there. □

Lemma 5.7. *For every $s \in \mathcal{S}_{\Lambda_{\mathrm{p}}^c}$, the kernel $\widehat{h}_s : \mathcal{A}_{\Lambda_{\mathrm{p}}^c} \to\!\!\!\!\bullet\ \mathcal{S}_{\Lambda_{\mathrm{p}}^c}$ is Feller continuous.*

Proof. We do the proof by case analysis on s. We present the most interesting cases only (the others can be found in the long version of this paper [Barthe et al. 2022]).

- If $s = M$, with $M \in \mathscr{V}_\sigma$, then we split $\mathcal{A}_{\Lambda_{\mathrm{p}}^c}$ as $\mathcal{A}_{\Lambda_{\mathrm{p}}^c} = \{\mathrm{eval}\} \sqcup \{\sigma\} \sqcup Z$ (observe that this splitting is indeed consistent with Definition 3.16), and we write: $\widehat{h}_s = \widehat{h}_s^{\{\mathrm{eval}\}} \sqcup \widehat{h}_s^{\{\sigma\}} \sqcup Z$, with

$\widehat{h}_s^{\{\mathrm{eval}\}} : \{\mathrm{eval}\} \to \mathcal{S}_{\Lambda_{\mathrm{P}}^c}$ and $\widehat{h}_s^{\{\sigma\}} : \{\sigma\} \to \mathcal{S}_{\Lambda_{\mathrm{P}}^c}$. It is immediate that those two kernels are Feller, since they are defined on one-point spaces, so we can conclude the thesis by Lemma 5.5.

- If $s = W$, with W of type $\sigma \to \tau$, then we see that using Lemma 5.5 as above, it is enough to show that the kernel $\widehat{h}_s^{app} : V \in \mathscr{V}_\sigma \mapsto \{WV^1\}$ is Feller. Observe that this kernel can be written as the deterministic kernel $\overline{f}$, with $f(V) = WV$. Since f is continuous, we can conclude the thesis because kernels deterministically generated by a continuous function are Feller continuous.

□

From Lemma 5.6 and Lemma 5.7, we immediately obtain Feller continuity of the applicative LMP for our continuous λ-calculus $\mathcal{M}_{\Lambda_{\mathrm{P}}^c}$:

Proposition 5.8. *$\mathcal{M}_{\Lambda_{\mathrm{P}}^c}$ is Feller continuous.*

Finally, combining Proposition 5.8 and Theorem 4.11, we obtain the main result of this section: the *full abstraction theorem*. The latter states that all notions of bisimilarity we consider coincide on $\mathcal{M}_{\Lambda_{\mathrm{P}}^c}$. Consequently, we obtain a characterization of bisimilarity by means of tests, and we can then use such a characterization to prove full abstraction of bisimilarity with respect to context equivalence, this way showing that all the notions of equivalence on Λ_{P}^c considered so far coincide.

Theorem 5.9 (Full Abstraction Theorem).

$$\overset{\mathcal{M}_{\Lambda_{\mathrm{P}}^c}}{\sim_{state}} = \overset{\mathcal{M}_{\Lambda_{\mathrm{P}}^c}}{\sim_{logic}} = \overset{\mathcal{M}_{\Lambda_{\mathrm{P}}^c}}{\sim_{event}} = \overset{\mathcal{M}_{\Lambda_{\mathrm{P}}^c}}{\sim_{test}} = \overset{\mathcal{M}_{\Lambda_{\mathrm{P}}^c}}{\equiv_{ctx}}.$$

Proof. By Proposition 5.8 and Theorem 4.11, on $\mathcal{M}_{\Lambda_{\mathrm{P}}^c}$ we have: $\sim_{state} = \sim_{logic} = \sim_{event} = \sim_{test}$. The inclusion $\overset{\mathcal{M}_{\Lambda_{\mathrm{P}}^c}}{\sim_{state}} \subseteq \overset{\Lambda_{\mathrm{P}}^c}{\equiv_{ctx}}$ is obtained exactly as in the proof of soundness of applicative bisimilarity for Λ_{P} [Dal Lago and Gavazzo 2019] (recall that state bisimilarity on $\mathcal{M}_{\Lambda_{\mathrm{P}}^c}$ corresponds to applicative bisimilarity for Λ_{P}^c as defined by Dal Lago and Gavazzo [2019]). To conclude the proof, we need to show that $\overset{\Lambda_{\mathrm{P}}^c}{\equiv_{ctx}} \subseteq \overset{\mathcal{M}_{\Lambda_{\mathrm{P}}^c}}{\sim_{test}}$. The proof is based on the following construction: for every test $\alpha \in \mathcal{T}$, we build by induction on α a pair of contexts $(\mathfrak{C}_\alpha^{\mathscr{V},\sigma})$ and $(\mathfrak{C}_\alpha^{\mathscr{T},\sigma})$ such that the following invariant holds, for all values $V \in \mathscr{V}_\sigma$ and terms $M \in \mathscr{T}_\sigma$.

$$\mathrm{Prob}(\alpha, (V,\sigma)) = [\![\mathfrak{C}_\alpha^{\mathscr{V},\sigma}[V]]\!](1) \qquad \mathrm{supp}([\![\mathfrak{C}_\alpha^{\mathscr{V},\sigma}[V]]\!]) \subseteq [0,1]$$
$$\mathrm{Prob}(\alpha, (M,\sigma)) = [\![\mathfrak{C}_\alpha^{\mathscr{T},\sigma}[M]]\!](1) \qquad \mathrm{supp}([\![\mathfrak{C}_\alpha^{\mathscr{T},\sigma}[V]]\!]) \subseteq [0,1].$$

The construction is similar to the one used in the proof of full abstraction of bisimilarity for discrete probabilistic λ-calculi [Crubillé and Dal Lago 2014; Crubillé et al. 2015], the most interesting case now being the one of $\alpha \overset{?}{=\!\leq} q \cdot \alpha'$, with $q \in \mathbb{Q}$. We take $\mathfrak{C}_\alpha^{\mathcal{U},\sigma} = \mathbf{0}$, if $\sigma \neq \mathbf{real}$ or $\mathcal{U} \neq \mathscr{V}$; and $\mathfrak{C}_\alpha^{\mathscr{V},\mathbf{real}} = \boldsymbol{op}_{\leq}(\mathbf{sample}, \boldsymbol{op}_{\leq}(q, [])) * \mathfrak{C}_{\alpha'}^{\mathscr{V},\mathbf{real}}[]$, otherwise. We then see that when two programs M, N of type σ are context equivalent, then they have the same probability of success for any $\alpha \in \mathcal{T}$, and from there we can deduce that they are test bisimilar. □

6 PERSPECTIVES ON THE GENERAL CASE

After having proved that continuity of the *first-order* functions implies coincidence between the four notions of equivalence we have introduced in Section 3, we go back to the general case, namely the one of Λ_{P}. The only difference with respect to Λ_{P}^c is that first-order functions are not necessarily continuous. This, as we are going to see, turns out to have very strong consequences on the nature of coinductive relational reasoning. Before proceeding further, let us consider an example, which very clearly shows where the crux is.

Example 6.1. Consider the following two programs:

$$M \triangleq \textbf{let } x = \textbf{sample in } (\lambda y.\textbf{if } x = y \textbf{ then } 1 \textbf{ else } 0)$$
$$N \triangleq \textbf{let } x = \textbf{sample in } \lambda y.0.$$

The programs M and N are contextually equivalent. The easiest way to prove that is by going through a denotational interpretation of $\Lambda_{\mathbf{P}}$. Vákár et al. [2019] have recently introduced quasi-Borel domains and have proved their adequacy as a denotational semantics for a Bayesian λ-calculus with recursive types subsuming $\Lambda_{\mathbf{P}}$. Quasi-Borel domains thus provides an adequate semantics for $\Lambda_{\mathbf{P}}$ too, and one can show the denotational interpretations of M and N to coincide.[4] As highlighted in Sabok et al. [2021], relying on a connection with the privacy rule in ν-calculus, the equivalence between M and N can be understood from a security viewpoint: the sampling of x can be seen as the generation of a *private name* — or in cryptographic terms, a private key. Indeed, no adversary — i.e. external context—can *a priori* guess the private key x with non-negligible probability. On the other hand, are these programs also state bisimilar? Actually, state bisimilarity of M and N seems very hard to prove. How, then, about event bisimilarity?

6.1 Event Bisimulation is *Unsound*

Dal Lago and Gavazzo [2019] have proved soundness of state bisimilarity with respect to context equivalence in $\mathcal{M}_{\Lambda_{\mathbf{P}}}$:

Proposition 6.2 ([Dal Lago and Gavazzo 2019]). $\sim^{\mathcal{M}_{\Lambda_{\mathbf{P}}}}_{state} \subseteq \equiv^{ctx}$.

At that point, the reader may wonder whether it is possible to use state bisimilarity and Proposition 6.2 to deal with Example 6.1. Unfortunately that would not be so easy, because state bisimilarity when applied to the programs from Example 6.1 makes seemingly intractable measurability problems to appear, as we illustrate further in Example 6.8 below. In turn, Theorem 3.14 tells us that state bisimilarity is included in event bisimilarity. Event bisimilarity, then, would seem to be a natural candidate to go towards contextual equivalence.

Example 6.3. If we look again at the programs M and N from Example 6.1 — that are context equivalent — we can show explicitly that they are also event bisimilar (for a formal proof, see [Barthe et al. 2022]). Indeed, recall from the testing characterization of event bisimilarity given by Theorem 3.14, that an "adversary" for event bisimilarity can only do three things: evaluating a program (that we see as an agent in some cryptographic game); passing messages to this agent; doing two separate *copies* of any current state of the agent. Moreover, the adversary has to decide *before the game starts* which sequence of actions to play. Intuitively, we can see that none of its admissible strategies allows the adversary to distinguish between M and N with nonnull probability.

Using the testing characterization from Theorem 3.14, we can in fact show that context equivalence is contained in event applicative bisimilarity.

Proposition 6.4. $\equiv^{ctx} \subseteq \sim^{\mathcal{M}_{\Lambda_{\mathbf{P}}}}_{event}$.

Proof. Recall that event bisimilarity is caracterised by tests, i.e. $\sim^{\mathcal{M}_{\Lambda_{\mathbf{P}}}}_{event} = \sim^{\mathcal{M}_{\Lambda_{\mathbf{P}}}}_{test}$. From there, the proof is done exactly as in the proof of Theorem (5.9), by building for every test a context that emulates it. □

[4] The proof of denotational equality of M and N (and thus of their contextual equivalence) has been communicated to the authors by Dario Stein based on previous work by Sabok et al. [2021]. Details can be found in the extended version of this paper [Barthe et al. 2022].

The problem with event bisimilarity, at least when spelled out for $\Lambda_{\mathbf{P}}$, is that you simply become too coarse, thus unsound for context equivalence. The problem here lies in the fact that programs can do strictly more than what an "adversary" for event bisimilarity does: indeed they can somehow perform *man-in-the-middle attacks*; by contrast once the adversary from event bisimilarity has created two copies of a process, it cannot make them *interact* with each other. This is exploited in the following example.

Example 6.5. Let us consider the following two programs:

$$M \triangleq \textbf{let } x = \textbf{sample in } \lambda y.((\textbf{if } x = y \textbf{ then } 1 \textbf{ else } 0) \oplus x)$$
$$N \triangleq \textbf{let } x = \textbf{sample in } \lambda y.(0 \oplus x),$$

where $\oplus$ is as in Example 2.1. The two programs first sample a real number x, to be thought of as a private key. Then, with probability $\frac{1}{2}$, they behave as the corresponding process of Example 6.1, but with probability $\frac{1}{2}$ they just *reveal* their secret key. It means that a context, seen as an adversary, can perform the following attack: first, it *initiates* a session, which means that the process needs to choose its secret key. Then the adversary creates two copies of the underlying process: from there, it obtains with probability $\frac{1}{4}$ the original process of Example 6.1, and the secret key, and it can conclude its attack by passing the key to the process. Indeed, it turns out that M and N are not context equivalent, and we can prove it by building a context that distinguishes them. We take $C = (\lambda z.z(z1))[\,]$ and observe (a formal proof for this can be found in [Barthe et al. 2022]) that: $[\![C[M]]\!] = \frac{1}{2}\lambda + \frac{1}{4}\{1^1\} + \frac{1}{4}\{0^1\}$ and $[\![C[N]]\!] = \frac{1}{2}\lambda + \frac{1}{2}\{0^1\}$, where λ denotes the uniform distribution on $[0,1]$. From there, it is enough to compose C with a context D that tests the equality to 1.

The programs M, N of Example 6.5 are designed as a counterexample to the soundness of event bisimilarity. We have shown in Example 6.5 that they are not context *equivalent*: now, we are going to show that they are event *bisimilar*, using our approximation Lemma 3.15.

Example 6.6. Let $\mathscr{W}$ be any countable set of values. First, we define a restriction of the LMP $\mathcal{M}_{\Lambda_{\mathbf{P}}}$ from Definition 3.16: we keep the same state space, but we restrict the transitions to the ones labelled by a countable restriction of $\mathcal{A}_{\Lambda_{\mathbf{P}}}$, that depends on $\mathscr{W}$. More formally, we take:

$$\mathcal{M}^{\mathrm{app}}_{\mathscr{W}} \triangleq (\mathcal{S}_{\Lambda_{\mathbf{P}}}, \mathcal{A}^{\mathrm{app}}_{\mathscr{W}}, \{h_a \mid a \in \mathcal{A}^{\mathrm{app}}_{\mathscr{W}}\}),$$

where $\mathcal{A}^{\mathrm{app}}_{\mathscr{W}}$ is the standard Borel space defined by taking the restriction of $\mathcal{A}_{\Lambda_{\mathbf{P}}}$ to the set $T \cup \mathscr{W} \cup \{\eval\} \cup \{\overset{?}{\leq} q \mid q \in \mathbb{Q}\} \cup \{\mathrm{case}(\hat{\imath}) \mid i \in I\} \cup \{\mathrm{unbox}\}$. We are going to show that the programs M and N are bisimilar in each of the $\mathcal{M}^{\mathrm{app}}_{\mathscr{W}}$ — that have countable labels and analytical state spaces, thus state bisimilarity and event bisimilarity coincide there (by Theorem 3.13). From this point, we will be able to conclude by Lemma 3.15 that M and N are also event bisimilar in $\mathcal{M}_{\Lambda_{\mathbf{P}}}$.

We write $\mathcal{R}_{\mathscr{W}}$ for the bisimilarity on $\mathcal{M}^{\mathrm{app}}_{\mathscr{W}}$. We define: $S = \{(M,N)\} \cup \{(\hat{f_M}(r), \hat{f_N}(r)) \mid r \in \mathbb{R}, r \notin \mathscr{W}\} \cup \{(r,r) \mid r \in \mathbb{R}\}$, where $f_M, f_N : \mathbb{R} \to \mathscr{V}$ are defined thus:

$$f_M(r) \triangleq \lambda y.((\textbf{if } r = y \textbf{ then } 1 \textbf{ else } 0) \oplus r); \qquad f_N(r) \triangleq \lambda y.(0 \oplus r);$$

and we show that the reflexive closure of S — that we note $\mathcal{R}$ — is a bisimulation. That is: for each $(a,b) \in S$ and action α, $a \to_\alpha \mu_a$, $b \to_\alpha \mu_b$ implies that $\mu_a \Gamma\mathcal{R} \mu_b$.

- Let $r \in \mathbb{R}$ such that $r \notin \mathscr{W}$, and α be a *relevant* action for $(f_M(r), f_N(r))$. It means that α is an applicative action with a value of real type, so is of the form $r' \in \mathbb{R}$ with $r' \in \mathscr{W}$. Observe that in particular, it means that $r' \neq r$. So we need to show that $[\![f_M(r)r']\!] \Gamma\mathcal{R} [\![f_N(r)r']\!]$. Since we know that $r \neq r'$, we can see $[\![f_M(r)r']\!] = [\![f_N(r)r']\!] = \frac{1}{2}\{0^1\} + \frac{1}{2}\{r^1\}$, which allows us—since $\mathcal{R}$ is reflexive—to immediately conclude that $[\![f_M(r)r']\!] \Gamma\mathcal{R} [\![f_N(r)r']\!]$.

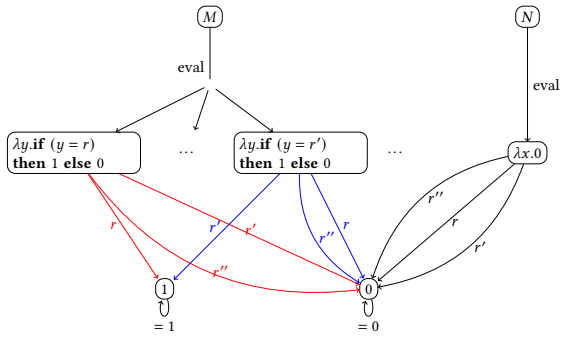

Fig. 5. The programs M, N: it holds that $M \nsim_{state} N$, $M \equiv^{ctx} N$, $M \sim_{event} N$.

- We look now at the pair $(M, N) \in S$. The only relevant action there is the eval action, so we can reformulate our goal as: $[\![M]\!]\Gamma\mathcal{R}[\![N]\!]$. To do that, we consider any $\mathcal{R}$-closed $A \in \Sigma^{\mathrm{app}}$. We need to show that $[\![M]\!](A) = [\![N]\!](A)$. First, looking at the operational semantics for M and N that we have computed in Example 6.5, we see that (notice that $\lambda(\mathscr{W}) = 0$):

$$[\![M]\!](A) = \lambda\{r \in \mathbb{R} \setminus \mathscr{W} \mid f_M(r) \in A\}; \qquad [\![N]\!](A) = \lambda\{r \in \mathbb{R} \setminus \mathscr{W} \mid f_N(r) \in A\}.$$

  Recall that A is R-closed: it means that for any $r \in \mathscr{W}$, $f_M(r) \in A$ if and only if $f_N(r) \in A$, thus $\{r \in \mathbb{R} \setminus \mathscr{W} \mid f_M(r) \in A\} = \{r \in \mathbb{R} \setminus \mathscr{W} \mid f_N(r) \in A\}$, which ends the proof.

Theorem 6.7. $\sim_{event}^{\mathcal{M}_{\Lambda_{\mathrm{P}}}} \not\subseteq \equiv^{ctx}$.

Proof. Directly from Example 6.5, Example 6.6, and our approximation Lemma 3.15. □

6.2 On Full Abstraction for State Bisimilarity

Unfortunately, $\mathcal{M}_{\Lambda_{\mathrm{P}}}$ is *not* one among the LMPs to which the correspondence results from Section 3 can be applied. Indeed, as we have just proved, logical equivalence, testing equivalence, and event bisimilarity are in fact *not sound*. Completeness of applicative bisimilarity was left as an open question in the work by Dal Lago and Gavazzo [2019]. We still cannot give a definite answer to that question, but some further light on the subject can indeed be given.

Example 6.8. Consider, once again, the programs M and N from Example 6.1. The relevant fragment of the underlying LMPs is in Figure 5, call it $\mathcal{M}_|$. Observe that $\mathcal{M}_|$ is a *stable fragment* of $\mathcal{M}_{\Lambda_{\mathrm{P}}}$, in the sense that from every state s in this fragment, the probability to end up again in this fragment after doing any action a is 1. Clearly, this is only a fragment of the whole applicative LMP, but *in this fragment*, it is easy to show that M and N are not bisimilar. Let $\mathcal{R}$ be a bisimulation. We can first observe easily that $(1, 0) \notin \mathcal{R}$. From there, we can see that for each state of the form:

$$L_r : \quad \lambda y.\mathbf{if}\ (y = r)\ \mathbf{then}\ 1\ \mathbf{else}\ 0,$$

it holds that $(L_r, \lambda x.0) \notin \mathcal{R}$: that is because the action $a = r$ goes to 1 with probability 1 when starting from the state L_r, and to 0 with probability 0 when starting from $\lambda x.0$. Looking at Figure 5, we see that there are actually no other states in $\mathcal{M}_|$ such that $(v, \lambda x.0) \in \mathcal{R}$. As a corollary, the set $\{\lambda x.0\}$ is $\mathcal{R}$-closed. Since moreover it is measureable, it means that $\{\lambda x.0\} \in \Sigma(\mathcal{R})$. From there, we end the proof by contradiction. Suppose that $M\mathcal{R}N$. We should have $h_{\mathrm{eval}}(M, \{\lambda x.0\}) = h_{eval}(N, \{\lambda x.0\})$, but it is not the case since $h_{\mathrm{eval}}(M, \{\lambda x.0\}) = 0$, while $h_{\mathrm{eval}}(N, \{\lambda x.0\}) = 1$.

It should be noted that to show that $\{\lambda x.0\}$ is $\mathcal{R}$-closed, we implicitly use the fact that there are no other states in the LMP, which is not the case if we consider the whole applicative LMP instead

of just a fragment of it. In order to adapt the proof for the whole LMP, we would need to show the existence of a $\mathcal{R}$-closed measurable set X that contains $\lambda x.0$, and not more that a negligible number of the L_r. We were however not able to show or disprove the existence of such a set X. A natural choice for X could seem to be the $\mathcal{R}$-equivalence class of $\lambda x.0$ instead of simply $\{\lambda x.0\}$. However, while this set is indeed $\mathcal{R}$-closed, it is however potentially not measurable. We can indeed show the existence of primitive functions which makes the equivalence class of $\{\lambda x.0\}$ non measurable (the proof, that can be found in the extended version [Barthe et al. 2022], use the same primitive function as in Proposition 2.5).

It would be appealing to be able to say that whenever two states are not bisimilar in a stable fragment, as in Example 6.8, then they are also not bisimilar in the whole LMP. We call this property a *locality property*: it formalizes the philosophical requirement that whenever two processes in a transition system are not equivalent, this should stay true when we add to the global system some other processes that do not interact at all with the original processes. From a more practical point of view, locality is appealing too, as no matter how simple the processes of interest are, if one uses non-local relations, then she is forced to consider the whole LMP with an uncountable state space.

We can see, using its logical characterization, that this locality property holds for $\sim_{event}$. Indeed, whether a formula ϕ holds for some state s does not depend on the behaviour of states independent from s. By contrast, we are not able to show locality in general for $\sim_{state}$ when both the states and the action space are standard Borel; crucially we are not able to show it in the special case of the LMP $\mathcal{M}_{\Lambda_P}$. We have not been able to find a concrete proof that locality does not hold for $\mathcal{M}_{\Lambda_P}$ either. The crux of the matter, as illustrated in Example 6.8, is the definition of the relator Γ from Definition 3.1. As mentioned in Definition 3.3, there exists an alternative notion of relator Θ based on *couplings*, that coincide with Γ in the restricted case of Borel relations (cf. Theorem 3.5). While the relation we would obtain when replacing Γ by Θ would be local, it is not know whether it is transitive, and thus an equivalence. Moreover this new state bisimilarity would not be complete, since it can be shown that it distinguishes the two programs of Example 6.8.

6.3 Bounding Non-Measurability in Descriptive Set-Theory

For context equivalence, we can go beyond our negative non-measurability result in Proposition 2.5, by using tools from descriptive set theory [Kechris 2012] to bound *how much* non-measurable $\equiv^{ctx}$ is. We can make the notion of non-measurability more precise relying on the so-called projective hierarchy [Kechris 2012] — the latter starting from Borel sets, and defining bigger and bigger classes using projections and complementation. More precisely, we can show that $\equiv^{ctx}$ is in the class of *co-analytic* sets, i.e. those that are defined as the complementary of an *analytic* set. All of that means that (the graph of) context equivalence is in the first layer of the projective hierarchy (see the extended version of this paper for more details).

We may wonder what is the position of the other notions of equivalence considered in this paper in the projective hierarchy: are their graphs measurable? Co-analytical? Are they in some levels of the projective hierarchy, at least? For event bisimilarity, we can show using its logical characterization that the situation is the same as for context equivalence: there exists a choice of primitive functions making (the graph of) $\sim_{event}$ not measurable. Nonetheless, $\sim_{event}$ is co-analytic. For state bisimilarity, instead, things are more complicated: as for event bisimilarity and contextual equivalence, we can show that there exists a choice of primitive functions making $\sim_{state}$ not measurable; but in contrast to the co-analiticity of $\sim_{event}$, we are unable to show that $\sim_{state}$ stays in some level of the projective hierarchy.

7 RELATED WORK

Programming languages with continuous probabilistic choices and higher-order functions have been the subject of numerous studies in the last ten years (e.g. [Borgström et al. 2016; Ehrhard et al. 2018; Staton et al. 2016; Vákár et al. 2019]). This, in particular, thanks to the potential applications to Bayesian programming, which however require the presence of conditioning operators, which we have not considered in this work. Program equivalence for higher-order languages with continuous probabilities has been studied mostly by means of denotational models, the latter being based on nontrivial mathematical structures — such as quasi-Borel spaces and domains [Staton 2017; Staton et al. 2016; Vákár et al. 2019], and measurable cones [Ehrhard et al. 2018] (in fact, measurable spaces do not form a closed category [Aumann 1961], and therefore they cannot provide an adequate semantics for higher-order languages). In this line of research, the recent work by Sabok et al. [2021] connecting probabilistic behaviours and name generation greatly inspired us in finding the examples from Section 6. Although all the aforementioned (denotational) models have been proved to be adequate for languages similar to (and sometimes even more expressive than [Vákár et al. 2019]) $\Lambda_{\mathbf{P}}$, to the best of the authors' knowledge full abstraction for such semantics is currently an open problem.

In contrast to the many results on denotationally-based program equivalence, only few works have focused on operationally-based notions of program equivalence. Among those, we mention the work by Wand et al. [2018] (see also the work by Culpepper and Cobb [2017]), where full abstraction of suitable bi-orthogonal logical relations on a stochastic λ-calculus with scoring is proved, and the work by Dal Lago and Gavazzo [2019], where soundness of a notion of applicative bisimilarity essentially coinciding with our notion of state bisimilarity on $\mathcal{M}_{\Lambda_{\mathbf{P}}}$ is proved. Even if fully abstract, bi-orthogonal logical relations have a high logical complexity: to logically relate two programs one needs to execute such programs inside arbitrary evaluation contexts. This makes bi-orthogonal logical relations oftentimes not readily usable to prove equivalence between programs. In contrast, the applicative bisimilarity by Dal Lago and Gavazzo [2019] is a lightweight technique, although it is currently unknown whether it is fully abstract.

Applicative bisimilarity [Abramsky 1990] is, together with open bisimilarity [Lassen 1999, 2005; Sangiorgi 1994], among the most effective and best studied notions of coinductive equivalence for programming languages with higher-order functions. Applicative bisimilarity has been studied for different languages, including nondeterministic [Lassen 1998; Ong 1993] and probabilistic [Crubillé and Dal Lago 2014; Dal Lago and Gavazzo 2019; Dal Lago et al. 2014] languages, and, more generally, languages with algebraic effects [Dal Lago et al. 2017, 2020; Gavazzo 2019]. For all those languages, applicative bisimilarity has been proved to be a sound technique for contextual equivalence. Full-abstraction results, instead, are more delicate. Even if applicative bisimilarity is known to be fully abstract in presence of certain kinds of effects, there are effects for which full abstraction fails, the prime example being nondeterminism [Lassen 1998]. Applicative bisimilarity is also known to be unsound for specific (non-algebraic) effects, such as local names [Koutavas et al. 2011]. The case of discrete probabilistic effects is among those for which full-abstraction results are possible, at least when programs are evaluated following the call-by-value discipline [Crubillé and Dal Lago 2014]. This is somehow surprising, since in the case of λ-calculi with nondeterministic choice, this correspondence no longer holds [Lassen 1998]. Probabilistic effects, in other words, appear to somehow *lie in between* determinism and nondeterminism. This phenomenon can be read through the lenses provided by characterizations of notions of bisimulation through *testing* [Larsen and Skou 1991; van Breugel et al. 2005]. When one moves from discrete to continuous distributions the situation changes. As already mentioned, applicative bisimilarity is sound in presence of continuous probabilities [Dal Lago and Gavazzo 2019], but full abstraction is currently an open problem.

There is a large body of work on generalization of the classic notion of probabilistic bisimilarity — and its logical and testing characterizations — as defined by Larsen and Skou [1991] to the setting of continuous distributions and LMPs [Panangaden 2009]. In particular, starting with the seminal work by Blute et al. [1997]; Desharnais et al. [1998, 2002], the notions of state and event bisimilarity [Danos et al. 2006, 2005] (as well as their metric counterparts [Desharnais et al. 1999, 2004]) have been proposed as notions of equivalence for LMPs. In that respect, there is a vast literature that investigates the characterization of the aforementioned notions of bisimilarity through logic-based and testing-based notions [Clerc et al. 2019; Danos et al. 2006, 2005; Fijalkow et al. 2017; van Breugel et al. 2005]. Even if dealing with LMPs (and thus with uncountable state spaces), all the aforementioned works consider LMPs with *countable* actions, and thus cannot be applied to the study of expressive higher-order languages, such as the ones studied in this paper. As an exception to that, we mention the work on bisimulation-based equivalences (and their characterizations) on LMPs with *continuous time* [Desharnais and Panangaden 2003; Fijalkow et al. 2017]. There, the set of actions of the LMP is replaced with a suitable notion of time, which usually has the cardinality of the continuum, and thus gives something closer to the LMPs we use in this paper. It should be noticed, however, that even if one can in principle see time as a continuous set of actions, the role played by these two notions is different, as different are the goals and challenges one has when working with LMPs with continuous time and with continuous actions.

8 CONCLUSION

This paper studies the nature of coinductive relational reasoning in the context of λ-calculi with continuous probabilistic choices. Surprisingly, the well-known correspondence between logical, testing, and bisimilarity equivalences is lost, with many equivalences turning out to be unsound for contextual equivalence. This, despite the fact that the theory of bisimilarity for probabilistic systems allows for a very smooth transition from the discrete to the continuous case. The deep reason for this hiatus comes from the fact that *uncountably* many actions are around, and this falsifies the characterization of bisimilarity by testing, namely the key step towards full-abstraction.

More on the positive side, this paper introduces a new class of probabilistic systems with uncountable actions in which the transition function is itself continuous, in the sense of Feller. For such systems, surprisingly, it is once again true that testing precisely characterizes bisimilarity, and this holds for both state and event bisimilarity. The aforementioned result is put at work in the context of a calculus in which real numbers can only be manipulated through continuous functions. This way the result of full abstraction is restored.

In this paper, we are *not* concerned with evaluating the expressive power of the thus obtained calculus, namely of $\Lambda^c_{\mathbf{P}}$. It is clear that not allowing a flow of information from reals to other types of data to happen — therefore preventing the value of any real number argument from altering the control flow — is a very strong restriction. However, it is not certain that this restriction is unacceptable, think for example at Bayesian programming. Along the same lines, we may wonder whether the $\Lambda^c_{\mathbf{P}}$ *contexts* are as powerful as the ones from $\Lambda_{\mathbf{P}}$, i.e., whether two $\Lambda^c_{\mathbf{P}}$-programs that are context equivalent in $\Lambda^c_{\mathbf{P}}$ are still equivalent when we consider *all* the contexts in $\Lambda_{\mathbf{P}}$. Such a full abstraction property does not seem to hold, since the higher-order nature of the considered languages could make it possible for a $\Lambda_{\mathbf{P}}$ context C to inject non-continuity inside a program M of $\Lambda^c_{\mathbf{P}}$. In any case, a thorough investigation along these lines is outside the scope of this paper, and left for future work.

REFERENCES

Samson Abramsky. 1990. The Lazy Lambda Calculus. In *Research Topics in Functional Programming*, D. Turner (Ed.). Addison Wesley, 65–117.

Robert J. Aumann. 1961. Borel structures for function spaces. *Illinois Journal of Mathematics* 5, 4 (1961), 614 – 630. https://doi.org/10.1215/ijm/1255631584

Roland Carl Backhouse and Paul F. Hoogendijk. 1993. Elements of a Relational Theory of Datatypes. In *Formal Program Development - IFIP TC2/WG 2.1 State-of-the-Art Report*. 7–42. https://doi.org/10.1007/3-540-57499-9_15

Hendrik Pieter Barendregt. 1984. *The lambda calculus: its syntax and semantics*. North-Holland.

Michael Barr. 1970. Relational algebras. *Lect. Notes Math.* 137 (1970), 39–55. https://doi.org/10.1007/BFb0060439

Gilles Barthe, Raphaëlle Crubillé, Ugo Dal Lago, and Francesco Gavazzo. 2022. On Feller Continuity and Full Abstraction (Long Version). *arXiv preprint arXiv:2207.10590* (2022).

Mathias Beiglböck, Christian Léonard, and Walter Schachermayer. 2009. A General Duality Theorem for the Monge–Kantorovich Transport Problem. *arXiv preprint arXiv:0911.4347* (2009).

Patrick Billingsley. 1986. *Probability and Measure* (second ed.). John Wiley and Sons.

Richard Blute, Josée Desharnais, Abbas Edalat, and Prakash Panangaden. 1997. Bisimulation for Labelled Markov Processes. In *Proc. of LICS 1997*. IEEE Computer Society, 149–158. https://doi.org/10.1109/LICS.1997.614943

Johannes Borgström, Ugo Dal Lago, Andrew D Gordon, and Marcin Szymczak. 2016. A lambda-calculus foundation for universal probabilistic programming. *Prof. of ICFP 2016*, 33–46. https://doi.org/10.1145/2951913.2951942

Florence Clerc, Nathanaël Fijalkow, Bartek Klin, and Prakash Panangaden. 2019. Expressiveness of probabilistic modal logics: A gradual approach. *Inf. Comput.* 267 (2019), 145–163. https://doi.org/10.1016/j.ic.2019.04.002

Raphaëlle Crubillé and Ugo Dal Lago. 2014. On Probabilistic Applicative Bisimulation and Call-by-Value λ-Calculi. In *Proc. of ESOP 2014 (LNCS)*, Vol. 8410. Springer, 209–228. https://doi.org/10.1007/978-3-642-54833-8_12

Raphaëlle Crubillé, Ugo Dal Lago, Davide Sangiorgi, and Valeria Vignudelli. 2015. On Applicative Similarity, Sequentiality, and Full Abstraction. In *Proc. of Symposium in Honor of Ernst-Rüdiger Olderog. (LNCS)*, Vol. 9360. Springer, 65–82. https://doi.org/10.1007/978-3-319-23506-6_7

Ryan Culpepper and Andrew Cobb. 2017. Contextual Equivalence for Probabilistic Programs with Continuous Random Variables and Scoring. In *Proc. of ESOP 2017 (LNCS)*, Hongseok Yang (Ed.), Vol. 10201. Springer, 368–392. https://doi.org/10.1007/978-3-662-54434-1_14

Ugo Dal Lago and Francesco Gavazzo. 2019. On Bisimilarity in Lambda Calculi with Continuous Probabilistic Choice. In *Proc. of MFPS 2019*. 121–141. https://doi.org/10.1016/j.entcs.2019.09.007

Ugo Dal Lago, Francesco Gavazzo, and Paul Blain Levy. 2017. Effectful applicative bisimilarity: Monads, relators, and Howe's method. In *Proc. of LICS 2017*. IEEE Computer Society, 1–12. https://doi.org/10.1109/LICS.2017.8005117

Ugo Dal Lago, Francesco Gavazzo, and Ryo Tanaka. 2020. Effectful applicative similarity for call-by-name lambda calculi. *Theor. Comput. Sci.* 813 (2020), 234–247. https://doi.org/10.1016/j.tcs.2019.12.025

Ugo Dal Lago, Davide Sangiorgi, and Michele Alberti. 2014. On coinductive equivalences for higher-order probabilistic functional programs. In *Proc. of POPL 2014*. 297–308. https://doi.org/10.1145/2535838.2535872

Vincent Danos, Josée Desharnais, François Laviolette, and Prakash Panangaden. 2006. Bisimulation and cocongruence for probabilistic systems. *Inf. Comput.* 204, 4 (2006), 503–523. https://doi.org/10.1016/j.ic.2005.02.004

Vincent Danos, François Laviolette, Josée Desharnais, and Prakash Panangaden. 2005. Almost Sure Bisimulation in Labelled Markov Processes. (2005). Unpublished note. Available at https://www.cs.mcgill.ca/~prakash/Pubs/new-neg.pdf.

Josée Desharnais, Abbas Edalat, and Prakash Panangaden. 1998. A Logical Characterization of Bisimulation for Labeled Markov Processes. In *Proc. of LICS 1998*. IEEE Computer Society, 478–487. https://doi.org/10.1109/LICS.1998.705681

Josée Desharnais, Abbas Edalat, and Prakash Panangaden. 2002. Bisimulation for Labelled Markov Processes. *Inf. Comput.* 179, 2 (2002), 163–193. https://doi.org/10.1006/inco.2001.2962

Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. 1999. Metrics for Labeled Markov Systems. In *Proc. of CONCUR 1999 (LNCS)*, Vol. 1664. Springer, 258–273. https://doi.org/10.1007/3-540-48320-9_19

Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. 2004. Metrics for labelled Markov processes. *Theor. Comput. Sci.* 318, 3 (2004), 323–354. https://doi.org/10.1016/j.tcs.2003.09.013

Josée Desharnais and Prakash Panangaden. 2003. Continuous stochastic logic characterizes bisimulation of continuous-time Markov processes. *J. Log. Algebraic Methods Program.* 56, 1-2 (2003), 99–115. https://doi.org/10.1016/S1567-8326(02)00068-1

Thomas Ehrhard, Michele Pagani, and Christine Tasson. 2018. Measurable cones and stable, measurable functions: a model for probabilistic higher-order programming. In *Proc. of POPL 2018*. 59:1–59:28. https://doi.org/10.1145/3158147

Nathanaël Fijalkow, Bartek Klin, and Prakash Panangaden. 2017. Expressiveness of probabilistic modal logics, revisited. In *Proc. of ICALP 2017 (LIPIcs)*, Vol. 80. 105:1–105:12. https://doi.org/10.4230/LIPIcs.ICALP.2017.105

Francesco Gavazzo. 2019. *Coinductive Equivalences and Metrics for Higher-order Languages with Algebraic Effects. (Equivalences coinductives et métriques pour les langages d'ordre supérieur avec des effets algébriques)*. Ph.D. Dissertation. University of Bologna, Italy.

Michèle Giry. 1982. A categorical approach to probability theory. In *Categorical Aspects of Topology and Analysis*. Springer Berlin Heidelberg, 68–85. https://doi.org/10.1007/BFb0092872

Shin-ya Katsumata, Tetsuya Sato, and Tarmo Uustalu. 2018. Codensity Lifting of Monads and its Dual. *Logical Methods in Computer Science* 14, 4 (2018). https://doi.org/10.23638/LMCS-14(4:6)2018

Shin-ya Katsumata and Tetsuya Sato. 2015. Codensity liftings of monads. In *Proc. of CALCO (LIPIcs)*, Vol. 35. 156–170. https://doi.org/10.4230/LIPIcs.CALCO.2015.156

Alexander S. Kechris. 2012. *Classical Descriptive Set Theory.* Springer New York.

Hans G. Kellerer. 1984. Duality theorems for marginal problems. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete* 67, 4 (1984), 399–432. https://doi.org/10.1007/BF00532047

Gregory M. Kelly. 2005. Basic Concepts of Enriched Category Theory. *Reprints in Theory and Applications of Categories* 10 (2005), 1–136.

Vasileios Koutavas, Paul Blain Levy, and Eijiro Sumii. 2011. From Applicative to Environmental Bisimulation. In *Proc. of MFPS 2011*. 215–235. https://doi.org/10.1016/j.entcs.2011.09.023

Kim G. Larsen and Arne Skou. 1991. Bisimulation through Probabilistic Testing. *Inf. Comput.* 94, 1 (1991), 1–28. https://doi.org/10.1016/0890-5401(91)90030-6

Søren Lassen. 1998. *Relational Reasoning about Functions and Nondeterminism.* PhD dissertation. Aarhus University. BRICS Dissertation Series BRICS DS-98-2.

Søren B. Lassen. 1999. Bisimulation in Untyped Lambda Calculus: Böhm Trees and Bisimulation up to Context. *Electr. Notes Theor. Comput. Sci.* 20 (1999), 346–374. https://doi.org/10.1016/S1571-0661(04)80083-5

Søren B. Lassen. 2005. Eager Normal Form Bisimulation. In *Proceedings of LICS 2005*. IEEE Computer Society, 345–354.

Paul Blain Levy, John Power, and Hayo Thielecke. 2003. Modelling environments in call-by-value programming languages. *Inf. Comput.* 185, 2 (2003), 182–210. https://doi.org/10.1016/S0890-5401(03)00088-9

Ernest G. Manes and Michael A. Arbib. 1986. *Algebraic Approaches to Program Semantics.* Springer.

James H. Morris. 1969. *Lambda Calculus Models of Programming Languages.* Ph.D. Dissertation. MIT.

C.-H. Luke Ong. 1988. *The Lazy Lambda Calculus: An Investigation Into the Foundations of Functional Programming.* Ph.D. Dissertation. Imperial College of Science and Technology.

C.-H. Luke Ong. 1993. Non-Determinism in a Functional Setting. In *Proc. of LICS 1993*. IEEE Computer Society, 275–286. https://doi.org/10.1109/LICS.1993.287580

Prakash Panangaden. 1999. The Category of Markov Kernels. *Electron. Notes Theor. Comput. Sci.* 22 (1999), 171–187. https://doi.org/10.1016/S1571-0661(05)80602-4

Prakash Panangaden. 2009. *Labelled Markov Processes.* Imperial College Press.

Kalyanapuram Rangachari Parthasarathy. 2005. *Probability measures on metric spaces.* Vol. 352. American Mathematical Soc. https://doi.org/10.1090/chel/352

Zbigniew Piotrowski. 1985. Separate and joint continuity. *Real Analysis Exchange* 11, 2 (1985), 293–322. https://doi.org/10.2307/44151750

Marcin Sabok, Sam Staton, Dario Stein, and Michael Wolman. 2021. Probabilistic programming semantics for name generation. In *Proc. of POPL 2021*. 1–29. https://doi.org/10.1145/3434292

Davide Sangiorgi. 1994. The Lazy Lambda Calculus in a Concurrency Scenario. *Inf. Comput.* 111, 1 (1994), 120–153. https://doi.org/10.1006/inco.1994.1042

Sam Staton. 2017. Commutative Semantics for Probabilistic Programming. In *Proc. of ESOP 2017 (LNCS)*, Vol. 10201. Springer, 855–879. https://doi.org/10.1007/978-3-662-54434-1_32

Sam Staton, Hongseok Yang, Frank D. Wood, Chris Heunen, and Ohad Kammar. 2016. Semantics for probabilistic programming: higher-order functions, continuous distributions, and soft constraints. In *Proc. of LICS 2016*. 525–534. https://doi.org/10.1145/2933575.2935313

Jan Swart and Anita Winter. 2013. Markov processes: Theory and Examples. (2013). Course Notes.

Pedro Sánchez Terraf. 2011. Unprovability of the logical characterization of bisimulation. *Inf. Comput.* 209, 7 (2011), 1048–1056. https://doi.org/10.1016/j.ic.2011.02.003

Matthijs Vákár, Ohad Kammar, and Sam Staton. 2019. A domain theory for statistical probabilistic programming. In *Proc. of POPL 2019*. 36:1–36:29. https://doi.org/10.1145/3290349

Franck van Breugel, Michael W. Mislove, Joël Ouaknine, and James Worrell. 2005. Domain theory, testing and simulation for labelled Markov processes. *Theor. Comput. Sci.* 333, 1-2 (2005), 171–197. https://doi.org/10.1016/j.tcs.2004.10.021

Ignacio D. Viglizzo. 2005. Final Sequences and Final Coalgebras for Measurable Spaces. In *Proc. of CALCO 2005 (LNCS)*, Vol. 3629. Springer, 395–407. https://doi.org/10.1007/11548133_25

Cédric Villani. 2008. *Optimal Transport: Old and New.* Springer Science & Business Media.

Mitchell Wand, Ryan Culpepper, Theophilos Giannakopoulos, and Andrew Cobb. 2018. Contextual equivalence for a probabilistic language with continuous random variables and recursion. *Proc. of ICFP 2018*, 1–30. https://doi.org/10.1145/3236782