Metric Reasoning About λ-Terms: The Affine Case

Raphaëlle Crubillé, Ugo Dal Lago

Metric Reasoning About λ-Terms: The Affine Case

Raphaëlle Crubillé
ENS Lyon
Email: raphaelle.crubille@ens-lyon.fr

Ugo Dal Lago
Università di Bologna & INRIA
Email: ugo.dallago@unibo.it

***Abstract*—Terms of Church's λ-calculus can be considered *equivalent* along many different definitions, but context equivalence is certainly the most direct and universally accepted one. If the underlying calculus becomes probabilistic, however, equivalence is too discriminating: terms which have totally unrelated behaviours are treated the same as terms which behave very similarly. We study the problem of evaluating the *distance* between affine λ-terms. A natural generalisation of context equivalence, is shown to be characterised by a notion of *trace* distance, and to be bounded from above by a coinductively defined distance based on the Kantorovich metric on distributions. A different, again fully-abstract, tuple-based notion of trace distance is shown to be able to handle nontrivial examples.**

I. Introduction

Probabilistic models are formidable tools when abstracting the behaviour of complicated, intractable systems by simpler ones, at the price of introducing uncertainty. But there is more: randomness can be seen as *a way to compute*; in modern cryptography, as an example, having access to a source of uniform randomness is essential to achieve security in an asymmetric setting [14]. Other domains where probabilistic models play a key role include machine learning [24], robotics [27], and linguistics [21].

Probabilistic models of computation have been studied not only directly, but also through concrete or abstract programming languages, which most often are extensions of their deterministic siblings. Among the many ways probabilistic choice can be captured in programming, the simplest one consists in endowing the language of programs with an operator modelling the flipping of a fair coin. This renders program evaluation a probabilistic process, and under mild assumptions the language becomes universal for probabilistic computation. Particularly fruitful in this sense has been the line of work on the functional paradigm, both at a theoretical [17], [26], [23] and at a more practical level [15].

In presence of higher-order functions, program equivalence can be captured by so-called *context equivalence*: two programs M and N are considered equivalent if they behave the same no matter how the environment interacts with them: for every context $\mathcal{C}$, it holds that $\text{Obs}(\mathcal{C}[M]) = \text{Obs}(\mathcal{C}[N])$. However, this definition has the drawback of being based on an universal quantification over *all* contexts: showing that two programs are equivalent requires considering their interaction with every possible context. The problem of giving handier characterisations of context equivalence can be approached in many different ways. As an example, coinductive methodologies for program equivalence have been studied thoroughly in deterministic [1], [25] and non-deterministic [19] computation, with new and exciting results appearing recently also for probabilistic languages: applicative bisimilarity, a coinductively defined notion of equivalence for functional programs, has been shown to be sound, and sometime even fully abstract, for probabilistic λ-calculi [6], [4].

In a probabilistic setting, however, equivalences are too discriminating if defined as above. Indeed, two programs are equivalent if their probabilistic behaviour is *exactly* the same (in every context). The actual value of probabilities in a probabilistic model often comes from statistical measurements, and should be considered more as an approximation to the actual probability law. Consequently, we would like to compare programs by appropriately reflecting small variations in them. Another scenario in which a richer, more informative way of comparing programs is needed is cryptography, where a central notion of equivalence, called *computational indistinguishability* [13] is indeed based on statistical distance rather than equality: the adversary *can* win the game, but with a *small* probability. Summing up, equivalences should be refined into metrics, and this is the path we will follow in this paper.

In probabilistic λ-calculi, the notion of observation $\text{Obs}(\cdot)$ is quantitative: it is either the *probability* of convergence to a certain observable base value (e.g. the empty string), or the probability of convergence *tout court*. One can then easily define a notion of *context distance* as the maximal distance contexts can achieve when separating two terms:

$$\delta^{\text{ctx}}(M,N) = \sup_{\mathcal{C}} \ |\text{Obs}(\mathcal{C}[M]) - \text{Obs}(\mathcal{C}[N])|.$$

This looks very close to computational indistinguishability, except for the absence of a security parameter: a scheme is secure if the advantage of any adversary in a given game (e.g., consisting in distinguishing between the case where the scheme is used, and the case where it is replaced by a truly random process) is "small" (e.g., negligible). Again, however, we find ourselves in front of a definition which risks to be useless in proofs, given that all contexts must be taken into account. But how difficult evaluating the distance between concrete higher-order terms really is? Are there ways to alleviate the burden of dealing with all contexts, like for equivalences? These are the questions we address in this

paper, and which have to the authors' knowledge not been investigated before.

As we will discuss in Section II below, finding handier characterisations of the context distance poses challenges which are simply different (and often harder) than the ones encountered in context equivalence. In particular, the context distance tends to *trivialise* and, perhaps worse, naively applying the natural generalisation of techniques known for equivalence is bound to lead to unsound methodologies. Indeed, one immediately realises that the number of times contexts access their argument is a crucial parameter, which must necessarily be dealt with. This is the reason why we work with an affine λ-calculus in this paper: this is a necessary first step, but also points to the right way to tame the general, non-linear case, as we hing in Section VI-C.

An extended version of this paper with more details is available [5]. The authors are partially supported by the ANR project 12IS02001 PACE.

Contributions

We introduce in this paper *three* distinct notions of distance for terms in an untyped, probabilistic, and affine λ-calculus. The first one is a notion of *trace distance*, in which terms are faced with *linear* tests, i.e. sequences of arguments. The distance between two terms is then defined as the greatest separation any linear test achieves. The first results of this paper are the non-expansiveness of the trace distance, which implies (given that any linear test can easily be implemented by an affine context) that the trace and context distances coincide. This is the topic of Section IV below.

Section V, instead, focuses on another notion of distance, which is coinductively defined following the well-known Kantorovich metric [18] for distributions of states in any labelled Markov chain (LMC in the following), and that we dub the *bisimulation distance*. This second notion of a distance is not only at least as discriminating as the trace distance, which is well expected, but non-expansive itself. This is proved by a variation on the Howe's method [16], a well-known technique for proving that bisimilarity is a congruence in an higher-order setting, and which has never been used for metrics before. On the other hand, the bisimulation distance *does not* coincide with the context distance, a fact that we do not only prove by giving a counterexample, but that we justify by relying on a test-based characterisation of the bisimulation distance known from the literature.

For the sake of simplicity, the trace and bisimulation distances are analysed on a purely applicative λ-calculus, keeping in mind that pairs could be very easily handled, and can even be encoded in the applicative fragment, as discussed in Section IV-C. The presence of pairs, however, allows us to form very interesting examples of distance problems, one of which will drive us throughout the paper but unfortunately turns out hard to handle neither by the trace distance nor by the bisimulation distance. This is the starting point for the third notion of distance introduced in this paper, which is the subject of Section VI, and which we call the *tuple distance*. Our third notion of distance can be proved to coincide with the trace distance, and thus with the context distance. But this is not the end of the story: in the tuple distance, not a single but *many* terms are compared, and this makes the distance between concrete terms much easier to evaluate: interaction is somehow internalised. In particular, our running example can be handled quite easily. The way the tuple distance is defined makes it adaptable to non-affine calculi, a topic which is outside the scope of this paper, but which we briefly discuss in Section VI-C.

Related Work

This is definitely not the first work on metrics for probabilistic systems: notions of coinductively defined metrics for LMCs, as an example, have been extensively studied (e.g. [10], [9], [28]). There have been, to our knowledge, not so many investigations on the meaning of metrics for concrete programming languages [12], and almost nothing on metric for higher-order languages.

If the key property notions of *equivalences* are required to satisfy consists in being *congruences*, the corresponding property for metrics has traditionally been taken as *non-expansiveness*. Indeed, many results from the literature (e.g. [10], [22]) have precisely the form of non-expansiveness results for metrics defined in various forms. The underlying language, however, invariably take the form of a process algebra without any higher-order feature. The work by Gebler, Tini, and co-authors shows that one could go beyond non-expansiveness and towards uniform continuity [12] but, again, higher-order functions remain out of scope.

Notions of *equivalence* for various forms of probabilistic λ-calculi have also been extensively studied, starting from the pioneering work by Plotkin and Jones [17], down to recent results on probabilistic applicative bisimulation [6], [4], logical relations [3], and probabilistic coherent spaces [7], [11]. None of the works above, however, go beyond equivalences and deals with notions of distance between terms.

II. The Anatomy of a Distance

In this section, we describe the difficulties one encounters when trying to characterise the context distance with either bisimulation or trace metrics.

Suppose we have two terms M and N of boolean type written in a probabilistic λ-calculus. As such, M and N do not evaluate to *a* value in the domain of booleans but to a *distribution* over the same domain. M evaluates to the distribution assigning **true** probability 1, while N evaluates to the uniform distribution over booleans, (i.e. the distribution which attributes probability $\frac{1}{2}$ to both **true** and **false**). Figure 1 depicts the relevant fragment of a LMC, whose induced notion of probabilistic bisimilarity has been proved to be sound for context equivalence [4]. M and N are not bisimilar. Indeed, $\widehat{\textbf{true}}$ and $\widehat{\textbf{false}}$ are trivially not bisimilar, while M and N go to equivalent states with different probabilities. The two terms are non-equivalent also contextually. But what *should be* the distance between M and N?

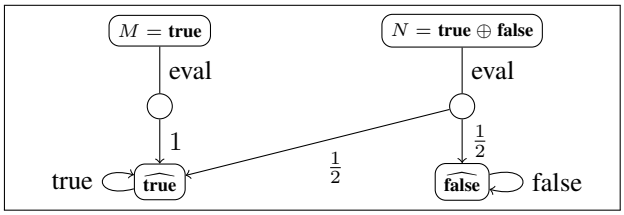

Fig. 1. M and N as States of a LMC.

For the moment, let us forget about the context distance, and concentrate on the notions of distance for LMCs we mentioned in Section I. In all cases we are aware of, we obtain that M and N are at distance $\frac{1}{2}$. As an example, if we consider a trace metric, we have to compare the success probability of linear tests, starting from M and N. More precisely, the tests of interest with respect to these two terms are:

$$t := \text{eval}; \qquad s := \text{eval} \cdot \text{true}; \qquad r := \text{eval} \cdot \text{false}.$$

Since neither M nor N has a non-zero divergence probability, they both pass the test t with probability 1. The success probability of the test s corresponds to the probability of evaluating to **true**: it is 1 for M and $\frac{1}{2}$ for N. Similarly, the success probability of r corresponds to the probability to obtain **false** after evaluation: it is 0 for M and $\frac{1}{2}$ for N. So we can see that the maximal separation linear tests can obtain is $\frac{1}{2}$. The situation is quite similar for bisimulation metrics [10], which attribute distance $\frac{1}{2}$ to M and N.

It is easy, however, to find a family of contexts $\{\mathcal{C}_n\}_{n\in\mathbb{N}}$ such that $\mathcal{C}_n[M]$ evaluates to **true** with probability 1, and $\mathcal{C}_n[N]$ evaluates to **false** with probability $1 - \frac{1}{2^n}$: define $\mathcal{C}_n$ as a context that copies its argument n times, returning **false** if *at least one* of the n copies evaluates to **false**, and otherwise returns **true**. As a consequence, the context distance between M and N is 1. In fact, this reasoning can be extended to any pair of programs which are not equivalent but whose probability of convergence is 1: out of a context which separates them of $\varepsilon > 0$, with ε very small, we can construct a context that separates them of 1 performing some statistical reasoning. The situation is more complicated if we take the probability of convergence as an observable: we cannot *always* construct contexts that discriminate terms based on their probability of convergence, although something can be done if the terms' probabilities of convergence are different but close to 1. The context metric, in other words, risks to be not continuous and close to trivial if contexts are too powerful. What the example above shows, however, is something even worse: if contexts are allowed to copy their arguments, then any metric defined upon the usual presentation of probabilistic λ-calculus as an LMC (a fragment of which is depicted in Figure 1) is bound to be *unsound* w.r.t. the context metric.

Whether bisimulation metrics are sound, how close they are to the context distance, and whether they are useful in relieving the burden of evaluating it, are however open and interesting questions even in absence of the copying capability, i.e., when the underlying language is *affine*. This is the main reason why we focus in this work on such a λ-calculus, whose expressive power is limited (although definitely non-trivial [20]) but which is anyway higher-order. We discover this way an elegant and deep theory in which trace and bisimulation metrics are indeed sound, At the end of this paper, some hints will be given about how the case of the untyped λ-calculus can be handled, a problem which we leave for future work.

Evaluating the context distance between affine terms is already an interesting and nontrivial problem. Consider, as an example, a sequence of terms $\{M_n\}_{n\in\mathbb{N}}$ defined inductively as follows (where Ω stands for a term with zero probability of converging):

$$M_0 = \langle \lambda x.\Omega, \lambda x.\Omega \rangle; \qquad M_{n+1} = \langle \lambda x.M_n, \lambda x.\Omega \rangle.$$

M_0 is the pair whose components are both equal to $\lambda x.\Omega$, and M_{n+1} is defined as a pair whose first component is the function which returns M_n whatever its argument is, and the second component is again $\lambda x.\Omega$. We are now going to define another sequence of terms $\{N_n\}_{n\in\mathbb{N}}$, which can be seen as a noisy variation on $\{M_n\}_{n\in\mathbb{N}}$. More precisely, N_0 is the same as M_0, and for each $n \in \mathbb{N}$, N_{n+1} is constructed similarly to M_{n+1}, but adding some negligible noise in *both* components:

$$N_0 = \langle \lambda x.\Omega, \lambda x.\Omega \rangle;$$
$$N_{n+1} = \langle \lambda x.N_n \oplus_{\frac{1}{2^{n+1}}} \Omega, \lambda x.\Omega \oplus_{\frac{1}{2^{n+1}}} I \rangle.$$

(I stands for the identity: $\lambda x.x$, while the term $L \oplus^p K$ has the same behaviour as L with probability $(1-p)$, and the same behaviour as K with probability p.) We would like to study how the distance between M_n and N_n evolves when n tends to infinity: do the little differences we apply at each step n accumulate, and how can we express this accumulation quantitatively?

Intuitively, it is easy for the environment to separate M_n and N_n of $\frac{1}{2^n}$: it is enough to consider a context $\mathcal{C}$ which simply takes the second component of the pair, passes any argument to it, and evaluates it: the convergence probability of $\mathcal{C}[M_n]$ is 0, while the convergence probability of $\mathcal{C}[N_n]$ is $\frac{1}{2^n}$. But the environment can also decide to take the *first* component of the pair, in order to use the fact that M_{n-1} and N_{n-1} can be distinguished: more precisely, let us suppose that we have a context $\mathcal{C}$ which separates M_{n-1} and N_{n-1}. Then we can construct a context $\mathcal{D}$ which takes the first element of the pair, passes any argument to it, tries to evaluate it, and if it succeeds, gives the result as an argument to $\mathcal{C}$. We would like to express the supremum of the separation that such a context can obtain as a function of the distance between M_{n-1} and N_{n-1}. Unfortunately, this is not so simple: if $\mathcal{C}$ is such that the convergence probability of $\mathcal{C}[M_{n-1}]$ is ε and the convergence probability of $\mathcal{C}[N_{n-1}]$ is ι, we can see that the convergence probability of $\mathcal{D}[M_n]$ is ε, whereas the convergence probability of $\mathcal{D}[N_n]$ is $(\iota \cdot (1 - \frac{1}{2^n}))$. But it is not possible to express $|\varepsilon - \iota \cdot (1 - \frac{1}{2^n})|$ as a function of $|\varepsilon - \iota|$ and n: intuitively, the separation that the context $\mathcal{D}$ can achieve depends not only on the separation that the context $\mathcal{C}$ can achieve, but also on *how* $\mathcal{C}$ achieves it. And moreover, the environment may of course

decide to use the two components of the pair, and to make them interact in an arbitrary way. Summing up, although the mechanism of construction of these terms seems to be locally easy to measure, it is complicated to have any idea about how the distance between them evolves when n tends to infinity.

III. Preliminaries

In this section, an affine and probabilistic λ-calculus, which is the object of study of this paper, will be introduced formally, together with a notion of context distance for it.

A. An Affine, Untyped, Probabilistic λ-Calculus

We endow the λ-calculus with a probabilistic choice operator $\oplus$, which corresponds to the possibility for the program to choose one between two arguments, each with the same probability. *Terms* are expressions generated by the following grammar:

$$M ::= x \;\Big|\; \lambda x.M \;\Big|\; MM \;\Big|\; M \oplus M \;\Big|\; \Omega,$$

where Ω models divergence[1], and x ranges over a countable set $\mathbf{X}$ of variables.

The class of affine terms, which model functions using their arguments at most once, can be isolated by way of a formal system, whose judgements are in the form $\Gamma \vdash M$ (where Γ is any finite set of variables) and whose rules are the following (where Γ, Δ stands for the union of two disjoints contexts):

$$\frac{}{\Gamma, x \vdash x} \qquad \frac{\Gamma, x \vdash M}{\Gamma \vdash \lambda x.M} \qquad \frac{\Gamma \vdash M \qquad \Delta \vdash N}{\Gamma, \Delta \vdash MN}$$

$$\frac{\Gamma \vdash M \qquad \Gamma \vdash N}{\Gamma \vdash M \oplus N} \qquad \frac{}{\Gamma \vdash \Omega}$$

A *program* is a term such that $\emptyset \vdash M$, and $\mathbf{P}$ is the set of all such terms. We will call them *closed terms*. We say that a program is a *value* if it is of the form $\lambda x.M$, and $\mathbf{V}$ is the set of all such programs. The semantics of the just defined calculus is expressed as a binary relation $\Downarrow$ between programs and *value subdistributions* (or simply *value distributions*), i.e. functions from values to real numbers whose sum is *smaller* or equal to 1. The relation $\Downarrow$ is inductively defined by the following rules:

$$\frac{}{\Omega \Downarrow \emptyset} \qquad \frac{V \in \mathbf{V}}{V \Downarrow \{V^1\}} \qquad \frac{M \Downarrow \mathscr{D} \qquad N \Downarrow \mathscr{E}}{M \oplus N \Downarrow \frac{1}{2}\mathscr{D} + \frac{1}{2}\mathscr{E}}$$

$$\frac{M \Downarrow \mathscr{D} \qquad N \Downarrow \mathscr{E} \qquad \{L\{V/x\} \Downarrow \mathscr{F}_{L,V}\}_{\lambda x.L \in \mathsf{S}(\mathscr{D}), V \in \mathsf{S}(\mathscr{E})}}{MN \Downarrow \sum \mathscr{D}(\lambda x.L) \cdot \mathscr{E}(V) \cdot \mathscr{F}_{L,V}}$$

where $\mathsf{S}(\mathscr{D})$ stands for the support of the distribution $\mathscr{D}$. The divergent program Ω, as expected, evaluates to the *empty* value distribution $\emptyset$ which assigns 0 to any value. The expression $\{V^1\}$ stands for the Dirac's value distribution on V; more generally the expression $\{V_1^{p_1}, \ldots, V_n^{p_n}\}$ indicates the value distribution assigning probability p_i to each V_i (and 0 to any other value).

[1]Since we only consider affine terms, we cannot encode divergence by the usual constructions of λ-calculus.

For every program M, there exists precisely *one* value distribution $\mathscr{D}$ such that $M \Downarrow \mathscr{D}$, that we note $[\![M]\!]$. This holds only because we restrict ourselves to affine terms. Moreover, $[\![M]\!]$ is always a finite distribution. The rule for application expresses the fact that the semantics is call-by-value: the argument is evaluated before being passed to the function. There is no special reason why we adopt call-by-value here, and all we are going to say also holds for (weak) call-by-name evaluation.

The way $\Downarrow$ is defined means that it is a *big-step* notion of semantics. In some circumstances, we need to have a more local view of how programs behave. We can define *small-step semantics*, again as a relation $\Rightarrow$ between programs and value distributions (itself defined on top of a relation $\rightarrow$ between programs and *program* distributions capturing a single evaluation step). The definition poses no significant problems [5]. Big-step and small-step semantics are equivalent: for every program M, there exists a unique distribution $\mathscr{D}$ such that $M \Rightarrow \mathscr{D}$, and moreover $\mathscr{D} = [\![M]\!]$.

B. Context Distance

We now want to define a notion of observation for programs which somehow measures the convergence probability of a program. We will do that following the previous literature on this subject. For any distribution $\mathscr{D}$ over a set A, its sum $\sum_{a \in A} \mathscr{D}(a)$ is indicated as $\sum \mathscr{D}$ and is said to be the *weight* of $\mathscr{D}$. The *convergence probability* of a term M, that we note $\mathcal{P}^{\mathrm{cv}}(M)$, is simply $\sum [\![M]\!]$, i.e., the weight of its semantics. For instance, the convergence probability of Ω is zero.

The environment, as usual, is modelled by the notion of a *context*, which is nothing more than a term with at most one occurrence of the *hole* $[\cdot]$. Contexts are generated by the following grammar:

$$\mathcal{C} ::= [\cdot] \;\Big|\; x \;\Big|\; \lambda x.\mathcal{C} \;\Big|\; \mathcal{C}M \;\Big|\; M\mathcal{C} \;\Big|\; \mathcal{C} \oplus \mathcal{C} \;\Big|\; \Omega.$$

Affine contexts can be identified by a formal system akin to the one for terms, that we elide due to lack of space (see [5]). We note as $\mathcal{C}[M]$ the program obtained by replacing $[\cdot]$ by the closed term M in $\mathcal{C}$. The interaction of a program M with a context $\mathcal{C}$ is the execution of the program $\mathcal{C}[M]$.

We now consider three different ways of comparing programs, based on their behaviour when interacting with the environment: a preorder $\leq^{\mathrm{ctx}}$, an equivalence relation $\equiv^{\mathrm{ctx}}$, and a map δ^{ctx}:

Definition 1 (Context Equivalence, Context Distance): Let M and N be two programs. Then we write that $M \leq^{\mathrm{ctx}} N$ if and only if for every context $\mathcal{C}$, it holds that $\mathcal{P}^{\mathrm{cv}}(\mathcal{C}[M]) \leq \mathcal{P}^{\mathrm{cv}}(\mathcal{C}[N])$. If $M \leq^{\mathrm{ctx}} N$ and $N \leq^{\mathrm{ctx}} M$, then we say that the two terms are *context equivalent*, and we write $M \equiv^{\mathrm{ctx}} N$. With the same hypotheses, we say the *context distance* between M and N is the real number $\delta^{\mathrm{ctx}}(M, N)$ defined as $\sup_{\mathcal{C}} |\mathcal{P}^{\mathrm{cv}}(\mathcal{C}[M]) - \mathcal{P}^{\mathrm{cv}}(\mathcal{C}[N])|$.

Please observe that, following [8], we only compare programs and not arbitrary terms. This is anyway harmless in an affine setting.

Example 1: Let I be the identity $\lambda x.x$. I and Ω are as far as two programs can be: $\delta^{\text{ctx}}(I, \Omega) = 1$. To prove that, finding a context which always converges for one of the terms, and always diverges for the other one, suffices. We can take $\mathcal{C} = [\cdot]$, and we have that $\mathcal{P}^{\text{cv}}(\mathcal{C}[I]) = 1$ and $\mathcal{P}^{\text{cv}}(\mathcal{C}[\Omega]) = 0$. Of course, I and Ω are not context equivalent. Throwing in probabilistic choice can complicate matters a bit. Consider the two terms $I \oplus \Omega$ and I. One can easily prove that $\delta^{\text{ctx}}(I \oplus \Omega, I) \geq \frac{1}{2}$: just consider $\mathcal{C} = [\cdot]$. However, showing that the above inequality is in fact an equality, requires showing that there cannot exist any context that separates more, which is possible, but definitely harder. This will be shown in the next section, using a trace-based characterisation of context distance.

C. On Pseudometrics

Which properties does the context distance satisfy, and which structure it then gives to the set of programs? This section answers these questions, and prepares the ground for the sequel by fixing some terminology.

Definition 2 (Pseudometrics): Let S be a set. A *premetric* on S is any function $\mu : S \to S$ such that $0 \leq \mu(s,t) \leq 1$ and $\mu(s,s) = 0$. A *pseudometric* on S is any premetric such that for every $s, t, u \in S$, it holds that $\mu(s,t) = \mu(t,s)$ and $\mu(s,t) \leq \mu(s,u) + \mu(u,t)$. The set of all pseudometrics on S is indicated with $\Delta(S)$.

Please observe that pseudometrics are not metrics in the usual sense, since $\mu(s,t) = 0$ does not necessarily imply that $s = t$. If we have a pseudometric μ, we can construct an equivalence relation by considering the *kernel* of μ, that is the set of those pairs (s,t) such that $\mu(s,t) = 0$. It is easy to prove that the context distance is indeed a pseudometric, and that its kernel is context equivalence. We would now want to define a preorder $\leq^{\text{metr}}$ on pseudometrics in such a way that if $\mu \leq^{\text{metr}} \rho$, then the kernel of μ is included in the kernel of ρ. The natural choice, then, is to take the following definition, which is the reverse of the pointwise order on $[0,1]$:

Definition 3 (Pseudometric Ordering): Let S be any set, and let μ and ρ be two metrics in $\Delta(S)$. Then we stipulate that $\mu \leq^{\text{metr}} \rho$ if and only if, for every $s, t \in S$ we have that $\rho(s,t) \leq \mu(s,t)$.

As is well-known, the relation $\leq^{\text{metr}}$ gives $\Delta(S)$ the structure of a complete lattice.

But when, precisely, can a pseudometric on programs be considered a sound notion of distance? First of all, we would like it to put two programs at least as far as the difference between their convergence probabilities, since this is precisely our notion of observation:

Definition 4 (Adequacy): Let μ be a pseudometric on the set of programs. Then μ is an *adequate pseudometric* if for any programs M and N, we have that $|\sum [\![M]\!] - \sum [\![N]\!]| \leq \mu(M,N)$.

Secondly, we are interested in how programs behave when interacting with the environment. Especially, if we have two terms M and N at a given distance ε, and we put them in an environment $\mathcal{C}$, we would like a pseudometric μ to give us some information about the distance between $\mathcal{C}[M]$ and $\mathcal{C}[N]$. This is the idea behind the following, standard, definition:

Definition 5 (Non-Expansiveness): Let μ be a pseudometric on programs. We say μ is *non-expansive* if for every pair of programs M and N and for every context $\mathcal{C}$, we have that $\mu(\mathcal{C}[M], \mathcal{C}[N]) \leq \mu(M,N)$.

Non-expansiveness is the natural generalisation of the notion of a *congruence*. By construction, δ^{ctx} is a non-expansive pseudometric. We can also adapt the notion of soundness to pseudometric; μ is said to be a *sound* pseudometric on programs if $\mu \leq^{\text{metr}} \delta^{\text{ctx}}$. Clearly, any adequate and non-expansive pseudometric is sound. In the rest of this paper, we will only deal with pseudometrics, but for the sake of simplicity we will refer to them simply as metrics.

IV. The Trace Distance

The first notion of metric we study is based on traces, i.e., linear tests. This is handier than the context distance, since contexts are replaced by objects with a simpler structure.

A. Definition

A *trace* s is a sequence in the form $@V_1 \cdots @V_n$, where $V_1, \ldots, V_n$ are values, and we note $\mathcal{T}r$ the set of traces. We define the probability that a program accepts a trace inductively on the length of the trace, as follows:

$$Pr(\lambda x.M, \epsilon) = 1;$$
$$Pr(\lambda x.M, @V \cdot s) = Pr(M\{V/x\}, s);$$
$$Pr(M, s) = \sum_V [\![M]\!](V) \cdot Pr(V, s) \quad \text{if } M \notin \mathbf{V}.$$

Please observe that the probability that a term M accepts a trace $s = @V_1 \cdots @V_n$ is the probability of convergence of $MV_1 \cdots V_n$. We are now going to define a metric, based on the probability that programs accept arbitrary traces:

Definition 6: Let M, N be two programs. Then we define the *trace distance* between them as $\delta^{\text{tr}}(M,N) = \sup_s |Pr(M,s) - Pr(N,s)|$. One can then define *trace equivalence* and the *trace preorder*, in the expected way [5].

Please observe that δ^{tr} is a pseudometric on programs in the sense of Definition 2, and that it is an adequate one. The kernel of δ^{tr} is nothing more than trace equivalence.

Example 2: The trace distance $\delta^{\text{tr}}(I \oplus \Omega, I)$ between $I \oplus \Omega$ and I is $\frac{1}{2}$. Showing that it is greater than $\frac{1}{2}$ is easy: it is sufficient to consider the empty trace. The other inequality, requires evaluating, for any trace s, the probability of accepting it. This is however much easier than dealing with all contexts, because we can now control the structure of the overall program we obtain: for any trace $s = @V_1 \cdots @V_n$, we can see that: $Pr(I \oplus \Omega, s) = \frac{1}{2} \cdot \sum [\![V_1 \cdots V_n]\!]$, and $Pr(I, s) = \sum [\![V_1 \cdots V_n]\!]$. The difference (in absolute value) between $Pr(I \oplus \Omega, s)$ and $Pr(I, s)$, then, cannot be greater than $\frac{1}{2}$.

The trace distance and the context distance indeed *coincide*, as well as the trace and context preorder, and the trace and

context equivalence. In the rest of this section, we will give the details of the proof for the pseudometric case, but the proof is similar for $\equiv^{\text{ctx}}$ and $\leq^{\text{ctx}}$. It is easy to realise that the context distance is a lower bound on the trace distance, since any trace $@V_1 \cdots @V_n$ can be seen as the context $[\cdot]V_1 \cdots V_n$. We thus focus on non-expansiveness.

B. Non-Expansiveness

Are there contexts that can separate strictly more than traces? In order to show that it is not the case, it is enough to show that δ^{tr} is non-expansive:

Theorem 1: Let M and N be two programs, and let $\mathcal{C}$ be a context. Then $\delta^{\text{tr}}(\mathcal{C}[M], \mathcal{C}[N]) \leq \delta^{\text{tr}}(M, N)$.

Since δ^{tr} is adequate, we can conclude that trace metric and context metric actually coincide:

Theorem 2: $\delta^{\text{ctx}} = \delta^{\text{tr}}$.

The rest of this section is devoted to an outline of the proof of Theorem 1 (see [5] for more details). The proof we give here is roughly inspired by the proof of congruence of trace equivalence for a non-deterministic λ-calculus [8]. The overall structure of the proof is the following: we first express the capacity of a program to do a trace by means of a labelled transition system (LTS in the following) whose states are distributions over programs. Then we consider *another* LTS, where the states are distributions over pairs of contexts and programs, that intuitively models the execution of $\mathcal{C}[M]$, but keeps the evolution of $\mathcal{C}$ and M apart. The first LTS, called $\mathcal{L}^{\text{tr}}$, has distributions over programs as states, and traces as actions. We indicate with $\dot{\Rightarrow}$ the transition relation associated to $\mathcal{L}^{\text{tr}}$. This LTS is itself defined on top of a labelled transition relation, noted $\xrightarrow{a}$, where $a \in \{\tau\} \bigcup \{@V \mid V \in \mathbf{V}\}$. Intuitively, a τ-step corresponds to an internal computation step for any term in the support of the distribution, while a $@V$-step corresponds to an interaction with the environment, which provides V as an argument. The relation $\dot{\Rightarrow}$ is defined as the accumulation of several steps of $\dot{\rightarrow}$. Both the relations $\dot{\Rightarrow}$ and $\dot{\rightarrow}$ are given in Figure 2. We write $\mathscr{D} \dot{+} \mathscr{E}$ for $\mathscr{D} + \mathscr{E}$ when we want to insist on $\mathscr{D}$ and $\mathscr{E}$ to have disjoint supports.

$$\frac{M \to \mathscr{E}}{\mathscr{D} \dot{+} \alpha \cdot \{M^1\} \xrightarrow{\tau} \mathscr{D} + \alpha \cdot \mathscr{E}}$$

$$\frac{\mathscr{D} \text{ value distribution}}{\mathscr{D} \xrightarrow{@V} \sum_M \mathscr{D}(\lambda x.M) \cdot \{M\{V/x\}^1\}} \qquad \frac{\mathscr{D} \text{ value distribution}}{\mathscr{D} \xrightarrow{\epsilon} \mathscr{D}}$$

$$\frac{\mathscr{D} \xrightarrow{\tau} \mathscr{E} \qquad \mathscr{E} \xrightarrow{t} \mathscr{F}}{\mathscr{D} \xrightarrow{t} \mathscr{F}} \qquad \frac{\mathscr{D} \xrightarrow{@V} \mathscr{E} \qquad \mathscr{E} \xrightarrow{t} \mathscr{F}}{\mathscr{D} \xrightarrow{@V \cdot t} \mathscr{F}}$$

Fig. 2. Small-step Trace Relations on Program Distributions.

Please observe that these relations are *not* probabilistic. The relation $\xrightarrow{\tau}$ is non-deterministic, since at each step we can decide which term of the distribution we want to reduce. However, $\xrightarrow{\tau}$ is strongly normalising and confluent. Moreover, we can show that $\dot{\Rightarrow}$ is in fact deterministic: for every program distribution $\mathscr{D}$, and every trace s, there exists a unique $\mathscr{E}$ such that $\mathscr{D} \overset{s}{\Rightarrow} \mathscr{E}$. The interest of the relation $\dot{\Rightarrow}$ is that it gives an alternate formulation for the probability of success in doing a trace: $Pr(M, s) = p$ if and only if there is a distribution $\mathscr{D}$ such that $p = \sum \mathscr{D}$ and $\{M^1\} \overset{s}{\Rightarrow} \mathscr{D}$. We can now use the relation $\dot{\Rightarrow}$ to give an equivalent formulation of Theorem 1: if M and N are such that $\delta^{\text{tr}}(M, N) \leq \varepsilon$, then for every trace s and context $\mathcal{C}$, if $\{\mathcal{C}[M]^1\} \overset{s}{\Rightarrow} \mathscr{D}$ and $\{\mathcal{C}[N]^1\} \overset{s}{\Rightarrow} \mathscr{E}$, then it holds that $|\sum \mathscr{D} - \sum \mathscr{E}| \leq \varepsilon$. This statement, however, cannot be proved directly, yet, because the way $\mathcal{C}$ and the argument terms interact is lost. It is then time to introduce our second LTS, called $\mathcal{L}^{\text{tr}}_{\mathbf{C} \times \mathbf{P}}$, which will allow us to relate $\{\mathcal{C}[M]^1\} \dot{\Rightarrow} \cdot$ to the behaviour of M: we want to talk about the evolution of a system consisting of the program M and the environment $\mathcal{C}$, while keeping the system and the environment as separate as possible. $\mathbf{C} \times \mathbf{P}$ is the set of pairs of the form $(\mathcal{C}, M)$, where $\mathcal{C}$ is a context and M is a program. We say that $(\mathcal{C}, M) \in \mathbf{C} \times \mathbf{P}$ is a value if $\mathcal{C}[M] \in \mathbf{V}$. The states of $\mathcal{L}^{\text{tr}}_{\mathbf{C} \times \mathbf{P}}$ are distributions over $\mathbf{C} \times \mathbf{P}$, while its labels are traces. We will note as $\dot{\Rightarrow}_{\mathbf{C} \times \mathbf{P}}$ the transition relation of $\mathcal{L}^{\text{tr}}_{\mathbf{C} \times \mathbf{P}}$. Similarly to what we have done for $\mathcal{L}^{\text{tr}}$, we first define an auxiliary relation $\xrightarrow{a}_{\mathbf{C} \times \mathbf{P}}$. The rules defining $\dot{\rightarrow}_{\mathbf{C} \times \mathbf{P}}$ and $\dot{\Rightarrow}_{\mathbf{C} \times \mathbf{P}}$ are in Figure 3. The relation $\xrightarrow{\tau}_{\mathbf{C} \times \mathbf{P}}$ is strongly normalising. As a consequence, $\dot{\Rightarrow}_{\mathbf{C} \times \mathbf{P}}$ is deterministic.

Intuitively, if s is a trace, the transition relation $\overset{s}{\Rightarrow}_{\mathbf{C} \times \mathbf{P}}$ corresponds to the transition relation $\overset{s}{\Rightarrow}$ starting from $\{\mathcal{C}[M]^1\}$. More precisely, if $\{\mathcal{C}[M]^1\} \overset{s}{\Rightarrow} \mathscr{D}$ and $\{(\mathcal{C}, M)^1\} \overset{s}{\Rightarrow}_{\mathbf{C} \times \mathbf{P}} \mathscr{E}$, then $\sum \mathscr{D} = \sum \mathscr{E}$. This allows us to give yet another equivalent formulation of Theorem 1: if $\delta^{\text{tr}}(M, N) \leq \varepsilon$, then if $\{(\mathcal{C}, M)^1\} \overset{s}{\Rightarrow}_{\mathbf{C} \times \mathbf{P}} \mathscr{D}$ and $\{(\mathcal{C}, N)^1\} \overset{s}{\Rightarrow}_{\mathbf{C} \times \mathbf{P}} \mathscr{E}$, it holds that $|\sum \mathscr{D} - \sum \mathscr{E}| \leq \varepsilon$. We are in fact going to show a stronger result, which uses the notion of ε-related distributions:

Definition 7: Let $\varepsilon \in [0, 1]$. We say that two distributions $\mathscr{D}$ and $\mathscr{E}$ over $\mathbf{C} \times \mathbf{P}$ are ε-related, and we note $\mathscr{D} \equiv^{\text{par}}_{\varepsilon} \mathscr{E}$ if there exist $n \in \mathbb{N}$, $\mathcal{C}_1, \ldots, \mathcal{C}_n$ distinct contexts, $p_1, \ldots, p_n$ positive real numbers with $\sum_i p_i \leq 1$, and $\mathscr{F}_1, \ldots, \mathscr{F}_n$, and $\mathscr{G}_1, \ldots, \mathscr{G}_n$ distributions over $\mathbf{P}$, such that:

- $\mathscr{D} = \sum_{1 \leq i \leq n} p_i \cdot (\mathcal{C}_i, \mathscr{F}_i)$;
- $\mathscr{E} = \sum_{1 \leq i \leq n} p_i \cdot (\mathcal{C}_i, \mathscr{G}_i)$;
- $\forall i$, $\delta^{\text{tr}}(\mathscr{F}_i, \mathscr{G}_i) \leq \varepsilon$.

Please observe that, if $\delta^{\text{tr}}(M, N) \leq \varepsilon$, then for every context $\mathcal{C}$, the distributions $\{(\mathcal{C}, M)^1\}$ and $\{(\mathcal{C}, N)^1\}$ are ε-related. In fact, the notion of ε-relatedness is a way to capture pairs of distributions over $\mathbf{C} \times \mathbf{P}$ representing *the same* environment, in which we put programs which are close for the trace pseudometric. The following can be seen as a stability result: if we start from ε-related distributions, and we do a trace s, we end up in two distributions which are still ε-related.

Lemma 1: Let $\mathscr{D}$, $\mathscr{E}$ be distributions over $\mathbf{C} \times \mathbf{P}$, and $\varepsilon \in [0, 1]$ such that $\mathscr{D} \equiv^{\text{par}}_{\varepsilon} \mathscr{E}$. Let s be a trace. Let $\mathscr{F}$ and $\mathscr{G}$ be such that: $\mathscr{D} \overset{s}{\Rightarrow}_{\mathbf{C} \times \mathbf{P}} \mathscr{F}$, and $\mathscr{E} \overset{s}{\Rightarrow}_{\mathbf{C} \times \mathbf{P}} \mathscr{G}$. Then $\mathscr{F} \equiv^{\text{par}}_{\varepsilon} \mathscr{G}$.

We can now see Theorem 1 as a direct consequence of Lemma 1. Indeed, let M and N be two programs at distance at most ε for the trace metric, and let $\mathscr{D}$ and $\mathscr{E}$ be such that

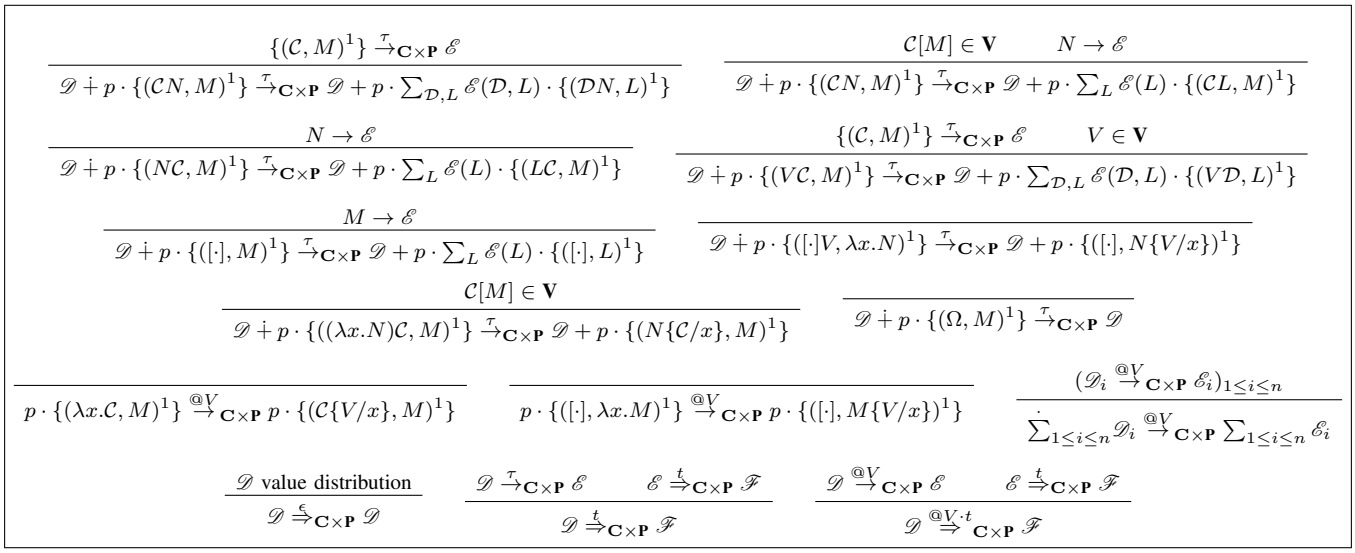

Fig. 3. Small-Step Trace Relations on Distributions over $\mathbf{C} \times \mathbf{P}$.

$\{(\mathcal{C}, M)^1\} \stackrel{s}{\Rightarrow}_{\mathbf{C}\times\mathbf{P}} \mathscr{D}$, and $\{(\mathcal{C}, N)^1\} \stackrel{s}{\Rightarrow}_{\mathbf{C}\times\mathbf{P}} \mathscr{E}$. Then, as we have already observed, $\{(\mathcal{C}, M)^1\}$ and $\{(\mathcal{C}, N)^1\}$ are ε-related. By Lemma 1, we can deduce that $\mathscr{D}$ and $\mathscr{E}$ are ε-related. And it is easy to see that this implies $|\sum \mathscr{D} - \sum \mathscr{E}| \leq \varepsilon$.

C. Adding Pairs to the Calculus

The trace distance and the results we have just presented about it can be extended to an affine λ-calculus *with pairs*, namely a calculus whose language of terms also includes the following two constructs:

$$M ::= \langle M, N\rangle \;\Big|\; \texttt{let}\ \langle x, y\rangle = M\ \texttt{in}\ N.$$

We assume that terms are typed in any linear type system guaranteeing the absence of deadlocks (e.g., simple or recursive types), and we generalize the operational semantics in a natural way [5]. We would now like to extend the definition of a trace accordingly: which action should we perform on a term in the form $\langle M, N\rangle$? The naïve solution would be to add projections to the trace language: $s ::= \pi_1 \cdot s \;\Big|\; \pi_2 \cdot s$, with trace interpretation extended in the expected way. However, this way the trace distance would not be sound for the context distance, anymore. Indeed, let us consider the following example:

Example 3: We are going to compare the following terms:

$$M := \langle \lambda z.(I \oplus \Omega), \lambda z.(I \oplus \Omega)\rangle; \qquad N := \langle \lambda z.I, \lambda z.I\rangle.$$

These two terms are at context distance at least $\frac{3}{4}$, since we can consider the context $\mathcal{C} := \texttt{let}\ \langle x, y\rangle = [\cdot]\ \texttt{in}\ (xI)(yI)$, and we can see that $\sum [\![\mathcal{C}[M]]\!] = \frac{1}{4}$, while $\sum [\![\mathcal{C}[N]]\!] = 1$. But we cannot find any trace that separates them more than $\frac{1}{2}$. The interesting case is when $s = \pi_i \cdot t$. But then:

$$\begin{aligned} |Pr(M, s) - Pr(N, s)| &= |Pr(\lambda z.(\Omega \oplus I), t) - Pr(\lambda z.I, t)| \\ &\leq \delta^{\mathrm{tr}}(\lambda z.(\Omega \oplus I), \lambda z.I). \end{aligned}$$

And it is easy to see that in the calculus with pairs we still have $\delta^{\mathrm{tr}}(\lambda z.(\Omega \oplus I), \lambda z.I) = \frac{1}{2}$.

The reason why we cannot recover the context distance by way of projections is that the `let` construct above allows us to access *both* components of a pair, and the distances each of them induce can *add up*. A way out consists in extending the trace language to pairs really following linearity, and considering a new action in the form $\otimes L$ with the following trace interpretation:

$$Pr(\langle M, N\rangle, \otimes L \cdot t) = \sum_{V,W} [\![M]\!](V) \cdot [\![N]\!](W) \cdot Pr(L\{V, W/x, y\}, t).$$

Remarkably, this is consistent with the well-known embedding of pairs into the applicative fragment (see, e.g., [2], and of course [5]).

This way of handling pairs allows the trace distance and the context distance to coincide, again. However, the trace distance loses its grip with respect to the context distance. Consider, for instance, the terms M and N from Example 3. Showing an upper bound on the distance between M and N is the same thing as showing an upper bound on $\delta^{\mathrm{tr}}(L\{\lambda z.(\Omega \oplus I), \lambda z.(\Omega \oplus I)/x, y\}, L\{\lambda z.I, \lambda z.I/x, y\})$ *for all terms* L such that $x, y \vdash L$, which is in fact not far away from what we should show if we were considering the context distance directly.

V. The Bisimulation Distance

As we realised in the last section, the trace metric can be a way to alleviate the burden of evaluating the context distance between terms but, in particular in presence of pairs, its usefulness is limited. In this section, we will look at another way to define the distance between programs which is genuinely coinductive, being based on the Kantorovich metric for distributions.

A. Definition

A labelled Markov chain (LMC) is a triple $\mathscr{M} = (S, \mathscr{L}, \mathscr{P})$, where S is a countable set of states, $\mathscr{L}$ is a

countable set of labels, and $\mathscr{P}$ is a transition probability matrix, that is a function: $\mathscr{P} : S \times \mathscr{L} \to \text{Distr}(S)$. Moreover, if the image of $\mathscr{P}$ only consists of distributions with *finite* support, we call $\mathscr{M}$ an *image-finite* LMC. We are now going to define, similarly to [10] (but in absence of non-determinism), the metric analogue to bisimulation. The idea is to define a metric on the set S of states of the LMC as the greatest fixed point of some monotone operator on metrics. Please recall that $(\Delta(S), \leq^{\text{metr}})$ is a complete lattice, and so any monotone operator has indeed a greatest fixed point.

Lifting Metrics to Distributions: We are going to define a way to turn any premetric over a set S into a metric over finite distribution over S.

Definition 8: Let μ be a premetric on a set S. We define the *lifting of* μ as the metric on the set of finite distributions over S defined as follows: for every $\mathscr{D}$, $\mathscr{E}$ finite distributions over S, $\mu(\mathscr{D}, \mathscr{E})$ is the optimum solution to the following linear program:

$$\min \sum_{(s,t)\in \mathsf{S}(\mathscr{D})\times\mathsf{S}(\mathscr{E})} h_{s,t}\cdot\mu(s,t) + \sum_s w_s + \sum_t z_t$$

$$\begin{aligned}\text{subject to}\quad & \textstyle\sum_{s\in\mathsf{S}(\mathscr{D})} h_{s,t} + z_t = \mathscr{E}(t);\\ & \textstyle\sum_{t\in\mathsf{S}(\mathscr{E})} h_{s,t} + w_s = \mathscr{D}(s);\\ & \forall (s,t)\in \mathsf{S}(\mathscr{D})\times\mathsf{S}(\mathscr{E}),\ h_{s,t},\ z_t,\ w_s \geq 0.\end{aligned}$$

Please observe that this linear program has an optimal solution. We can make use of the notion of duality from linear programming, and obtain an alternative characterisation of lifting:

Theorem 3: Let μ be a premetric on S and Let $\mathscr{D}$, $\mathscr{E}$ be finite distributions over S. Then:

$$\mu(\mathscr{D},\mathscr{E}) = \max \sum_s a_s\cdot\mathscr{D}(s) + b_s\cdot\mathscr{E}(s)$$

$$\begin{aligned}\text{subject to}\quad & \forall s\in S,\ a_s \leq 1;\\ & \forall s\in S,\ b_s \leq 1;\\ & \forall s,\ t\in S,\ a_s + b_t \leq \mu(s,t).\end{aligned}$$

The interest of the lifting construction comes from the fact that the lifting of a metric μ behaves coherently with the original metric μ. In particular, if we know the lifting of μ, we are able to recover μ by considering Dirac distributions:

Lemma 2: Let μ be a premetric on S, and $s, t \in S$. Then $\mu(\{s^1\},\{t^1\}) = \mu(s,t)$.

Moreover, the metric's structure is preserved when we consider the lifting: indeed, if a premetric on states is symmetric and verifies the triangular inequality, the same holds for its lifting.

Metrics as Fixpoints: In a non-probabilistic setting, a relation R is a bisimulation if every pair of states s, t such that $s\,R\,t$ can do the same actions and end up into states which are themselves in the relation. In order to obtain a quantitative counterpart of the scheme above, we define an operator F on the set of metrics over the states of a LMC: intuitively, given a metric μ, we define a new metric $F(\mu)$ which corresponds to the distance obtained by first doing a step of the transition relation, and then applying the lifting of μ to the resulting distributions.

Definition 9: Let $\mathscr{M} = (S, \mathscr{L}, \mathscr{P})$ be an image-finite LMC. We define an operator F on $\Delta(S)$ as

$$F(\mu)(s,t) = \sup\{\mu(\mathscr{P}(s,a),\mathscr{P}(t,a)) \mid a\in\mathscr{L}\}.$$

Please observe that F is a monotone operator on the complete lattice of metrics. We define the *the bisimulation metric* $\delta^{\mathsf{b}}_{\mathscr{M}}$ on $\mathscr{M}$ as the greatest fixpoint of F.

Bisimulation Metric and the Affine λ-Calculus: We are now going to consider a specific LMC $\mathscr{M}^{\Lambda}$, which captures the interactive behaviour of our calculus.

Definition 10: We define the LMC $\mathscr{M}^{\Lambda} = (S^{\Lambda}, \mathscr{L}^{\Lambda}, \mathscr{P}^{\Lambda})$ where:

- The set of states S^{Λ} is defined as follows: $\mathbf{P}\uplus\mathbf{V}$. A value V in the second component of S^{Λ} is distinguished from one in the first by using the notation $\hat{V}$.
- The set of labels $\mathscr{L}^{\Lambda}$ is taken to be
$$\mathscr{L}^{\Lambda} = \{@V \mid V\in\mathbf{V}\}\bigcup\{\text{eval}\}.$$
- The transition probability matrix $\mathscr{P}^{\Lambda}$ is such that: for every $M\in\mathbf{P}$, and any value $V\in\mathsf{S}([\![M]\!])$, it holds that $\mathscr{P}^{\Lambda}(M,\text{eval})(\widehat{V}) = [\![M]\!](V)$, and that for every term M such that $\lambda x.M\in\mathbf{P}$, and $V\in\mathbf{V}$, it holds that $\mathscr{P}^{\Lambda}(\widehat{\lambda x.M}, @V)(M\{V/x\}) = 1$.

The results we have proved previously in this section apply to $\mathscr{M}^{\Lambda}$. In particular, one can define the bisimulation metric on $\mathscr{M}^{\Lambda}$. The *bisimulation distance* on programs, which we indicate as δ^{b}, is defined to be the restriction of $\delta^{\mathsf{b}}_{\mathscr{M}^{\Lambda}}$ to programs.

We can see easily that δ^{b} is an adequate metric. But there is more, since the bisimulation metric is well-known to be a lower bound on the trace distance: the bisimulation distance is a *sound* metric. In the next section, we anyway show non-expansiveness for it, which is stronger.

B. Non-Expansiveness

Proving the non-expansiveness of δ^{b} cannot be done directly, by a plain induction on contexts. Our strategy towards the result is the Howe's technique [16], a way of proving congruence of coinductively-defined equivalences which has been widely used for deterministic and non-deterministic languages, and that we here adapt to metrics.

The idea, then, is to start from δ^{b}, construct another metric ${\delta^{\mathsf{b}}}^{H}$ on top of δ^{b} (which turns out to be non-expansive by construction), and show that ${\delta^{\mathsf{b}}}^{H} = \delta^{\mathsf{b}}$. We first need to transform our metric δ^{b} on programs into a metric on (potentially open) terms. Any metric μ on programs can be extended into a metric on open terms, which by abuse of notation we continue to call μ and which is defined as follows

$$\mu(M,N) = \sup_{V_1,\ldots,V_n\in\mathbf{V}} \mu(M\{V_1,\ldots,V_n/x_1,\ldots,x_n\},\\ N\{V_1,\ldots,V_n/x_1,\ldots,x_n\}),$$

where $x_1, \ldots, x_n$ are the variables occurring free in either M or N. The Howe's lifting of a metric μ is defined based on a formal system for judgements in the form $\Gamma \vdash \mu^H(M,N) \leq \varepsilon$, where ε is a real number between 0 and 1. Its rules are given in Figure 4. Please observe that, potentially, there are several different ε such that $\Gamma \vdash \mu^H(M,N) \leq \varepsilon$. We are finally in a

$$\frac{\mu(x,M) \leq \varepsilon \qquad x, \Gamma \vdash M}{x, \Gamma \vdash \mu^H(x,M) \leq \varepsilon}$$

$$\frac{\Gamma \vdash \mu^H(M,K) \leq \varepsilon \quad \Gamma \vdash \mu^H(N,T) \leq \iota \qquad \mu(K \oplus T, L) \leq \gamma \quad \Gamma \vdash L}{\Gamma \vdash \mu^H(M \oplus N, L) \leq \frac{\varepsilon+\iota}{2} + \gamma}$$

$$\frac{x, \Gamma \vdash \mu^H(M,K) \leq \varepsilon \qquad \mu(\lambda x.K, L) \leq \iota \quad \Gamma \vdash L}{\Gamma \vdash \mu^H(\lambda x.M, L) \leq \varepsilon + \iota}$$

$$\frac{\Gamma \vdash \mu^H(M,K) \leq \varepsilon \quad \Delta \vdash \mu^H(N,T) \leq \iota \qquad \mu(KT, L) \leq \gamma \quad \Gamma, \Delta \vdash L}{\Gamma, \Delta \vdash \mu^H(MN, L) \leq \varepsilon + \iota + \gamma}$$

Fig. 4. Howe's Rules.

position to define the Howe's lifting of μ:

Definition 11: Let μ be a metric on terms. We define a premetric μ^H on terms by:

$$\mu^H(M,N) = \inf\left(\{\varepsilon \mid \exists \Gamma, \Gamma \vdash \mu^H(M,N) \leq \varepsilon\} \bigcup \{1\}\right).$$

We can see that $\delta^{\mathfrak{b}H}$ is a premetric on open terms. Please observe that it is not necessarily a metric, since its construction entails neither symmetry nor the triangular inequality. The interest of this construction is that the metric $\delta^{\mathfrak{b}H}$ is (more or less by construction) non-expansive:

Lemma 3 (Non-expansiveness of $\delta^{\mathfrak{b}H}$): For every context $\mathcal{C}$ and for every terms M, N it holds that $\delta^{\mathfrak{b}H}(\mathcal{C}[M], \mathcal{C}[N]) \leq \delta^{\mathfrak{b}H}(M,N)$.

The goal now is to show that $\delta^{\mathfrak{b}H} \leq^{\text{metr}} \delta^{\mathfrak{b}}$. Since $\delta^{\mathfrak{b}}$ is the greatest fixed point of F for our LMC $\mathscr{M}^\Lambda$, we are going to show that $\delta^{\mathfrak{b}H}$ can be extended into a metric on the states of $\mathscr{M}^\Lambda$, obtaining a fixed point for the operator F. First we extend $\delta^{\mathfrak{b}H}$ to a premetric on S^Λ:

Definition 12: We define the extension of $\delta^{\mathfrak{b}H}$ to S^Λ (that we note still $\delta^{\mathfrak{b}H}$ by abuse of notation), by:

$$\begin{aligned}
\delta^{\mathfrak{b}H}(M,N) &:= \delta^{\mathfrak{b}H}(M,N);\\
\delta^{\mathfrak{b}H}(\widehat{V},\widehat{W}) &:= \delta^{\mathfrak{b}H}(V,W);\\
\delta^{\mathfrak{b}H}(M,\widehat{W}) &:= 1.
\end{aligned}$$

Since $\delta^{\mathfrak{b}H}$ isn't guaranteed to be a metric, we are forced to further refine it, by adding rules corresponding to symmetry and to the triangular inequality: we define $\delta^{\mathfrak{b}H}_\triangle$ over S^Λ by the rules of Figure 5.

We can see easily that $\delta^{\mathfrak{b}} \leq^{\text{metr}} \delta^{\mathfrak{b}H} \leq^{\text{metr}} \delta^{\mathfrak{b}H}_\triangle$ with respect to the preorder on terms. We want to show that $\delta^{\mathfrak{b}H} = \delta^{\mathfrak{b}}$. In

$$\frac{\delta^{\mathfrak{b}H}(s,t) \leq \varepsilon}{\vdash \delta^{\mathfrak{b}H}_\triangle(s,t) \leq \varepsilon}$$

$$\frac{\vdash \delta^{\mathfrak{b}H}_\triangle(s,t) \leq \varepsilon \qquad \vdash \delta^{\mathfrak{b}H}_\triangle(t,s) \leq \iota}{\vdash \delta^{\mathfrak{b}H}_\triangle(s,t) \leq \min(\varepsilon, \iota)}$$

$$\frac{\vdash \delta^{\mathfrak{b}H}_\triangle(s,t) \leq \varepsilon \qquad \vdash \delta^{\mathfrak{b}H}_\triangle(t,u) \leq \iota}{\vdash \delta^{\mathfrak{b}H}_\triangle(s,u) \leq \varepsilon + \iota}$$

Fig. 5. Symmetric and Transitive Closure of $\delta^{\mathfrak{b}H}$.

order to have that, we will show that $\delta^{\mathfrak{b}H}_\triangle \leq^{\text{metr}} \delta^{\mathfrak{b}}$. That is a direct consequence of the following theorem:

Theorem 4: $\delta^{\mathfrak{b}H}_\triangle$ is a pre-fixpoint of F.

Proof. We need to show that $\delta^{\mathfrak{b}H}_\triangle \leq^{\text{metr}} F(\delta^{\mathfrak{b}H}_\triangle)$. Please remember that the preorder on metrics corresponds to the reverse of the point-wise preorder for states. So if we read this inequality on metrics as an inequality on the states of $\mathscr{M}^\Lambda$, we see that it is equivalent to: for every $s,t \in S^\Lambda$, $F(\delta^{\mathfrak{b}H}_\triangle)(s,t) \leq \delta^{\mathfrak{b}H}_\triangle(s,t)$. If we unfold the definition of the operator F on metrics, we can see that it means that for every $a \in \mathscr{L}^\Lambda$, $\delta^{\mathfrak{b}H}_\triangle(\mathscr{P}^\Lambda(s,a), \mathscr{P}^\Lambda(t,a)) \leq \delta^{\mathfrak{b}H}_\triangle(s,t)$. Please remember that there are two kinds of actions in our LMC: the action eval of evaluating a program to obtain a value distribution, and the action $@V$, which corresponds to passing the value V to a distinguished value. If we consider separately each of these actions, we see that the result we want to have is equivalent to:

- Let M, N be closed terms. Then $\delta^{\mathfrak{b}H}_\triangle(\widehat{[\![M]\!]}, \widehat{[\![N]\!]}) \leq \delta^{\mathfrak{b}H}_\triangle(M,N)$;
- Let M, N be such that $x \vdash M$ and $x \vdash N$, and let V be a value. Then it holds that: $\delta^{\mathfrak{b}H}_\triangle(M\{V/x\}, N\{V/x\}) \leq \delta^{\mathfrak{b}H}_\triangle(\widehat{\lambda x.M}, \widehat{\lambda x.N})$.

The first point is the so-called Key Lemma, while the second one is a form of substitutivity. Both are non-trivial to prove (see [5] for more details). Remarkably, the Key Lemma can be proved exploiting the duality between the two presentations of Kantorovich lifting (see Theorem 3). □

Since $\delta^{\mathfrak{b}H}$ is non-expansive by construction, we now have the result we were aiming for:

Theorem 5: $\delta^{\mathfrak{b}}$ is non-expansive.

C. On Full-Abstraction and Pairs

The bisimulation distance is a sound approximation of the context distance. But how about full-abstraction? Is there any hope to prove that the two coincide? The answer is negative: there are terms whose distance is *strictly* higher in the bisimulation metric than in the context (or trace) metric.

Example 4: Consider the following terms: M corresponds to the program that takes an argument, and then returns I with probability $\frac{1}{2}$, and diverges with probability $\frac{1}{2}$. N corresponds to the program which chooses first between the function which

return I whenever it is called, and the function which diverges whenever called. Formally:

$$M := \lambda x.(I \oplus \Omega); \qquad N := (\lambda x.I) \oplus (\lambda x.\Omega).$$

These two terms are at distance 0 for the context distance: since the calculus is linear, the step where the choice is done is irrelevant. However, $\delta^{\mathrm{b}}(M,N) = \frac{1}{2}$: the proof, whose details can be found in [5] use the characterisation of bisimulation distance by testing from [9], in which not only linear tests, but also more complicated tests (like *threshold tests*) are available.

But how about pairs? Indeed, for the sake of simplicity, we have presented the metatheory of the bisimulation metric for a purely applicative λ-calculus. Following the lines of our discussion in Section IV-C, however, the LMC $\mathscr{M}^{\Lambda}$ can be extended into one handling pairs in a relatively simple way. The difficulties we encountered when trying to evaluate the (trace, or context) distance between pairs of terms unfortunately remain: it is not clear whether coinduction could provide any additional *advantage* over contextual distance.

VI. The Tuple Distance

The two metrics we have just defined have been shown to be non-expansive, even if the calculus is extended with pairs. In that case, however, they do not represent so much of an improvement with respect to the context distance. Please recall *where* the problem comes from: we would like to define actions starting from $\langle M, N\rangle$, and respecting the affine paradigm. We have seen that taking *projections* as actions lead to an unsound metric, and we have circumvented the problem by considering an action $\otimes L$, following [8]. Intuitively the action $\otimes L$ corresponds to replacing the free variables of L (which are supposed to be included in $\{x, y\}$) by the components of the pair: if for instance V and W are values, we have that $\langle V, W\rangle \overset{\otimes L}{\rightsquigarrow} \{L\{V, W/x, y\}^1\}$. But what can any environment L do if we give it V and W as two values to interact with? Let us suppose that both V and W are functions, and remember that we are in an affine setting. The environment can (probabilistically) pass some arguments to V, and independently some other arguments to W, and then possibly pass to one of the two programs an argument that contains the other one. The idea behind the construction we present in this section, then, is to *keep* the information about the two components of the pairs in the states until they really interact with each other.

Our idea can be made concrete by introducing another LMC, whose states are not closed terms anymore, but *tuples* in the form $[V_1, \ldots, V_n]$, where $V_1, \ldots, V_n$ are values. The possible actions the environment can perform on a tuple $[V_1, \ldots, V_n]$ correspond to the choice of an index $i \in \{1, \ldots, n\}$ and of an action to apply to the value V_i. If V_i is a pair, the only possible action is to split it into two components. We call this action unfold^i. If V_i is a function, the environment can pass it an argument, which can possibly be constructed using other V_j's. More precisely, the argument is built by way of an open term $\mathcal{C}$, and a typing context Γ, such that $\Gamma \vdash \mathcal{C}$, and Γ is a subset of $\{x_j \mid j \neq i\}$: the free variables of Γ represent the places where other values V_j, with $j \neq i$, are used. Moreover, we ask that for any values $W_1, \ldots, W_n$, the term obtained in substituting x_j by W_j is a value: it means that $\mathcal{C}$ is one of the x_j, or of the form $\lambda y.\mathcal{D}$. We call a pair $(\Gamma, \mathcal{C})$ which verifies these conditions a (n,i)-*open value*. Formally, the LMC $\mathscr{M}^{\Lambda}_{\mathrm{mul}} = (S^{\Lambda}_{\mathrm{mul}}, \mathscr{A}^{\Lambda}_{\mathrm{mul}}, \mathscr{P}^{\Lambda}_{\mathrm{mul}})$ is defined in Figure 6.

A. The Metric

We are going to define a metric on closed terms which corresponds to linear tests in $\mathscr{M}^{\Lambda}_{\mathrm{mul}}$. First, we define *tuple traces* simply as words over $\mathscr{A}^{\Lambda}_{\mathrm{mul}}$. The probability to succeed in doing a trace s starting from a tuple $K \in S^{\Lambda}_{\mathrm{mul}}$ can be naturally defined, and paves the way to defining a metric on tuples of values:

$$Pr^{\mathrm{mul}}(K, \varepsilon) = 1;$$
$$Pr^{\mathrm{mul}}(K, a \cdot s) = \sum_H \mathscr{P}(K, a)(H) \cdot Pr^{\mathrm{mul}}(H, s);$$
$$\delta^{\mathrm{mul}}(K, H) = \sup_s |Pr^{\mathrm{mul}}(K, s) - Pr^{\mathrm{mul}}(H, s)|.$$

What we need, however, is a metric on *programs*. Please remember that states of the LMC $\mathscr{M}^{\Lambda}_{\mathrm{mul}}$ are tuples of *values*. Any program M, however, can be viewed as the distribution of values obtained by evaluating it, i.e. its semantics $[\![M]\!]$:

$$\delta^{\mathrm{mul}}(M,N) = \sup_s \Big|\sum [\![M]\!](V) \cdot Pr^{\mathrm{mul}}([V], s) - \sum [\![N]\!](W) \cdot Pr^{\mathrm{mul}}([W], s)\Big|.$$

The just introduced metric should at least be put in relation to the context metric for it to be useful. We know from Section IV that the context metric coincides with the trace metric. The following theorem relates the trace metric δ^{tr} and the metric δ^{mul}:

Theorem 6: Let I be any finite set of variables, and $\{V_x\}_{x\in I}$ and $\{W_x\}_{x\in I}$ any two collections of values. For any open term $\mathcal{C}$ such that $I \vdash \mathcal{C}$, it holds that:

$$\delta^{\mathrm{tr}}(\mathcal{C}\{V_x/x\}_{(x\in I)}, \mathcal{C}\{W_x/x\}_{(x\in I)}) \leq \delta^{\mathrm{mul}}([V_x]_{(x\in I)}, [W_x]_{(x\in I)}).$$

Theorem 6 can be read as a non-expansiveness result: if we have a system $\mathcal{E}$, playing the role of the environment, and which is prepared to interact with n components, and moreover we have two tuples K and H of length n, then the tuple distance between K and H gives us an upper bound on the trace distance between the system composed of $\mathcal{E}$ interacting with K, and the system composed of $\mathcal{E}$ interacting with H.

We can now see that δ^{mul} coincides with the context metric: one inequality comes from Theorem 6, the other comes from the fact that any trace s over $\mathscr{A}^{\Lambda}_{\mathrm{mul}}$ and designed to start from a single value, can be simulated by a context.

Theorem 7: On programs, $\delta^{\mathrm{mul}} = \delta^{\mathrm{ctx}}$.

$$S^{\Lambda}_{\text{mul}} = \{[V_1, \ldots, V_n] \mid V_1, \ldots, V_n \text{ closed values } \};$$

$$\mathscr{A}^{\Lambda}_{\text{mul}} = \left\{\text{unfold}^i \mid i \in \mathbb{N}\right\} \cup \left\{ @(\Gamma, \mathcal{C})^i \mid i \in \mathbb{N}, (\Gamma, \mathcal{C}) \text{ a } (n,i)\text{-open-value } \right\};$$

$$\mathscr{P}^{\Lambda}_{\text{mul}}([s_1, \ldots, \langle N, L\rangle, \ldots, s_n], \text{unfold}^i)([s_1, \ldots, s_{i-1}, V, W, s_{i+1}, \ldots, s_n]) = [\![N]\!](V) \cdot [\![L]\!](W);$$

$$\mathscr{P}^{\Lambda}_{\text{mul}}([s_1, \ldots, \lambda y.N, \ldots, s_n], @(\Gamma, \mathcal{C})^i)([s_{h_1}, \ldots, W, \ldots, s_{h_m}]) = [\![N\{\left(\mathcal{C}\{s_{j_l}/x_{j_l}\}_{(1\leq l\leq k)}\right)/y\}]\!](W);$$

$$\text{with } \{1, \ldots, n\} = \{i\} \cup \{j_1, \ldots, j_k\} \cup \{h_1, \ldots, h_m\},\ n = 1 + k + m, \text{ and } \Gamma = x_{j_1}, \ldots, x_{j_k}.$$

Fig. 6. The Tuple LMC.

B. Examples

The tuple distance, that we have just proved to be fully-abstract, can be seen as yet another presentation of the context distance. But there is much more: it allows to evaluate the distance between concrete programs, even when the latter contain pairs, in a relatively easy way. In this section, we will give two examples.

1) A Simple Example: Consider the terms M and N defined in Example 3. We can prove that $\delta^{\text{mul}}(M, N) = \frac{3}{4}$. The proof of the fact that $\delta^{\text{mul}}(M, N) \leq \frac{3}{4}$ can be found in [5]. Here we are only going to show that $\delta^{\text{mul}}(M, N) \geq \frac{3}{4}$. In order to show that, we are going to present a particular trace s such that $|Pr^{\text{mul}}([M], s) - Pr^{\text{mul}}([N], s)| = \frac{3}{4}$. More precisely, we take $s = \text{unfold}^1 \cdot @(\emptyset, I)^1 \cdot @(\emptyset, I)^2$: it corresponds to first separating the two components of the pair, and then passing I as an argument to the first and to the second component. The relevant fragment of $\mathscr{M}^{\Lambda}_{\text{mul}}$ can be found in Figure 7. In particular, we can see that $Pr([M], s) = 1$, and $Pr([N], s) = \frac{1}{4}$.

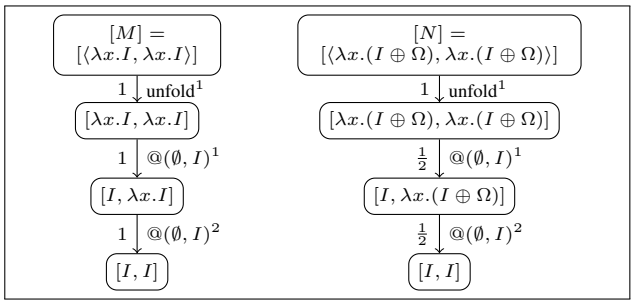

Fig. 7. A Fragment of the Tuple LMC.

2) A More Complicated Example: Please recall the example we presented in Section II. We note $\{u_n\}_{n\in\mathbb{N}}$ the sequence defined as: $u_n = \prod_{1\leq i\leq n}(1 - \frac{1}{2^i})$.

Theorem 8: For every $n \in \mathbb{N}$, $\delta^{\text{mul}}(M_n, N_n) = 1 - u_n$.

Proof. We first show that $\delta^{\text{mul}}(M_n, N_n) \geq 1 - u_n$. As in the previous example, we do that by finding, for each $n \in \mathbb{N}$, a trace s_n such that $|Pr([M_n], s_n) - Pr([N_n], s_n)| = 1 - u_n$. We define the sequence $(s_n)_{n\in\mathbb{N}}$ inductively as follows:

$$s_0 = \epsilon; \qquad s_{n+1} = \text{unfold}^1 \cdot @(\emptyset, I)^1 \cdot s_n.$$

s_0 is the trace which always succeeds, whatever the starting state is. s_{n+1} corresponds to separating the two components of the pair which is in first position in the tuple, then passing the identity as an argument to the first component of this pair, and then executing s_n. For this sequence of traces, the recursive equations of Figure 8 are verified (the proof can be found in [5]). We can see by solving these equations that for every

$$Pr([M_0], s_0) = 1; \qquad Pr([N_0], s_0) = 1;$$

$$Pr([M_{n+1}], s_{n+1}) = Pr([M_n], s_n);$$

$$Pr([N_{n+1}], s_{n+1}) = (1 - \frac{1}{2^{n+1}}) \cdot Pr([N_n], s_n).$$

Fig. 8. Recursive Equations Verified by s_n.

$n \in \mathbb{N}$, $Pr(M_n, s_n) = 1$ and $Pr(N_n, s_n) = u_n$. As a direct consequence, we obtain the result. We want now to show that $\delta^{\text{mul}}(M_n, N_n) \leq 1 - u_n$. To do that, we need to establish that there doesn't exist a trace t such that $|Pr([M_n], t) - Pr([N_n], t)| > 1 - u_n$. We're in fact going to show something stronger: for every $n \in \mathbb{N}$, we're going to define a set A_n of pairs of tuple, which contains the pair $([M]_n, [N_n])$, and such that for every $(K, H) \in A_n$, for every trace t, $|Pr(K, t) - Pr(H, t)| \leq 1 - u_n$. Intuitively, the idea behind the sequence $\{A_n\}_{n\in\mathbb{N}}$ is the following: if we start from $[M_n]$, follow a trace of even length, and end up in a tuple K with a non-zero probability, and when we follow *the same trace* starting from $[N_n]$ end up in the tuple H, then the pair of tuples (K, H) is in one of the A_j, with j smaller than n.

Definition 13: Let be $n \in \mathbb{N}$. Let A_n be the set of (K, H) such that: there exist $m \in \mathbb{N}$, and $k_i \geq n+1$ (for $1 \leq i \leq m$), where:

$$K = [M_n, [\lambda x.\Omega]^m];$$

$$H = \left[N_n, [\lambda x.\Omega \oplus_{\frac{1}{2^{k_i}}} I]_{1\leq i\leq m}\right].$$

We want now to give an upper bound to the separation between K and H any trace can induce, if $(K, H) \in A_n$.

Lemma 4: For every $n \in \mathbb{N}$, for every $(K, H) \in A_n$, we can partition the set of traces as:

$$\mathcal{T}r = \{s \mid Pr(K, s) = 0 \text{ and } Pr(H, s) \leq \frac{1}{2}\} \bigcup \{s \mid Pr(K, s) = 1 \text{ and } Pr(H, s) \geq u_n\}.$$

The proof of Lemma 4 can be found in [5]. The result we want to show is a direct consequence of Lemma 4: we can see easily that for any trace s, if $(K, H) \in A_n$, the separation s can induce is smaller or equal to $1 - u_n$. □

C. On Tuples and Copying

The tuple distance naturally suggests a way to handle λ-calculi in which copying is indeed allowed. Although the details are clearly outside the scope of this paper, we anyway want to give some hints about why this is the case.

What makes the trace and behavioural distances unsound in presence of copying is their inability to capture an environment which can access the program at hand *more than once*. In our view, however, the problem does not come from the way those distances are defined in the abstract, but rather in the way *the underlying LMC* reflects the operational semantics of the calculus at hand. In a sense, it is in the responsibility of the LMC to guarantee that the environment can access terms multiple times. The LMC $\mathscr{M}^{\Lambda}$ we introduced in this paper (which is close to the ones from the literature [4], [6], [8]), as an example, is not adequate.

Suppose, however, to extend $\mathscr{M}^{\Lambda}_{\mathsf{mul}}$ to an LMC for a λ-calculus in the style of Wadler's linear λ-calculus [29]: there, the grammar of terms includes a construct $!M$ whose purpose is marking those subterms which can indeed be duplicated. The actions the environment can perform on a term in the form $!M$ simply reflects the above: the environment can create a *new copy* of $!M$, *but also keeps* the possibility to access $!M$ in the future. One immediately realises that tuples are indeed the right way to model the access to both $!M$ and M.

VII. Conclusions

We have initiated the study of metrics in higher-order languages, starting with the relatively easy case of affine λ-terms, where copying capabilities are simply not available. We showed that three different notions of distance are sound (and sometime fully-abstract) for the context distance, the natural generalisation of Morris' observational equivalence. One of them, the tuple distance, reflects the inherently monoidal structure of the underlying calculus, this way allowing to solve some nontrivial distance problems.

We are actively working on extending the results described here to the non-affine case, which for various reasons turns out to be more difficult, as discussed in Section II. We are in particular quite optimistic about the possibility of generalising the tuple distance to a metric reflecting copying. The real challenge, however, consists in handling the case in which copying is indeed available, but the number of copies of a given term the environment can have access to is somehow bounded, maybe polynomially in the value of a security parameter. That would indeed be a way to get closer to computational indistinguishability, a central notion in modern cryptography.

References

[1] S. Abramsky. The Lazy λ-Calculus. In D. Turner, editor, *Research Topics in Functional Programming*, pages 65–117. Addison Wesley, 1990.

[2] A. Asperti and L. Roversi. Intuitionistic light affine logic. *ACM Trans. Comput. Log.*, 3(1):137–175, 2002.

[3] A. Bizjak and L. Birkedal. Step-indexed logical relations for probability. In *FoSSaCS*, pages 279–294, 2015.

[4] R. Crubillé and U. Dal Lago. On probabilistic applicative bisimulation and call-by-value λ-calculi. In *ESOP*, pages 209–228, 2014.

[5] R. Crubillé and U. Dal Lago. Metric reasoning about λ-terms: the affine case (long version). Available at http://arxiv.org/abs/1505.03638, 2015.

[6] U. Dal Lago, D. Sangiorgi, and M. Alberti. On coinductive equivalences for higher-order probabilistic functional programs. In *POPL*, pages 297–308, 2014.

[7] V. Danos and T. Ehrhard. Probabilistic coherence spaces as a model of higher-order probabilistic computation. *Inf. Comput.*, 209(6):966–991, 2011.

[8] Y. Deng and Y. Zhang. Program equivalence in linear contexts. To appear in Theoretical Computer Science, 2014.

[9] J. Desharnais, V. Gupta, R. Jagadeesan, and P. Panangaden. Metrics for labeled markov systems. In *CONCUR*, 1999.

[10] J. Desharnais, R. Jagadeesan, V. Gupta, and P. Panangaden. The metric analogue of weak bisimulation for probabilistic processes. In *LICS*, pages 413–422, 2002.

[11] T. Ehrhard, C. Tasson, and M. Pagani. Probabilistic coherence spaces are fully abstract for probabilistic PCF. In *POPL*, pages 309–320, 2014.

[12] D. Gebler and S. Tini. Fixed-point characterization of compositionality properties of probabilistic processes combinators. In *EXPRESS-SOS*, pages 63–78, 2014.

[13] O. Goldreich. *Modern Cryptography, Probabilistic Proofs and Pseudorandomness*, volume 17 of *Algorithms and Combinatorics*. Springer, 1998.

[14] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984.

[15] N. D. Goodman, V. K. Mansinghka, D. M. Roy, K. Bonawitz, and J. B. Tenenbaum. Church: a language for generative models. In *UAI 2008*, pages 220–229, 2008.

[16] D. J. Howe. Proving congruence of bisimulation in functional programming languages. *Inf. Comput.*, 124(2):103–112, 1996.

[17] C. Jones and G. D. Plotkin. A probabilistic powerdomain of evaluations. In *LICS*, pages 186–195, 1989.

[18] L. V. Kantorovich. On the transfer of masses. In *Dokl. Akad. Nauk. SSSR*, volume 37, pages 227–229, 1942.

[19] S. B. Lassen. Relational reasoning about contexts. In *Higher Order Operational Techniques in Semantics, Publications of the Newton Institute*, pages 91–135. Cambridge University Press, 1998.

[20] H. G. Mairson. Linear lambda calculus and ptime-completeness. *J. Funct. Program.*, 14(6):623–633, 2004.

[21] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*, volume 999. MIT Press, 1999.

[22] J. C. Mitchell, A. Ramanathan, A. Scedrov, and V. Teague. A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. *Theor. Comput. Sci.*, 353(1-3):118–164, 2006.

[23] S. Park, F. Pfenning, and S. Thrun. A probabilistic language based on sampling functions. *ACM Trans. Program. Lang. Syst.*, 31(1), 2008.

[24] J. Pearl. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, 1988.

[25] A. M. Pitts. Operationally-based theories of program equivalence. In *Semantics and Logics of Computation*, pages 241–298. Cambridge University Press, 1997.

[26] N. Ramsey and A. Pfeffer. Stochastic lambda calculus and monads of probability distributions. In *POPL*, pages 154–165, 2002.

[27] S. Thrun. Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, pages 1–35, 2002.

[28] F. van Breugel and J. Worrell. A behavioural pseudometric for probabilistic transition systems. *Theor. Comput. Sci.*, 331(1):115–142, 2005.

[29] P. Wadler. A syntax for linear logic. In *MFPS*, pages 513–529, 1993.