Amplification Rate of Contextual Distances in Randomised Programming

Houssein Mansour*[1] and Raphaëlle Crubillé†[2]

[1]Università di Bologna
[2]CNRS, Aix-Marseille Université

A central question in the study of higher-order programming languages is that of *program equivalence*: when can two syntactically different programs be considered as equivalent? For instance, when applying a program transformation for optimisation purposes, it is important to ensure that the transformed program remains, in some sense, equivalent to the original. A widely accepted definition is *contextual equivalence*, introduced by Morris in 1969 [Jr.69]: two programs are deemed equivalent when they behave identically in all possible contexts. This definition is made self-contained by modelling contexts as programs written in the same language:

$$M \sim N \quad \text{if and only if} \quad \forall C \text{ context},\ \mathrm{Obs}(C[M]) = \mathrm{Obs}(C[N]),$$

where $\mathrm{Obs}(\cdot)$ refers to a notion of *observable behaviour*: for pure lambda-calculus, for instance, a natural choice of observation is whether a program terminates. In this work, we are interested in *probabilistic* higher-order programming languages, where the notion of observation becomes intrinsically *quantitative*: for instance we might define $\mathrm{Obs}(M) \in [0,1]$ as the probability that M terminates. In such a quantitative setting, it becomes natural to look at *quantitative refinements* of program equivalence, with the aim of expressing that two non-equivalent programs are nonetheless very close, in the sense where they behave similarly with high probability. This motivation led to the introduction of the notion of *distances* for randomised programming languages [CL15a, CL15b, CL17, Ehr22, LHP23]. In the same spirit, quantitative notions have been developed in *computational security* [CL15a, BK22], where two programs are said to be *computationally indistinguishable* if no admissible adversary can tell them apart with more than negligible probability. Program distances have also been studied in the context of *differential privacy* [dAGHK19].

Unlike equivalence, however, there is no universally accepted definition of contextual distance. A naive generalisation of contextual equivalence, as proposed in [CL15b], would be to measure the maximum observable difference, across all contexts:

$$d^{\mathrm{ctx}}(M,N) := \sup_{C \text{ context}} \left|\mathrm{Obs}(C[M]) - \mathrm{Obs}(C[N])\right|.$$

This is indeed a *refinement* of contextual equivalence, since one can recover program equivalence from the pseudo-metric d^{ctx} by looking at its kernel. However, it is known that the above definition behaves unexpectedly as soon as contexts are granted *copying capabilities*: indeed in many cases the

*houssein.mansour@unibo.it
†raphaelle.crubille@lis-lab.fr

distances between two non-equivalent but close looking programs becomes unexpectedly 1. Intuitively, what happens is that if some context C is able to distinguish two programs M and N with some very small non-zero probability, it is often possible to build *amplification* contexts that increase this probability by doing many times the same experiment as C, and then doing statistic reasoning on the results of the successive experiments. In particular, in randomised languages that includes copying, together with a type system ensuring termination, a phenomenon known as *trivialisation* occurs [CL15b]: the distance between any pair of non-equivalent programs is always 1. Several alternative definitions of contextual distance have been proposed to overcome this problem:

- The *Amortised Contextual Distance* (δ^r_{ctx}), introduced by Thomas Ehrhard [Ehr22], is based on the addition of an *amortising factor* $r \in (0,1)$ to the definition of the naive contextual distance: intuitively, each time the context decides to access a new copy of its argument M, this access can now fail with probability r. Formally, this idea is modelled as:

$$\delta^r_{\mathrm{ctx}}(M_1, M_2) = \sup_{C \text{ context}} \left|\mathrm{Obs}\left(C\left[M_1 \oplus^r \Omega\right]\right) - \mathrm{Obs}\left(C\left[M_2 \oplus^r \Omega\right]\right)\right|,$$

  where Ω is a kind of error term that is added to the language, thus $M \oplus^r \Omega$ is a term that chooses at runtime between M and failure with probability r. This mechanism prevents the trivialisation effect, which asymptotically appears as $r \to 1$: then $\delta^r_{\mathrm{ctx}}(M, N)$ tends toward the naive contextual distance $d^{\mathrm{ctx}}(M, N)$.

- Another approach is to restrict the number of times a context can use its argument, typically via a linear type system. The first instance of this idea was the *affine contextual distance* introduced by Crubillé and Dal Lago [CL15b]. A more general approach, inspired by the works on *Fuzz*[dAGHK19], consists in defining a *graded* family of distances $(\delta^n(M, N))_{n \in \mathbb{N}}$, where $\delta^n(M, N)$ denotes the maximal observable difference that an *n-affine context* — a context that uses its argument at most n times—can enforce.

In this work, we propose a more *probability-theoretic* proof of the trivialisation property for the naive contextual distance, based on the *(weak) law of large numbers*, from where we are then able to extract this new proof bounds on the amortised and graded contextual distances. More precisely, we are able to (lower) bound them using *affine* contextual distances, i.e. expressions that use only the actions of *affine* contexts. We see our results as a first step towards bridging the gap between the kind of quantitative tools that are standard in probability and statistics, and the more elementary probabilistic reasoning that is currently used in (discrete) probabilistic semantics.

1 Applying the law of large numbers to prove trivialisation

1.1 The Weak Law of Large Numbers

The *weak law of large numbers*(LLN) is a result from early probability theory – already proved by Bernoulli [Ber13] in the case of boolean-valued distributions. Let's fix $\mathcal{D}$ a probability distribution on real numbers, and suppose that we sample many times independently from $\mathcal{D}$, and then take the mean of all these samplings. The LLN essentially says that we'll get with high probability a real very close from $\mathbb{E}(\mathcal{D})$–the expected value of $\mathcal{D}$. More precisely, it says that the empiric means *converges in probability* towards the expected value, which means that when the number of samplings goes to infinity, the probability of getting into a fixed width window around the expectancy goes to 1. From an operational point of view, it follows that testing the empirical mean is equivalent to testing the expectation itself, as soon as we have access to a sufficient number of samples.

1.2 Proving trivialisation with the LLN

We present now our trivialisation proof technique in the case of a probabilistic language where all programs terminate with probability 1, and moreover only ground type programs are observable. To simplify as far as possible the presentation, we additionally suppose that this ground type is **Bool**: it means that Obs(M) is the probability that M returns **true**[1]. Our overall approach is as follows:

1. First, we associate to any program of type **Bool** an infinite sequence of $\{0,1\}$-valued random variables $(X_P^n)_{n\in\mathbb{N}}$, such that the X_P^i are independent identically distributed (i.i.d) of law $[\![P]\!] := \text{Prob}(P \downarrow \textbf{false})\cdot\delta_0 + \text{Prob}(P \downarrow \textbf{true})\cdot\delta_1$–where δ_0 and δ_1 denote the Dirac distributions centred at 0 and 1, respectively. In other terms, for any $i_1,\dots,i_N$, and any $n\in\mathbb{N}$, $\text{Proba}(X_P^1 = i_1 \wedge \dots \wedge X_P^N = i_N)$ is exactly the probability that the successive evaluation of N copies of P returns successively $i_1,\dots,i_N$.

2. Secondly, for each $N\in\mathbb{N}$, and for every rational number $q>0$, we build a context C_N^q that calls N times its input, computes $\#\{i \mid \text{ the } i\text{-th call returns } 1\}$, and finally compare the result to the target value $N\cdot q$. In our random variables formalism, it becomes:

$$\text{Proba}(C_N^q[P]\downarrow \textbf{true}) = \text{Proba}(\overline{X_P}^N \leq q) \text{ where } \overline{X_P}^N := \sum_{1\leq i\leq N} X_P^i.$$

   Observe that this step uses crucially the fact that the context language is able to *copy* its input. $\overline{X_P}^N$ is the *N-th empiric mean* of the sequence of i.i.d variable (X_P), thus the LLN tells us how $C_n[P]$ behaves when n goes towards infinity:

$$\text{Proba}(C_n^q[P]\downarrow \textbf{true}) \to_{n\to\infty} \begin{cases} 1 \text{ when } q > \text{Proba}(P\downarrow \textbf{true}) \\ 0 \text{ when } q < \text{Proba}(P\downarrow \textbf{true}) \end{cases}. \tag{1}$$

We are now ready to prove trivialisation for any pair of non-equivalent programs M, N of type **Bool**. The trick consists in taking as target value q the real number which is *in the middle* between $\text{Proba}(M\downarrow \textbf{true})$ and $\text{Proba}(N\downarrow \textbf{true})$, i.e. $q_{M,N} := \frac{1}{2}(\text{Proba}(M\downarrow \textbf{true}) + \text{Proba}(N\downarrow \textbf{true}))$. If M and N are not equivalent, q will be *strictly* between $\mathbb{E}(X_M)$ and $\mathbb{E}(X_N)$, and thus when n becomes big enough, the empiric mean of n copies of (e.g.) M will be smaller than the target value q with overwhelming probability, while the empiric mean of N will be bigger than q with overwhelming probability. This situation is represented in Figure **??**. From there, and using (1) from above, we obtain the following proposition:

Proposition 1. *Let M, N closed programs of type Bool. Then:*

$$\begin{aligned}\delta_{ctx}(M,N) \geq & |\mathit{Proba}(C_n^{q_{M,N}}[M]\downarrow \textbf{\textit{true}}) - \mathit{Proba}(C_n^{q_{M,N}}[N]\downarrow \textbf{\textit{true}})| \\ = & |\mathit{Proba}\left(\overline{X_M}^n \leq q_{M,N}\right) - \mathit{Proba}\left(\overline{X_N}^n \leq q_{M,N})\right)| \to_{n\to+\infty} 1 \textit{ when } M \not\sim N\end{aligned}$$

[1] Our ideas could nonetheless be extended to other settings, as long as copying by contexts is allowed, and the termination guarantee holds–which are the two essential assumptions for trivialisation, see [CL15b]. In particular, with very minor modifications, our proof scheme would apply to the probabilistic variant of system $\mathbb{T}$ considered in [BLH17].

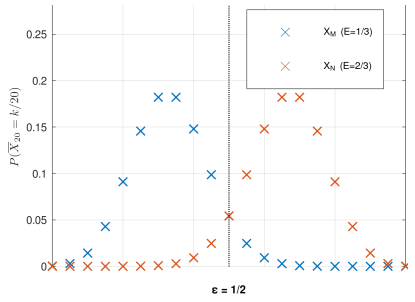

Figure 1: Probability distributions of the empiric mean for i.i.d. sequences (X_M), (X_N) with $\mathbb{E}(X_M) < \mathbb{E}(X_N)$.

Finally, we need to extend the trivialisation result from pairs of programs of type **Bool** to pairs of programs of any (identical) type. The same trick as in [CL15b] works there: if two programs M, N of some type σ are not equivalent, then there exists a context D that distinguishes them, and since this context returns an observable event it means that $D[M]$ and $D[N]$ are programs of type Bool, that moreover are non-equivalent, and thus we can conclude by applying the trivialisation result to $D[M], D[N]$.

2 Lower bounds on the graded and amortised distances

Both the graded and the amortised contextual distances are tamed versions of the naive contextual distance, where the copying abilities of the contexts are *controlled* in some ways. Moreover, the naive distance can be recovered from them *at the limit*, i.e. when the controls are made as weak as possible: when the graduation $n \to +\infty$ for the graded distance, or when the amortising factor $r \to +\infty$ for the amortised one. Our aim in this – and future – work is to extract information about the tamed distances from the contexts we built in Section 1.

Recall that by construction, C_n uses its argument exactly n times. So C_n contributes to the computation of the graded distance only at grade n. Moreover, it contributes with a factor r^n to the amortised distance, where r is the global amortising factor, because during its execution C_n is forced to pay n times the unitary cost r to obtain n copies of its argument. We want now to replicate the trick that allowed us to deduce trivialisation of the naive distance at all types from trivialisation at **Bool** type, but we need to be careful: indeed when considering contexts D that transform programs of type σ into programs of type Bool, we are forced to restrict ourselves to *linear* D. In the graded case, that's simply because we want the composite context $C_n[D]$ to still use its result at most n times: it is guaranteed by choosing D to be affine. It's subtler in the amortised case: what we need is that the transferring context D *does not* pay an additional amortising cost, and that's also guaranteed by choosing D to be linear.

Theorem 1. *For two programs M, N of some type σ:*

$$\delta^r_{ctx}(M,N) \geq \sup_{D \text{ linear}, n\in\mathbb{N}} r^n \cdot \delta^n_{LLN}([\![D[M]]\!],[\![D[N]]\!])$$
$$\forall n \in \mathbb{N}, \quad \delta^n(M,N) \geq \sup_{D \text{ affine}} \delta^n_{LLN}([\![D[M]]\!],[\![D[N]]\!]).$$

where $\delta^n_{LLN}([\![M]\!],[\![N]\!]) := |Proba(\overline{X_M}^n \leq q_{M,N}) - Proba(\overline{X_N}^n \leq q_{M,N})|$ – observe that this quantity depends indeed only on the $\{0,1\}$-valued probability distributions $[\![M]\!]$ and $[\![N]\!]$.

Observe that given $[\![M]\!], [\![N]\!]$ two probability distributions on $\{0,1\}$, and $n \in \mathbb{N}$, $\delta^n_{LLN}([\![M]\!],[\![N]\!])$ is *computable*, while it's less clear for $\sup_{n\in\mathbb{N}} r^n \cdot \delta^n_{LLN}([\![M]\!],[\![N]\!])$. We are working on establishing computable upper approximations of $\sup_{n\in\mathbb{N}} r^n \cdot \delta^n_{LLN}([\![M]\!],[\![N]\!])$, and also asymptotic approximations for $\delta^n_{LLN}([\![M]\!],[\![N]\!])$ and $(\sup_{n\in\mathbb{N}} r^n \cdot \delta^n_{LLN}([\![M]\!],[\![N]\!])$ when $n \to +\infty$ and $r \to 1$ respectively. We're not quite there yet but we obtained some preliminary results using results from probability

theory (or statistics) that describe the speed at which the empirical mean converges to the expected value: the *Central Limit Theorem (CLT)* [Bil95], the *Berry–Esseen inequality* [Ber41, Ess42], and the *Chernoff–Hoeffding inequality* [Che52].

3 Non termination and parallel convergence testers.

In our previous developments, it was actually crucial to consider a language where all programs *terminate with probability* 1: otherwise, the context C_n *does not* compute the empirical mean of the evaluation of n copies of its argument. It was shown in [CL17] that trivialisation of the naive contextual distance does not necessarily hold for a language without termination guarantees: for instance, in untyped randomised λ-calculus[2], the distance between $\lambda x.x$ and $\lambda x.x \oplus^{1/2} \Omega$ is $1/2$, where Ω here is any term terminating with 0 probability, and $\oplus^{1/2}$ is the fair probabilistic choice. Trivialisation for non-terminating languages, however, can be recovered, as shown in [CL17] by adding to the contexts language a *parallel convergence tester*, as introduced by Abramsky [Abr87], i.e. a construct $[\cdot \parallel \cdot]$ such that $[M \parallel N]$ terminates if and only if one of the two programs M and N terminate. The proof of trivialisation from [CL17] relies on the construction–using parallel convergence testers–of a family of contexts that simulates certain Boolean functions, known as *Tribes functions*, defined by partitioning the n input variables into m disjoint groups (called "tribes") and returning true if at least one tribe contains only true inputs.

3.1 Proving trivialisation via the LLN with parallel convergence testers

We looked at how to extend our approach from the terminating setting to the one of a non-terminating language extended with parallel convergence tester. Our goal was to use the parallel tester to define again a function that simulates the computation of the empirical mean of n copies of its input, with the aim of again applying the law of large numbers. Surprisingly, we found a way to build such contexts only when adding to the contexts language *all* n-ary m-convergence testers–meaning $[M_1 \parallel \ldots \parallel M_n]$ terminates if and only if at least m amongst the n terms terminate. While in a non-probabilistic calculus such generalised convergence testers could easily be simulated by the binary convergence tester, we found no way to do this in the probabilistic case. So while our LLN-driven scheme for trivialisation proof works for a probabilistic language with all n-ary m-convergence testers, the case of only binary convergence testers seems – for now at least – out of the scope of this technique.

3.2 Threshold theory and Trivialisation

We discovered that the trivialisation problem can in fact be presented as a problem from *threshold theory* [DC18], a research area that studies phenomena where a minor change in the parameters induces a major change in behaviour. A typical problem there is to consider a family of *monotonous* Boolean functions $f_n : \{0,1\}^n \to \{0,1\}$, and to sample its inputs independently according to a Bernoulli distribution with parameter $p \in [0,1]$–i.e. each bit X_i is set to 1 with probability p, and to 0 otherwise. A *threshold phenomenon* occurs when $\hat{f}_n(p) := \text{Proba}(f_n(X_1, \ldots, X_n) = 1 \mid X_i \leftarrow p \cdot \delta_0 + (1-p) \cdot \delta_1)$ transitions sharply from near 0 to near 1 as p crosses a critical value–where here

[2]Even though the trivialisation phenomenon doesn't necessarily happen in non-terminating randomised lambda-calculi with copying, the naive contextual distance isn't considered to be meaningful here either, because the amplification phenomenon still occurs for many pair of programs, in particular all the programs that encode randomised Booleans that terminate with probability 1. The graded distance [Ehr22] was actually designed first for PCF_p, a randomised variant of PCF.

sharply means that when n tends towards ∞, the width of the transition windows tends towards 0. Threshold theory is concerned about when such threshold exists, how fast they are reached, the width of the window where they occur etc....

Our LLN-based proof of trivialisation in Section 1 can be rewritten in the world of threshold theory: First we built for every target value q a family of monotonous boolean functions, that can be simulated by a context of the language :

$$f_n^q : (x_1, \dots, x_n) \mapsto \begin{cases} 1 \text{ if } \frac{1}{n} \cdot \sum x_i \leq q, \\ \text{and 0 otherwise.} \end{cases}$$

Then we proved, using the LLN, that for every $0 < q < 1$, the family f_n^q has a sharp threshold.All proofs of trivialisation in the literature– i.e. the original ones in [CL17], as well as ours in the present work – can be reframed to fit in this framework, for some well-chosen family (f_n^q). Besides, the construction of the f_n^q is usually driven by the expressive power of contexts, and as a consequence the difficult bit consists in proving that a given family of $(f_n^q)_{n\in\mathbb{N}}$ has a sharp threshold. That's also a crucial question in threshold theory, and thus there are there a number of non-trivial results on which kinds of family of boolean functions have threshold.

In particular, it is known [DC18] that any family of *weakly symmetric* monotone Boolean functions admits a *sharp threshold*, where weakly symmetric here means that f_n is invariant under the action of some transitive sub-group of the permutations group. This result can in particular be applied to the functions (f_n^q) we used in Section 1, even though the LLN give a more intuitive explanation. More interestingly, it can also be applied to the *Tribes* functions, that indeed form a weakly symmetric family — being invariant under any permutation that reorders the tribes and permutes the bits within each tribe. This gives us a more elegant proof that the one in [CL17] of the fact that the Tribes function has a sharp thresshold, and thus that the naive contextual distance is trivial for a language with copying and binary parallel convergence testers.

Moreover, threshold theory has also been interested in the *speed* of the sharpening of the threshold, which makes us hope to be able to use this proof of trivialisation in order to obtain bounds on the graded and amortised distances also in this setting.

References

[Abr87] S. Abramsky. Observation equivalence as a testing equivalence. *Theoretical Computer Science*, 53(2-3):225–241, 1987.

[Ber13] J. Bernoulli. *Ars Conjectandi*. Impensis Thurnisiorum, fratrum, 1713.

[Ber41] A. C. Berry. The accuracy of the gaussian approximation to the sum of independent variates. *Transactions of the American Mathematical Society*, 49(1):122–136, 1941.

[Bil95] P. Billingsley. *Probability and Measure*. John Wiley & Sons, 3rd edition, 1995.

[BK22] A. Broadbent and M. Karvonen. Categorical composable cryptography. In *International Conference on Foundations of Software Science and Computation Structures*, pages 161–183. Springer International Publishing Cham, 2022.

[BLH17] F. Breuvart, U. Dal Lago, and A. Herrou. On higher-order probabilistic subrecursion. In *Foundations of Software Science and Computation Structures - 20th International Conference, FOSSACS 2017, Held as Part of the European Joint Conferences on Theory*

and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings, pages 370–386, 2017.

[Che52] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, 23(4):493–507, 1952.

[CL15a] A. Cappai and U. Dal Lago. On equivalences, metrics, and polynomial time. In *Proc. of FCT*, pages 311–323, 2015.

[CL15b] R. Crubillé and U. Dal Lago. Metric reasoning about λ-terms: The affine case. In *Proc. of LICS*, pages 633–644, 2015.

[CL17] R. Crubillé and U. Dal Lago. Metric reasoning about λ-terms: The general case. In *European Symposium on Programming*, pages 341–367. Springer, 2017.

[dAGHK19] A. A. de Amorim, M. Gaboardi, J. Hsu, and S.-Y. Katsumata. Probabilistic relational reasoning via metrics. In *34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–19. IEEE, 2019.

[DC18] H. Duminil-Copin. Sharp threshold phenomena in statistical physics. In *Proceedings of the Takagi Lectures 2017*, 2018.

[Ehr22] T. Ehrhard. Differentials and distances in probabilistic coherence spaces. *Logical Methods in Computer Science*, 18(3), 2022.

[Ess42] C.-G. Esseen. On the liapounoff limit of error in the theory of probability. *Arkiv för Matematik, Astronomi och Fysik*, 28A(9):1–19, 1942.

[Jr.69] J. H. Morris Jr. *Lambda-Calculus Models of Programming Languages*. PhD thesis, Massachusetts Institute of Technology, 1969.

[LHP23] U. Dal Lago, N. Hoshino, and P. Pistone. On the lattice of program metrics. In M. Gaboardi and F. van Raamsdonk, editors, *8th International Conference on Formal Structures for Computation and Deduction (FSCD 2023)*, volume 260 of *LIPIcs*, pages 20:1–20:19. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2023.