On Computational Indistinguishability and Logical Relations

Ugo Dal Lago, Zeinab Galal, Giulia Giusti

On Computational Indistinguishability and Logical Relations

Ugo Dal Lago[1,2][0000−0001−9200−070X], Zeinab Galal[1,2][0009−0008−6402−3531], and Giulia Giusti[3][0000−0002−6533−8307]

[1] University of Bologna, Italy
[2] INRIA Sophia Antipolis, France
[3] ENS Lyon, France

Abstract. A λ-calculus is introduced in which all programs can be evaluated in probabilistic polynomial time and in which there is sufficient structure to represent sequential cryptographic constructions and adversaries for them, even when the latter are oracle-based. A notion of observational equivalence capturing computational indistinguishability and a class of approximate logical relations are then presented, showing that the latter represent a sound proof technique for the former. The work concludes with the presentation of an example of a security proof in which the encryption scheme induced by a pseudorandom function is proven secure against active adversaries in a purely equational style.

Keywords: Computational Indistinguishability · Probabilistic Effects · Metrics · Logical Relations

Introduction

The two predominant models in cryptography, namely the computational [27] and the symbolic [19] models, have had very different fates with respect to the application of language-based verification techniques to them. In the symbolic model, which does not account for complexity nor for probability, the application of classic verification methodologies (e.g. model checking [21], rewriting [42] and abstract interpretation [1]) is natural and has been extensively done. In the computational model, instead, all this is notoriously more problematic.

An interesting line of work, which has given rise to an increasing number of contributions in the last 25 years (see, e.g., [33, 40, 30, 15]), consists in the application of classical program equivalence theories to programming languages specifically designed to capture the reference notion of complexity in the computational model, namely that of a probabilistic polynomial time algorithm (PPT below). Once this is done, the gold standard notion of equivalence in cryptography, namely *computational indistinguishability* [35, 26], becomes a form of observational equivalence, thus paving the way towards the study of computational indistinguishability via standard tools from programming language theory, like

logical relations [50, 49] and applicative bisimilarity [2, 47], which are sound by construction (although not necessarily complete) for observational equivalence.

This is precisely the direction we explore in this work; our objective is to define a typed λ-calculus with references and probabilistic choice able to naturally capture the complexity constraints mentioned above through a form of graded modality, at the same time allowing to easily express primitives, experiments and reductions, which are the building blocks on which game-based proofs are based. The language we introduce, called λ**BLL**, can be seen as derived from Bounded Linear Logic [25]. Its syntax and operational properties are analyzed in detail in Section 1.

In Section 2, we then move on to define a notion of logical relation for λ**BLL** and demonstrate that it is *sound* for an approximate observational equivalence precisely capturing computational indistinguishability. A crucial aspect is that the proposed logical relation, in fact based on a logical *metric*, is *approximate* and therefore manages to capture programs that do not behave *exactly* the same way.

Section 3 is devoted to showing how a set of equations all justifiable through the introduced logical relations allows us to prove the security of an intrinsically second-order cryptographic construction, i.e. the proof that the encryption scheme induced by a pseudorandom function is CPA-secure, a classic result in cryptography. Notably, this proof intrinsically relies on approximate notions of equivalence. Moreover, parts of it make essential use of references.

1 λBLL: a Calculus Capturing PPT

We define a language λ**BLL**, inspired by graded λ-calculi [45] and CBPV [38, 20], and expressive enough to model complex cryptographic experiments requiring to keep track of the messages on which the oracle is queried by the adversary.

Types At the level of types, λ**BLL** has a linear type system (Figure 1) with a correspondence to Bounded Linear Logic (**BLL**) [25] and graded-calculi [45] with indexed comonadic types. In our case, the grades are polynomials and serve to keep the complexity of the attackers under control. They are built from positive natural numbers ($\mathbb{N}_{\geq 1}$), addition and multiplication, but also contain a polynomial variable i (corresponding to the security parameter), allowing us to reason on indexed families of types and terms as in **BLL** [25]. Ground types are generated from unit $\mathbb{U}$, booleans $\mathbb{B}$ and binary strings $\mathbb{S}[p]$ of length p for some polynomial p. We distinguish between positive types and general types in the CBPV style [38, 20] to restrict the argument of an application to be of positive type. To model references, we use effect typing [24] and annotate the bang and arrow types with reference contexts providing information on which memory cells are used during program execution. We also consider two types of contexts to distinguish between term variables x and memory references r.

Ground types	$G ::= \mathbb{U} \mid \mathbb{B} \mid \mathbb{S}[p]$	**Ground values**	$W ::= \star \mid \mathbf{t} \mid \mathbf{f} \mid s$
Positive types	$P ::= G \mid P \otimes P \mid !^{\Theta}_{p} A$	**Positive values**	$Z ::= x \mid W \mid \langle Z, Z \rangle \mid !M$
Types	$A ::= P \mid P \multimap^{\Theta} A$	**Values**	$\mathcal{V} \ni V ::= Z \mid \lambda x.M$
		Computations	$\Lambda \ni M ::= \mathtt{return}\, V \mid \mathtt{der}(Z) \mid MZ$
Variable contexts	$\Gamma ::= \varnothing \mid x : P, \Gamma$		$\mid \mathtt{let}\, x = N \,\mathtt{in}\, M \mid f_p(Z_1, \ldots, Z_m)$
Reference contexts	$\Theta ::= \varnothing \mid r : G, \Theta$		$\mid \mathtt{loop}\, V\, p\, \mathtt{times}\, \mathtt{from}\, M \mid \mathtt{set}\, r\, Z$
			$\mid \mathtt{if}\, Z\, \mathtt{then}\, M\, \mathtt{else}\, N \mid \mathtt{get}\, r$
Polynomials	$p ::= 1 \mid i \mid p + p \mid p \times p$		$\mid \mathtt{let}\, \langle x, y \rangle = Z \,\mathtt{in}\, M$

Fig. 1: Syntax of λ**BLL**

Terms Grammars for values and computations are in Figure 1. Memory references can only store values of ground type, and are handled in a simple way via reading and writing operators on locations. The term $\mathtt{set}\, r\, V$ corresponds to updating the memory location referenced by r with the value V and $\mathtt{get}\, r$ returns the value under the reference r.

We enrich the grammar of λ-calculus with function symbols computing probabilistic polytime functions, which are the basic building blocks of any cryptographic protocol. We fix a set of function symbols $\mathcal{F}$ and each function symbol f in $\mathcal{F}$ comes equipped with:

- a type denoted $\mathsf{typeof}(f)$ of the form $G_1 \times \cdots \times G_m \to G$ where $G_1, \ldots, G_m$ and G are ground types;
- for every polynomial p in $\mathbb{N}_{\geq 1}[i]$, a term constructor f_p of arity m.

For example, we will consider the function symbol $\mathtt{random}$ with $\mathsf{typeof}(\mathtt{random}) = \mathbb{S}[i]$ and arity 0 interpreted as a map randomly generating a string in $\{0,1\}^i$, and the function symbol $\mathtt{xor}$ with $\mathsf{typeof}(\mathtt{xor}) = \mathbb{S}[i] \times \mathbb{S}[i] \to \mathbb{S}[i]$ and arity 2 interpreted as a map computing the *bitwise exclusive-or* of binary strings.

Furthermore, in order to make λ**BLL** expressive enough to model experiments involving an adversary that can access an oracle a polynomial number of times, the grammar of computations includes an iterator $\mathtt{loop}$.

Typing Rules The typing rules for λ**BLL** are given in Figure 2. We have two kinds of typing judgements:

$$\Gamma \vdash V : A \quad \text{and} \quad \Gamma; \Theta \vdash M : A$$

for values and computations respectively, where $\Gamma = x_1 : P_1, \ldots, x_n : P_n$ is a context assigning positive types to term variables and $\Theta = r_1 : G_1, \ldots, r_n : G_n$ is a reference context assigning ground types to reference variables. The operation of polynomial addition induces a binary partial operation $\boxplus$ on positive types defined by induction below:

$$\begin{aligned} G \boxplus G &:= G \\ (P \otimes Q) \boxplus (R \otimes S) &:= (P \boxplus R) \otimes (Q \boxplus S) \\ (!^{\Theta}_{p} A) \boxplus (!^{\Theta}_{q} A) &:= !^{\Theta}_{p+q} A. \end{aligned}$$

To account for polynomial multiplication, we also define for every polynomial $p \in \mathbb{N}_{\geq 1}[i]$, a total unary operation on positive types by induction:

$$\begin{aligned} p * G &:= G \\ p * (P \otimes Q) &:= (p * P) \otimes (p * Q) \\ p * (!^{\Theta}_{q} A) &:= !^{\Theta}_{p \times q} A. \end{aligned}$$

It is important to note that on ground types, the identities $G \boxplus G = G$ and $p * G = G$ mean that ground values (unit $\star$, booleans $\mathbf{t}, \mathbf{f}$ and binary strings $s \in \{0,1\}^*$) are duplicable whereas we keep track of the polytime complexity for higher-order applications and effects similarly to [17]. The partial operation $\boxplus$ on positive types can be extended to a total operation on variable contexts:

$$\begin{aligned} \varnothing \boxplus \varnothing &:= \varnothing \\ (x : P, \Gamma) \boxplus \Delta &:= \begin{cases} x : P, \Gamma \boxplus \Delta & \text{if } x \text{ does not occur in } \Delta \\ x : P \boxplus Q, \Gamma \boxplus \Sigma & \text{if } \Delta = x : Q, \Sigma \end{cases} \end{aligned}$$

We also extend the operation $p * (-)$ on positive types to a total operation on term variables contexts:

$$\begin{aligned} p * \varnothing &:= \varnothing \\ p * (x : P, \Gamma) &:= (x : p * P), p * \Gamma \end{aligned}$$

For a polynomial p in $\mathbb{N}_{\geq 1}[i]$ and a type A, we write Ap for the type $A[p/i]$ where we substitute all the occurrences of the security parameter i by p. Similarly, for a term M, we write Mp for the term $M[p/i]$.

Probability Distributions Our calculus incorporates probabilistic effects with references by combining the distribution monad and the state monad. Recall that for a set X, a *(finite) probability distribution* is a function $\mu : X \to [0,1]$ with finite support, *i.e.* the set $\mathbf{supp}(\mu) := \{x \in X \mid \mu(x) > 0\}$ is finite, and such that $\sum_{x \in X} \mu(x) = 1$. We denote by $\delta_x : X \to [0,1]$ the *Dirac distribution* mapping an element y in X to 1 if $y = x$ and to 0 otherwise. Any probability distribution μ is then equal to

$$\sum_{1 \leq k \leq m} a_k \cdot \delta_{x_k} \text{ where } \{x_1, \ldots, x_m\} = \mathbf{supp}(\mu) \text{ and } a_k = \mu(x_k)$$

for $1 \leq k \leq m$. We denote by $\mathbf{D}(X)$ the set of all probability distributions over X. It induces a monad $(\mathbf{D}, \eta_{\mathbf{D}}, \gg=_{\mathbf{D}})$ on the category $\mathbf{Set}$ of sets and functions (we will omit the subscripts if there is no ambiguity). The unit has components $\eta_X : x \mapsto \delta_x$ given by Dirac distributions and the bind operator

$$\gg= : \mathbf{D}(X) \times \mathbf{Set}(X, \mathbf{D}(Y)) \to \mathbf{D}(Y)$$

maps a distribution $\mu = \sum_k a_k \delta_{x_k} \in \mathbf{D}(X)$ and a function $f : X \to \mathbf{D}(Y)$ to the pushforward distribution $\mu \gg= f := \sum_k a_k \delta_{f(a_k)}$.

$$\frac{}{\Gamma, x : P \vdash x : P}\ \text{VAR} \qquad \frac{}{\Gamma \vdash \mathbf{t} : \mathbb{B}}\ \text{TRUE} \qquad \frac{}{\Gamma \vdash \mathbf{f} : \mathbb{B}}\ \text{FALSE}$$

$$\frac{\text{typeof}(f) = G_1 \times \cdots \times G_m \to G \quad (\Gamma_k p \vdash Z_k : G_k p)_{1 \le k \le m} \quad p \in \mathbb{N}_{\ge 1}[i]}{\boxplus_k \Gamma_k p; \Theta \vdash f_p(Z_1, \ldots, Z_m) : Gp}\ \text{FUN}$$

$$\frac{s \in \{0,1\}^c \quad p : i \mapsto c \text{ is a constant polynomial}}{\Gamma \vdash s : \mathbb{S}[p]}\ \text{STRING} \qquad \frac{\Gamma \vdash Z_1 : P \quad \Delta \vdash Z_2 : Q}{\Gamma \boxplus \Delta \vdash \langle Z_1, Z_2 \rangle : P \otimes Q}\ \text{TENSOR}$$

$$\frac{\Gamma \vdash Z : P \otimes Q \quad x : P, y : Q, \Delta; \Theta \vdash M : A}{\Gamma \boxplus \Delta; \Theta \vdash \mathtt{let}\ \langle x, y \rangle = Z\ \mathtt{in}\ M : A}\ \text{LET} \qquad \frac{}{\Gamma \vdash \star : \mathbb{U}}\ \text{UNIT} \qquad \frac{\Gamma; \Theta \vdash M : A}{p * \Gamma \vdash\ !M : !_p^{\Theta} A}\ \text{BANG}$$

$$\frac{\Gamma \vdash Z : !_1^{\Theta} A}{\Gamma; \Theta \vdash \mathtt{der}(Z) : A}\ \text{DER} \qquad \frac{\Gamma; \Theta \vdash M : P \overset{\Theta}{\multimap} A \quad \Delta \vdash Z : P}{\Gamma \boxplus \Delta; \Theta \vdash MZ : A}\ \text{APP} \qquad \frac{\Gamma, x : P; \Theta \vdash M : A}{\Gamma \vdash \lambda x.M : P \overset{\Theta}{\multimap} A}\ \text{LAM}$$

$$\frac{\Gamma \vdash V : A}{\Gamma; \Theta \vdash \mathtt{return}\, V : A}\ \text{ETA} \qquad \frac{\Gamma; \Theta \vdash N : P \quad x : P, \Delta; \Theta \vdash M : A}{\Gamma \boxplus \Delta; \Theta \vdash \mathtt{let}\, x = N\ \mathtt{in}\, M : A}\ \text{LET}$$

$$\frac{\Gamma \vdash V : P \overset{\Theta}{\multimap} P \quad \Delta; \Theta \vdash M : P}{(p * \Gamma) \boxplus \Delta; \Theta \vdash \mathtt{loop}\, V\, p\, \mathtt{times\, from}\, M : P}\ \text{LOOP} \qquad \frac{\Gamma \vdash Z : G}{\Gamma; \Theta, r : G \vdash \mathtt{set}\, r\, Z : \mathbb{U}}\ \text{SET}$$

$$\frac{}{\Gamma; \Theta, r : G \vdash \mathtt{get}\, r : G}\ \text{GET} \qquad \frac{\Gamma; \Theta \vdash Z : \mathbb{B} \quad \Delta; \Theta \vdash M : A \quad \Delta; \Theta \vdash N : A}{\Gamma \boxplus \Delta; \Theta \vdash \mathtt{if}\, Z\, \mathtt{then}\, M\, \mathtt{else}\, N : A}\ \text{CASE}$$

Fig. 2: λ**BLL** typing rules

Combining Probability with References The general idea is that a *store* is a map from memory reference variables to values that preserves typing. More precisely, for a fixed closed reference context $\Theta = r_1 : G_1, \ldots, r_m : G_m$ (meaning that the security parameter variable i does not occur in the types $G_1, \ldots G_m$), we denote by St_Θ the set of functions $e : \{r_1, \ldots, r_m\} \to \mathcal{V}$ such that $e(r_j) \in \{V \in \mathcal{V} \mid \cdot \vdash V : G_j\}$ for all $1 \le j \le m$.

We associate to every closed Θ a corresponding monad $(\mathbf{T}_\Theta, \eta_\Theta, \gg\!=_\Theta)$ on **Set** given by the tensor product [32] of the distribution monad with the state monad $\mathbf{T}_\Theta := (\mathbf{D}(- \times \mathrm{St}_\Theta))^{\mathrm{St}_\Theta}$, similarly to [4]. The unit of $\mathbf{T}_\Theta$ has components $X \to \mathbf{D}(X \times \mathrm{St}_\Theta)^{\mathrm{St}_\Theta}$ mapping $x \in X$ and $e \in \mathrm{St}_\Theta$ to the Dirac distribution $\delta_{(x,e)}$. The bind operator

$$\gg\!=_\Theta : \mathbf{T}_\Theta X \times \mathbf{Set}(X, \mathbf{T}_\Theta Y) \to \mathbf{T}_\Theta Y$$

takes $\varphi \in \mathbf{T}_\Theta X$ and $f : X \to \mathbf{T}_\Theta Y$ to the map $\boldsymbol{\lambda} e.(\varphi(e) \gg\!=_{\mathbf{D}} \mathbf{eval} \circ (f \times \mathrm{id}_{\mathrm{St}_\Theta}))$ where $\boldsymbol{\lambda}$ and **eval** are respectively the Currying operator and the evaluation map induced by the Cartesian closed structure of **Set**.

From Sets to Indexed Families To work with general term sequents where the security parameter i *may occur freely*, we generalize the discussion above from sets to families of sets. Let **ISet** be the category whose objects are families $X = \{X_n\}_{n \ge 1}$ of sets indexed by $\mathbb{N}_{\ge 1}$ and a morphism from $X = \{X_n\}_{n \ge 1}$ to $Y = \{Y_n\}_{n \ge 1}$ is a family of functions $\{f_n : X_n \to Y_n\}_{n \ge 1}$.

In our calculus, probabilistic effects are generated via the function symbols in $\mathcal{F}$. For each $f \in \mathcal{F}$ with $\mathsf{typeof}(f) = G_1 \times \cdots \times G_m \to G$, we assume that:

- there is a family $[\![f]\!] = \{[\![f]\!]_n\}_{n\geq 1}$ of set-functions $[\![f]\!]_n : [\![G_1]\!]_n \times \cdots \times [\![G_m]\!]_n \to \mathbf{D}([\![G]\!]_n)$ indexed over the security parameter $n \geq 1$ where $[\![\mathbb{S}[p]]\!]_n := \{0,1\}^{p(n)}$, $[\![\mathbb{B}]\!]_n := \{\mathbf{t}, \mathbf{f}\}$ and $[\![\mathbb{U}]\!]_n := \{\star\}$.
- these functions can be evaluated in probabilistic polynomial time: there exists a PPT algorithm $\mathsf{alg}(f)$ such that for every $n \geq 1$, if $\mathsf{alg}(f)$ is fed with input 1^n and a tuple $t \in [\![G_1]\!]_n \times \cdots \times [\![G_m]\!]_n$, it returns $x \in [\![G]\!]_n$ with probability $[\![f]\!]_n(t)(x)$. This can be achieved by taking function symbols from a language guaranteeing the aforementioned complexity bounds [40, 18]. A very small amount of these would however be sufficient for completeness.

Now, for a general reference context Θ (whose types may contain i), we define a monad on **ISet** mapping an indexed family $X = \{X_n\}_{n\geq 1}$ to the family

$$\{\mathbf{T}_{\Theta n}(X_n)\}_{n\geq 1} = \{(\mathbf{D}(X_n \times \mathrm{St}_{\Theta n}))^{\mathrm{St}_{\Theta n}}\}_{n\geq 1}$$

which we will use for the operational semantics of $\lambda\mathbf{BLL}$.

Operational Semantics For every variable context Γ, reference context Θ and type A, we define indexed families $\Lambda^{\Theta}(\Gamma; A) = \{\Lambda^{\Theta}_n(\Gamma; A)\}_{n\geq 1}$ and $\mathcal{V}(\Gamma; A) = \{\mathcal{V}_n(\Gamma; A)\}_{n\geq 1}$ of typable terms and values respectively as

$$\Lambda^{\Theta}_n(\Gamma; A) := \{M \in \Lambda \mid \Gamma n; \Theta n \vdash M : An\} \text{ and}$$
$$\mathcal{V}_n(\Gamma; A) := \{V \in \mathcal{V} \mid \Gamma n \vdash V : An\}.$$

If the variable context Γ is empty, we write $\Lambda^{\Theta}_n(A)$ and $\mathcal{V}_n(A)$ for $\Lambda^{\Theta}_n(\varnothing; A)$ and $\mathcal{V}_n(\varnothing; A)$ respectively.

For a fixed reference context Θ, the small step operational semantics (Figure 3) is an indexed relation $\longrightarrow = \{\longrightarrow_n\}_{n\geq 1}$ with

$$\longrightarrow_n \subseteq (\Lambda^{\Theta}_n \times \mathrm{St}_{\Theta n}) \times \mathbf{D}(\Lambda^{\Theta}_n \times \mathrm{St}_{\Theta n})$$

where $\Lambda^{\Theta}_n := \{M \in \Lambda \mid \Gamma n; \Theta n \vdash M : An \text{ for some } \Gamma, A\}$. For a triple $(M, e, \mathscr{D})$ in $\longrightarrow_n$, we write $(M, e) \longrightarrow_n \mathscr{D}$ and for ease of readability, we denote a probability distribution $\sum_{1\leq k\leq m} a_k \delta_{x_k}$ in set theoretic fashion $\{x_1^{a_1} \ldots, x_m^{a_m}\}$. For the case $\mathtt{set}\, r\, V$, $e[V/r]$ denotes the store mapping a reference r' to $e(r')$ if $r' \neq r$ and to V if $r' = r$.

Our calculus is strongly normalizing and in addition to the small step operational semantics for one step reductions, we also provide a final or big step semantics for the convergence behavior of terms. For a fixed reference context Θ and a type A, the final semantics $(\!|-|\!)^{\Theta,A}$ for closed $\lambda\mathbf{BLL}$-terms is a map in **ISet** corresponding to the indexed family of functions

$$\{(\!|-|\!)^{\Theta,A}_n : \Lambda^{\Theta}_n(A) \to \mathbf{D}(\mathcal{V}_n(A) \times \mathrm{St}_{\Theta n}))^{\mathrm{St}_{\Theta n}}\}_{n\geq 1}$$

obtained in two steps:

$$\frac{}{(\mathtt{let}\, x = \mathtt{return}\, V\, \mathtt{in}\, M, e) \longrightarrow_n \{(M[V/x], e)^1\}}$$

$$\frac{(N, e) \longrightarrow_n \{(N_k, e_k)^{a_k}\}}{(\mathtt{let}\, x = N\, \mathtt{in}\, M, e) \longrightarrow_n \{(\mathtt{let}\, x = N_k\, \mathtt{in}\, M, e_k)^{a_k}\}}$$

$$\frac{}{(\mathtt{let}\, \langle x, y\rangle = \mathtt{return}\, \langle V, W\rangle\, \mathtt{in}\, M, e) \longrightarrow_n \{(M[V/x, W/y], e)^1\}}$$

$$\frac{}{((\lambda x.M)V, e) \longrightarrow_n \{(M[V/x], e)^1\}} \qquad \frac{(M, e) \longrightarrow_n \{(M_k, e_k)^{a_k}\}}{(MV, e) \longrightarrow_n \{(M_k V, e_k)^{a_k}\}}$$

$$\frac{}{(\mathtt{der}(!M), e) \longrightarrow_n \{(M, e)^1\}} \qquad \frac{}{(\mathtt{set}\, r\, V, e) \longrightarrow_n \{(\star, e[V/r])^1\}}$$

$$\frac{}{(\mathtt{get}\, r, e) \longrightarrow_n \{(e(r), e)^1\}} \qquad \frac{}{(\mathtt{if}\, \mathbf{t}\, \mathtt{then}\, M\, \mathtt{else}\, N, e) \longrightarrow_n \{(M, e)^1\}}$$

$$\frac{}{(\mathtt{if}\, \mathbf{f}\, \mathtt{then}\, M\, \mathtt{else}\, N, e) \longrightarrow_n \{(N, e)^1\}}$$

$$\frac{}{(f_p(W_1, \ldots, W_m), e) \longrightarrow_n \{([\![f]\!]_{p(n)}(W_1, \ldots, W_m), e)^1\}}$$

$$\frac{}{(\mathtt{loop}\, (\lambda x.M)\, 1\, \mathtt{times\, from}\, N, e) \longrightarrow_n \{(\mathtt{let}\, x = N\, \mathtt{in}\, M, e)^1\}}$$

$$\frac{}{(\mathtt{loop}\, (\lambda x.M)\, k+1\, \mathtt{times\, from}\, N, e) \longrightarrow_n \{(\mathtt{let}\, x = (\mathtt{loop}\, (\lambda x.M)\, k\, \mathtt{times\, from}\, N)\, \mathtt{in}\, M, e)^1\}}$$

Fig. 3: Small step semantics of $\lambda\mathbf{BLL}$

1. We first use the fact that the monad $\mathbf{T}_\Theta$ on $\mathbf{Set}$ extends to the category of ω-complete partial orders with a bottom element (ω-cppo) and Scott-continuous morphisms (it follows easily from the fact that both the distribution and the state monads extend to ω-cppo's). It allows us to define inductively a family $(\!|-|\!)_n^{\Theta,A} : \Lambda_n(A) \to \mathbf{T}_{\Theta n}^{\perp}(\mathcal{V}_n(A))$ where for a set X, $\mathbf{T}_{\Theta n}^{\perp} := \mathbf{T}_{\Theta n}(X \uplus \{\perp\}, \leq)$ is the image of the flat ordering ($\perp \leq x$ for all $x \in X$) under $\mathbf{T}_{\Theta n}$ (the bottom element $\perp$ is added to account for computations which are possibly non-terminating). Similarly to [37], each map $(\!|-|\!)_n^{\Theta,A}$ is obtained as the supremum $\bigvee_{k\in\omega}(\!|-|\!)_{n,k}^{\Theta,A}$ where $(\!|-|\!)_{n,k}^{\Theta,A}$ is defined inductively below:

$$(\!|M|\!)_{n,0}^{\Theta,A} := \perp \qquad (\!|\mathtt{return}\, V|\!)_{n,k+1}^{\Theta,A} := \eta_{\mathcal{V}_n(A)}(V)$$

$$(\!|\mathtt{if}\, \mathbf{t}\, \mathtt{then}\, M\, \mathtt{else}\, N|\!)_{n,k+1}^{\Theta,A} := (\!|M|\!)_{n,k}^{\Theta,A} \qquad (\!|\mathtt{if}\, \mathbf{f}\, \mathtt{then}\, M\, \mathtt{else}\, N|\!)_{n,k+1}^{\Theta,A} := (\!|N|\!)_{n,k}^{\Theta,A}$$

$$(\!|\mathtt{set}\, r\, Z|\!)_{n,k+1}^{\Theta,A}(e) := \delta_{(\star, e[Z/r])} \qquad (\!|\mathtt{get}\, r|\!)_{n,k+1}^{\Theta,A}(e) := \delta_{(e(r), e)}$$

$$(\!|\mathtt{der}(!M)|\!)_{n,k+1}^{\Theta,A} := (\!|M|\!)_{n,k}^{\Theta,A}$$

$$(\!|\mathtt{let}\, x = N\, \mathtt{in}\, M|\!)_{n,k+1}^{\Theta,A} := (\!|N|\!)_{n,k}^{\Theta,P} \gg\!= (U \mapsto (\!|M[U/x]|\!)_{n,k}^{\Theta,A})$$

$$(\!|\mathtt{let}\, \langle x, y\rangle = \langle Z, Z'\rangle\, \mathtt{in}\, M|\!)_{n,k+1}^{\Theta,A} := (\!|M[Z/x, Z'/y]|\!)_{n,k}^{\Theta,A}$$

$$(\!|MZ|\!)_{n,k+1}^{\Theta,A} := (\!|M|\!)_{n,k}^{\Theta, P \overset{\Theta}{\Rightarrow} A} \gg\!= (\lambda x.N \mapsto (\!|N[Z/x]|\!)_{n,k}^{\Theta,A})$$

For the case $(\!|\mathtt{loop}\,\lambda x.M\,m\,\mathtt{times\,from}\,N|\!)^{\Theta,A}_{n,k+1}$, if $m = 1$, we define it to be $(\!|N|\!)^{\Theta,A}_{n,k} \gg\!= (U \mapsto (\!|M[U/x]|\!)^{\Theta,A}_{n,k})$ and if $m > 1$, we take

$$(\!|\mathtt{loop}\,\lambda x.M\,(m-1)\,\mathtt{times\,from}\,N|\!)^{\Theta,A}_{n,k} \gg\!= (U \mapsto (\!|M[U/x]|\!)^{\Theta,A}_{n,k}).$$

For a function symbol f with $\mathsf{typeof}(f) = G_1 \times \cdots \times G_m \to G$ and a polynomial p, $(\!|f_p(W_1, \ldots, W_m)|\!)^{\Theta,A}_{n,k+1}(e)$ is the mapping

$$(V, e') \mapsto \delta_e(e')[\![f]\!]_{p(n)}(W_1, \ldots, W_m)(V).$$

2. We prove that for any $M \in \Lambda^{\Theta}_n(A)$ and $e \in \mathrm{St}_{\Theta n}$, (M, e) reduces to some distribution $\mathscr{D}$ in polynomial time. As a corollary, we obtain that the final semantics map $(\!|-|\!)^{\Theta,A}_n$ can in fact be restricted to $\Lambda^{\Theta}_n(A) \to \mathbf{T}_{\Theta n}(\mathcal{V}_n(A))$ since $\lambda\mathbf{BLL}$ is strongly normalizing.

To express the standard correspondence between the final (big step) semantics and the transitive closure of the small step semantics, we define an indexed relation

$$\Downarrow = \{\Downarrow^m_n \subseteq (\Lambda^{\Theta}_n \times \mathrm{St}_{\Theta n}) \times \mathbf{D}(\Lambda^{\Theta}_n \times \mathrm{St}_{\Theta n})\}_{n \geq 1, m \geq 0}$$

where the additional natural number m models the number of steps in the small step semantics:

$$\frac{}{(\mathtt{return}\,V, e) \Downarrow^0_n \{(\mathtt{return}\,V, e)^1\}} \qquad \frac{(M, e) \longrightarrow_n \mathscr{D} \quad \{E \Downarrow^{m_k}_n \mathscr{E}_E\}_{E \in \mathbf{supp}(\mathscr{D})}}{(M, e) \Downarrow^{1+\max_k m_k}_n \sum_E \mathscr{D}(E) \cdot \mathscr{E}_E}$$

and formulate the result as follows:

Lemma 1. *For a fixed reference context Θ, type A and security parameter n, the following are equivalent: for any term $M \in \Lambda^{\Theta}_n(A)$, store $e \in \mathrm{St}_{\Theta n}$ and distribution $\mathscr{D} \in \mathbf{D}(\mathcal{V}_n(A) \times \mathrm{St}_{\Theta n})$:*

$$(\!|M|\!)^{\Theta,A}_n(e) = \mathscr{D} \quad \Leftrightarrow \quad \exists k \in \mathbb{N},\ (\!|M|\!)^{\Theta,A}_{n,k}(e) = \mathscr{D} \quad \Leftrightarrow \quad \exists m \in \mathbb{N}, (M, e) \Downarrow^m_n \mathscr{D}$$

Soundness and Completeness for Polynomial Time A calculus like $\lambda\mathbf{BLL}$ makes sense, particularly in view of the cryptographic applications that we will present in the last part of this article, if there is a correspondence with the concept of probabilistic polynomial time. This section is dedicated to giving evidence that such a correspondence indeed holds.

Before moving on to the description of soundness and completeness, however, it is worth outlining what is meant in this context by probabilistic polynomial time. In fact, what we mean by a PPT function can be deduced from how we defined function symbols in $\mathcal{F}$: these are families of functions, indexed on natural numbers, which possibly return a distribution and are computable by a probabilistic Turing machine working in polynomial time on the value of the underlying parameter. That basic functions are PPT holds by hypothesis, but that the same remains true for any term definable in the calculus needs to be proved. Moreover, the fact that any such function can be represented in $\lambda\mathbf{BLL}$ has to be proved as well.

Soundness for PPT The goal is to show that there exists a polynomial bound on the length of reduction sequences for any term of λ**BLL**:

Theorem 1 (Polytime Soundness). *For every type derivation π of a term M in λ**BLL**, there exists a polynomial q_π such that for every natural numbers $n \geq 1, m \geq 0$ and store e, if $(Mn, e) \Downarrow_n^m \mathscr{D}$, then $m \leq q_\pi(n)$.*

Similarly to Section 4.2 in [15], the proof of Theorem 1 is structured into three steps:

- We assign a polynomial q_π to every type derivation π defined by induction on the structure of π.
- We prove that $q_{(\cdot)}$ is stable under polynomial substitution: for every type derivation π with conclusion $\Gamma; \Theta \vdash M : A$ and for every polynomial p, there is a type derivation ζ with conclusion $\Gamma p; \Theta p \vdash Mp : Ap$ such that $q_\zeta(n) = q_\pi(p(n))$ for all $n \geq 1$.
- Finally, we prove that $q_{(\cdot)}$ strictly decreases along term reduction: if π derives N and $(N, e) \longrightarrow_n \mathscr{D}$, then for all (N', e') in $\mathbf{supp}(\mathscr{D})$, there exists a type derivation ζ for N' such that $q_\pi > q_\zeta$.

Completeness for PPT Theorem 1 implicitly tells us that algorithms formulated as typable λ**BLL** terms are PPT, since the number of reduction steps performed is polynomially bounded and reduction can be simulated by a Turing machine [16, 3]. One can further prove that all PPT functions can be represented by λ**BLL** terms. Given the freedom we have about picking more and more basic function symbols, this does not seem surprising: we are anyway allowed to throw in new basic function symbols whenever needed. However, one can prove that completeness for PPT can be achieved with a very minimal set of basic functions symbols only including cyclic shift functions on strings and functions testing the value of the first bit in a string. Noticeably, soundness and completeness as presented above scale to *second-order* PPT, namely a notion of probabilistic polynomial time function accessing an oracle. This is quite relevant in our setting, given our emphasis on cryptographic constructions, and the fact that adversaries for some of those have oracle access to the underlying primitive.

2 Computational Indistinguishability

In this section, we define logical relations which we show to be sound for computational indistinguishability and which we will use in Section 3 to prove security of the private key encryption scheme induced by a pseudorandom function. Our approach is to first define a *logical metric* on terms from which we derive the indistinguishability logical relation containing terms whose distance is negligible with respect to this metric.

Term Relations In Section 1, we have considered indexed families $\Lambda^\Theta(\Gamma; A)$ and $\mathcal{V}(\Gamma; A)$ of sets containing terms that are closed for the security parameter

variable i. On the other hand, computational indistinguishability, which is the main focus of this paper, is a relation between terms where the security parameter is a free variable and can be instantiated for every positive natural number n.

For a variable context Γ, a location context Θ and a type A, we let $\Lambda_o^\Theta(\Gamma; A)$ and $\mathcal{V}_o(\Gamma; A)$ be the sets of derivable computation terms and values respectively which are open for the security parameter variable i:

$$\Lambda_o^\Theta(\Gamma; A) := \{M \in \Lambda \mid \Gamma; \Theta \vdash M : A\} \qquad \mathcal{V}_o(\Gamma; A) := \{V \in \mathcal{V} \mid \Gamma \vdash V : A\}.$$

Lemma 2. *The following rules are derivable for all $n \geq 1$:*

$$\frac{\Gamma; \Theta \vdash M : A}{\Gamma n; \Theta n \vdash Mn : An} \qquad \frac{\Gamma \vdash V : A}{\Gamma n \vdash Vn : An}$$

It implies that if M is in $\Lambda_o^\Theta(\Gamma; A)$, then Mn is in $\Lambda_n^\Theta(\Gamma; A)$ for all $n \geq 1$ and a similar statement holds for values in $\mathcal{V}_o(\Gamma; A)$.

Definition 1. *An* open term relation $\mathcal{R}$ *is an indexed family of pairs of relations $\{(\mathcal{RC}^\Theta(\Gamma; A), \mathcal{RV}(\Gamma; A))\}_{\Gamma,\Theta,A}$ with*

$$\mathcal{RC}^\Theta(\Gamma; A) \subseteq \Lambda_o^\Theta(\Gamma; A) \times \Lambda_o^\Theta(\Gamma; A) \qquad \mathcal{RV}(\Gamma; A) \subseteq \mathcal{V}_o(\Gamma; A) \times \mathcal{V}_o(\Gamma; A).$$

A closed *(for term variables)* term relation $\mathcal{R}$ *is an indexed family of pairs of relations $\{(\mathcal{RC}^\Theta(A), \mathcal{RV}(A))\}_{\Theta,A}$ with $\mathcal{RC}^\Theta(A) \subseteq \Lambda_o^\Theta(A) \times \Lambda_o^\Theta(A)$ and $\mathcal{RV}(A) \subseteq \mathcal{V}_o(A) \times \mathcal{V}_o(A)$.*

Every open term relation induces a closed term relation by restricting to empty term variable contexts. For the other direction, we use the standard notion of *open extension* of a closed relation via substitutions with positive values.

Contextual Indistinguishability The notion of behavioral equivalence we consider here is *computational indistinguishability* with respect to a polytime adversary represented as a λ**BLL**-context. Recall that a function which grows asymptotically slower than the inverse of any polynomial is called negligible [10]:

Definition 2. *A function $\varepsilon : \mathbb{N} \to \mathbb{R}_+$ is* negligible *if for all $k \in \mathbb{N}$, there exists $N \in \mathbb{N}$ such that for all $n \geq N$, $\varepsilon(n) < \frac{1}{n^k}$.*

Definition 3. *For terms M, N in $\Lambda_o^\Theta(\Gamma; A)$, we say that M and N are* contextually indistinguishable *if for every closing context C such that $C[M]$ and $C[N]$ are in $\Lambda_o^\Xi(\mathbb{B})$ for some reference context Ξ, there exists a negligible function $\varepsilon : \mathbb{N} \to \mathbb{R}_+$ such that for every $n \geq 1$, $e \in \mathrm{St}_{\Xi n}$ and subset $X \subseteq \{\mathbf{t}, \mathbf{f}\} \times \mathrm{St}_{\Xi n}$,*

$$|(\!|C[M]n|\!)_n^{\Xi,\mathbb{B}}(e)(X) - (\!|C[N]n|\!)_n^{\Xi,\mathbb{B}}(e)(X)| \leq \varepsilon(n).$$

We adopt a coinductive characterization of contextual indistinguishability following the approach in [28, 36] in the case of contextual equivalence for applicative bisimilarity. The contextual indistinguishability relation can indeed be alternatively defined as the largest open λ**BLL**-term relation that is both *compatible* and *adequate*. Compatibility means that the relation is closed under contexts: if (M, N) is in $\mathcal{R}$ and C is a context, then $(C[M], C[N])$ is also in $\mathcal{R}$. Adequacy on the other hand depends on the observational behavior we consider, it is typically termination for contextual equivalence or probability of convergence for non-deterministic calculi. In our case, the notion of interest in computational indistinguishability:

Definition 4. *We define an indexed relation* $\approx = \{\approx_\Theta \subseteq \Lambda_o^\Theta(\mathbb{B}) \times \Lambda_o^\Theta(\mathbb{B})\}_\Theta$ *on closed (for term variables) terms of Boolean type which are indistinguishable: a pair* (M, N) *is in the relation* $\approx_\Theta$ *if and only if there exists a negligible function* $\varepsilon : \mathbb{N} \to \mathbb{R}_+$ *such that for every* $n \geq 1$, $e \in \mathrm{St}_{\Theta n}$ *and subset* $X \subseteq \{\mathbf{t}, \mathbf{f}\} \times \mathrm{St}_{\Theta n}$,

$$|(\!|Mn|\!)_n^{\Theta,\mathbb{B}}(e)(X) - (\!|Nn|\!)_n^{\Theta,\mathbb{B}}(e)(X)| \leq \varepsilon(n).$$

We call an open λ**BLL**-term relation $\mathcal{R}$ *adequate* if it is included in $\approx$ for closed Boolean terms, *i.e.* $\mathcal{R}^\Theta(\mathbb{B}) \subseteq \approx_\Theta$ for all reference contexts Θ and we obtain that the predicate of adequacy on open λ**BLL**-term relations is closed under countable unions and relational composition.

Lemma 3. *Contextual indistinguishability is the largest adequate compatible open* λ**BLL***-relation and we denote it by* $\sim$.

This coinductive characterization provides a useful proof principle to show soundness: any open relation $\mathcal{R}$ that is both compatible and adequate must be included in $\sim$ and is therefore *sound* for contextual indistinguishability (*i.e.* any pair of terms (M, N) in $\mathcal{R}$ are contextually indistinguishable).

Background on Metrics

Weighted Relations For metric reasoning on λ**BLL**, we consider distances valued in the unit real interval $[0, 1]$ equipped with the operation of *truncated addition* $x \oplus y := \min\{1, x + y\}$ for $x, y \in [0, 1]$. We have in particular that $1 = 1 \oplus 1$ which has a direct correspondence with the fact that values of ground type are arbitrarily duplicable in λ**BLL** (see Remark 1).

Recall that a (unital) *quantale* is a tuple $(\mathcal{Q}, \leq, \otimes, 1)$ where $(\mathcal{Q}, \leq)$ is a complete lattice, $(\mathcal{Q}, \otimes, 1)$ is a monoid and $\otimes$ distributes over arbitrary joins [46]. The unit interval with the opposite of the natural order (the natural order is defined as: $x \leq y$ if and only if there exists z such that $x \oplus z = y$) can be equipped with a quantale structure $\mathcal{L} = ([0, 1], \geq, \oplus, 0)$, called the *Łukasiewicz quantale*.

For sets X and Y, an $\mathcal{L}$*-weighted relation* $R : X \nrightarrow Y$ from X to Y consists of a function $X \times Y \to [0, 1]$. They form a category, which we denote by $\mathbf{Rel}_\mathcal{L}$, where the identity $\mathrm{id}_X : X \nrightarrow X$ maps a pair (x, y) to 0 if $x = y$ and 1 otherwise. The composite of two relations $R : X \nrightarrow Y$ and $S : Y \nrightarrow Z$ is the relation

$S \circ R : X \nrightarrow Z$ mapping a pair (x, z) to $\inf_{y \in Y} R(x, y) \oplus S(y, z)$. The category $\mathbf{Rel}_{\mathcal{L}}$ can be equipped with a *dual* (transpose) operation mapping a relation $R : X \nrightarrow Y$ to the relation $R^{\mathrm{op}} : Y \nrightarrow X$ which simply maps (y, x) to $R(x, y)$. Any function $f : X \to Y$, induces a $\mathcal{L}$-relation via its graph $\mathbf{gr}(f) : X \nrightarrow Y$ mapping a pair (x, y) to 0 if $f(x) = y$ and 1 otherwise.

Pseudo-metric Spaces If we restrict to the special case of weighted endo-relations $R : X \nrightarrow X$ that are reflexive ($R \leq \mathrm{id}_X$), symmetric ($R = R^{\mathrm{op}}$) and transitive ($R \leq R \circ R$), we obtain the notion of pseudo-metric space:

Definition 5. *A* pseudo-metric space *consists of a pair (X, d_X) where X is a set and d_X is a function from $X \times X \to [0, 1]$ satisfying the following axioms:*

- *reflexivity: for all x in X, $d_X(x, x) = 0$;*
- *symmetry: for all x, y in X, $d_X(x, y) = d(y, x)$;*
- *triangular inequality: for all x, y, z in X, $d_X(x, z) \leq d_X(x, y) \oplus d_X(y, z)$*

If d_X further satisfies the separation axiom (for all x, y, $d_X(x, y) = 0$ implies $x = y$), then (X, d_X) is a metric space.

For the rest of the paper, even if we do not assume that the separation axiom holds, we will just say metric space instead of pseudo-metric space. Note that metric space with the *discrete metric* $\mathrm{disc} : X \times X \to [0, 1]$ mapping a pair (x, y) to 0 if $x = y$ and 1 otherwise corresponds exactly to the identity weighted relation defined above.

Definition 6. *For two metric spaces (X, d_X) and (Y, d_Y), a function $f : X \to Y$ is said to be* non-expansive *if for all x, x' in X, $d_Y(f(x), f(x')) \leq d_X(x, x')$. We denote by* **PMet** *the category of pseudo-metric spaces and non-expansive maps.*

The category **PMet** is equivalent to the category of $\mathcal{L}$-enriched categories and $\mathcal{L}$-enriched functors between them [31]. We recall below some properties of **PMet** which we will use to define the logical metric in the following section, they are all instances of more general statements on quantale-enriched categories and we refer the reader to [31] for a complete account. **PMet** is symmetric monoidal closed with tensor product $(X, d_X) \otimes (Y d_Y)$ given by $(X \times Y, d_{X \otimes Y})$ where for all $x, x' \in X$ and $y, y' \in Y$,

$$d_{X \otimes Y}((x, y), (x', y')) := d_X(x, x') \oplus d_Y(y, y').$$

The unit is given by $\mathbf{1} = (\{\star\}, \mathrm{disc})$ and the linear hom $X \multimap Y$ has underlying set $\mathbf{PMet}(X, Y)$ (the set of non-expansive maps from X to Y) and distance $d_{X \multimap Y}(f, g) := \sup_{x \in X} d_Y(f(x), g(x))$. For any $k \geq 1$ and $x \in [0, 1]$, we define inductively $k \cdot x$ as $1 \cdot x := x$ and $(k + 1) \cdot x := (k \cdot x) \oplus x$. This operation induces a *scaling* operation $k \cdot (X, d_X) := (X, k \cdot d_X)$ on **PMet** which we use to model the graded bang of λ**BLL**. Note that if $f : (X, d_X) \to (Y, d_Y)$ is non-expansive, then $f : (X, k \cdot d_X) \to (Y, k \cdot d_Y)$ is also non-expansive for all $k \in \mathbb{N}$.

Extending Monadic Effects from Sets to Metric Spaces In order to define the logical metric on computation terms, we need to extend the effect monad $\mathbf{T}_\Theta$ defined in Section 1 from sets to metric spaces. To do so, we follow the standard approach of monad extensions from sets to quantale weighted relations [31, 7]. It is well-known that the distribution monad $\mathbf{D}$ (and therefore the monads $\mathbf{T}_\Theta$ as well) on sets only *laxly* extends to weighted relations via *Kantorovich lifting* [34, 12, 8]. The Kantorovich lifting for distributions fits into the more general framework of *Barr extensions* for monads from sets to quantale relations [31].

In this section, we only give the explicit definition of how the Barr lax extension $\overline{\mathbf{T}}_\Theta$ of the effect monad acts on metric spaces and we refer the reader to [22, 51] for more background on lax extensions for weighted relations. The Kantorovich lifting can be formulated in terms of couplings for probability distributions:

Definition 7. *For sets X, Y and distributions $\mu \in \mathbf{D}(X)$, $\psi \in \mathbf{D}(Y)$, a* coupling over μ and ν *is a distribution $\gamma \in \mathbf{D}(X \times Y)$ such that*

$$\forall x \in X, \mu(x) = \sum_{y \in Y} \gamma(x, y) \text{ and } \forall y \in Y, \nu(y) = \sum_{x \in X} \gamma(x, y).$$

We denote by $\Omega(\mu, \nu)$ the set of all couplings over μ and ν.

For a metric space (X, d_X), the Kantorovich lifting of the distance d_X is the distance $\mathbf{K}(d_X)$ on $\mathbf{D}(X)$ mapping distributions $\mu, \nu \in \mathbf{D}(X)$ to

$$\mathbf{K}(d_X)(\mu, \nu) := \inf_{\gamma \in \Omega(\mu,\nu)} \sum_{x_1, x_2 \in X} \gamma(x_1, x_2) \cdot d_X(\mu(x_1), \nu(x_2)).$$

While there are many other possible choices of metrics on distribution spaces besides the Kantorovich distance $\mathbf{K}(d_X)$ [23], it is the smallest among the ones which laxly extends to weighted relations and it also coincides with the *statistical distance* (or *total variation distance*) when d_X is the discrete metric.

Definition 8. *For a set X, the* statistical distance $d_{\text{stat}} : \mathbf{D}(X) \times \mathbf{D}(X) \to [0, 1]$ *maps two distributions $\mu, \nu \in \mathbf{D}(X)$ to*

$$d_{\text{stat}}(\mu, \nu) := \frac{1}{2} \cdot \sum_{x \in X} |\mu(x) - \nu(x)| = \sup_{A \subseteq X} |\mu(A) - \nu(A)|.$$

For a closed (for the security parameter variable) location context Θ, the action of the lax extension $\overline{\mathbf{T}}_\Theta$ on a metric space (X, d_X) is the metric space with underlying set $\mathbf{T}_\Theta(X)$ and distance $\overline{\mathbf{T}}_\Theta(d_X)$ mapping functions $\varphi, \psi : \text{St}_\Theta \to \mathbf{D}(X \times \text{St}_\Theta)$ to

$$\overline{\mathbf{T}}_\Theta(d_X)(\varphi, \psi) := \sup_{e \in \text{St}_\Theta} \mathbf{K}(d_{X \otimes \text{St}})(\varphi(e), \psi(e)). \tag{1}$$

Logical Metric We now have all the ingredients to define a logical metric for λ**BLL**-terms using the lax extension of the monad $\mathbf{T}_\Theta$ to metric spaces.

Definition 9. *We define a family of metrics on closed computations and values indexed by the security parameter:*

$$\mathbf{dV}_n^A : \mathcal{V}_n(A) \times \mathcal{V}_n(A) \to [0,1] \quad \textit{and} \quad \mathbf{dC}_n^{\Theta,A} : \Lambda_n^\Theta(A) \times \Lambda_n^\Theta(A) \to [0,1]$$

by mutual induction on the type A:

$$\mathbf{dV}_n^{\mathbb{S}[p]}(s,s') := \mathrm{disc}_{\mathbb{S}[p(n)]}(s,s') \qquad \mathbf{dV}_n^{\mathbb{B}}(W,W') := \mathrm{disc}_{\mathbb{B}}(W,W')$$

$$\mathbf{dV}_n^{\mathbb{U}}(\star,\star) := \mathrm{disc}_{\mathbb{U}}(\star,\star) = 0 \qquad \mathbf{dV}_n^{!_p^\Theta A}(!M,!N) := \oplus_{p(n)} \mathbf{dC}_n^{\Theta,A}(M,N)$$

$$\mathbf{dV}_n^{P \otimes Q}(\langle U, V\rangle, \langle U', V'\rangle) := \mathbf{dV}_n^P(U,U') \oplus \mathbf{dV}_n^Q(V,V')$$

$$\mathbf{dV}_n^{P \multimap_\Theta A}(\lambda x.M, \lambda y.N) := \sup_{V \in \mathcal{V}_n(P)} \mathbf{dC}_n^{\Theta,A}(M[V/x], N[V/y])$$

$$\mathbf{dC}_n^{\Theta,A}(M,N) := \overline{\mathbf{T}}_\Theta(\mathbf{dV}_n^A)((\!|M|\!)_n^{\Theta,A}, (\!|N|\!)_n^{\Theta,A})$$

where disc *denotes the discrete metric and $\overline{\mathbf{T}}_\Theta$ is the lax extension of the functor $\mathbf{T}_\Theta : \mathbf{Set} \to \mathbf{Set}$ defined in (1).*

In our setting, the metric version of the fundamental lemma states that substitution by positive value terms is a non-expansive operation:

Lemma 4 (Fundamental Lemma for Logical Metrics). *For a context $\Gamma = x_1 : P_1, \ldots, x_m : P_m$ and a term M in $\Lambda_n^\Theta(\Gamma; A)$ with $n \geq 1$, for every closed positive values $Z_j, Z_j' \in \mathcal{V}_n(P_j)$ with $1 \leq j \leq m$, we have*

$$\mathbf{dC}_n^{\Theta,A}(M\rho, M\rho') \leq \bigoplus_{1 \leq j \leq m} \mathbf{dV}_n^{P_j}(Z_j, Z_j')$$

where $\rho := [Z_1/x_1, \ldots, Z_m/x_m]$ and $\rho' := [Z_1'/x_1, \ldots, Z_m'/x_m]$. A similar statement holds for open value terms in $\mathcal{V}_n(\Gamma; A)$.

Remark 1. A key ingredient in the proof of the fundamental lemma is the equality $\mathbf{dV}_n^{P \boxplus Q}(V,W) = \mathbf{dV}_n^P(V,W) \oplus \mathbf{dV}_n^Q(V,W)$ for closed values V, W, it allows to keep a precise track of how distances are amplified when contexts are added $\Gamma \boxplus \Delta$ in rules such as `let` or $\otimes$ for example. We can see here that the main motivation behind using truncated addition $\oplus$ in our setting is that it allows for the additional flexibility of having ground types being duplicable without loosing the ability to measure distances for higher types: if P and Q are equal to some ground type G, and $V \neq W$, then the equality above indeed rewrites to $1 = 1 \oplus 1$ which would not be possible if we had considered for example the Lawvere quantale with regular addition instead of the Łukasiewicz quantale with truncated addition.

Indistinguishability Logical Relation We now define a closed (for term variables) λ**BLL**-relation $\mathbf{Ind} = (\mathbf{IndC}, \mathbf{IndV})$ with

$$\mathbf{IndC}^{\Theta}(A) \subseteq \Lambda_o^{\Theta}(A) \times \Lambda_o^{\Theta}(A) \quad \text{and} \quad \mathbf{IndV}(A) \subseteq \mathcal{V}_o(A) \times \mathcal{V}_o(A).$$

For terms M, N in $\Lambda_o^{\Theta}(A)$, the pair (M, N) is in $\mathbf{IndC}^{\Theta}(A)$ if there exists a negligible function $\varepsilon : \mathbb{N} \to \mathbb{R}_+$ such that for all $n \geq 1$,

$$\mathbf{dC}_n^{\Theta,A}(Mn, Nn) \leq \varepsilon(n).$$

The relation on values $\mathbf{IndV}(A)$ is defined similarly via the logical metric on values $\mathbf{dV}^A$.

The fundamental lemma for the indistinguishability logical relation can now be directly derivable from the non-expansiveness of the logical metric (Lemma 4) and basic closure properties of negligible functions:

Lemma 5. *For a variable context $\Gamma = x_1 : P_1, \ldots, x_m : P_m$ and closed positive values (Z_k, Z'_k) in $\mathbf{IndV}(P_k)$ with $1 \leq k \leq m$, we have for all $M \in \Lambda_o^{\Theta}(\Gamma; A)$ and $U \in \mathcal{V}_o(\Gamma; A)$,*

$$(M\rho, M\rho') \in \mathbf{IndC}^{\Theta}(A) \text{ and } (U\rho, U\rho') \in \mathbf{IndV}(A)$$

where $\rho := [Z_1/x_1, \ldots, Z_m/x_m]$ and $\rho' := [Z'_1/x_1, \ldots, Z'_m/x_m]$. In particular, for a closed term $M \in \Lambda^{\Theta}(A)$, we have $(M, M) \in \mathbf{IndC}^{\Theta}(A)$.

Theorem 2. *The open extension of* **Ind** *is adequate and compatible.*

Since the contextual indistinguishability relation $\sim$ is the largest compatible adequate relation, it contains **Ind** by Lemma 3, which implies that **Ind** is *sound* for contextual indistinguishability. Full abstraction on the other hand is not possible within our framework: since base types are equipped with the discrete metric whose Kantorovich lifting coincides with statistical distance, we cannot hope to capture the whole contextual indistinguishability relation as it is well-known that statistical closeness is strictly included in computational indistinguishability (*e.g.* Proposition 3.2.3 in [26]).

3 Proving Encryption Scheme Secure Equationally

This section is devoted to the presentation of a game-based proof [48] of security against active attacks for the encryption scheme Π_F induced by any pseudorandom function F. The proof is rather standard and a less formal version of it can be found in many cryptography textbooks (see, *e.g.*, [35]). Following the advice of the anonymous reviewers, we are keeping the presentation as self-contained as possible.

Pseudorandom Functions and Private-key Encryption Schemes A *pseudorandom function* [35] is a function computed by any deterministic polytime algorithm taking two strings in input, and producing a string as output, in such a way that when the first of the two parameters is picked at random, the unary function obtained through currying is indistinguishable from a random one, all this to the eyes of adversaries working in probabilistic polynomial time. The notion of a pseudorandom function is closely related to that of a secure block-cipher.

Private key encryption schemes [35], instead, are triples of algorithms in the form (Gen, Enc, Dec), where Gen is responsible for generating a private key at random, Enc is responsible for turning a message into a ciphertext and Dec is responsible for turning a ciphertext into a message. Both Enc and Dec make essential use of a private shared key. One way to construct private key encryption schemes is by way of pseudorandom functions: given one such function F, the scheme Π_F is such that Enc encrypts a message m as the pair $(r, F_k(r) \oplus m)$, where r is a random string generated on the fly and $\oplus$ is the bitwise exclusive-or operator. The algorithm Gen, instead, simply returns a string picked uniformly at random between those whose length is equal to that of the input. When written down as λ**BLL** terms, the algorithms Enc for encryption and Gen for key generation have the types in Figure 4.

Defining Security The security of any encryption scheme, and of Π_F in particular, is defined on the basis of a so-called *cryptographic experiment*, which following [35] we call $PrivKCPA^F$. Such an experiment allows the scheme Π_F and a generic adversary Adv to interact. The experiment proceeds by first allowing Adv the possibility of generating two distinct messages m_0 and m_1, then encoding m_b (where b is picked at random) with a fresh key k, passing the obtained ciphertext c to Adv, and asking it to determine which one between m_0 and m_1 the ciphertext c corresponds to. The experiment $PrivKCPA^F$ then returns 1 if and only if Adv succeeds in this task. In doing all this, the adversary is *active*, i.e. it has the possibility of accessing an oracle for $Enc_k(\cdot)$. Consequently, the adversary is naturally modeled as a second-order term, see again Figure 4. Obviously, how Adv works internally is not known, but the considerations in Section 1 allow us to conclude that all PPT functions of that type can be encoded in λ**BLL**. The security of Π_F can be expressed as the fact that for every such Adv, the probability that $PrivKCPA^F$ returns 1 is at most $\frac{1}{2} + \varepsilon(n)$, where ε is a negligible function. This depends, in an essential way, on the fact that the function F is indeed pseudorandom.

Proving Security How is the security of Π_F actually *proved*? In fact, the proof is, like most cryptographic proofs, done *by reduction*. In other words, it proceeds contrapositively, turning any hypothetical adversary Adv for Π_F into a *distinguisher* D for F (namely an algorithm designed to distinguish F from a truly random function). If D can be proved successful whenever Adv is successful,

$$\vdash \mathit{Gen} : \mathbb{U} \multimap \mathbb{S}[p_k]$$
$$\vdash \mathit{Enc} : \mathbb{S}[p_k] \otimes \mathbb{S}[p_m] \multimap \mathbb{S}[p_c]$$
$$\vdash \mathit{Oracle} : \mathbb{S}[p_k] \multimap \mathbb{S}[p_m] \multimap \mathbb{S}[p_c]$$
$$\vdash \mathit{Adv} : !_q(\mathbb{S}[p_m] \multimap \mathbb{S}[p_c]) \multimap \mathbb{S}[p_m] \otimes \mathbb{S}[p_m] \otimes !_1(\mathbb{S}[p_c] \multimap \mathbb{B})$$
$$\vdash \mathit{PrivKCPA}^F : \mathbb{B}$$
$$\vdash D : !_q(\mathbb{S}[p_m] \multimap \mathbb{S}[p_c]) \multimap \mathbb{B}$$

Fig. 4: Types for Terms in the CPA-security Proof

we can conclude that Π_F is secure whenever F is pseudorandom, both notions being spelled out as the *non-existence* of adversaries of the appropriate kind.

The aforementioned reduction can actually be organized as follows. First of all, we have to define how a distinguisher D can be defined with Adv as a subroutine. The idea is to design D in such a way as to create the right environment around Adv, letting it believe that it is interacting with the experiment $\mathit{PrivKCPA}$, and exploiting its capabilities for the sake of distinguishing F from a random function. In the context of $\lambda\mathbf{BLL}$, the distinguisher D becomes an ordinary term having the type in Figure 4.

Then, we have to form two instances on D namely that interacting with the pseudorandom function F, which we indicate as D^F, and that interacting with a genuinely random function f, indicated as D^f. Both F and f can be assumed to be terms of $\lambda\mathbf{BLL}$, but while the former can be taken as a term which does not use any reference, the latter can only be captured by a stateful computation — one cannot hope to pick uniformly at random a function on n-bit strings in polynomial time in n without the help of some bookkeeping mechanism. The latter will actually be implemented as a reference, call it *ledger*, whose purpose is to keep track of the previous strings on which the function has been queried, so that randomness can be generated *only when needed*. Since the type of f reflects the presence of *ledger*, the type of D is to be updated accordingly, as we are going to describe in the next paragraph.

For an arbitrary type A and a location context Ξ whose variables do not occur in A, we define $A \cdot \Xi$ inductively as follows:

$$G \cdot \Xi := G \qquad (P \otimes Q) \cdot \Xi := (P \cdot \Xi) \otimes (Q \cdot \Xi)$$
$$(!_p^{\Theta} A) \cdot \Xi := !_p^{\Theta,\Xi}(A \cdot \Xi) \qquad (P \xrightarrow{\Theta} A) \cdot \Xi := (P \cdot \Xi) \xrightarrow{\Theta,\Xi} (A \cdot \Xi)$$

This operation can be easily extended to term variable contexts as follows:

$$\varnothing \cdot \Xi := \varnothing \qquad (\Gamma, x : P) \cdot \Xi := \Gamma \cdot \Xi, x : P \cdot \Xi$$

Lemma 6. *For every derivable judgments $\Gamma; \Theta \vdash M : A$ and $\Gamma \vdash V : A$ and every location context Ξ whose variables do not occur in Γ, Θ and A, we obtain that the jugdments $\Gamma \cdot \Xi; \Theta, \Xi \vdash M : A \cdot \Xi$ and $\Gamma \cdot \Xi \vdash V : A \cdot \Xi$ are also derivable.*

We also have to define an encryption scheme Π_f which is structurally identical to Π_F, but which works with truly random functions (as opposed to *pseudorandom* ones) as keys. Accordingly, one can form a variation $PrivKCPA^f$ on $PrivKCPA^F$.

Now, the security of Π^F becomes the equation $PrivKCPA^F \asymp \texttt{flipcoin}$, whereas `flipcoin` is the term, of boolean type, returning each possible result with probability $\frac{1}{2}$, while $\asymp$ is a relation coarser than $\approx$ defined by observing, through marginals, only the actual boolean value returned by the computation, without looking at the underlying store. The aforementioned equation can be proved under the hypothesis that F is pseudorandom, and this last condition also becomes an equation. This time, however, the terms to be compared are D^F and D^f.

The security proof then proceeds by contraposition, as explained schematically in Figure 5: from the negation of the thesis, the negation of the hypothesis is derived and this is done by proving that both on the right and on the left sides of the diagram it is possible to *link* the terms through the relation $\approx$. In this context, it is clear that the use of observational indistinguishability and logical relations becomes useful. In particular, a number of equations can be used, as discussed in the following section. Noticeably, all of them can be proved sound for observational indistinguishability through logical relations.

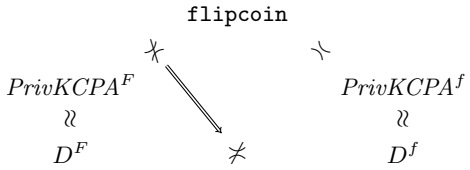

Fig. 5: Outline Proof of Security

Examples of Indistinguishability Equivalences Term equations and behavioral equivalences have been generalized to the setting of metric spaces via quantitative (in)equations $M =_\varepsilon N$ [39] and behavioral metrics [8, 22]. In our case, terms $M = \{M_n\}_n$ are families indexed by the security parameter and compared to the previous approaches, the contextual indistinguishability relation $M \sim N$ intuitively means that for every n, $M_n =_{\varepsilon(n)} N_n$ for some negligible function ε.

We present below two typical examples of pairs of terms which are in the contextual indistinguishability relation and are used for proving security properties. A first example is the pair

$$\texttt{return}\,\mathbf{f} \sim \texttt{let}\, y = M\, \texttt{in}\, \texttt{let}\, x = \texttt{random}\, \texttt{in}\, \texttt{equal}(x, y) \qquad \text{(randF)}$$

where M is any computation term $\vdash M : \mathbb{S}[i]$. Intuitively, it means that for a given binary string represented here by M, testing equality with a randomly generated string returns true with a negligible probability.

The function symbol `random` is interpreted as the *uniform* distribution in $\mathbf{D}(\{0,1\}^n)$ given by $[\![\texttt{random}]\!]_n : s \mapsto \frac{1}{2^n}$ for $n \geq 1$. The function symbol `equal`

is interpreted as the function mapping a pair of strings $(s_1, s_2) \in \{0,1\}^n$ to $\delta_{\mathbf{t}}$ if $s_1 = s_2$ and to $\delta_{\mathbf{f}}$ otherwise for all $n \geq 1$. Therefore, we obtain that for all $n \geq 1$, the final (big step) semantics of $\mathtt{return}\,\mathbf{f}$ is given by $(\!|\mathtt{return}\,\mathbf{f}|\!)_n = \delta_{\mathbf{f}} \in \mathbf{D}(\{\mathbf{t}, \mathbf{f}\})$ and for $N := \mathtt{let}\, y = M\, \mathtt{in}\, \mathtt{let}\, x = \mathtt{random}\, \mathtt{in}\, \mathtt{equal}(x, y)$, we have:

$$(\!|N|\!)_n = \sum_{s \in \{0,1\}^n} (\!|M|\!)_n(s) \cdot \left(\frac{1}{2^n} \delta_{\mathbf{t}} + \left(1 - \frac{1}{2^n}\right) \delta_{\mathbf{f}} \right) = \frac{1}{2^n} \delta_{\mathbf{t}} + \left(1 - \frac{1}{2^n}\right) \delta_{\mathbf{f}}$$

We can easily see that $\mathtt{return}\,\mathbf{f}$ and N are *not* contextually equivalent since they reduce to different distributions. They are however contextually indistinguishable which would be quite difficult to prove directly since it requires to quantify over all closing contexts which can possibly copy their argument.

Instead, we use the logical metric defined in Section 2, which here coincides with statistical distance (Definition 8) and obtain that for all $n \geq 1$, $\mathbf{dC}_n^{\mathbb{B}}(\mathtt{return}\,\mathbf{f}, N) = \frac{1}{2^n}$. Since the function $\varepsilon : n \mapsto \frac{1}{2^n}$ is negligible, the pair $(\mathtt{return}\,\mathbf{f}, N)$ is in $\mathbf{IndC}(\mathbb{B})$ and we obtain $\mathtt{return}\,\mathbf{f} \sim N$ by soundness (Theorem 2).

Another required equation states that sampling a random string is equivalent to random sampling followed by a performing a $\mathtt{xor}$ operation by a fixed string (represented by a computation term $\vdash M : \mathbb{S}[i]$):

$$\mathtt{random} \sim \mathtt{let}\, y = M\, \mathtt{in}\, \mathtt{let}\, x = \mathtt{random}\, \mathtt{in}\, \mathtt{xor}(x, y) \qquad \text{(randXOR)}$$

The equation above is an example of *Kleene equivalence* as the two terms $\mathtt{random}$ and $P := \mathtt{let}\, y = M\, \mathtt{in}\, \mathtt{let}\, x = \mathtt{random}\, \mathtt{in}\, \mathtt{xor}(x, y)$ have the same final (big step) semantics. The function symbol $\mathtt{xor}$ is interpreted by the standard *exclusive-or* function on binary strings mapping a pair (s_1, s_2) to $\delta_{\mathtt{xor}(s_1, s_2)}$. The final semantics of $(\!|P|\!)_n$ for $n \geq 1$ is therefore given by:

$$(\!|P|\!)_n(s) = \sum_{s_2} (\!|M|\!)_n(s_2) \cdot \left(\sum_{s_1} \frac{1}{2^n} \delta_{\mathtt{xor}(s_1, s_2)}(s) \right) = \sum_{s_2} (\!|M|\!)_n(s_2) \cdot \frac{1}{2^n} = \frac{1}{2^n}$$

where the penultimate equality holds since $\delta_{\mathtt{xor}(s_1, s_2)}(s) = 1$ if $s = \mathtt{xor}(s_1, s_2)$ (or equivalently $s_1 = \mathtt{xor}(s, s_2)$) and $\delta_{\mathtt{xor}(s_1, s_2)}(s) = 0$ otherwise. Since $(\!|P|\!)_n = (\!|\mathtt{random}|\!)_n$, the distance between the two terms is therefore equal to 0 for the logical metric and we can conclude immediately that they are in particular contextually indistinguishable. We can prove more generally that if two closed terms are Kleene equivalent, then they are contextually indistinguishable.

Related Work

Although the literature regarding formal methods for the security analysis of protocols and primitives is much more abundant in the symbolic model than in the computational one, it certainly cannot be said that the latter has not been the subject of attention by the research community. The work on probabilistic

relational Hoare logic which gave rise to the `EasyCrypt` tool [13], must certainly be mentioned. The result of Bana and Comon Lundt on inconsistency proofs as security proofs [9], which in turn gave rise to the `Squirrel` tool [6], is another pertinent example. In both cases, the model provides for the possibility of higher-order constructions, which however are not fully-fledged. In particular, managing complexity aspects and higher-order functions at the same time turns out to be hard.

This last direction is the one followed by the work on CSLR and its formalization [44]. In this case we find ourselves faced with a λ-calculus for polynomial time and its application to the study of cryptographic primitives. There are two differences with this work. First of all, the greater expressiveness of λ**BLL** allows to capture PPT even for second-order constructions. Furthermore, the logical relations introduced here effectively give rise to a notion of metric, while in CSLR the underlying equational theories are exact, even though a notion of observational equivalence similar to ours has been introduced.

Another attempt that goes in the same direction as ours is the work by Mitchell et al. [41], who introduced a process algebra in the style of Milner's CCS capable of modeling cryptographic protocols. Unlike ours, the resulting calculus is concurrent and this gives rise to a series of complications. Once again, despite the underlying notion of observational equivalence being approximate and therefore adhering to computational indistinguishability, the proposed notion of bisimulation is exact and as such much finer.

Logical relations [50, 43] are a powerful tool for relational reasoning about higher-order terms. They are known to work well in calculi with effects and in particular in presence of probabilistic choice effects [29, 11, 5]. It is also known that metric versions of logical relations can be given, and that they are useful for sensitivity analysis [45, 14]. The possibility of applying logical relations to calculi such as the cryptographic λ-calculus is well-known [30], but the underlying calculus turns out to be fundamentally different from ours, being in the tradition of the symbolic model and abstracting away from probabilistic effects and complexity constraints.

Conclusion

This work shows how an approximate form of logical relation can be defined and proved sound for computational indistinguishability in a higher order λ-calculus with probabilistic effects and references. This allows cryptographic proofs to be carried out in a purely equational way by justifying the equations used.

Possible topics for future work include the transition to a logic in the style of higher-order logic, this way enabling the combination of relational and logical reasoning, in the sense of the work of Aguirre and co-authors [4].

Acknowledgments. The first two authors are partially supported by the MUR FARE project CAFFEINE, and by the ANR PRC project PPS (ANR-19-CE48-0014). The third author is partially supported by Fondation CFM.

References

1. Abadi, M.: Security Protocols: Principles and Calculi. In: Aldini, A., Gorrieri, R. (eds.) Foundations of security analysis and design IV, pp. 1–23 (2006). https://doi.org/10.1007/978-3-540-74810-6_1
2. Abramsky, S.: The lazy lambda calculus. In: Research Topics in Functional Programming, p. 65–116. Addison-Wesley Longman Publishing Co., Inc. (1990)
3. Accattoli, B., Dal Lago, U.: (Leftmost-Outermost) Beta Reduction is Invariant, Indeed. Logical Methods in Computer Science **12**(1) (2016). https://doi.org/10.2168/LMCS-12(1:4)2016
4. Aguirre, A., Barthe, G., Gaboardi, M., Garg, D., Katsumata, S.y., Sato, T.: Higher-order probabilistic adversarial computations: categorical semantics and program logics. In: Proc. of ICFP 2021. pp. 1–30. ACM (2021). https://doi.org/10.1145/3473598
5. Aguirre, A., Birkedal, L.: Step-Indexed Logical Relations for Countable Nondeterminism and Probabilistic Choice. In: Proc. of POPL 2023. vol. 7, pp. 33–60. ACM (2023). https://doi.org/10.1145/3571195
6. Baelde, D., Jacomme, C.: The Squirrel Prover and its Logic. ACM SIGLOG News **11**(2), 62–83 (2024). https://doi.org/10.1145/3665453.3665461
7. Balan, A., Kurz, A., Velebil, J.: Extending set functors to generalised metric spaces. Logical Methods in Computer Science **15**(1) (2019). https://doi.org/10.23638/LMCS-15(1:5)2019
8. Baldan, P., Bonchi, F., Kerstan, H., König, B.: Coalgebraic Behavioral Metrics. Logical Methods in Computer Science **14**(3) (2018). https://doi.org/10.23638/LMCS-14(3:20)2018
9. Bana, G., Comon-Lundh, H.: A Computationally Complete Symbolic Attacker for Equivalence Properties. In: Proc. of CCS 2014. pp. 609–620 (2014). https://doi.org/10.1145/2660267.2660276
10. Bellare, M.: A Note on Negligible Functions. Journal of Cryptology **15**(4), 271–284 (2002). https://doi.org/10.1007/s00145-002-0116-x
11. Bizjak, A., Birkedal, L.: Step-Indexed Logical Relations for Probability. In: Proc. of FoSSaCS 2015. pp. 279–294. Springer (2015). https://doi.org/10.1007/978-3-662-46678-0_18
12. van Breugel, F.: The metric monad for probabilistic nondeterminism. Draft available at http://www. cse. yorku. ca/ franck/research/drafts/monad. pdf (2005)
13. Canetti, R., Stoughton, A., Varia, M.: EasyUC: Using EasyCrypt to Mechanize Proofs of Universally Composable Security. In: Proc. of CSF 2019. pp. 167–183. IEEE (2019). https://doi.org/10.1109/CSF.2019.00019
14. Dal Lago, U., Gavazzo, F.: A relational theory of effects and coeffects. In: Proc. of POPL 2022. vol. 6, pp. 1–28. ACM (2022). https://doi.org/10.1145/3498692
15. Dal Lago, U., Giusti, G.: On Session Typing, Probabilistic Polynomial Time, and Cryptographic Experiments. In: Proc. of CONCUR 2022. vol. 243, pp. 37:1–37:18 (2022). https://doi.org/10.4230/LIPICS.CONCUR.2022.37
16. Dal Lago, U., Martini, S.: On Constructor Rewrite Systems and the Lambda Calculus. Logical Methods in Computer Science (2012). https://doi.org/10.1007/978-3-642-02930-1_14
17. Dal Lago, U., Petit, B.: Linear dependent types in a call-by-value scenario. In: Proc. of PPDP 2012. pp. 115–126. ACM (2012). https://doi.org/10.1145/2370776.2370792

18. Dal Lago, U., Zuppiroli, S., Gabbrielli, M.: Probabilistic Recursion Theory and Implicit Computational Complexity. Scientific Annals of Computer Science pp. 177–216 (2014). https://doi.org/10.7561/SACS.2014.2.177
19. Dolev, D., Yao, A.: On the security of public key protocols. IEEE Transactions on information theory **29**(2), 198–207 (1983). https://doi.org/10.1109/TIT.1983.1056650
20. Ehrhard, T., Tasson, C.: Probabilistic call by push value. Logical Methods in Computer Science **15**(1) (2019). https://doi.org/10.23638/LMCS-15(1:3)2019
21. Fiore, M., Abadi, M.: Computing symbolic models for verifying cryptographic protocols. In: Proc. of CSFW 2001. pp. 160–173. IEEE (2001). https://doi.org/10.1109/CSFW.2001.930144
22. Gavazzo, F.: Quantitative Behavioural Reasoning for Higher-order Effectful Programs: Applicative Distances. In: Proc. of LICS 2018. pp. 452–461. ACM (2018). https://doi.org/10.1145/3209108.3209149
23. Gibbs, A.L., Su, F.E.: On Choosing and Bounding Probability Metrics. International Statistical Review **70**(3), 419–435 (2002). https://doi.org/10.2307/1403865
24. Gifford, D.K., Lucassen, J.M.: Integrating functional and imperative programming. In: Proc. of the 1986 ACM Conference on LISP and Functional Programming. pp. 28–38. ACM (1986). https://doi.org/10.1145/319838.319848
25. Girard, J.Y., Scedrov, A., Scott, P.J.: Bounded linear logic: a modular approach to polynomial-time computability. Theoretical Computer Science **97**(1), 1–66 (1992). https://doi.org/10.1016/0304-3975(92)90386-T
26. Goldreich, O.: Foundations of cryptography. Vol. 1: Basic tools. Cambridge Univ. Press, Cambridge (2007)
27. Goldwasser, S., Micali, S.: Probabilistic encryption. Journal of Computer and System Sciences **28**(2), 270–299 (1984). https://doi.org/10.1016/0022-0000(84)90070-9
28. Gordon, A.D.: Operational equivalences for untyped and polymorphic object calculi. In: Higher-Order Operational Techniques in Semantics, Publications of the Newton Institute, pp. 9–54. Cambridge University Press (1998)
29. Goubault-Larrecq, J., Lasota, S., Nowak, D.: Logical Relations for Monadic Types. Mathematical Structures in Computer Science pp. 1169–1217 (2008). https://doi.org/10.1007/3-540-45793-3_37
30. Goubault-Larrecq, J., Lasota, S., Nowak, D., Zhang, Y.: Complete Lax Logical Relations for Cryptographic Lambda-Calculi. In: Proc. of CSL 2004. pp. 400–414. Springer (2004). https://doi.org/10.1007/978-3-540-30124-0_31
31. Hofmann, D., Seal, G.J., Tholen, W.: Monoidal Topology: A Categorical Approach to Order, Metric, and Topology. Cambridge University Press (2014)
32. Hyland, M., Plotkin, G., Power, J.: Combining effects: Sum and tensor. Theoretical Computer Science **357**(1), 70–99 (2006). https://doi.org/10.1016/j.tcs.2006.03.013
33. Impagliazzo, R., Kapron, B.M.: Logics for reasoning about cryptographic constructions. Journal of Computer and System Sciences **72**(2), 286–320 (2006). https://doi.org/10.1016/j.jcss.2005.06.008
34. Kantorovich, L.V.: On the Translocation of Masses. Journal of Mathematical Sciences **133**(4), 1381–1382 (2006). https://doi.org/10.1007/s10958-006-0049-2
35. Katz, J., Lindell, Y.: Introduction to modern cryptography. CRC press (2020)
36. Lassen, S.B.: Relational reasoning about contexts. In: Higher-Order Operational Techniques in Semantics, Publications of the Newton Institute, vol. 91, pp. 91–136. Cambridge University Press (1998)
37. Leroy, X., Grall, H.: Coinductive big-step operational semantics. Information and Computation **207**(2), 284–304 (2009). https://doi.org/10.1016/j.ic.2007.12.004

38. Levy, P.B.: Call-by-push-value: A Functional/imperative Synthesis, vol. 2. Springer Science & Business Media (2012)
39. Mardare, R., Panangaden, P., Plotkin, G.: Quantitative Algebraic Reasoning. In: Proc. of LICS 2016. pp. 700–709. ACM (2016). https://doi.org/10.1145/2933575.2934518
40. Mitchell, J., Mitchell, M., Scedrov, A.: A linguistic characterization of bounded oracle computation and probabilistic polynomial time. In: Proc. of SFCS 1998. pp. 725–733. IEEE (1998). https://doi.org/10.1109/SFCS.1998.743523
41. Mitchell, J.C., Ramanathan, A., Scedrov, A., Teague, V.: A probabilistic polynomial-time process calculus for the analysis of cryptographic protocols. Theoretical Computer Science **353**(1-3) (2006). https://doi.org/10.1016/j.tcs.2005.10.044
42. Mitchell, J.C.: Multiset Rewriting and Security Protocol Analysis. In: Proc. of RTA 2002. pp. 19–22. Springer (2002). https://doi.org/10.1007/3-540-45610-4_2
43. Mitchell, J.C., Scedrov, A.: Notes on sconing and relators. In: Proc. of CSL 1992. pp. 352–378. Springer (1992). https://doi.org/10.1007/3-540-56992-8_21
44. Nowak, D., Zhang, Y.: A Calculus for Game-Based Security Proofs. In: Proc. of ProvSec 2010. Springer (2010). https://doi.org/10.1007/978-3-642-16280-0_3
45. Reed, J., Pierce, B.C.: Distance makes the types grow stronger: a calculus for differential privacy. In: Proc. of ICFP 2010. pp. 157–168. ACM (2010). https://doi.org/10.1145/1863543.1863568
46. Rosenthal, K.I.: Quantales and their applications. Wiley (1990)
47. Sangiorgi, D., Rutten, J.: Advanced topics in bisimulation and coinduction. Cambridge University Press (2011)
48. Shoup, V.: Sequences of games: a tool for taming complexity in security proofs. IACR Cryptol. ePrint Arch. p. 332 (2004), `http://eprint.iacr.org/2004/332`
49. Statman, R.: Logical relations and the typed λ-calculus. Information and control **65**(2-3), 85–97 (1985). https://doi.org/10.1016/S0019-9958(85)80001-2
50. Tait, W.W.: Intensional interpretations of functionals of finite type I. The journal of symbolic logic **32**(2), 198–212 (1967). https://doi.org/10.2307/2271658
51. Wild, P., Schröder, L.: Characteristic Logics for Behavioural Hemimetrics via Fuzzy Lax Extensions. Logical Methods in Computer Science (2022). https://doi.org/10.46298/lmcs-18(2:19)2022