Differential logical relations, Part I: The simply-typed case

Differential Logical Relations

Part I: The Simply-Typed Case

Ugo Dal Lago
University of Bologna, Italy
INRIA Sophia Antipolis, France
ugo.dallago@unibo.it

Francesco Gavazzo
IMDEA Software, Spain
francesco.gavazzo@gmail.com

Akira Yoshimizu
INRIA Sophia Antipolis, France
akiray@bp.iij4u.or.jp

Abstract

We introduce a new form of logical relation which, in the spirit of metric relations, allows us to assign each pair of programs a quantity measuring their distance, rather than a boolean value standing for their being equivalent. The novelty of differential logical relations consists in measuring the distance between terms not (necessarily) by a numerical value, but by a mathematical object which somehow reflects the interactive complexity, i.e. the type, of the compared terms. We exemplify this concept in the simply-typed lambda-calculus, and show a form of soundness theorem. We also see how ordinary logical relations and metric relations can be seen as instances of differential logical relations. Finally, we show that differential logical relations can be organised in a cartesian closed category, contrarily to metric relations, which are well-known *not* to have such a structure, but only that of a monoidal closed category.

2012 ACM Subject Classification Computability → Lambda calculus; Logic → Equational Logic and Rewriting

Keywords and phrases Logical Relations λ-Calculus Program Equivalence Semantics.

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.XXX

Related Version http://arxiv.org/abs/1904.12137

Funding The authors are partially supported by the ERC Consolidator Grant DLV-818616 DIAPASoN, as well as by the ANR projects 14CE250005 ELICA and 16CE250011 REPAS.

1 Introduction

Modern software systems tend to be heterogeneous and complex, and this is reflected in the analysis methodologies we use to tame their complexity. Indeed, in many cases the only way to go is to make use of compositional kinds of analysis, in which *parts* of a large system can be analysed in isolation, without having to care about the rest of the system, the *environment*. As an example, one could consider a component A and replace it with another (e.g. more efficient) component B without looking at the context $\mathcal{C}$ in which A and B are supposed to operate, see Figure 1. Of course, for this program transformation to be safe, A should be *equivalent* to B or, at least, B should be a *refinement* of A.

Program equivalences and refinements, indeed, are the cruxes of program semantics, and have been investigated in many different programming paradigms. When programs have an interactive behaviour, like in concurrent or higher-order languages, even *defining* a notion of program equivalence is not trivial, while coming out with handy methodologies for *proving* concrete programs to be equivalent can be quite challenging, and has been one of the major

research topics in programming language theory, stimulating the development of techniques like logical relations [23, 20], applicative bisimilarity [1], and to some extent denotational semantics [26, 27] itself.

Coming back to our example, may we say anything about the case in which A and B are *not* equivalent, although behaving very similarly? Is there anything classic program semantics can say about this situation? Actually, the answer is negative: the program transformation turning such an A into B cannot be justified, simply because there is no guarantee about what the possible negative effects that turning A into B could have on the overall system formed by $\mathcal{C}$ and A. There are, however, many cases in which program transformations like the one we just described are indeed of interest, and thus desirable. Many examples can be, for instance, drawn from the field of *approximate computing* [21], in which equivalence-breaking program transformations are considered as beneficial *provided* the overall behaviour of the program is not affected too much by the transformation, while its intensional behaviour, e.g. its performance, is significantly improved.

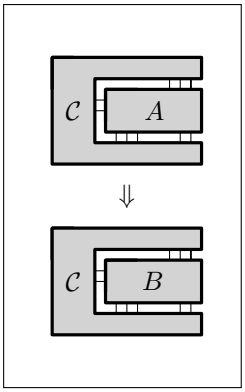

Figure 1 Replacing A with B.

One partial solution to the problem above consists in considering program *metrics* rather than program *equivalences*. This way, any pair of programs are dubbed being at a certain numerical distance rather than being merely equivalent (or not). This, for example, can be useful in the context of differential privacy [24, 6, 32] and has also been studied in the realms of domain theory [13, 5, 14, 16, 4] (see also [28] for an introduction to the subject) and coinduction [30, 29, 15, 9]. The common denominator among all these approaches is that on the one hand, the notion of a congruence, crucial for compositional reasoning, is replaced by the one of a *Lipschitz-continuous* map: any context should not amplify (too much) the distance between any pair of terms, when it is fed with either the former or the latter:

$$\delta(C[M], C[N]) \leq c \cdot \delta(M, N).$$

This enforces compositionality, and naturally leads us to consider metric spaces and Lipschitz functions as the underlying category. As is well known, this is not a cartesian closed category, and thus does *not* form a model of typed λ-calculi, unless one adopts linear type systems, or type systems in which the number of uses of each variable is kept track of, like FUZZ [24]. This somehow limits the compositionality of the metric approach [13, 17].

Even if one considers affine calculi, there are program transformations which are intrinsically unjustifiable in the metric approach. Consider the following two programs of type $REAL \to REAL$

$$M_{SIN} := \lambda x.\mathtt{sin}(x) \qquad\qquad M_{ID} := \lambda x.x.$$

The two terms compute two very different functions on the real numbers, namely the sine trigonometric function and the identity on $\mathbb{R}$, respectively. The euclidean distance $|\sin x - x|$ is unbounded when x ranges over $\mathbb{R}$. As a consequence, comparing M_{SIN} and M_{ID} using the so-called sup metric[1] as it is usually done in metric logical relations [24, 13] and applicative distances [17, 10], we see that their distance is infinite, and that the program transformation turning M_{SIN} into M_{ID} cannot be justified this way, for very good reasons. As highlighted

[1] Recall that given (pseudo)metric spaces (X, d_X), (Y, d_Y) we can give the set Y^X of non-expansive maps between X and Y a (pseudo)metric space structure setting $d_{Y^X}(f, g) = \sup_{x \in X} d_Y(f(x), g(x))$

by Westbrook and Chaudhuri [31], this is not the end of the story, at least if the environment in which M_{SIN} and M_{ID} operate feed either of them *only with* real numbers close to 0. If this is the case, M_{SIN} can be substituted with M_{ID} without affecting *too much* the overall behaviour of the system.

The key insight by Westbrook and Chaudhuri is that justifying program transformations like the one above requires taking the difference $\delta(M_{SIN}, M_{ID})$ between M_{SIN} and M_{ID} not merely as a number, but as a more structured object. What they suggest is to take $\delta(M_{SIN}, M_{ID})$ as *yet another program*, which however describes the difference between M_{SIN} and M_{ID}:

$$\delta(M_{SIN}, M_{ID}) := \lambda x.\lambda\varepsilon.|\sin x - x| + \varepsilon.$$

This reflects the fact that the distance between M_{SIN} and M_{ID}, namely the discrepancy between their output, depends not only on the discrepancy on the input, namely on ε, but also *on the input itself*, namely on x. It both x and ε are close to 0, $\delta(M_{SIN}, M_{ID})$ is itself close to 0.

In this paper, we develop Westbrook and Chaudhuri's ideas, and turn them into a framework of *differential logical relations*. We will do all this in a simply-typed λ-calculus with real numbers as the only base type. Starting from such a minimal calculus has at least two advantages: on the one hand one can talk about meaningful examples like the one above, and on the other hand the induced metatheory is simple enough to highlight the key concepts.

The contributions of this paper can be summarised as follows:

- After introducing our calculus $ST^{\lambda}_{\mathbb{R}}$, we define differential logical relations inductively on types, as ternary relations between pairs of programs and *differences*. The latter are mere set theoretic entities here, and the nature of differences between terms depends on terms' types.
- We prove a soundness theorem for differential logical relations, which allows us to justify compositional reasoning about terms' differences. We also prove a *finite difference theorem*, which stipulates that the distance between two simply-typed λ-terms is finite if mild conditions hold on the underlying set of function symbols.
- We give embeddings of logical and metric relations into differential logical relations. This witnesses that the latter are a generalisation of the former two.
- Finally, we show that generalised metric domains, the mathematical structure underlying differential logical relations, form a cartesian closed category, contrarily to the category of metric spaces, which is well known not to have the same property.

Due to space constraints, many details have to be omitted, but can be found in an Extended Version of this work [12].

2 A Simply-Typed λ-Calculus with Real Numbers

In this section, we introduce a simply-typed λ-calculus in which the only base type is the one of real numbers, and constructs for iteration and conditional are natively available. The choice of this language as the reference calculus in this paper has been made for the sake of simplicity, allowing us to concentrate on the most crucial aspects, at the same time guaranteeing a minimal expressive power.

$$\frac{x:\tau \in \Gamma}{\Gamma \vdash x:\tau} \qquad \frac{}{\Gamma \vdash r : REAL} \qquad \frac{f_n \in \mathcal{F}_n}{\Gamma \vdash f_n : REAL^n \to REAL} \qquad \frac{\Gamma, x:\tau \vdash M : \rho}{\Gamma \vdash \lambda x.M : \tau \to \rho}$$

$$\frac{\Gamma \vdash M : \tau \to \rho \quad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \rho} \qquad \frac{\Gamma \vdash M : \tau \quad \Gamma \vdash N : \rho}{\Gamma \vdash \langle M, N\rangle : \tau \times \rho} \qquad \frac{}{\Gamma \vdash \pi_1 : \tau \times \rho \to \tau} \qquad \frac{}{\Gamma \vdash \pi_2 : \tau \times \rho \to \rho}$$

$$\frac{\Gamma \vdash M : \tau \quad \Gamma \vdash N : \tau}{\Gamma \vdash \texttt{iflz}\ M\ \texttt{else}\ N : REAL \to \tau} \qquad \frac{\Gamma \vdash M : \tau \to \tau \quad \Gamma \vdash N : \tau}{\Gamma \vdash \texttt{iter}\ M\ \texttt{base}\ N : REAL \to \tau}$$

Figure 2 Typing rules for $ST^{\lambda}_{\mathbb{R}}$.

Terms and Types

$ST^{\lambda}_{\mathbb{R}}$ is a typed λ-calculus, so its definition starts by giving the language of *types*, which is defined as follows:

$$\tau, \rho ::= REAL \mid \tau \to \rho \mid \tau \times \rho.$$

The expression τ^n stands for $\underbrace{\tau \times \cdots \times \tau}_{n \text{ times}}$. The set of *terms* is defined as follows:

$$M, N ::= x \mid r \mid f_n \mid \lambda x.M \mid MN \mid \langle M, N\rangle \mid \pi_1 \mid \pi_2 \mid \texttt{iflz}\ M\ \texttt{else}\ N \mid \texttt{iter}\ M\ \texttt{base}\ N$$

where x ranges over a set $\mathbb{V}$ of variables, r ranges over the set $\mathbb{R}$ of real numbers, n is a natural number, and f_n ranges over a set $\mathcal{F}_n$ of total real functions of arity n. We do not make any assumption on $\{\mathcal{F}_n\}_{n\in\mathbb{N}}$, apart from the predecessor $pred_1$ being part of $\mathcal{F}_1$. The family, in particular, could in principle contain non-continuous functions. The expression $\langle M_1, \ldots, M_n\rangle$ is simply a shortcut for $\langle \ldots \langle\langle M_1, M_2\rangle, M_3\rangle \ldots, M_n\rangle$. All constructs are self-explanatory, except for the `iflz` and `iter` operators, which are conditional and iterator combinators, respectively. An *environment* Γ is a set of assignments of types to variables in $\mathbb{V}$ where each variable occurs at most once. A *type judgment* has the form $\Gamma \vdash M : \tau$ where Γ is an environment, M is a term, and τ is a type. Rules for deriving correct typing judgments are in Figure 2, and are standard. The set of terms M for which $\cdot \vdash M : \tau$ is derivable is indicated as $CT(\tau)$.

Call-by-Value Operational Semantics

A static semantics is of course not enough to give meaning to a paradigmatic programming language, the dynamic aspects being captured only once an *operational* semantics is defined. The latter turns out to be very natural. *Values* are defined as follows:

$$V, W ::= r \mid f_n \mid \lambda x.M \mid \langle M, N\rangle \mid \pi_1 \mid \pi_2 \mid \texttt{iflz}\ M\ \texttt{else}\ N \mid \texttt{iter}\ M\ \texttt{base}\ N$$

The set of closed values of type τ is $CV(\tau) \subseteq CT(\tau)$, and the evaluation of $M \in CT(\tau)$ produces a value $V \in CV(\tau)$, as formalised by the rules in Figure 3, through the judgment $M \Downarrow V$. We write $M \Downarrow$ if $M \Downarrow V$ is derivable *for some* V. The absence of full recursion has the nice consequence of guaranteeing a form of termination:

▶ **Theorem 1.** *The calculus $ST^{\lambda}_{\mathbb{R}}$ is terminating: if $\cdot \vdash M : \tau$ then $M \Downarrow$.*

Theorem 1 can be proved by way of a standard reducibility argument. Termination implies the following.

▶ **Corollary 2.** *If $\cdot \vdash M : REAL$ then there exists a unique $r \in \mathbb{R}$ satisfying $M \Downarrow r$, which we indicate as $NF(M)$.*

$$\frac{}{V \Downarrow V} \qquad \frac{M \Downarrow f_n \quad N \Downarrow \langle L_1, \ldots, L_n \rangle \quad L_i \Downarrow r_i}{MN \Downarrow f(r_1, \ldots, r_n)} \qquad \frac{M \Downarrow \lambda x.L \quad N \Downarrow V \quad L\{V/x\} \Downarrow W}{MN \Downarrow W}$$

$$\frac{M \Downarrow \pi_1 \quad N \Downarrow \langle L, P \rangle \quad L \Downarrow V}{MN \Downarrow V} \qquad \frac{M \Downarrow \pi_2 \quad N \Downarrow \langle L, P \rangle \quad P \Downarrow V}{MN \Downarrow V}$$

$$\frac{M \Downarrow \texttt{iflz}\ L\ \texttt{else}\ P \quad N \Downarrow r \quad r < 0 \quad L \Downarrow V}{MN \Downarrow V} \qquad \frac{M \Downarrow \texttt{iflz}\ L\ \texttt{else}\ P \quad N \Downarrow r \quad r \geq 0 \quad P \Downarrow V}{MN \Downarrow V}$$

$$\frac{M \Downarrow \texttt{iter}\ L\ \texttt{base}\ P \quad N \Downarrow r \quad r < 0 \quad P \Downarrow V}{MN \Downarrow V}$$

$$\frac{M \Downarrow \texttt{iter}\ L\ \texttt{base}\ P \quad N \Downarrow r \quad r \geq 0 \quad L((\texttt{iter}\ L\ \texttt{base}\ P)(pred_1(r))) \Downarrow V}{MN \Downarrow V}$$

Figure 3 Operational semantics for $ST^{\lambda}_{\mathbb{R}}$.

Context Equivalence

A *context* C is nothing more than a term containing a single occurrence of a placeholder $[\cdot]$. Given a context C, $C[M]$ indicates the term one obtains by substituting M for the occurrence of $[\cdot]$ in C. Typing rules in Figure 2 can be lifted to contexts by generalising judgments to the form $\Gamma \vdash C[\Delta \vdash \cdot : \tau] : \rho$, by which one captures that whenever $\Delta \vdash M : \tau$, it holds that $\Gamma \vdash C[M] : \rho$. Two terms M and N such that $\Gamma \vdash M, N : \tau$ are said to be *context equivalent* [22] if for every C such that $\emptyset \vdash C[\Gamma \vdash \cdot : \tau] : REAL$ it holds that $NF(C[M]) = NF(C[N])$. Context equivalence is the largest adequate congruence, and is thus considered as the coarsest "reasonable" equivalence between terms. It can also be turned into a pseudometric [11, 10] — called *context distance* — by stipulating that

$$\delta(M, N) = \sup_{\emptyset \vdash C[\Gamma \vdash \cdot : \tau] : REAL} |NF(C[M]) - NF(C[M])|.$$

The obtained notion of distance, however, is bound to trivialise [11], given that $ST^{\lambda}_{\mathbb{R}}$ is not affine. Trivialisation of context distance highlights an important limit of the metric approach to program difference which, ultimately, can be identified with the fact that program distances are sensitive to interactions with the environment. Our notion of a differential logical relation tackles such a problem from a different perspective, namely refining the concept of program distance which is not just a number, but is now able to take into account interactions with the environment.

Set-Theoretic Semantics

Before introducing differential logical relations, it is useful to remark that we can give $ST^{\lambda}_{\mathbb{R}}$ a standard set-theoretic semantics. To any type τ we associate the set $[\![\tau]\!]$, the latter being defined by induction on the structure of τ as follows:

$$[\![REAL]\!] = \mathbb{R}; \qquad [\![\tau \to \rho]\!] = [\![\tau]\!] \to [\![\rho]\!]; \qquad [\![\tau \times \rho]\!] = [\![\tau]\!] \times [\![\rho]\!].$$

This way, any closed term $M \in CT(\tau)$ is interpreted as an element $[\![M]\!]$ of $[\![\tau]\!]$ in a natural way (see, e.g. [20]). Up to now, everything we have said about $ST^{\lambda}_{\mathbb{R}}$ is absolutely standard, and only serves to set the stage for the next sections.

3 Making Logical Relations Differential

Logical relations can be seen as one of the *many* ways of defining when two programs are to be considered equivalent. Their definition is type driven, i.e., they can be seen as a *family*

$\{\delta_\tau\}_\tau$ of binary relations indexed by types such that $\delta_\tau \subseteq CT(\tau) \times CT(\tau)$. This section is devoted to showing how all this can be made into differential logical relations.

The first thing that needs to be discussed is how to define the space of *differences* between programs. These are just boolean values in logical relations, become real numbers in ordinary metrics, and is type-dependent itself. A function $(\!|\cdot|\!)$ that assigns a set to each type is defined as follows:

$$(\!|REAL|\!) = \mathbb{R}^\infty_{\geq 0}; \qquad (\!|\tau \to \rho|\!) = [\![\tau]\!] \times (\!|\tau|\!) \to (\!|\rho|\!); \qquad (\!|\tau \times \rho|\!) = (\!|\tau|\!) \times (\!|\rho|\!);$$

where $\mathbb{R}^\infty_{\geq 0} = \mathbb{R}_{\geq 0} \cup \{\infty\}$. The set $(\!|\tau|\!)$ is said to be the *difference space* for the type τ and is meant to model the outcome of comparisons between closed programs of type τ. As an example, when τ is $REAL \to REAL$, we have that $(\!|\tau|\!) = \mathbb{R} \times \mathbb{R}^\infty_{\geq 0} \to \mathbb{R}^\infty_{\geq 0}$. This is the type of the function $\delta(M, N)$ we used to compare the two programs described in the Introduction.

Now, which structure could we endow $(\!|\tau|\!)$ with? First of all, we can define a partial order $\leq_\tau$ over $(\!|\tau|\!)$ for each type τ as follows:

$$\begin{aligned} r &\leq_{REAL} s && \text{if } r \leq s \text{ as the usual order over } \mathbb{R}^\infty_{\geq 0};\\ f &\leq_{\tau\to\rho} g && \text{if } \forall x \in [\![\tau]\!].\forall t \in (\!|\tau|\!).f(x,t) \leq_\rho g(x,t);\\ (t,u) &\leq_{\tau\times\rho} (s,r) && \text{if } t \leq_\tau s \text{ and } u \leq_\rho r. \end{aligned}$$

This order has least upper bounds and greater lower bounds, thanks to the nice structure of $\mathbb{R}^\infty_{\geq 0}$:

▶ **Proposition 3.** *For each type τ, $((\!|\tau|\!), \leq_\tau)$ forms a complete lattice.*

The fact that $(\!|\tau|\!)$ has a nice order-theoretic structure is not the end of the story. For every type τ, we define a binary operation $*_\tau$ as follows:

$$\begin{aligned} r *_{REAL} s &= r + s \text{ if } r, s \in \mathbb{R}_{\geq 0}; && (f *_{\tau\to\rho} g)(V,t) = f(V,t) *_\rho g(V,t);\\ r *_{REAL} s &= \infty \text{ if } r = \infty \vee s = \infty; && (t,s) *_{\tau\times\rho} (u,r) = (t *_\tau u, s *_\rho r). \end{aligned}$$

This is precisely what it is needed to turn $(\!|\tau|\!)$ into a *quantale*[2] [25].

▶ **Proposition 4.** *For each type τ, $(\!|\tau|\!)$ forms a commutative unital non-idempotent quantale.*

The fact that $(\!|\tau|\!)$ is a quantale means that it has, e.g., the right structure to be the codomain of generalised metrics [19, 18]. Actually, a more general structure is needed for our purposes, namely the one of a generalised metric domain, which will be thoroughly discussed in Section 6 below. For the moment, let us concentrate our attention to programs:

▶ **Definition 5** (Differential Logical Relations)**.** *We define a* differential logical relation *$\{\delta_\tau \subseteq \Lambda_\tau \times (\!|\tau|\!) \times \Lambda_\tau\}_\tau$ as a set of ternary relations indexed by types satisfying*

$$\begin{aligned} \delta_{REAL}(M, r, N) &\Leftrightarrow |NF(M) - NF(N)| \leq r;\\ \delta_{\tau\times\rho}(M, (d_1, d_2), N) &\Leftrightarrow \delta_\tau(\pi_1 M, d_1, \pi_1 N) \wedge \delta_\rho(\pi_2 M, d_2, \pi_2 N)\\ \delta_{\tau\to\rho}(M, d, N) &\Leftrightarrow (\forall V \in CV(\tau).\ \forall x \in (\!|\tau|\!).\ \forall W \in CV(\tau).\\ &\qquad \delta_\tau(V, x, W) \Rightarrow \delta_\rho(MV, d([\![V]\!], x), NW) \wedge \delta_\rho(MW, d([\![V]\!], x), NV)). \end{aligned}$$

An intuition behind the condition required for $\delta_{\tau\to\rho}(M, d, N)$ is that $d([\![V]\!], x)$ overapproximates both the "distance" between MV and NW and the one between MW and NV, this *whenever* W is within the error x from V.

[2] Recall that a quantale $\mathbb{Q} = (Q, \leq_Q, 0_Q, *_Q)$ consists of a complete lattice $(Q, \leq_Q)$ and a monoid $(Q, 0_Q, *_Q)$ such that the lattice and monoid structures properly interact (meaning that monoid multiplication distributes over joins). We refer to [25, 18] for details.

3.1 A Fundamental Lemma

Usually, the main result about any system of logical relations is the so-called *Fundamental Lemma*, which states that any typable term is in relation *with itself*. But how would the Fundamental Lemma look like here? Should any term be at *somehow minimal distance* to itself, in the spirit of what happens, e.g. with metrics [24, 13]? Actually, there is no hope to prove anything like that for differential logical relations, as the following example shows.

▶ **Example 6.** Consider again the term $M_{ID} = \lambda x.x$, which can be given type $\tau = REAL \to REAL$ in the empty context. Please recall that $(\!|\tau|\!) = \mathbb{R} \times \mathbb{R}^{\infty}_{\geq 0} \to \mathbb{R}^{\infty}_{\geq 0}$. Could we prove that $\delta_\tau(M_{ID}, 0_\tau, M_{ID})$, where 0_τ is the constant-0 function? The answer is negative: given two real numbers r and s at distance ε, the terms $M_{ID}r$ and $M_{ID}s$ are themselves ε apart, thus at nonnull distance. The best one can say, then, is that $\delta_\tau(M_{ID}, f, M_{ID})$, where $f(x, \varepsilon) = \varepsilon$.

As the previous example suggests, a term M being at self-distance d is a witness of M being *sensitive to changes* to the environment according to d. Indeed, the only terms which are at self-distance 0 are the constant functions. This makes the underlying theory more general than the one of logical or metric relations, although the latter can be proved to be captured by differential logical relations, as we will see in the next section.

Coming back to the question with which we opened the section, we can formulate a suitable fundamental lemma for differential logical relations.

▶ **Theorem 7** (Fundamental Lemma, Version I)**.** *For every $\cdot \vdash M : \tau$ there is a $d \in (\!|\tau|\!)$ such that $(M, d, M) \in \delta_\tau$.*

Proof sketch. The proof proceeds, as usual, by induction on the derivation of $\cdot \vdash M : \tau$. In order to deal with e.g. λ-abstractions we have to strengthen our statement taking into account *open* terms. This turned out to be non-trivial and requires to extend our notion of a differential logical relation to arbitrary terms. First of all, we need to generalise $(\!|\cdot|\!)$ and $[\![\cdot]\!]$ to environments. For instance, $(\!|\Gamma|\!)$ is the set of families in the form $\alpha = \{\alpha_x\}_{(x:\rho)\in\Gamma}$, where $\alpha_x \in (\!|\rho|\!)$. Similarly for $[\![\Gamma]\!]$. This way, a natural space for differences between terms $\Gamma \vdash M, N : \tau$ can be taken as $(\!|\tau|\!)^{[\![\Gamma]\!]\times(\!|\Gamma|\!)}$, namely the set of maps from $[\![\Gamma]\!] \times (\!|\Gamma|\!)$ to $(\!|\tau|\!)$. Given an environment Γ, a family $\mathbf{V} = \{V_x\}_{(x:\rho_x)\in\Gamma}$ such that $V_x \in CV(\rho_x)$ is said to be a Γ-*family of values*. Such a Γ-family of values can naturally be seen as a substitution $\mathbf{V}$ mapping each variable $(x : \rho) \in \Gamma$ to $V_x \in CV(\rho_x)$. As it is customary, for a term $\Gamma \vdash M : \tau$ we write $M\mathbf{V}$ for the closed term of type τ obtained applying the substitution $\mathbf{V}$ to M. We denote by $CV(\Gamma)$ the set of all Γ-family of values. Given a set Z, an environment Γ, and two Γ-indexed families $\alpha = \{\alpha_x\}_{(x:\rho)\in\Gamma}$, $\beta = \{\beta_x\}_{(x:\rho)\in\Gamma}$ over Z (meaning that e.g. $\alpha_x \in Z$, for each $(x : \rho) \in \Gamma$), we introduce the following notational convention. For a Γ-indexed family $B = \{b_x\}_{(x:\rho)\in\Gamma}$ such that $b_x \in \{0, 1\}$, we can construct a 'choice' Γ-indexed family B^α_β as follows:

$$(B^\alpha_\beta)_x = \begin{cases} \alpha_x & \text{if } b_x = 0 \\ \beta_x & \text{if } b_x = 1. \end{cases}$$

Moreover, given a family B as above, we can construct the *inverse* family $\overline{B}$ as the family $\{1 - b_x\}_{(x:\rho)\in\Gamma}$. We can now talk about *open terms*, and from a differential logical relation $\{\delta_\tau \subseteq \Lambda_\tau \times (\!|\tau|\!) \times \Lambda_\tau\}_\tau$ construct a family of relations $\{\delta^\Gamma_\tau \subseteq \Lambda^\Gamma_\tau \times (\!|\tau|\!)^{[\![\Gamma]\!]\times(\!|\Gamma|\!)} \times \Lambda^\Gamma_\tau\}_{\tau,\Gamma}$ by stipulating that $\delta^\Gamma_\tau(M, d, N)$ iff

$$\delta_\Gamma(\mathbf{V}, Y, \mathbf{W}) \Longrightarrow \forall B \in \{0, 1\}^\Gamma.\delta_\tau(MB^{\mathbf{V}}_{\mathbf{W}}, d([\![\mathbf{V}]\!], Y), N\overline{B}^{\mathbf{V}}_{\mathbf{W}}).$$

We now prove the following strengthening of our main thesis: for any term $\Gamma \vdash M : \tau$, there is a $d \in (\!|\tau|\!)^{[\![\Gamma]\!]\times(\!|\Gamma|\!)}$ such that $\delta^\Gamma_\tau(M, d, M)$. At this point the proof is rather standard, and proceeds by induction on the derivation of $\Gamma \vdash M : \tau$. ◀

But what do we gain from Theorem 7? In the classic theory of logical relations, the Fundamental Lemma has, as an easy corollary, that logical relations are compatible: it suffices to invoke the theorem with any context C *seen as a term* $C[x]$, such that $x : \tau, \Gamma \vdash C[x] : \rho$. Thus, ultimately, logical relations are proved to be a *compositional* methodology for program equivalence, in the following sense: if M and N are equivalent, then $C[M]$ and $C[N]$ are equivalent, too.

In the realm of differential logical relations, the Fundamental Lemma plays a similar role, although with a different, *quantitative* flavor: once C has been proved sensitive to changes according to d, and V, W are proved to be at distance e, then, e.g., the impact of substituting V with W in C can be measured by composing d and e (and $[\![V]\!]$), i.e. by computing $d([\![V]\!], e)$. Notice that the sensitivity analysis on C and the relational analysis on V and W are decoupled. What the Fundamental Lemma tells you is that d can *always* be found.

3.2 Our Running Example, Revisited

It is now time to revisit the example we talked about in the Introduction. Consider the following two programs, both closed and of type $REAL \to REAL$:

$$M_{SIN} = \lambda x.sin_1(x); \qquad M_{ID} = \lambda x.x.$$

First of all, let us observe that, as already remarked, comparing M_{SIN} and M_{ID} using the sup metric on $\mathbb{R} \to \mathbb{R}$, as it is done in metric logical relations and applicative distances, naturally assigns them distance ∞, the euclidean distance $|x - sin(x)|$ being unbounded when x ranges over $\mathbb{R}$.

Let us now prove that $(M_{SIN}, f, M_{ID}) \in \delta_{REAL \to REAL}$, where $f(x, y) = y + |x - \sin x|$. Consider any pair of real numbers $r, s \in \mathbb{R}$ such that $|r - s| \leq \varepsilon$, where $\varepsilon \in \mathbb{R}^{\infty}_{\geq 0}$. We have that:

$$\begin{aligned} |\sin r - s| &= |\sin r - r + r - s| \leq |\sin r - r| + |r - s| \leq |\sin r - r| + \varepsilon = f(r, \varepsilon) \\ |\sin s - r| &= |\sin s - \sin r + \sin r - r| \leq |\sin s - \sin r| + |\sin r - r| \leq |s - r| + |\sin r - r| \\ &\leq \varepsilon + |\sin r - r| = f(r, \varepsilon). \end{aligned}$$

The fact that $|\sin s - \sin r| \leq |s - r|$ is a consequence of sin being 1-Lipschitz continuous (see, e.g., [12] for a simple proof).

Now, consider a context C which makes use of either M_{SIN} or M_{ID} by feeding them with a value close to 0, call it θ. Such a context could be, e.g., $C = (\lambda x.x(x\theta))[\cdot]$. C can be seen as a term having type $\tau = (REAL \to REAL) \to REAL$. A self-distance d for C can thus be defined as an element of

$$(\!|\tau|\!) = [\![REAL \to REAL]\!] \times (\!|REAL \to REAL|\!) \to \mathbb{R}^{\infty}_{\geq 0}.$$

namely $F = \lambda\!\!\lambda\langle g, h\rangle.h(g(\theta), h(\theta, 0))$. This allows for compositional reasoning about program distances: the overall impact of replacing M_{SIN} by M_{ID} can be evaluated by computing $F([\![M_{SIN}]\!], f)$. Of course the context C needs to be taken into account, but *once and for all*: the functional F can be built without knowing with which term(s) it will be fed with.

any access to either M_{SIN} or M_{ID}.

4 Logical and Metric Relations as DLRs

The previous section should have convinced the reader about the peculiar characteristics of differential logical relations compared to (standard) metric and logical relations. In

this section we show that despite the apparent differences, logical and metric relations can somehow be retrieved as specific kinds of program differences. This is, however, bound to be nontrivial. The naïve attempt, namely seeing program equivalence as being captured by *minimal* distances in logical relations, fails: the distance between a program *and itself* can be nonnull.

How should we proceed, then? Isolating those distances which witness program equivalence is indeed possible, but requires a bit of an effort. In particular, the sets of those distances can be, again, defined by induction on τ. For every τ, we give $(\!|\tau|\!)^0 \subseteq (\!|\tau|\!)$ by induction on the structure of τ:

$$(\!|REAL|\!)^0 = \{0\}; \qquad (\!|\tau \times \rho|\!)^0 = (\!|\tau|\!)^0 \times (\!|\rho|\!)^0;$$

$$(\!|\tau \to \rho|\!)^0 = \{f \in (\!|\tau \to \rho|\!) \mid \forall x \in [\![\tau]\!].\forall y \in (\!|\tau|\!)^0.f(x,y) \in (\!|\rho|\!)^0\}.$$

Notice that $(\!|\tau \to \rho|\!)^0$ is not defined as $[\![\tau]\!] \times (\!|\tau|\!)^0 \to (\!|\rho|\!)^0$ (doing so would violate $(\!|\tau \to \rho|\!)^0 \subseteq (\!|\tau \to \rho|\!)$). The following requires some effort, and testifies that, indeed, program equivalence in the sense of logical relations precisely corresponds to being at a distance in $(\!|\tau|\!)^0$:

▶ **Theorem 8.** *Let $\{\mathcal{L}_\tau\}_\tau$ be a logical relation. There exists a differential logical relation $\{\delta_\tau\}_\tau$ satisfying $\mathcal{L}_\tau(M,N) \iff \exists d \in (\!|\tau|\!)^0.\delta_\tau(M,d,N)$.*

What if we want to generalise the argument above to metric relations, as introduced, e.g., by Reed and Pierce [24]? The set $(\!|\tau|\!)^0$ becomes a set of distances parametrised by a single real number:

$$(\!|REAL|\!)^r = \{r\}; \qquad (\!|\tau \times \rho|\!)^r = (\!|\tau|\!)^r \times (\!|\rho|\!)^r;$$

$$(\!|\tau \to \rho|\!)^r = \{f \in (\!|\tau \to \rho|\!) \mid \forall x \in [\![\tau]\!].\forall y \in (\!|\tau|\!)^s.f(x,y) \in (\!|\rho|\!)^{r+s}\}.$$

A result similar to Theorem 8 is unfortunately outside the scope of this paper, but can be found in the Extended Version [12]. In particular, metric relations are only available in calculi, like FUZZ [24], which rely on *linear* type systems, thus more refined than the one we endow $ST^\lambda_{\mathbb{R}}$ with.

5 Strengthening the Fundamental Theorem through Finite Distances

Let us now ask ourselves the following question: given any term $M \in CT(\tau)$, what can we say about its sensitivity, i.e., about the values $d \in (\!|\tau|\!)$ such that $\delta_\tau(M,d,M)$? Two of the results we have proved about $ST^\lambda_{\mathbb{R}}$ indeed give partial answers to the aforementioned question. On the one hand, Theorem 7 states that such a d can *always* be found. On the other hand, Theorem 8 tells us that such a d can be taken in $(\!|\tau|\!)^0$. Both these answers are not particularly informative, however. The mere existence of such a $d \in (\!|\tau|\!)$, for example, is trivial since d can always be taken as d_∞, the maximal element of the underlying quantale. The fact that such a d can be taken from $(\!|\tau|\!)^0$ tells us that, e.g. when $\tau = \rho \to \xi$, M returns equivalent terms when fed with equivalent arguments: there is no quantitative guarantee about the behaviour of the term when fed with non-equivalent arguments.

Is this the best one can get about the sensitivity of $ST^\lambda_{\mathbb{R}}$ terms? The absence of full recursion suggests that we could hope to prove that infinite distances, although part of the underlying quantale, can in fact be useless. In other words, we are implicitly suggesting that self-distances could be elements of $(\!|\tau|\!)^{<\infty} \subset (\!|\tau|\!)$, defined as follows:

$$(\!|REAL|\!)^{<\infty} = \mathbb{R}_{\geq 0}; \qquad (\!|\tau \times \rho|\!)^{<\infty} = (\!|\tau|\!)^{<\infty} \times (\!|\rho|\!)^{<\infty};$$

$$(\!|\tau \to \rho|\!)^{<\infty} = \{f \in (\!|\tau \to \rho|\!) \mid \forall x \in [\![\tau]\!].\forall t \in (\!|\tau|\!)^{<\infty}.f(x,t) \in (\!|\rho|\!)^{<\infty}\}.$$

Please observe that $(\!|\tau|\!)^{<\infty}$ is in general a much larger set of differences than $\bigcup_{r\in\mathbb{R}^{\infty}_{\geq 0}}(\!|\tau|\!)^{r}$: the former equals the latter only when τ is $REAL$. Already when τ is $REAL \to REAL$, the former includes, say, functions like $f(r,\varepsilon) = (r+\varepsilon)^2$, while the latter does not.

Unfortunately, there are terms in $ST^{\lambda}_{\mathbb{R}}$ which cannot be proved to be at self-distance in $(\!|\tau|\!)^{<\infty}$, and, surprisingly, this is *not* due to the higher-order features of $ST^{\lambda}_{\mathbb{R}}$, but to $\{\mathcal{F}_n\}_{n\in\mathbb{N}}$ being arbitrary, and containing functions which do not map finite distances to finite distances, like

$$h(r) = \begin{cases} 0 & \text{if } r = 0 \\ \frac{1}{r} & \text{otherwise} \end{cases}$$

(see Figure 4). Is this phenomenon *solely* responsible for the necessity of finite self-distances in $ST^{\lambda}_{\mathbb{R}}$? The answer is positive, and the rest of this section is devoted precisely to formalising and proving the aforementioned conjecture.

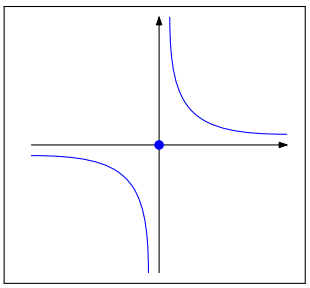

Figure 4 A total, but highly discontinuous, function.

First of all, we need to appropriately axiomatise the absence of unbounded discontinuities from $\{\mathcal{F}_n\}_{n\in\mathbb{N}}$. A not-so-restrictive but sufficient axiom turns out to be weak boundedness: a function $f_n : \mathbb{R}^n \to \mathbb{R}$ is said to be *weakly bounded* if and only if it maps bounded subsets of $\mathbb{R}^n$ into bounded subsets of $\mathbb{R}$. As an example, the function h above is *not* weakly bounded, because $h([-\varepsilon,\varepsilon])$ is

$$\left(-\infty, -\frac{1}{\varepsilon}\right] \cup \{0\} \cup \left[\frac{1}{\varepsilon}, \infty\right)$$

which is unbounded for any as $\varepsilon > 0$. Any term M is said to be weakly bounded iff any function symbol f_n occurring in M is itself weakly bounded. Actually, this is precisely what one needs to get the strengthening of the Fundamental Theorem we are looking for.

▶ **Theorem 9** (Fundamental Theorem, Version II). *For any weakly bounded term $\cdot \vdash M : \tau$, there is $d \in (\!|\tau|\!)^{<\infty}$ such that $(M, d, M) \in \delta_\tau$.*

The reader may have wondered about how restrictive a condition weak boundedness really is. In particular, whether it corresponds to some form of continuity. In fact, the introduced condition only rules out unbounded discontinuities. In other words, weak boundedness can be equivalently defined by imposing local boundedness *at any point* in the domain $\mathbb{R}$. This is weaker than asking for boundedness, which requires the existence of a global bound.

6 A Categorical Perspective

Up to now, differential logical relations have been treated very concretely, without looking at them through the lens of category theory. This is in contrast to, e.g., the treatment of metric relations from [13], in which soundness of metric relations for FUZZ is obtained as a byproduct of a proof of symmetric monoidal closedness for the category **MET** of pseudometric spaces and Lipschitz functions.

But what could take the place of pseudometric spaces in a categorical framework capturing differential logical relations? The notion of a metric needs to be relaxed along at least two axes. On the one hand, the codomain of the "metric" δ is not necessarily the set of real numbers, but a more general structure, namely a quantale. On the other, as we already noticed, it is not necessarily true that equality implies indistancy, but rather than indistancy implies inequality. What comes out of these observations is, quite naturally, the notion

of a generalized metric domain, itself a generalisation of partial metrics [7]. The rest of this section is devoted to proving that the category of generalised metric domains is indeed cartesian closed, thus forming a model of simply typed λ-calculi.

Formally, given a quantale $\mathbb{Q} = (Q, \leq_Q, 0_Q, *_Q)$[3], a *generalised metric domain* on $\mathbb{Q}$ is a pair (A, δ_A), where A is a set and δ_A is a subset of $A \times \mathbb{Q} \times A$ satisfying some axioms akin to those of a metric domain:

$$\delta_A(x, 0_Q, y) \Rightarrow x = y; \qquad \text{(Indistancy Implies Equality)}$$

$$\delta_A(x, d, y) \Rightarrow \delta_A(y, d, x); \qquad \text{(Symmetry)}$$

$$\delta_A(x, d, y) \wedge \delta_A(y, e, y) \wedge \delta_A(y, f, z) \Rightarrow \delta_A(x, d * e * f, z). \qquad \text{(Triangularity)}$$

Please observe that δ_A is a *relation* rather than a function. Moreover, the first axiom is dual to the one typically found in, say, pseudometrics. The third axiom, instead, resembles the usual triangle inequality for pseudometrics, but with the crucial difference that since objects can have non-null self-distance, such a distance has to be taken into account. Requiring equality to imply indistancy (and thus $\delta_A(x, 0_Q, y) \Leftrightarrow x = y$), we see that (Triangularity) gives exactly the usual triangle inequality (properly generalised to quantale and relations [18, 19]).

In this section we show that generalised metric domains form a cartesian closed category, unlike that of metric spaces (which is known to be non-cartesian closed). As a consequence, we obtain a firm categorical basis of differential logical relations. The category of generalised metric domain, denoted by **GMD**.

▶ **Definition 10.** *The category* **GMD** *has the following data.*

- *An object $\mathcal{A}$ is a triple $(A, \mathbb{Q}, \delta)$ where $\mathbb{Q}$ is a quantale and (A, δ) is a generalized metric domain on $\mathbb{Q}$.*
- *An arrow $(A, \mathbb{Q}, \delta) \to (B, \mathbb{S}, \rho)$ is a pair (f, ζ) consisting of a function $f \colon A \to B$ and another function $\zeta \colon Q \times A \to S$ satisfying $\forall a, a' \in A. \forall q \in Q. \delta(a, q, a') \Rightarrow \rho(f(a), \zeta(q, a), f(a'))$ and $\rho(f(a), \zeta(q, a'), f(a'))$.*

We can indeed give **GMD** the structure of a category. In fact, the identity on the object $\mathcal{A} = (A, \mathbb{Q}, \delta)$ in **GMD** is given by $(\mathrm{id}_{\mathcal{A}}, \mathrm{id}'_{\mathcal{A}})$ where $\mathrm{id}_{\mathcal{A}} \colon A \to A$ is the set-theoretic identity on A and $\mathrm{id}'_{\mathcal{A}} \colon Q \times A \to Q$ is defined by $\mathrm{id}'_{\mathcal{A}}(q, a) = q$. The composition of two arrows $(f, \zeta) \colon (A, \mathbb{Q}, \delta) \to (B, \mathbb{S}, \rho)$ and $(g, \eta) \colon (B, \mathbb{S}, \rho) \to (C, \mathbb{T}, \nu)$ is the pair (h, θ) where $h \colon A \to C$ is given by the function composition $g \circ f \colon A \to C$ and $h \colon Q \times A \to T$ is given by $\theta(q, a) = \eta(\zeta(q, a), f(a))$. Straightforward calculations show that composition is associative, and that the identity arrow behaves as its neutral element.

Most importantly, we can give **GMD** a cartesian closed structure, as shown by the following result[4].

▶ **Theorem 11.** **GMD** *is a cartesian closed category.*

Proof sketch. Before entering details, it is useful to remark that the cartesian product of two quantales is itself a quantale (with lattice and monoid structure defined pointwise). Similarly, for any quantale $\mathbb{Q}$ and set X, the function space $\mathbb{Q}^X$ inherits a quantale structure from $\mathbb{Q}$ pointwise. Let us now show that **GMD** is cartesian closed. We begin showing that **GMD** has a terminal object and binary products. The former is defined as $(\{*\}, \mathbb{O}, \delta_0)$, where $\mathbb{O}$

[3] When unambiguous, we will omit subscripts in $\leq_Q$, 0_Q, and $*_Q$.
[4] See [12] for a detailed proof.

is the one-element quantale $\{0\}$, and $\delta_0 = \{(*, 0, *)\}$ (notice that $(\{*\}, \delta_0)$ is a generalized metric domain on $\mathbb{O}$), whereas the binary product $\mathcal{A} \times \mathcal{B}$ of two objects $\mathcal{A}$ and $\mathcal{B}$ in **GMD** is given by a triple $(A \times B, \mathbb{Q} \times \mathbb{S}, \delta \times \rho)$. Finally, we define exponentials in **GMD**. Given $\mathcal{C}$, $\mathcal{B}$ in **GMD**, their exponential $\mathcal{C}^{\mathcal{B}}$ is the triple $(C^B, \mathbb{T}^{\mathbb{S}\times B}, \nu^\rho)$, where C^B is the function space $\{f \mid f \colon B \to C\}$, $\mathbb{T}^{\mathbb{S}\times B}$ is the exponential quantale, and ν^ρ is a ternary relation over $C^B \times T^{S\times B} \times C^B$ defined by: if $\rho(b, s, b')$ then $\nu(f(b), d(s, b), f'(b'))$ and $\nu(f(b), d(s, b'), \zeta(b'))$. Please notice that the relation ν^ρ is indeed a differential logical relation. ◀

Interestingly, the constructions of product and exponential objects in the proof of Theorem 11 closely match the definition of a differential logical relation. In other words, differential logical relations as given in Definition 5 can be seen as providing a denotational model of $ST^\lambda_{\mathbb{R}}$ in which base types are interpreted by the generalised metric domain corresponding to the Euclidean distance.

7 Conclusion

In this paper, we introduced differential logical relations as a novel methodology to evaluate the "distance" between programs of higher-order calculi akin to the λ-calculus. We have been strongly inspired by some unpublished work by Westbrook and Chaudhuri [31], who were the first to realise that evaluating differences between interactive programs requires going beyond mere real numbers. We indeed borrowed our running examples from the aforementioned work.

This paper's contribution, then consists in giving a simple definition of differential logical relations, together with some results about their underlying metatheory: two formulations of the Fundamental Lemma, a result relating differential logical relations and ordinary logical relations, and a proof that generalised metric domains — the metric structure corresponding to differential logical relations — form a cartesian closed category. Such results give evidence that, besides being *more expressive* than metric relations, differential logical relations are somehow *more canonical*, naturally forming a model of simply-typed λ-calculi.

As the title of this paper suggests, we see the contributions above just as a very first step towards understanding the nature of differences in a logical environment. In particular, at least two directions deserve to be further explored.

- The first one concerns *language features*: admittedly, the calculus $ST^\lambda_{\mathbb{R}}$ we consider here is very poor in terms of its expressive power, lacking full higher-order recursion and thus not being universal. Moreover, $ST^\lambda_{\mathbb{R}}$ does not feature any form of effect, including probabilistic choices, in which evaluating differences between programs would be very helpful. Addressing such issues seems to require to impose a domain structure on generalised metric domain, on one hand, and to look at monads on **GMD**, on the other hand (for the latter, the literature on monadic lifting for quantale-valued relations might serve as a guide [18]).
- The second one is about *abstract differences*: defining differences as functions with *the same rank* as that of the compared programs implies that reasoning about them is complex. Abstracting differences so as to facilitate differential reasoning could be the way out, given that deep connections exist between logical relations and abstract interpretation [2]. Another way to understand program difference better is to investigate whether differential logical relations can be related to abstract structures for differentiation, as in [3]. Indeed, Example 6 suggests that an interesting distance between a program and itself can be taken as its derivative, the latter being defined as in [8].

References

1. S. Abramsky. The lazy lambda calculus. In D. Turner, editor, *Research Topics in Functional Programming*, pages 65–117. Addison Wesley, 1990.
2. Samson Abramsky. Abstract interpretation, logical relations and Kan extensions. *J. Log. Comput.*, 1(1):5–40, 1990.
3. Mario Alvarez-Picallo and C.-H. Luke Ong. Change actions: Models of generalised differentiation. In *Proc. of FOSSACS 2019*, pages 45–61, 2019.
4. A. Arnold and M. Nivat. Metric interpretations of infinite trees and semantics of non deterministic recursive programs. *Theor. Comput. Sci.*, 11:181–205, 1980.
5. C. Baier and M.E. Majster-Cederbaum. Denotational semantics in the CPO and metric approach. *Theor. Comput. Sci.*, 135(2):171–220, 1994.
6. Gilles Barthe, Marco Gaboardi, Justin Hsu, and Benjamin C. Pierce. Programming language techniques for differential privacy. *SIGLOG News*, 3(1):34–53, 2016.
7. Michael A. Bukatin, Ralph Kopperman, Steve Matthews, and Homeira Pajoohesh. Partial metric spaces. *The American Mathematical Monthly*, 116(8):708–718, 2009.
8. Yufei Cai, Paolo G. Giarrusso, Tillmann Rendel, and Klaus Ostermann. A theory of changes for higher-order languages: incrementalizing λ-calculi by static differentiation. In *Proc. of PLDI*, pages 145–155, 2014.
9. Konstantinos Chatzikokolakis, Daniel Gebler, Catuscia Palamidessi, and Lili Xu. Generalized bisimulation metrics. In *CONCUR 2014 - Concurrency Theory - 25th International Conference, CONCUR 2014, Rome, Italy, September 2-5, 2014. Proceedings*, pages 32–46, 2014.
10. Raphaëlle Crubillé and Ugo Dal Lago. Metric reasoning about λ-terms: The affine case. In *Proc. of LICS 2015*, pages 633–644, 2015.
11. Raphaëlle Crubillé and Ugo Dal Lago. Metric reasoning about λ-terms: The general case. In *Proc. of ESOP 2017*, pages 341–367, 2017.
12. Ugo Dal Lago, Francesco Gavazzo, and Akira Yoshimizu. Differential logical relations, Part I: The simply-typed case (extended version). 2018. URL: `https://arxiv.org/abs/1904.12137`.
13. A.A. de Amorim, M. Gaboardi, J. Hsu, S. Katsumata, and I. Cherigui. A semantic account of metric preservation. In *Proc. of POPL 2017*, pages 545–556, 2017.
14. J.W. de Bakker and J.I. Zucker. Denotational semantics of concurrency. In *STOC*, pages 153–158, 1982.
15. Josee Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labelled markov processes. *Theor. Comput. Sci.*, 318(3):323–354, 2004.
16. M.H. Escardo. A metric model of pcf. In *Workshop on Realizability Semantics and Applications*, 1999.
17. Francesco Gavazzo. Quantitative behavioural reasoning for higher-order effectful programs: Applicative distances. In *Proc. of LICS 2018*, pages 452–461, 2018.
18. D. Hofmann, G.J. Seal, and W. Tholen, editors. *Monoidal Topology. A Categorical Approach to Order, Metric, and Topology.* Number 153 in Encyclopedia of Mathematics and its Applications. Cambridge University Press, 2014.
19. F.W. Lawvere. Metric spaces, generalized logic, and closed categories. *Rend. Sem. Mat. Fis. Milano*, 43:135–166, 1973.
20. John C. Mitchell. *Foundations for Programming Languages.* MIT Press, 1996.
21. Sparsh Mittal. A survey of techniques for approximate computing. *ACM Comput. Surv.*, 48(4), 2016.
22. J. Morris. *Lambda Calculus Models of Programming Languages.* PhD thesis, MIT, 1969.
23. Gordon D. Plotkin. Lambda-definability and logical relations. Memorandum SAI-RM-4, University of Edinburgh, 1973.
24. J. Reed and B.C. Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In *Proc. of ICFP 2010*, pages 157–168, 2010.
25. K.I. Rosenthal. *Quantales and their applications.* Pitman research notes in mathematics series. Longman Scientific & Technical, 1990.

436 **26** Dana Scott. Outline of a mathematical theory of computation. Technical Report PRG02,
437 OUCL, November 1970.
438 **27** Dana Scott and Christopher Strachey. Toward a mathematical semantics for computer
439 languages. Technical Report PRG06, OUCL, August 1971.
440 **28** F. Van Breugel. An introduction to metric semantics: operational and denotational models
441 for programming and specification languages. *Theor. Comput. Sci.*, 258(1-2):1–98, 2001.
442 **29** F. Van Breugel and J. Worrell. A behavioural pseudometric for probabilistic transition systems.
443 *Theor. Comput. Sci.*, 331(1):115–142, 2005.
444 **30** Franck van Breugel and James Worrell. Towards quantitative verification of probabilistic
445 transition systems. In *Proc. of ICALP 2001*, pages 421–432, 2001.
446 **31** Edwin M. Westbrook and Swarat Chaudhuri. A semantics for approximate program trans-
447 formations. *CoRR*, abs/1304.5531, 2013. URL: `http://arxiv.org/abs/1304.5531`.
448 **32** Lili Xu, Konstantinos Chatzikokolakis, and Huimin Lin. Metrics for differential privacy in
449 concurrent systems. In *Proc. of FORTE 2014*, pages 199–215, 2014.