The differential lambda-calculus: From semantics to syntax

Thomas Ehrhard (IRIF - Université Paris Cité, CNRS, Inria)
Laurent Regnier (I2M - Université d'Aix-Marseille, CNRS)

Tallinn - July 10, 2024

What is differentiation?

Approximate functions by linear maps:

$$f : \mathbb{R}^n \to \mathbb{R} \quad \rightsquigarrow \quad \begin{array}{c} Df : \mathbb{R}^n \to \mathcal{L}(\mathbb{R}^n, \mathbb{R}) \\ f(x+u) = f(x) + Df(x).u + o(u) \end{array}$$

Numerical linearization of f

But what if $M : A \Rightarrow B$ is a program?

$$M : A \Rightarrow B \quad \rightsquigarrow \quad DM : A \Rightarrow (A \multimap B)$$

Linear logical linearization of M

The lambda-calculus: a syntax for functions

A syntax to denote functions, just as terms of set theory denote sets: this was the original idea of Alonzo Church's Type Theory

- Given a set of variables $\mathcal{V} = \{x, y, x_1, \dots\}$
- if $x \in \mathcal{V}$ then x is a term
- is M and N are terms then $M\,N$ is a term (function M applied to argument N)
- if $x \in \mathcal{V}$ and M is a term then $\lambda x\, M$ is a term (function $x \mapsto M$). The variable x is bound

Only one rule for computing with these terms: β-reduction

$$\overbrace{(\lambda x\, M)N}^{\text{redex}} \to \overbrace{M\,[N/x]}^{\text{reduct}}$$

Computing $=$ rewriting

Fact

λ-calculus is a deterministic and Turing complete model of computation

Do λ-terms denote functions (or morphisms)?

Yes! It suffices to find a cartesian closed category $\mathcal{C}$ and, in that category, an object U together with two morphisms

$$e^+ \in \mathcal{C}(U \Rightarrow U, U)$$
$$e^- \in \mathcal{C}(U, U \Rightarrow U)$$

such that $e^- \circ e^+ = \mathsf{Id}_{U \Rightarrow U}$

- Impossible if $\mathcal{C}$ is the category of sets and functions, for cardinality reasons
- Dana Scott (1968): possible in the category of complete lattices and directed sup preserving functions

Many other examples since then: denotational models of the λ-calculus

An example

$\mathbf{Rel}_!$ is the following category:

- objects of $\mathbf{Rel}_!$: all sets
- $\mathbf{Rel}_!(E, F) = \mathcal{P}\left(\mathcal{M}_{\mathrm{fin}}(E) \times F\right)$ where $\mathcal{M}_{\mathrm{fin}}(E)$ is the set of all finite multisets $[a_1, \ldots, a_n]$ of elements of E

with

- identity at E: $\mathsf{Id}_E = \{([a], a) \mid a \in E\}$
- composition: if $R \in \mathbf{Rel}_!(E, F)$ and $S \in \mathbf{Rel}_!(F, G)$ then

$$\begin{aligned} S \circ R = \{(m_1 + \cdots + m_n, c) \mid \exists b_1, \ldots, b_n \in F \\ ((m_i, b_i) \in R)_{i=1}^{n} \text{ and } ([b_1, \ldots, b_n], c) \in S\} \end{aligned}$$

Intuition

Elements of E = "basis" of a vector space (or rather of a free semi-module on the semi-ring $\{0, 1\}$ with $1 + 1 = 1$); multiset $[a_1, \dots a_k]$ = monomial $x_{a_1} \dots x_{a_k}$

Then $R \in \mathbf{Rel}_!(E, F)$ is a powerseries (with parameters indexed by E and output spanned by F)

$$\begin{aligned} \lambda x^{\mathbf{Bool}} \,.\, x \wedge \neg x &= \{([\mathbf{t}, \mathbf{t}], \mathbf{f}), ([\mathbf{f}, \mathbf{f}], \mathbf{f}), ([\mathbf{t}, \mathbf{f}], \mathbf{f}), ([\mathbf{t}, \mathbf{f}], \mathbf{t})\} \\ &= (x_{\mathbf{t}}^2 + x_{\mathbf{f}}^2 + x_{\mathbf{t}} x_{\mathbf{f}}).\mathbf{f} + x_{\mathbf{t}} x_{\mathbf{f}}.\mathbf{t} \end{aligned}$$

The definition of identities and composition is compatible with this intuition

This is a CCC:

- the categorical product is disjoint union
- the internal hom of E and F is $(E \Rightarrow F) = \mathcal{M}_{\mathrm{fin}}(E) \times F$

The simplest model of the λ-calculus

Given a non-empty set E_0 (none elements of which are pairs), we can define a monotone family of sets:

- $U_0 = \emptyset$
- $U_{n+1} = E_0 \cup (\mathcal{M}_{\mathrm{fin}}(U_n) \times U_n)$

and then

$$U = \bigcup_{n=0}^{\infty} U_n \quad \text{satisfies} \quad U = E_0 \cup (U \Rightarrow U)$$

Linearity in $\mathbf{Rel}_!$

Even if the category $\mathbf{Rel}_!$ is very simple, it has an interesting feature: some morphisms are linear

Definition

A morphism $R \in \mathbf{Rel}_!(E, F) = \mathcal{P}(\mathcal{M}_{\mathrm{fin}}(E) \times F)$ is linear if all the elements of R are of shape $([a], b)$

Identity morphisms are linear and linear morphisms are stable under composition

$\rightsquigarrow$ the subcategory of sets and linear morphisms is (isomorphic) to **Rel**, the category of sets and relations

This category is monoidal symmetric, that is there is a well behaved tensor product which is simply $E \otimes F = E \times F$

Rel is a well-known model of Linear Logic

Differentiation

We have actually an internal linear hom in **Rel**:

$$\mathbf{Rel}(G \otimes E, F) = \mathbf{Rel}(G, E \multimap F)$$

namely $E \multimap F = E \times F$

As a powerseries any $R \in \mathbf{Rel}_!(E, F) = \mathcal{P}\left(\mathcal{M}_{\mathrm{fin}}(E) \times F\right)$ has a derivative

$$R' = \{(m, (a, b)) \mid (m + [a], b) \in R\} \in \mathbf{Rel}_!(E, E \multimap F)$$

Satisfies all the expected algebraic properties of a derivative (Leibniz, chain rule etc)

In the semantic universe $\mathbf{Rel}_!$ we have at the same time:

- the λ-calculus
- and differentiation

This suggests to extend the lambda-calculus with differentiation
this is exactly what we did in 2002

The differential λ-calculus

We introduce a new construction in the λ-calculus:

- if M and N are terms, then $\mathrm{D}M \cdot N$ is a term, the differential application of M to N

Intuitively: M denotes a morphism $R \in \mathbf{Rel}_!(U, U)$, so we have $R' \in \mathbf{Rel}_!(U, U \multimap U)$, and so, swapping the arguments we get

$$S \in \mathbf{Rel}(U, U \Rightarrow U)$$

$\mathrm{D}M \cdot N$ denotes the (linear) application of S to the denotation of N, which $\in \mathcal{P}(U)$

This differential application yields an element of $\mathcal{P}(U \Rightarrow U) = \mathbf{Rel}_!(U, U)$

Convenient feature of this syntax: derivatives are easy to iterate

$$\mathrm{D}^k M \cdot (N_1, \ldots, N_k) = \mathrm{D}(\cdots \mathrm{D} M \cdot N_1 \cdots) \cdot N_k$$

The differential redex

The main idea of the differential λ-calculus is to say that

$$\mathrm{D}(\lambda x\, M) \cdot N$$

is a new redex: the differential β-redex

$$\mathrm{D}(\lambda x\, M) \cdot N \to \lambda x \left(\frac{\partial M}{\partial x} \cdot N\right)$$

where $\frac{\partial M}{\partial x} \cdot N$ is a kind of "substitution" defined by induction on M

Intuition: replace exactly one occurrence of x in M

- In $\frac{\partial M}{\partial x} \cdot N$, the variable x is still free (there are remaining occurrences of x not yet substituted) $\leadsto$ the λx remaining in the reduct
- In $y\,x$ one cannot say that x has exactly one occurrence because y can be replaced with a function which duplicates or erases its argument

We need two more operations on terms to define $\frac{\partial M}{\partial x} \cdot N$:

- a constant 0
- and if M and N are terms, a new term $M + N$

Good news

We don't need anything more

$$\frac{\partial x}{\partial x} \cdot N = N \qquad\qquad \frac{\partial y}{\partial x} \cdot N = 0 \text{ if } y \in \mathcal{V} \setminus \{x\}$$

$$\frac{\partial \lambda y\, M}{\partial x} \cdot N = \lambda y \left(\frac{\partial M}{\partial x} \cdot N \right)$$

$$\frac{\partial \mathrm{D} P \cdot Q}{\partial x} \cdot N = \mathrm{D}\left(\frac{\partial P}{\partial x} \cdot N \right) \cdot Q + \mathrm{D} P \cdot \left(\frac{\partial Q}{\partial x} \cdot N \right)$$

$$\frac{\partial (P\, Q)}{\partial x} \cdot N = \left(\frac{\partial P}{\partial x} \cdot N \right) Q + \left(\mathrm{D} P \cdot \left(\frac{\partial Q}{\partial x} \cdot N \right) \right) Q$$

One must also say that all constructs commute with 0 and $+$, but the argument component of ordinary application

So we consider terms up to the following equalities

$$\lambda x\,0 = 0 \qquad \lambda x\,(M_1 + M_2) = \lambda x\,M_1 + \lambda x\,M_2$$
$$0\,N = 0 \qquad (M_1 + M_2)\,N = M_1\,N + M_2\,N$$
$$\mathrm{D}0 \cdot N = 0 \qquad \mathrm{D}(M_1 + M_2) \cdot N = \mathrm{D}M_1 \cdot N + \mathrm{D}M_2 \cdot N$$
$$\mathrm{D}M \cdot 0 = 0 \qquad \mathrm{D}M \cdot (N_1 + N_2) = \mathrm{D}M \cdot N_1 + \mathrm{D}M \cdot N_2$$

We do not have

$$M\,0 = 0 \qquad M\,(N_1 + N_2) = M\,N_1 + M\,N_2$$

Example

$$\frac{\partial (y\,x)}{\partial x} \cdot N = (\mathrm{D} y \cdot N)\, x$$

We also need

Schwarz

$$\mathrm{D}^2 M \cdot (N_1, N_2) = \mathrm{D}^2 M \cdot (N_2, N_1)$$

Sums come in even if not invited

$$(\mathrm{D}^2(\lambda x^{\textbf{Bool}} . \, x \wedge \neg x) \cdot (\mathbf{t}, \mathbf{f}))\, 0 \quad \to^* \quad \mathbf{t} + \mathbf{f}$$

Syntactic Taylor expansion

If we accept infinite sums and rational coefficients, we can write a Taylor expansion of the application:

$$\sum_{n=0}^{\infty} \frac{1}{n!} (\mathrm{D}^n M \cdot (N, \ldots, N))0 \quad \text{instead of} \quad M\,N$$

We can apply this to all the applications in an (ordinary) λ-term

Then the only ordinary applications we use are of shape $M\,0$

Differential resource calculus

- If $x \in \mathcal{V}$ then x is a term;
- if $x \in \mathcal{V}$ and t is a term then $\lambda x\, t$ is a term;
- if s is a term and T is a multiset $[t_1, \dots, t_n]$ of terms, then $\langle s \rangle\, T$ is a term

$\Delta =$ the set of all resource terms

Intuition:

$$\langle s \rangle\, T = (\mathrm{D}^n s \cdot (t_1, \dots, t_n))\, 0$$

All the constructions are (multi)linear

Example of multilinearity

$$\langle s \rangle\, [t_1 + t_1', t_2, \dots, t_n] = \langle s \rangle\, [t_1, \dots, t_n] + \langle s \rangle\, [t_1', t_2, \dots, t_n]$$

In a resource term s, all the occurrences of a variable x are linear occurrences (in contrast with the ordinary λ-calculus)

It make sense to define

$$\deg_x s = \text{number of occurrences of } x \text{ in } s$$

Differential β-reduction becomes:

$$\langle \lambda x\, s \rangle \left[t_1, \ldots, t_n\right] \to \begin{cases} \sum_{\sigma \in \mathfrak{S}_n} s \left[t_{\sigma(1)}/x_1, \ldots, t_{\sigma(n)}/x_n \right] & \text{if } n = \deg_x s \\ 0 & \text{otherwise} \end{cases}$$

where $x_1, \ldots, x_n$ are the n occurrences of x in s

Normalization of resource terms

Contrarily to the λ-calculus, reduction in the resource calculus always terminates, but can yield any sum (including 0)

Fact

Any resource term s reduces to a unique normal form $\mathsf{NF}(s)$ which is a finite sum of resource terms which have no redexes

Example:

$$\mathsf{NF}(\langle \lambda x \langle x \rangle [x, x] \rangle [y, z, z]) = 2 \langle y \rangle [z, z] + 4 \langle z \rangle [y, z]$$

Then we can define the complete Taylor expansion M^* of a λ-term M as a (generally infinite) linear combination of resource terms with ≥ 0 rational coefficients:

$$\begin{aligned} x^* &= x \\ (\lambda x\, M)^* &= \lambda x\, M^* = \sum_{s \in \Delta} M^*_s\, \lambda x\, s \\ (M\, N)^* &= \sum_{n=0}^{\infty} \frac{1}{n!}\ \langle M^* \rangle\, [\overbrace{N^*, \ldots, N^*}^{n}] \end{aligned}$$

If we develop these expressions, we get

$$M^* = \sum_{s \in \mathcal{T}(M)} \frac{1}{m(s)}\, s$$

where

$$\begin{aligned}
\mathcal{T}(x) &= \{x\} \\
\mathcal{T}(\lambda x\, M) &= \{\lambda x\, s \mid s \in \mathcal{T}(M)\} \\
\mathcal{T}(M\, N) &= \{\langle s\rangle\, [t_1, \ldots, t_n] \mid n \in \mathbb{N},\ s \in \mathcal{T}(M) \\
&\qquad \text{and } t_1, \ldots, t_n \in \mathcal{T}(N)\}
\end{aligned}$$

and $m(s) \in \mathbb{N} \setminus \{0\}$ depends only on s

A theorem

If $M \downarrow x$, then there is exactly one $s \in \mathcal{T}(M)$ such that $\mathsf{NF}(s) \neq 0$ and this s satisfies

$$\mathsf{NF}(s) = m(s)\,x$$

This s is the "trace" of execution of M in the Krivine machine, a realistic model of λ-term execution

s is a "decoration" of M with the multiplicities expressing how many times the various subterms are used

Example: $M = (\lambda y\, y\,(y\,x))\,\lambda z\, z$

Then $s = \langle \lambda y\, \langle y \rangle\, [\langle y \rangle\, [x]] \rangle\, [\lambda z\, z, \lambda z\, z]$ and $m(s) = 2$

Remark: This theorem may be generalized

Linear logic in a nutshell

A ressource aware typing discipline of programs:

$M : A \multimap B$ means M uses its input (typed by A) once and only once to produce its output (typed by B)

Examples

$$\lambda x^{\mathbf{Bool}} \,.\, x \wedge \neg x : \mathbf{Bool} \Rightarrow \mathbf{Bool}$$

$$\mathbf{App}_\ell = \lambda x^{A} \,.\, \lambda f^{A \multimap B} \,.\, f\, x : A \multimap (A \multimap B) \multimap B$$

$$\mathbf{App} = \lambda x^{A} \,.\, \lambda f^{A \Rightarrow B} \,.\, f\, x : A \Rightarrow (A \Rightarrow B) \multimap B$$

Exponentials

Exponential modalities for typing non linear programs:

$$A \Rightarrow B = {!A} \multimap B$$

(syntactic version of $\mathbf{Rel}_!(E, F)$ seen above)

Embedding linear programs into general ones

$$!A \multimap A \quad \text{(dereliction)}$$

And coping with erasing and duplication:

$$!A \multimap 1 \quad \text{erasing (weakening)}$$
$$!A \multimap {!A} \otimes {!A} \quad \text{duplication (contraction)}$$

Differential linear logic (DiLL)

Codereliction

$$A \multimap\ !A$$

Differentiation at 0:
$F : A \Rightarrow B \quad \rightsquigarrow \quad \lambda x^A \,.\, (\mathrm{D}F \cdot x)\, 0 : A \multimap B$

Coweakening

$$1 \multimap\ !A$$

Evaluation at 0: $F : A \Rightarrow B \quad \rightsquigarrow \quad F\, 0 : B$

Cocontraction

$$!A \otimes !A \multimap\ !A$$

Evaluation on a sum:
$F : A \Rightarrow B \ \rightsquigarrow \ \lambda x^A \,.\, \lambda y^A \,.\, F\,(x + y) : A \Rightarrow A \Rightarrow B$

Differential λ-Calculus and Differential Linear Logic, 20 Years Later (a conference at CIRM - Marseille 2024)

- Pierre Clairambault (CNRS, Aix-Marseille Université) Quantitative semantics in game models
- Ugo Dal Lago (University of Bologna) Reasoning Operationally about Probabilistic Higher-Order Programs
- Thomas Ehrhard (CNRS, Paris Cité) Coherent differentiation
- Zeinab Galal (Bologna) Stable species
- Nicola Gambino (University of Manchester) Generalized species
- Brenda Johnson (Union College) Differential Categories from Functor Calculus
- Marie Kerjean (CNRS, Sorbonne Paris Nord) Introduction to DiLL

DiλLL 2024, continued

- Delia Kesner (Paris Cité) Non-idempotent intersection types
- Giulio Manzonetto (Paris Cité) Taylor expansion and Böhm trees
- Guy McCusker (Bath) Weighted models
- Jean-Simon Pacaud Lemay (Macquarie University) Differential categories
- Michele Pagani (ENS de Lyon) Automatic differentiation
- Luc Pellissier (Paris-Est Créteil) Taylor expansion in proof nets
- Christine Tasson (ISAE-Supaero) Probabilistic coherence spaces