Coherent Differentiation

Resources in Computation — project meeting

21-23 September 2022

Thomas Ehrhard
IRIF, CNRS and Université Paris Cité

DiLL and Coherent Differentiation

Linear Logic: algebraic viewpoint on DS

Girard's LL (1986) reflects the fact that denotational models have an underlying linear structure featuring operations very similar to those of *linear algebra*: tensor product, direct product, linear function space, dual etc.

Of course such models have also non linear morphisms.

Linear Logic: algebraic viewpoint on DS

Girard's LL (1986) reflects the fact that denotational models have an underlying linear structure featuring operations very similar to those of *linear algebra*: tensor product, direct product, linear function space, dual etc.

Of course such models have also non linear morphisms.

The *exponential resource modality* of LL explains the connection between the linear and the non-linear worlds (categories).

Basic principle: we can forget that a function is linear, this is *dereliction*.

Differentiation in LL

Differential LL axiomatizes the converse operation:

$$\begin{aligned} \text{dereliction} &: \text{linear} \to \text{non-linear} \\ \text{differentiation} &: \text{non-linear} \to \text{linear} \end{aligned}$$

reformulating the standard laws of the differential calculus.

Then differentiation becomes a generic *logical* operation. In the differential λ-calculus:

$$\frac{\Gamma \vdash M : A \Rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash \mathrm{D}M \cdot N : A \Rightarrow B}$$

And $\mathrm{D}M \cdot N$ is linear in N (and also in M).

Intuition

The derivative of M should be $M' : A \Rightarrow (A \multimap B)$. Then intuitively

$$\mathrm{D}M \cdot N = \lambda x : A \cdot (M'\, x)(N)$$

Until recently DiLL was strongly non-deterministic, there was a deduction rule

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash A}{\Gamma \vdash A} \; (+)$$

apparently required to take into account the Leibniz rule $(uv)' = u'v + uv'$, Until recently DiLL was strongly non-deterministic, there was a deduction rule

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash A}{\Gamma \vdash A}\ (+)$$

apparently required to take into account the Leibniz rule $(uv)' = u'v + uv'$, or more generally

$$\frac{df(x,x)}{dx} \cdot u = f_1'(x,x) \cdot u + f_2'(x,x) \cdot u$$

Leibniz results from the interaction between differentiation and contraction.

$\rightsquigarrow$ models of DiLL are additive categories

Because of Leibniz, the categorical models $\mathcal{L}$ of DiLL are *additive* categories:

- $\mathcal{L}(X, Y)$ is a commutative monoid (with additive notations) for each objects X, Y of $\mathcal{L}$
- morphism composition is bilinear.

Remark

If $\mathcal{L}$ is cartesian and additive then the cartesian product is also a coproduct, the terminal object is initial: $\& = \oplus$.

From the viewpoint of LL, DiLL is somehow degenerate!

But there are many models of LL which *are not* additive categories.

Remark

There are even models of LL such as **Pcoh** where non-linear morphisms are obviously differentiable because they are analytic functions,

But there are many models of LL which *are not* additive categories.

Remark

There are even models of LL such as **Pcoh** where non-linear morphisms are obviously differentiable because they are analytic functions,

and which are not additive categories!

But there are many models of LL which *are not* additive categories.

Remark

There are even models of LL such as **Pcoh** where non-linear morphisms are obviously differentiable because they are analytic functions,

and which are not additive categories!

Question

Can we axiomatize logically the differentials in such models?

But there are many models of LL which *are not* additive categories.

Remark

There are even models of LL such as **Pcoh** where non-linear morphisms are obviously differentiable because they are analytic functions,

and which are not additive categories!

Question

Can we axiomatize logically the differentials in such models?

At first sight this seems difficult...

. . . a concrete example

In **Pcoh** the type 1 of LL is interpreted as

$$[0, 1] \subseteq \mathbb{R}$$

. . . a concrete example

In **Pcoh** the type 1 of LL is interpreted as

$$[0,1] \subseteq \mathbb{R}$$

A non-linear morphism, that is, an element of $\mathcal{L}_!(1,1)$, is an analytic function

$$\begin{aligned} f : [0,1] &\to [0,1] \\ x &\mapsto \sum_{n=0}^{\infty} a_n x^n \end{aligned}$$

for a (uniquely determined) sequence $(a_n)_{n\in\mathbb{N}}$ of elements of $\mathbb{R}_{\geq 0}$ such that $\sum_{n\in\mathbb{N}} a_n \leq 1$.

. . . a concrete example

In **Pcoh** the type 1 of LL is interpreted as

$$[0,1] \subseteq \mathbb{R}$$

A non-linear morphism, that is, an element of $\mathcal{L}_!(1,1)$, is an analytic function

$$\begin{aligned} f : [0,1] &\to [0,1] \\ x &\mapsto \sum_{n=0}^{\infty} a_n x^n \end{aligned}$$

for a (uniquely determined) sequence $(a_n)_{n\in\mathbb{N}}$ of elements of $\mathbb{R}_{\geq 0}$ such that $\sum_{n\in\mathbb{N}} a_n \leq 1$.

Problem

$f'(x) = \sum_{n=0}^{\infty}(n+1)a_{n+1}x^n$ has no reason to satisfy $f' \in \mathcal{L}_!(1,1)$.

For instance if $f(x) = x^k$ then $f \in \mathbf{Pcoh}_!(1, 1)$, and $f'(x) = kx^{k-1}$ so that $f' \notin \mathcal{L}_!(1, 1)$ if $k > 1$.

For instance if $f(x) = x^k$ then $f \in \mathbf{Pcoh}_!(1,1)$, and $f'(x) = kx^{k-1}$ so that $f' \notin \mathcal{L}_!(1,1)$ if $k > 1$.

Even worse: f defined by $f(x) = 1 - \sqrt{1-x}$ belongs to $\mathbf{Pcoh}_!(1,1)$ but

$$f'(x) = \frac{1}{2\sqrt{1-x}}$$

is not event defined on the whole of $[0,1]$ and is not bounded on $[0,1)$...

For instance if $f(x) = x^k$ then $f \in \mathbf{Pcoh}_!(1,1)$, and $f'(x) = kx^{k-1}$ so that $f' \notin \mathcal{L}_!(1,1)$ if $k > 1$.

Even worse: f defined by $f(x) = 1 - \sqrt{1-x}$ belongs to $\mathbf{Pcoh}_!(1,1)$ but

$$f'(x) = \frac{1}{2\sqrt{1-x}}$$

is not event defined on the whole of $[0,1]$ and is not bounded on $[0,1)$...

... and we cannot reject f because it is the interpretation of a program!

Key observation

If $f \in \mathbf{Pcoh}_!(1,1)$ and

$$x, u \in [0,1] \text{ satisfy } x + u \in [0,1]$$

then by the Taylor formula at x

$$f(x+u) = f(x) + f'(x)u + \frac{1}{2}f''(x)u^2 + \cdots \in [0,1]$$

Key observation

If $f \in \mathbf{Pcoh}_!(1,1)$ and

$$x, u \in [0,1] \text{ satisfy } x + u \in [0,1]$$

then by the Taylor formula at x

$$f(x+u) = f(x) + f'(x)u + \frac{1}{2}f''(x)u^2 + \cdots \in [0,1]$$

and all these derivatives are ≥ 0, so we have

$$f(x) + f'(x)u \in [0,1]\,.$$

This is true even if $x = 1$ and $f'(x) = \infty$ if we stipulate that $\infty 0 = 0$ (NB: required for multiplication to be Scott continuous on $\overline{\mathbb{R}}$).

So if we define $S = \{(x, u) \in [0, 1]^2 \mid x + u \in [0, 1]\}$ we can define

$$\begin{aligned} \mathrm{D}f : S &\to S \\ (x, u) &\mapsto (f(x), f'(x)u) \end{aligned}$$

exactly as $\mathrm{T}f$ in differential geometry, tangent categories etc.

So if we define $S = \{(x, u) \in [0, 1]^2 \mid x + u \in [0, 1]\}$ we can define

$$\begin{aligned} \mathrm{D}f : S &\to S \\ (x, u) &\mapsto (f(x), f'(x)u) \end{aligned}$$

exactly as $\mathrm{T}f$ in differential geometry, tangent categories etc.

Fact

S can be seen as an object of **Pcoh** *and*

$$\forall f \in \mathbf{Pcoh}_!(1, 1) \quad \mathrm{D}f \in \mathbf{Pcoh}_!(S, S)$$

Moreover this observation is not limited to 1 but can be extended to all the objects of **Pcoh***.*

Canonical summability structure

In any categorical model $\mathcal{L}$ of LL (Seely category) with 0-morphisms and where 1 is the unit of the $\otimes$:

- $(1 \multimap X) \simeq X$.

Canonical summability structure

In any categorical model $\mathcal{L}$ of LL (Seely category) with 0-morphisms and where 1 is the unit of the $\otimes$:

- $(1 \multimap X) \simeq X$.
- $((1 \oplus 1) \multimap X) \simeq X \mathbin{\&} X$ is the object of pairs because $\otimes$ distributes over $\oplus$.

Canonical summability structure

In any categorical model $\mathcal{L}$ of LL (Seely category) with 0-morphisms and where 1 is the unit of the $\otimes$:

- $(1 \multimap X) \simeq X$.
- $((1 \oplus 1) \multimap X) \simeq X \mathbin{\&} X$ is the object of pairs because $\otimes$ distributes over $\oplus$.
- What about $((1 \mathbin{\&} 1) \multimap X)$?

Canonical summability structure

In any categorical model $\mathcal{L}$ of LL (Seely category) with 0-morphisms and where 1 is the unit of the $\otimes$:

- $(1 \multimap X) \simeq X$.
- $((1 \oplus 1) \multimap X) \simeq X \& X$ is the object of pairs because $\otimes$ distributes over $\oplus$.
- What about $((1 \& 1) \multimap X)$? In many cases it is "the object of pairs (x, u) of elements of X such that $x + u \in X$".

Canonical summability structure

In any categorical model $\mathcal{L}$ of LL (Seely category) with 0-morphisms and where 1 is the unit of the $\otimes$:

- $(1 \multimap X) \simeq X$.
- $((1 \oplus 1) \multimap X) \simeq X \mathbin{\&} X$ is the object of pairs because $\otimes$ distributes over $\oplus$.
- What about $((1 \mathbin{\&} 1) \multimap X)$? In many cases it is "the object of pairs (x, u) of elements of X such that $x + u \in X$".

We define the functor $\mathbf{S}X = ((1 \mathbin{\&} 1) \multimap X)$.

Canonical summability structure

In any categorical model $\mathcal{L}$ of LL (Seely category) with 0-morphisms and where 1 is the unit of the $\otimes$:

- $(1 \multimap X) \simeq X$.
- $((1 \oplus 1) \multimap X) \simeq X \,\&\, X$ is the object of pairs because $\otimes$ distributes over $\oplus$.
- What about $((1 \,\&\, 1) \multimap X)$? In many cases it is "the object of pairs (x, u) of elements of X such that $x + u \in X$".

We define the functor $\mathbf{S}X = ((1 \,\&\, 1) \multimap X)$.

Remark

If $\mathcal{L}$ is additive then $1 \,\&\, 1 = 1 \oplus 1$ so that $\mathbf{S}X = X \,\&\, X$ and we retreive the fact that addition is always possible.

In general $\mathbf{S}X$ is somewhere in between $X \oplus X$ and $X \,\&\, X$.

$\mathbf{S}X = ((1 \mathbin{\&} 1) \multimap X)$ in several models

[For those who know these models, if you don't, don't worry!]

- In **Rel**, $\mathcal{P}(\mathbf{S}X) = \mathcal{P}(X) \times \mathcal{P}(X)$: **Rel** is an additive category.

$\mathbf{S}X = ((1 \mathbin{\&} 1) \multimap X)$ in several models

[For those who know these models, if you don't, don't worry!]

- In **Rel**, $\mathcal{P}(\mathbf{S}X) = \mathcal{P}(X) \times \mathcal{P}(X)$: **Rel** is an additive category.
- In the category **Coh** of Girard's coherence spaces

$$\mathrm{Cl}(\mathbf{S}X) = \{(x, u) \in \mathrm{Cl}(X)^2 \mid x \cup u \in \mathrm{Cl}(X) \text{ and } x \cap u = \emptyset\}\,.$$

$\mathbf{S}X = ((1 \mathbin{\&} 1) \multimap X)$ in several models

[For those who know these models, if you don't, don't worry!]

- In **Rel**, $\mathcal{P}(\mathbf{S}X) = \mathcal{P}(X) \times \mathcal{P}(X)$: **Rel** is an additive category.
- In the category **Coh** of Girard's coherence spaces

$$\mathrm{Cl}(\mathbf{S}X) = \{(x, u) \in \mathrm{Cl}(X)^2 \mid x \cup u \in \mathrm{Cl}(X) \text{ and } x \cap u = \emptyset\} .$$

- In the (more confidential but well-behaved) category of non-uniform coherence spaces

$$\mathrm{Cl}(\mathbf{S}X) = \{(x, u) \in \mathrm{Cl}(X)^2 \mid \forall a \in x \, \forall b \in u \;\; a \frown_X b\} .$$

[Difference wrt. Girard's coherence spaces: coherence is *not required* to be reflexive. So we can have $(x, u) \in \mathrm{Cl}(\mathbf{S}X)$ and $x \cap u \neq \emptyset$. Main benefit: $|!X| = \mathcal{M}_{\mathrm{fin}}(|X|)$.]

- In the category **Pcoh**: an object is a pair $X = (|X|, \mathrm{P}X)$ where $|X|$ is a set and $\mathrm{P}X \subseteq (\mathbb{R}_{\geq 0})^{|X|}$ subject to some closure properties.

- In the category **Pcoh**: an object is a pair $X = (|X|, \mathsf{P}X)$ where $|X|$ is a set and $\mathsf{P}X \subseteq (\mathbb{R}_{\geq 0})^{|X|}$ subject to some closure properties. Then

$$\mathsf{P}(\mathbf{S}X) = \{(x, u) \in \mathsf{P}X^2 \mid x + u \in \mathsf{P}X\}\,.$$

Example: in this model the type $1 \oplus 1$ of booleans is

$$\begin{aligned} |1 \oplus 1| &= \{\mathbf{t}, \mathbf{f}\} \\ \mathsf{P}(1 \oplus 1) &= \{x \in (\mathbb{R}_{\geq 0})^{\{\mathbf{t},\mathbf{f}\}} \mid x_{\mathbf{t}} + x_{\mathbf{f}} \leq 1\}\,. \end{aligned}$$

Main properties of $\mathbf{S}$

So we have this functor $\mathbf{S} : \mathcal{L} \to \mathcal{L}$, what are its basic properties?

- There are 2 obvious projections $\pi_i \in \mathcal{L}(\mathbf{S}X, X)$, for $i = 0, 1$.
- And there is an "addition" $\sigma \in \mathcal{L}(\mathbf{S}X, X)$.

Main properties of $\mathbf{S}$

So we have this functor $\mathbf{S} : \mathcal{L} \to \mathcal{L}$, what are its basic properties?

- There are 2 obvious projections $\pi_i \in \mathcal{L}(\mathbf{S}X, X)$, for $i = 0, 1$.
- And there is an "addition" $\sigma \in \mathcal{L}(\mathbf{S}X, X)$.

These are natural transformations satisfying further properties.

Main properties of $\mathbf{S}$

So we have this functor $\mathbf{S} : \mathcal{L} \to \mathcal{L}$, what are its basic properties?

- There are 2 obvious projections $\pi_i \in \mathcal{L}(\mathbf{S}X, X)$, for $i = 0, 1$.
- And there is an "addition" $\sigma \in \mathcal{L}(\mathbf{S}X, X)$.

These are natural transformations satisfying further properties.

Summability

$f_0, f_1 \in \mathcal{L}(Y, X)$ are summable if there is $h \in \mathcal{L}(Y, \mathbf{S}X)$ such that and $\pi_i\ h = f_i$ for $i = 0, 1$.

So we have this functor $\mathbf{S} : \mathcal{L} \to \mathcal{L}$, what are its basic properties?

- There are 2 obvious projections $\pi_i \in \mathcal{L}(\mathbf{S}X, X)$, for $i = 0, 1$.
- And there is an "addition" $\sigma \in \mathcal{L}(\mathbf{S}X, X)$.

These are natural transformations satisfying further properties.

Summability

$f_0, f_1 \in \mathcal{L}(Y, X)$ are summable if there is $h \in \mathcal{L}(Y, \mathbf{S}X)$ such that and $\pi_i\ h = f_i$ for $i = 0, 1$.

And then one defines $f_0 + f_1 = \sigma\, h$.

Main properties of $\mathbf{S}$

So we have this functor $\mathbf{S} : \mathcal{L} \to \mathcal{L}$, what are its basic properties?

- There are 2 obvious projections $\pi_i \in \mathcal{L}(\mathbf{S}X, X)$, for $i = 0, 1$.
- And there is an "addition" $\sigma \in \mathcal{L}(\mathbf{S}X, X)$.

These are natural transformations satisfying further properties.

Summability

$f_0, f_1 \in \mathcal{L}(Y, X)$ are summable if there is $h \in \mathcal{L}(Y, \mathbf{S}X)$ such that and $\pi_i\ h = f_i$ for $i = 0, 1$.

And then one defines $f_0 + f_1 = \sigma\, h$.

Our axioms guarantee that h is uniquely determined by f_0, f_1 when it exists, that $(\mathcal{L}(X, Y), 0, +)$ is a partial commutative monoid etc.

NB: as in **Coh**, not all pairs of morphisms $X \to Y$ are necessarily summable!

Differentiation

If $f \in \mathcal{L}_!(X, Y) = \mathcal{L}(!X, Y)$ is a *non-linear* morphism, we would like to define its derivative

$$\mathbf{D}f \in \mathcal{L}_!(\mathbf{S}X, \mathbf{S}Y) = \mathcal{L}(!\mathbf{S}X, \mathbf{S}Y)$$

Differentiation

If $f \in \mathcal{L}_!(X, Y) = \mathcal{L}(!X, Y)$ is a *non-linear* morphism, we would like to define its derivative

$$\mathbf{D}f \in \mathcal{L}_!(\mathbf{S}X, \mathbf{S}Y) = \mathcal{L}(!\mathbf{S}X, \mathbf{S}Y)$$

Of course we have $\mathbf{S}f \in \mathcal{L}(\mathbf{S}!X, \mathbf{S}Y)$, ok for the target but not for the source!

If $f \in \mathcal{L}_!(X, Y) = \mathcal{L}(!X, Y)$ is a *non-linear* morphism, we would like to define its derivative

$$\mathbf{D}f \in \mathcal{L}_!(\mathbf{S}X, \mathbf{S}Y) = \mathcal{L}(!\mathbf{S}X, \mathbf{S}Y)$$

Of course we have $\mathbf{S}f \in \mathcal{L}(\mathbf{S}!X, \mathbf{S}Y)$, ok for the target but not for the source!

Main idea

Differentiation = distributive law $\partial_X \in \mathcal{L}(!\mathbf{S}X, \mathbf{S}!X)$ between $\mathbf{S}$ (which is actually a monad) and the LL resource comonad "!".

Differentiation

If $f \in \mathcal{L}_!(X, Y) = \mathcal{L}(!X, Y)$ is a *non-linear* morphism, we would like to define its derivative

$$\mathbf{D}f \in \mathcal{L}_!(\mathbf{S}X, \mathbf{S}Y) = \mathcal{L}(!\mathbf{S}X, \mathbf{S}Y)$$

Of course we have $\mathbf{S}f \in \mathcal{L}(\mathbf{S}!X, \mathbf{S}Y)$, ok for the target but not for the source!

Main idea

Differentiation = distributive law $\partial_X \in \mathcal{L}(!\mathbf{S}X, \mathbf{S}!X)$ between $\mathbf{S}$ (which is actually a monad) and the LL resource comonad "!".

Fact

Such a structure is available in many models of LL (coherence spaces, non-uniform coherence spaces, probabilistic coherence spaces).

Following the standard "extension to the Kleisli category" allowed by a distributive law:

$$\mathbf{D}X = \mathbf{S}X$$
$$\mathbf{D}f = \mathbf{S}f\ \partial_X \in \mathcal{L}(!\mathbf{S}X, \mathbf{S}Y) = \mathcal{L}_!(\mathbf{S}X, \mathbf{S}Y).$$

Fact

Differentiation becomes a functor $\mathbf{D}$*, actually a commutative strong monad, on the CCC* $\mathcal{L}_!$*.*

∂ in canonical models

Remember, in a canonical model of CD we take

$$\mathbf{S}X = (1 \mathbin{\&} 1 \multimap X)$$

∂ in canonical models

Remember, in a canonical model of CD we take

$$\mathbf{S}X = (1 \mathbin{\&} 1 \multimap X)$$

so $\partial_X \in \mathcal{L}(!\mathbf{S}X, \mathbf{S}!X)$ means

$$\partial_X : !(1 \mathbin{\&} 1 \multimap X) \to (1 \mathbin{\&} 1 \multimap !X)$$

and comes from a $\partial^0 \in \mathcal{L}(1 \mathbin{\&} 1, !(1 \mathbin{\&} 1))$ which is a !-coalgebra structure on $1 \mathbin{\&} 1$.

∂ in **Rel**

In **Rel** $1 \mathbin{\&} 1 = \{0, 1\}$ and then

$$(i, [i_1, \ldots, i_k]) \in \partial^0 \Leftrightarrow i = i_1 + \cdots + i_k$$

that is $i = 0 = i_1 = \cdots = i_k$ or $i = 1$ and $i_j = 0$ for all j but exactly one for which $i_j = 1$.

∂ in **Rel**

In **Rel** $1 \,\&\, 1 = \{0, 1\}$ and then

$$(i, [i_1, \dots, i_k]) \in \partial^0 \Leftrightarrow i = i_1 + \cdots + i_k$$

that is $i = 0 = i_1 = \cdots = i_k$ or $i = 1$ and $i_j = 0$ for all j but exactly one for which $i_j = 1$.

Then

$$\partial_X = \{([(i_1, a_1), \dots, (i_k, a_k)], (i, [a_1, \dots, a_k])) \mid k \in \mathbb{N}, \\ a_1, \dots, a_k \in X \;\; i, i_1, \dots, i_k \in \{0, 1\} \text{ and } i = i_1 + \cdots + i_k\}$$

In other words:

$$\begin{aligned}\partial_X = \{([(0,a_1),\dots,(0,a_k)],(0,[a_1,\dots,a_k])) \mid \\ k\in\mathbb{N}, a_1,\dots,a_k\in X\} \\ \cup\,\{([(0,a_1),\dots,(0,a_k)]+[(1,a)],(0,[a_1,\dots,a_k,a])) \mid \\ k\in\mathbb{N}, a_1,\dots,a_k,a\in X\}\end{aligned}$$

In **Pcoh**, ∂^0 is identical (now a matrix with $0,1\in\mathbb{R}_{\geq 0}$ coefficients) and ∂ is similar to the above, with integer coefficients corresponding to the k which appears in the derivative kx^{k-1} of x^k.

Question

Since the CCC $\mathcal{L}_!$ is typically a model of PCF, this means that we have a semantics for a differential extension of PCF.

What could it look like?

A differential PCF: Λ_{CD}

Types of Λ_{CD}

For simplicity only, just one data type ι of integers.

We could also have a type of booleans and many more discrete data types (recursive types): all these things exist in the models.

- Ground types: $\mathrm{D}^d\iota$ for each $d \in \mathbb{N}$. The type of *integers at depth* d.
- $A \Rightarrow B$ is a type if A, B are types.

Then one extends D to all types

$$\mathsf{D}(\mathsf{D}^d\iota) = \mathsf{D}^{d+1}\iota \qquad \mathsf{D}(A \Rightarrow B) = (A \Rightarrow \mathsf{D}B)$$

Intuition

$\mathsf{D}A$ is the type of pairs (u, v) with $u, v : A$ and $u + v : A$.

Then one extends D to all types

$$\mathsf{D}(\mathsf{D}^d\iota) = \mathsf{D}^{d+1}\iota \qquad\qquad \mathsf{D}(A \Rightarrow B) = (A \Rightarrow \mathsf{D}B)$$

Intuition

$\mathsf{D}A$ is the type of pairs (u, v) with $u, v : A$ and $u + v : A$.

So an $u : \mathsf{D}^d A$ should be thought of as a balanced tree with 2^d leaves labeled by elements $(u_\delta)_{\delta\in\{0,1\}^d}$ such that $\sum_{\delta\in\{0,1\}^d} u_\delta : A$

Then one extends D to all types

$$\mathsf{D}(\mathsf{D}^d\iota) = \mathsf{D}^{d+1}\iota \qquad\qquad \mathsf{D}(A \Rightarrow B) = (A \Rightarrow \mathsf{D}B)$$

Intuition

$\mathsf{D}A$ is the type of pairs (u, v) with $u, v : A$ and $u + v : A$.

So an $u : \mathsf{D}^d A$ should be thought of as a balanced tree with 2^d leaves labeled by elements $(u_\delta)_{\delta\in\{0,1\}^d}$ such that $\sum_{\delta\in\{0,1\}^d} u_\delta : A$

Warning: this is an *additive* and not a *multiplicative* tree!

Term syntax: 3 kinds of construct

- λ-calculus
- arithmetics
- differentiation and tree management.

$$
\begin{aligned}
M, N, \cdots := {} & x \mid \lambda x : A \cdot M \mid (M)N \mid \mathsf{Y} M \\
& \mid \underline{n} \mid \mathsf{ifz}^d(M, P, Q) \mid \mathsf{succ}^d(M) \mid \cdots \\
& \mid \mathsf{D} M \mid \pi_i^d(M) \mid \iota_i^d(M) \mid \theta^d(M) \mid \mathsf{c}_l^d(M) \mid 0^A \mid M + N
\end{aligned}
$$

The exponents d express at which depth in the tree $u : \mathsf{D}^e A$ (with $e \geq d$) the corresponding construct should be applied.

Ordinary typing rules

The λ-calculus rules are the usual ones.

$$\frac{i \in \{1, \ldots, n\}}{(x_1 : A_1, \ldots, x_n : A_n) \vdash x_i : A_i}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A \cdot M : A \Rightarrow B} \qquad \frac{\Gamma \vdash M : A \Rightarrow B \qquad \Gamma \vdash N : A}{\Gamma \vdash (M)N : B}$$

$$\frac{\Gamma \vdash M : A \Rightarrow A}{\Gamma \vdash \mathsf{Y} M : A}$$

The arithemtic rules must take depth into account, for instance

$$\frac{}{\Gamma \vdash \underline{n} : \iota} \qquad \frac{\Gamma \vdash M : \mathrm{D}^d \iota}{\Gamma \vdash \mathrm{succ}^d(M) : \mathrm{D}^d \iota}$$

$$\frac{\Gamma \vdash M : \mathrm{D}^d \iota \qquad \Gamma \vdash P : A \qquad \Gamma \vdash Q : A}{\Gamma \vdash \mathrm{ifz}^d(M, P, Q) : \mathrm{D}^d A}$$

Intuition for the typing of $\mathrm{succ}^d(M)$

If $d = 2$ and M represents $((u_{00}, u_{01}), (u_{10}, u_{11})$ then $\mathrm{succ}^2(M)$ represents $((\mathrm{succ}(u_{00}), \mathrm{succ}(u_{01})), (\mathrm{succ}(u_{10}), \mathrm{succ}(u_{11}))$.

Some differential / tree typing rules

$$\frac{\Gamma \vdash M : A \Rightarrow B}{\Gamma \vdash \mathrm{D}M : \mathrm{D}A \Rightarrow \mathrm{D}B}$$

Intuition

$\mathrm{D}M$ maps (x, u) to $(M(x), M'(x) \cdot u)$.

NB: just as in DiLL, differentiation makes sense at all types.

Some differential / tree typing rules

$$\frac{\Gamma \vdash M : A \Rightarrow B}{\Gamma \vdash \mathrm{D}M : \mathrm{D}A \Rightarrow \mathrm{D}B}$$

Intuition

$\mathrm{D}M$ maps (x, u) to $(M(x), M'(x) \cdot u)$.

NB: just as in DiLL, differentiation makes sense at all types.

$$\frac{\Gamma \vdash M : \mathrm{D}^{d+2}A}{\Gamma \vdash \theta^d(M) : \mathrm{D}^{d+1}A}$$

Intuition

If $M : \mathrm{D}^2A$ represents $((u_{00}, u_{01}), (u_{10}, u_{11}))$ then $\theta^0(M) : \mathrm{D}A$ represents $(u_{00}, u_{01} + u_{10})$

$$\frac{\Gamma \vdash M : \mathsf{D}^d A}{\Gamma \vdash \iota_0^d(M) : \mathsf{D}^{d+1} A}$$

Intuition

If $M : A$ represents u then $\iota_0^0(M)$ represents $(u, 0)$.

$$\frac{\Gamma \vdash M : \mathrm{D}^d A}{\Gamma \vdash \iota_0^d(M) : \mathrm{D}^{d+1} A}$$

Intuition

If $M : A$ represents u then $\iota_0^0(M)$ represents $(u, 0)$.

Similarly for $\iota_1^d(M)$ on the other side.

$$\frac{\Gamma \vdash M : \mathrm{D}^d A}{\Gamma \vdash \iota_0^d(M) : \mathrm{D}^{d+1} A}$$

Intuition

If $M : A$ represents u then $\iota_0^0(M)$ represents $(u, 0)$.

Similarly for $\iota_1^d(M)$ on the other side.

Dually

$$\frac{\Gamma \vdash M : \mathrm{D}^{d+1} A}{\Gamma \vdash \pi_i^d(M) : \mathrm{D}^d A}$$

implements the obvious projections for $i = 0, 1$.

$$\frac{\Gamma \vdash M : \mathsf{D}^{d+l+2} A}{\Gamma \vdash \mathsf{c}_l^d(M) : \mathsf{D}^{d+l+2} A}$$

which implements a *circular permutation* of length $l+2$ at depth d in the access words in the tree represented by M.

Example

If $d = 0$, $l = 1$ and $M : \mathsf{D}^3 A$ represents

$$(((u_{000}, u_{001}), (u_{010}, u_{011})), ((u_{100}, u_{101}), (u_{110}, u_{111})))$$

then $\mathsf{c}_1^0(M)$ represents

$$(((u_{000}, u_{100}), (u_{001}, u_{101})), ((u_{010}, u_{110}), (u_{011}, u_{111})))$$

Cf the *standard flip* in tangent categories.

Reduction rules

All rules are derived from the categorical semantics of CD.

The syntax itself of Λ_{CD} is strongly suggested by the semantics.

The differential reduction

Assuming $\Gamma, x : A \vdash M : B$

$$\mathrm{D}(\lambda x : A \cdot M) \to \lambda x : \mathrm{D}A \cdot \partial(x, M)$$

where $\partial(x, M)$ is an operation defined by induction on M such that $\Gamma, x : \mathrm{D}A \vdash \partial(x, M) : \mathrm{D}B$.

Remark

The definition of $\partial(x, M)$ is quasi-homomorphic wrt. the structure of terms.

The differential reduction

Assuming $\Gamma, x : A \vdash M : B$

$$\mathrm{D}(\lambda x : A \cdot M) \to \lambda x : \mathrm{D}A \cdot \partial(x, M)$$

where $\partial(x, M)$ is an operation defined by induction on M such that $\Gamma, x : \mathrm{D}A \vdash \partial(x, M) : \mathrm{D}B$.

Remark

The definition of $\partial(x, M)$ is quasi-homomorphic wrt. the structure of terms.

Sums induced by Leibniz are not performed immediately, their position are marked by the construct $\theta^d(\_)$.

The differential reduction

Assuming $\Gamma, x : A \vdash M : B$

$$\mathrm{D}(\lambda x : A \cdot M) \to \lambda x : \mathrm{D}A \cdot \partial(x, M)$$

where $\partial(x, M)$ is an operation defined by induction on M such that $\Gamma, x : \mathrm{D}A \vdash \partial(x, M) : \mathrm{D}B$.

Remark

The definition of $\partial(x, M)$ is quasi-homomorphic wrt. the structure of terms.

Sums induced by Leibniz are not performed immediately, their position are marked by the construct $\theta^d(\_)$.

Major difference wrt. the definition of $\frac{\partial M}{\partial x} \cdot N$ in the differential λ-calculus, which is a symbolic differentiation involving a lot of "$+$".

Some cases of the def. of $\partial(x, M)$

- $\partial(x, x) = x$
- $\partial(x, y) = \iota_0^0(y)$

Some cases of the def. of $\partial(x, M)$

- $\partial(x, x) = x$
- $\partial(x, y) = \iota_0^0(y)$
- $\partial(x, \lambda y : B \cdot M) = \lambda y : B \cdot \partial(x, M)$

We have $\Gamma, y : B, x : A \vdash M : C$ and hence by inductive hypothesis $\Gamma, y : B, x : \mathrm{D}A \vdash \partial(x, M) : \mathrm{D}C$ so

$$\frac{\Gamma, y : B, x : \mathrm{D}A \vdash \partial(x, M) : \mathrm{D}C}{\Gamma, x : \mathrm{D}A \vdash \lambda y : B \cdot \partial(x, M) : B \Rightarrow \mathrm{D}C}$$

and remember that $\mathrm{D}(B \Rightarrow C) = (B \Rightarrow \mathrm{D}C)$.

- $\partial(x, (M)N) = (\theta^0(\mathsf{D}\partial(x, M)))\partial(x, N)$

- $\partial(x,(M)N) = (\theta^0(\mathrm{D}\partial(x,M)))\partial(x,N)$

Indeed if $\Gamma, x : A \vdash M : B \Rightarrow C$ and $\Gamma, x : A \vdash N : B$

$$\dfrac{\dfrac{\dfrac{\Gamma, x : \mathrm{D}A \vdash \partial(x,M) : B \Rightarrow \mathrm{D}C}{\Gamma, x : \mathrm{D}A \vdash \mathrm{D}\partial(x,M) : \mathrm{D}B \Rightarrow \mathrm{D}^2 C}}{\Gamma, x : \mathrm{D}A \vdash \theta^0(\mathrm{D}\partial(x,M)) : \mathrm{D}B \Rightarrow \mathrm{D}C} \quad \Gamma, x : \mathrm{D}A \vdash \partial(x,N) : \mathrm{D}B}{\Gamma, x : \mathrm{D}A \vdash (\theta^0(\mathrm{D}\partial(x,M)))\partial(x,N) : \mathrm{D}C}$$

Notice that $(\mathrm{D}B \Rightarrow \mathrm{D}^2 C) = \mathrm{D}^2(\mathrm{D}B \Rightarrow C)$.

Compare

$$\partial(x,(M)N) = (\theta^0(\mathrm{D}\partial(x,M)))\partial(x,N)$$

with the DiLL definition

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

Remark

In $\partial(x,(M)N)$ the sum is not performed, $\theta^0(\_)$ indicates that it should be performed at some point.

Compare

$$\partial(x,(M)N) = (\theta^0(\mathrm{D}\partial(x,M)))\partial(x,N)$$

with the DiLL definition

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

Remark

In $\partial(x,(M)N)$ the sum is not performed, $\theta^0(\_)$ indicates that it should be performed at some point.

$\partial(x,(M)N)$ involves a double derivative (D and ∂) but the $\theta^0(\_)$ drops the "degree 2" component.

Works also for fixpoints.

- $\partial(x, \mathrm{Y}M) = \mathrm{Y}\theta^0(\mathrm{D}\partial(x, M))$

Remark

Apart **Rel** or models with ∞ coefficients (*eg.* profunctor models) there are no “standard” models of DiLL with general fixpoints.

The $\mathsf{c}_l^d(M)$ construct plays a crucial role, for instance

- $\partial(x, \mathrm{D}M) = \mathsf{c}_0^0(\mathrm{D}\partial(x, M))$

typed as follows, assuming that $\Gamma, x : A \vdash M : B \Rightarrow C$ so that $\Gamma, x : A \vdash \mathrm{D}M : \mathrm{D}B \Rightarrow \mathrm{D}C$

$$\dfrac{\dfrac{\Gamma, x : \mathrm{D}A \vdash \partial(x, M) : B \Rightarrow \mathrm{D}C}{\Gamma, x : \mathrm{D}A \vdash \mathrm{D}\partial(x, M) : \mathrm{D}B \Rightarrow \mathrm{D}^2 C}}{\Gamma, x : \mathrm{D}A \vdash \mathsf{c}_0^0(\mathrm{D}\partial(x, M)) : \mathrm{D}B \Rightarrow \mathrm{D}^2 C}$$

The $c_l^d(M)$ construct plays a crucial role, for instance

- $\partial(x, \mathrm{D}M) = c_0^0(\mathrm{D}\partial(x, M))$

typed as follows, assuming that $\Gamma, x : A \vdash M : B \Rightarrow C$ so that $\Gamma, x : A \vdash \mathrm{D}M : \mathrm{D}B \Rightarrow \mathrm{D}C$

$$\dfrac{\dfrac{\Gamma, x : \mathrm{D}A \vdash \partial(x, M) : B \Rightarrow \mathrm{D}C}{\Gamma, x : \mathrm{D}A \vdash \mathrm{D}\partial(x, M) : \mathrm{D}B \Rightarrow \mathrm{D}^2 C}}{\Gamma, x : \mathrm{D}A \vdash c_0^0(\mathrm{D}\partial(x, M)) : \mathrm{D}B \Rightarrow \mathrm{D}^2 C}$$

Remark

Semantically, $\partial(x, \mathrm{D}M) = c_0^0(\mathrm{D}\partial(x, M))$ is the Schwarz lemma:

$$\frac{\partial^2 f(x, y)}{\partial x \partial y} \cdot (u, v) = \frac{\partial^2 f(x, y)}{\partial y \partial x} \cdot (v, u)$$

Why do basic operations act at arbitrary depths d?

This is required by the definition of $\partial(x, M)$.

The case of the successor.

- $\partial(x, \mathrm{succ}^d(M)) = \mathrm{succ}^{d+1}(\partial(x, M))$

Assuming $\Gamma, x : A \vdash M : \mathrm{D}^d\iota$ we have:

$$\frac{\Gamma, x : \mathrm{D}A \vdash \partial(x, M) : \mathrm{D}^{d+1}\iota}{\Gamma, x : \mathrm{D}A \vdash \mathrm{succ}^{d+1}(\partial(x, M)) : \mathrm{D}^{d+1}\iota}$$

Reflects the linearity of $\mathrm{succ}(\_)$.

Guided again by the semantics we set:

- $\partial(x, \mathrm{if}^d(M, P_1, P_2)) = \theta^0(\mathrm{c}^0_d(\mathrm{if}^{d+1}(\partial(x, M), \partial(x, P_1), \partial(x, P_2))))$

Intuitively

The $\mathrm{c}^0_d(\_)$ cyclic flip of length $d + 2$ is required for having the two levels at which $\theta^0(\_)$ acts close to one another.

Other reduction rules

Standard β-rules:

- $(\lambda x : A \cdot M)N \to M[N/x]$
- $\mathsf{Y}M \to (M)\mathsf{Y}M$

Other reduction rules

Standard β-rules:

- $(\lambda x : A \cdot M)N \to M[N/x]$
- $\mathrm{Y}M \to (M)\mathrm{Y}M$

δ-rules for computing with integers, for instance:

- $\mathrm{succ}^{0}(\underline{n}) \to \underline{n+1}$
- $\mathrm{if}^{0}(\underline{n+1}, P_1, P_2) \to P_2$

Projection rules

Many tree reduction rules, explaining how the projection $\pi_i^d(\_)$ construct interact with the other ones (including itself). We mention only a few of them.

- $\pi_0^d(\theta^d(M)) \to \pi_0^d(\pi_0^d(M))$
- $\pi_1^d(\theta^d(M)) \to \pi_1^d(\pi_0^d(M)) + \pi_0^d(\pi_1^d(M))$: what remains from the non-determinism of DiLL.
- $\pi_i^d(\iota_i^d(M)) \to M$
- $\pi_i^d(\iota_{1-i}^d(M)) \to 0$

Denotational Semantics

Any model of CD $\mathcal{L}$ with some additional *local completeness* properties (true in most known models such as **Coh**, **Pcoh** etc) is a denotational model of Λ_{CD}:

- type $A \rightsquigarrow$ object $[\![A]\!]$ of $\mathcal{L}$
- M with $x_1 : A_1, \ldots, x_n : A_n \vdash M : B \rightsquigarrow$ $[\![M]\!] \in \mathcal{L}_!([\![A_1]\!] \mathbin{\&} \cdots \mathbin{\&} [\![A_n]\!], [\![B]\!])$

Denotational Semantics

Any model of CD $\mathcal{L}$ with some additional *local completeness* properties (true in most known models such as **Coh**, **Pcoh** etc) is a denotational model of Λ_{CD}:

- type $A \rightsquigarrow$ object $[\![A]\!]$ of $\mathcal{L}$
- M with $x_1 : A_1, \ldots, x_n : A_n \vdash M : B \rightsquigarrow$ $[\![M]\!] \in \mathcal{L}_!([\![A_1]\!] \,\&\, \cdots \,\&\, [\![A_n]\!], [\![B]\!])$

Soundness

$M \to M' \Rightarrow [\![M]\!] = [\![M']\!]$

Denotational Semantics

Any model of CD $\mathcal{L}$ with some additional *local completeness* properties (true in most known models such as **Coh**, **Pcoh** etc) is a denotational model of Λ_{CD}:

- type $A \rightsquigarrow$ object $[\![A]\!]$ of $\mathcal{L}$
- M with $x_1 : A_1, \ldots, x_n : A_n \vdash M : B \rightsquigarrow$ $[\![M]\!] \in \mathcal{L}_!([\![A_1]\!] \,\&\, \cdots \,\&\, [\![A_n]\!], [\![B]\!])$

Soundness

$M \to M' \Rightarrow [\![M]\!] = [\![M']\!]$

The question of completeness

If $\vdash M : \iota$ satisfies $[\![M]\!] = \nu \in \mathbb{N}$, is it true that $M \to^* \underline{\nu}$?

Denotational Semantics

Any model of CD $\mathcal{L}$ with some additional *local completeness* properties (true in most known models such as **Coh**, **Pcoh** etc) is a denotational model of Λ_{CD}:

- type $A \rightsquigarrow$ object $[\![A]\!]$ of $\mathcal{L}$
- M with $x_1 : A_1, \ldots, x_n : A_n \vdash M : B \rightsquigarrow$ $[\![M]\!] \in \mathcal{L}_!([\![A_1]\!] \mathbin{\&} \cdots \mathbin{\&} [\![A_n]\!], [\![B]\!])$

Soundness

$M \to M' \Rightarrow [\![M]\!] = [\![M']\!]$

The question of completeness

If $\vdash M : \iota$ satisfies $[\![M]\!] = \nu \in \mathbb{N}$, is it true that $M \to^* \underline{\nu}$?

In other words: do we have enough reduction rules?

A Krivine machine

A complete reduction strategy described as a Krivine machine.

The machine can be made fully deterministic, this is a great novelty wrt. DiLL!

States of the machine

$$c := (\delta, M, s) \mid 0 \mid c_1 + c_2$$

States of the machine

$$c := (\delta, M, s) \mid 0 \mid c_1 + c_2$$

- $\delta = (\delta_1, \ldots, \delta_d) \in \{0,1\}^d$ for some $d \in \mathbb{N}$, the access word.

States of the machine

$$c := (\delta, M, s) \mid 0 \mid c_1 + c_2$$

- $\delta = (\delta_1, \ldots, \delta_d) \in \{0, 1\}^d$ for some $d \in \mathbb{N}$, the access word.
- M is a term such that $\vdash M : \mathrm{D}^d F$ where F is a sharp type, that is a type which is not of shape $\mathrm{D}A$, that is

$$F = (A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow \iota)$$

States of the machine

$$c := (\delta, M, s) \mid 0 \mid c_1 + c_2$$

- $\delta = (\delta_1, \ldots, \delta_d) \in \{0,1\}^d$ for some $d \in \mathbb{N}$, the access word.
- M is a term such that $\vdash M : \mathrm{D}^d F$ where F is a sharp type, that is a type which is not of shape $\mathrm{D}A$, that is

$$F = (A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow \iota)$$

- s is a stack such that $s : F \vdash \iota$ to be understood as a *linear* context $s[\ ]$ of type ι with a hole of type F.

States of the machine

$$c := (\delta, M, s) \mid 0 \mid c_1 + c_2$$

- $\delta = (\delta_1, \ldots, \delta_d) \in \{0,1\}^d$ for some $d \in \mathbb{N}$, the access word.
- M is a term such that $\vdash M : \mathrm{D}^d F$ where F is a sharp type, that is a type which is not of shape $\mathrm{D}A$, that is

$$F = (A_1 \Rightarrow \cdots \Rightarrow A_n \Rightarrow \iota)$$

- s is a stack such that $s : F \vdash \iota$ to be understood as a *linear* context $s[\ ]$ of type ι with a hole of type F.

The sate (δ, M, s) represents the term

$$\vdash s[\pi^0_{\delta_1}(\cdots \pi^0_{\delta_d}(M) \cdots)] : \iota$$

Syntax and typing of stacks

$$\frac{}{() : \iota \vdash \iota} \qquad \frac{s : \iota \vdash \iota}{\mathsf{succ} \cdot s : \iota \vdash \iota} \qquad \frac{s : \iota \vdash \iota}{\mathsf{pred} \cdot s : \iota \vdash \iota}$$

$$\frac{s : E \vdash \iota \qquad \vdash M_0 : \mathrm{D}^d E \qquad \vdash M_1 : \mathrm{D}^d E \qquad \mathrm{len}(\delta) = d}{\mathsf{if}(\delta, M_0, M_1) \cdot s : \iota \vdash \iota}$$

$$\frac{\vdash M : A \qquad s : E \vdash \iota}{\mathsf{arg}(M) \cdot s : A \Rightarrow E \vdash \iota}$$

$$\frac{s : \mathrm{D}A \Rightarrow E \vdash \iota \qquad i \in \{0, 1\}}{\mathrm{D}(i) \cdot s : A \Rightarrow E \vdash \iota}$$

Standard PCF reduction

$$
\begin{aligned}
(\delta, (M)N, s) &\to (\delta, M, \mathrm{arg}(N) \cdot s) \\
(\delta, \lambda x : A \cdot M, \mathrm{arg}(N) \cdot s) &\to (\delta, M\,[N/x]\,, s) \\
(\delta, \mathrm{Y}M, s) &\to (\delta, M, \mathrm{arg}(\mathrm{Y}M) \cdot s)
\end{aligned}
$$

Standard PCF reduction

$$
\begin{aligned}
(\delta, (M)N, s) &\to (\delta, M, \mathrm{arg}(N)\cdot s)\\
(\delta, \lambda x : A \cdot M, \mathrm{arg}(N)\cdot s) &\to (\delta, M\,[N/x]\,, s)\\
(\delta, \mathrm{Y}M, s) &\to (\delta, M, \mathrm{arg}(\mathrm{Y}M)\cdot s)
\end{aligned}
$$

$$
\begin{aligned}
(\delta, \mathrm{succ}^d(M), s) &\to (\delta, M, \mathrm{succ}\cdot s) \quad \text{with } d = \mathrm{len}(\delta)\\
(\langle\,\rangle, \underline{n}, \mathrm{succ}\cdot s) &\to (\langle\,\rangle, \underline{n+1}, s)
\end{aligned}
$$

With $\vdash M : \mathrm{D}^d\iota$ and $s : \iota \vdash \iota$.

Remember

In $\mathrm{succ}^d(M)$, succ acts at depth d in the tree represented by M.

And δ specifies exactly one leaf of this tree.

$$(\varepsilon\delta, \mathsf{if}^d(M, P_0, P_1), s) \to (\delta, M, \mathsf{if}(\varepsilon, P_0, P_1) \cdot s)$$
$$\text{with } d = \mathsf{len}(\delta)$$

With $\vdash M : \mathsf{D}^d\iota$ and $\vdash P_i : \mathsf{D}^e F$ with $e = \mathsf{len}(\varepsilon)$ and F sharp.

$$(\langle\,\rangle, \underline{0}, \mathsf{if}(\varepsilon, P_0, P_1) \cdot s) \to (\varepsilon, P_0, s)$$
$$(\langle\,\rangle, \underline{n+1}, \mathsf{if}(\varepsilon, P_0, P_1) \cdot s) \to (\varepsilon, P_1, s)$$

Interpretation

The access path ε to the result of P_0 or P_1 is stored in the stack, together with P_0 or P_1, the two possible continuations.

Differential reductions

$$(\delta i, \mathrm{D}M, s) \to (\delta, M, \mathrm{D}(i) \cdot s)$$

With $\vdash M : A \Rightarrow \mathrm{D}^d F$ (and F is sharp) and hence $\vdash \mathrm{D}M : \mathrm{D}A \Rightarrow \mathrm{D}^{d+1}F$, $s : \mathrm{D}A \Rightarrow F \vdash \iota$

Interpretation

We store in the stack:

- the instruction that M must be differentiated
- the index $i \in \{0, 1\}$ of the component of this differential which must be kept.

$$(\delta, \lambda x : A \cdot M, \mathsf{D}(i) \cdot s) \to (\delta i, \lambda x : \mathsf{D}A \cdot \partial(x, M), s)$$

Interpretation

We have the function explicitly as a λ so we can apply the differential reduction.

$$(\delta, \lambda x : A \cdot M, \mathsf{D}(i) \cdot s) \to (\delta i, \lambda x : \mathsf{D}A \cdot \partial(x, M), s)$$

Interpretation

We have the function explicitly as a λ so we can apply the differential reduction.

We use the operation $\partial(x, M)$ defined by induction on M, just as the substitution $M\,[N/x]$ in the standard β step $(\delta, \lambda x : A \cdot M, \mathrm{arg}(N) \cdot s) \to (\delta, M\,[N/x]\,, s)$.

$$(\delta, \lambda x : A \cdot M, \mathrm{D}(i) \cdot s) \to (\delta i, \lambda x : \mathrm{D}A \cdot \partial(x, M), s)$$

Interpretation

We have the function explicitly as a λ so we can apply the differential reduction.

We use the operation $\partial(x, M)$ defined by induction on M, just as the substitution $M\,[N/x]$ in the standard β step $(\delta, \lambda x : A \cdot M, \mathrm{arg}(N) \cdot s) \to (\delta, M\,[N/x]\,, s)$.

Remember: these two operations are homomorphic wrt. M.

Access word management steps

$$
\begin{aligned}
(\varepsilon i\delta, \iota_i^d(M), s) &\to (\varepsilon\delta, M, s)\\
(\varepsilon i\delta, \iota_{1-i}^d(M), s) &\to 0
\end{aligned}
$$

with $d = \mathrm{len}(\delta)$.

Interpretation

The projection meets an injection, at depth d.

The state 0 corresponds to the 0 of the semantics and can be interpreted as an error state.

If $d = \mathrm{len}(\delta)$,

$$(\varepsilon 0\delta, \theta^d(M), s) \to (\varepsilon 00\delta, M, s)$$
$$(\varepsilon 1\delta, \theta^d(M), s) \to (\varepsilon 01\delta, M, s) + (\varepsilon 10\delta, M, s)$$

Interpretation

If $\Gamma \vdash P : \mathrm{D}^2 A$ represents $((u_{00}, u_{01}), (u_{10}, u_{11}))$
then $\Gamma \vdash \theta^0(P) : \mathrm{D}A$ represents $(u_{00}, u_{01} + u_{10})$.

If $d = \mathrm{len}(\delta)$,

$$(\varepsilon 0\delta, \theta^d(M), s) \to (\varepsilon 00\delta, M, s)$$
$$(\varepsilon 1\delta, \theta^d(M), s) \to (\varepsilon 01\delta, M, s) + (\varepsilon 10\delta, M, s)$$

Interpretation

If $\Gamma \vdash P : \mathrm{D}^2 A$ represents $((u_{00}, u_{01}), (u_{10}, u_{11}))$
then $\Gamma \vdash \theta^0(P) : \mathrm{D}A$ represents $(u_{00}, u_{01} + u_{10})$.

Remark

This is the very last bit of differential "non-determinism" left.

If $d = \mathrm{len}(\delta)$,

$$(\varepsilon 0\delta, \theta^d(M), s) \to (\varepsilon 00\delta, M, s)$$
$$(\varepsilon 1\delta, \theta^d(M), s) \to (\varepsilon 01\delta, M, s) + (\varepsilon 10\delta, M, s)$$

Interpretation

If $\Gamma \vdash P : \mathrm{D}^2 A$ represents $((u_{00}, u_{01}), (u_{10}, u_{11}))$
then $\Gamma \vdash \theta^0(P) : \mathrm{D}A$ represents $(u_{00}, u_{01} + u_{10})$.

Remark

This is the very last bit of differential "non-determinism" left.

This $+$ occurs at ground type so the denotational semantics tells us that at most one of the two summands can produce sthg $\neq 0$.

If $d = \mathrm{len}(\delta)$,

$$(\varepsilon 0\delta, \theta^d(M), s) \to (\varepsilon 00\delta, M, s)$$
$$(\varepsilon 1\delta, \theta^d(M), s) \to (\varepsilon 01\delta, M, s) + (\varepsilon 10\delta, M, s)$$

Interpretation

If $\Gamma \vdash P : \mathrm{D}^2 A$ represents $((u_{00}, u_{01}), (u_{10}, u_{11}))$ then $\Gamma \vdash \theta^0(P) : \mathrm{D}A$ represents $(u_{00}, u_{01} + u_{10})$.

Remark

This is the very last bit of differential "non-determinism" left.

This $+$ occurs at ground type so the denotational semantics tells us that at most one of the two summands can produce sthg $\neq 0$.

A slight change in the machine allows to get rid of it, thank you Guillaume Geoffroy! (*Hint:* make the access word writable.)

$$(\varepsilon\alpha\delta, \mathsf{c}_l^d(M), s) \to (\varepsilon \underset{\rightarrow}{\alpha} \delta, M, s)$$

with $\mathsf{len}(\delta) = d$ and $\mathsf{len}(\alpha) = l + 2$ and

$$\text{if } \alpha = \langle \alpha_1, \ldots, \alpha_{l+2} \rangle$$
$$\text{then } \underset{\rightarrow}{\alpha} = \langle \alpha_{l+2}, \alpha_1, \ldots, \alpha_{l+1} \rangle$$

$$(\varepsilon\alpha\delta, \mathsf{c}_l^d(M), s) \to (\varepsilon \underset{\rightarrow}{\alpha} \delta, M, s)$$

with $\mathrm{len}(\delta) = d$ and $\mathrm{len}(\alpha) = l + 2$ and

$$\text{if } \alpha = \langle \alpha_1, \ldots, \alpha_{l+2} \rangle$$
$$\text{then } \underset{\rightarrow}{\alpha} = \langle \alpha_{l+2}, \alpha_1, \ldots, \alpha_{l+1} \rangle$$

Remark

Even if we run an initial state $(\langle\rangle, N, ())$ with $\vdash N : \iota$ and N contains no $\mathsf{c}_l^d(M)$, such constructs will be created by the $\partial(x, \_)$ operation during the execution if N contains $\mathsf{D}_\_$ constructs.

$$(\varepsilon\delta, \pi_i^d(M), s) \to (\varepsilon i\delta, M, s) \quad \text{with } d = \mathrm{len}(\delta)$$

Interpretation

Insertion of the projection identifier (a bit) at the right place in the access word.

Termination and determinism

Let $c = (\langle\,\rangle, M, ())$ with $\vdash M : \iota$.

Let $c = (\langle\,\rangle, M, ())$ with $\vdash M : \iota$.

Applying the reductions above we get *formal sums* (= finite multisets) $C = c_1 + \cdots + c_n$ of "simple" states $(c_i = (\delta_i, M_i, s_i))_{i=1}^{n}$.

Let $c = (\langle\,\rangle, M, ())$ with $\vdash M : \iota$.

Applying the reductions above we get *formal sums* (= finite multisets) $C = c_1 + \cdots + c_n$ of "simple" states $(c_i = (\delta_i, M_i, s_i))_{i=1}^n$.

In **Pcoh** one can interpret any such C as an element

$$[\![C]\!] = \sum_{i=1}^{n} [\![c_i]\!] = u \in \mathsf{PN}$$

where u is a probability subdistribution on $\mathbb{N}$, and we have

$$u = [\![M]\!]$$

(invariance of the reduction).

Determinism

Moreover we know that all the probabilities in this distribution are integers (simple analysis of the interpretation of terms)!

This means that

- either $u = 0$
- or $\exists! \nu \in \mathbb{N}$ such that $u = e_\nu$ (the probability distribution such that $e_\nu(\nu') = \delta_{\nu,\nu'}$).

Theorem

In the second case $(\langle\,\rangle, M, ()) \to^* (\langle\,\rangle, \underline{\nu}, ())$.

The proof uses an adaptation of the reducibility method involving all iterated derivatives of terms.

Conclusion

- DiLL has shown its relevance *eg.* as a foundation for Taylor expanding proofs and programs.

- DiLL has shown its relevance *eg.* as a foundation for Taylor expanding proofs and programs.
- Though, its operational meaning has always been unclear. Mainly because of its Leibniz related non-determinism.

- DiLL has shown its relevance *eg.* as a foundation for Taylor expanding proofs and programs.
- Though, its operational meaning has always been unclear. Mainly because of its Leibniz related non-determinism.
- CD proposes a new paradigm combining DiLL-like differentiation with full determinism of computations, and compatible with arbitrary recursive definitions.

- DiLL has shown its relevance *eg.* as a foundation for Taylor expanding proofs and programs.
- Though, its operational meaning has always been unclear. Mainly because of its Leibniz related non-determinism.
- CD proposes a new paradigm combining DiLL-like differentiation with full determinism of computations, and compatible with arbitrary recursive definitions.
- CD has also purely "domain theoretic" models such as (non-uniform) coherence spaces, without numerical coefficients.

- DiLL has shown its relevance *eg.* as a foundation for Taylor expanding proofs and programs.
- Though, its operational meaning has always been unclear. Mainly because of its Leibniz related non-determinism.
- CD proposes a new paradigm combining DiLL-like differentiation with full determinism of computations, and compatible with arbitrary recursive definitions.
- CD has also purely “domain theoretic” models such as (non-uniform) coherence spaces, without numerical coefficients.
- This suggests that differentiation has a general programming meaning.

- DiLL has shown its relevance *eg.* as a foundation for Taylor expanding proofs and programs.
- Though, its operational meaning has always been unclear. Mainly because of its Leibniz related non-determinism.
- CD proposes a new paradigm combining DiLL-like differentiation with full determinism of computations, and compatible with arbitrary recursive definitions.
- CD has also purely "domain theoretic" models such as (non-uniform) coherence spaces, without numerical coefficients.
- This suggests that differentiation has a general programming meaning.
- It remains to understand which one!