Differential Combinatory Algebras

June 26, 2012

Beginning (was on board)

1

[1] The beginning of this talk was given on the board. Here we try to recap the interactive feel of the board.

Beginning (was on board)

Applicative systems

Beginning (was on board)

Applicative systems

$$\_ \bullet \_ : A \times A \longrightarrow A$$

Beginning (was on board)

Applicative systems

$$\underbrace{\_}_{\text{“function”}} \quad \bullet\_ : A \times A \to A$$

Beginning (was on board)

Applicative systems

$$\underbrace{\_}_{\text{“function”}} \quad \bullet\_ : A \times A \to A$$

$$\frac{m : a \to b \quad n : a}{mn : a \to b}$$

How to make linear resource use available (on board)

$$m : a \to b$$

How to make linear resource use available (on board)

$$\frac{m : a \to b}{m : !a \multimap b}\ \text{Linear}\quad \text{Decompose}$$

How to make linear resource use available (on board)

$$\dfrac{\dfrac{m : a \longrightarrow b}{m :\, !a \multimap b}}{\delta[m] : (a \times\, !a) \multimap b}\ \text{Coderlict}$$

How to make linear resource use available (on board)

$$\dfrac{\dfrac{\dfrac{m : a \longrightarrow b}{m : !a \multimap b}}{\delta[m] : (a \times !a) \multimap b}}{\delta[m] : a \multimap (!a \multimap b)}\ \text{Curry}$$

How to make linear resource use available (on board)

$$\dfrac{\dfrac{\dfrac{\dfrac{m : a \to b}{m :\, !a \multimap b}}{\delta[m] : (a \times\, !a) \multimap b}}{\delta[m] : a \multimap (!a \multimap b)}}{\delta[m] : a \multimap (a \to b)}$$

How to make linear resource use available (on board)

$$\dfrac{\dfrac{\dfrac{\dfrac{m : a \to b}{m :\, !a \multimap b}}{\delta[m] : (a \times\, !a) \multimap b}}{\delta[m] : a \multimap (!a \multimap b)}}{\delta[m] : a \multimap (a \to b)}$$

Given that $m : a \to b$, $\delta[m]$ makes a linear use of one thing of type a, and then returns the rest of the function $a \to b$.

How to make linear resource use available (on board)

$$\dfrac{\dfrac{\dfrac{\dfrac{m : a \longrightarrow b}{m :\, !a \multimap b}}{\delta[m] : (a \times\, !a) \multimap b}}{\delta[m] : a \multimap (!a \multimap b)}}{\delta[m] : a \multimap (a \longrightarrow b)}$$

There are two options for adding linear resource use.

How to make linear resource use available (on board)

$$\dfrac{\dfrac{\dfrac{\dfrac{m : a \to b}{m :!a \multimap b}}{\delta[m] : (a \times !a) \multimap b}}{\delta[m] : a \multimap (!a \multimap b)}}{\delta[m] : a \multimap (a \to b)}$$

Option 1: Add a linear application

$$\dfrac{m : a \to b \quad v : a \quad p : a}{\mathrm{D}(m, v, p) : b}\ \text{Lin app}$$

How to make linear resource use available (on board)

$$\dfrac{\dfrac{\dfrac{\dfrac{m : a \to b}{m :!a \multimap b}}{\delta[m] : (a \times !a) \multimap b}}{\delta[m] : a \multimap (!a \multimap b)}}{\delta[m] : a \multimap (a \to b)}$$

Option 2: Add a combinator

$$\mathsf{d} : (a \to b) \to (a \multimap (a \to b))$$

How to use linear application (on board)

I will be a bit suggestive with the notation

How to use linear application (on board)

We use a special kind of substitution operation which tries to use v in a linear way and p in a more classical way.

$$\mathsf{D}(\lambda x.m, v, p) \xrightarrow{D[\beta]} \frac{\partial m}{\partial x}(p) \cdot v$$

How to use linear application (on board)

We use a special kind of substitution operation which tries to use v in a linear way and p in a more classical way.

$$\mathsf{D}(\lambda x.m, v, p) \xrightarrow{D[\beta]} \frac{\partial m}{\partial x}(p) \cdot v$$

This notation turns out to be more than suggestive. The categorical models of this system are Blute *et al*'s Differential Categories: examples are traditional systems of derivatives.

The derivative of application (on board)

Recall

$$\frac{m : a \to b \quad n : a}{m\,n : b}$$

The derivative of application (on board)

$$\frac{m : a \to b \quad n : a}{m\,n : b}$$

To be able to support this, application must be linear in the first variable.

The derivative of application (on board)

$$\frac{m : a \to b \quad n : a}{m\,n : b}$$

No assumptions are made about the second argument because we have no useful information about it.

The derivative of application (on board)

$$\frac{m : a \to b \quad n : a}{m\,n : b}$$

If $x \notin n$ then we can simplify what linear in the first argument means:

$$x \notin n \implies \frac{\partial m\,n}{\partial x}(p) \cdot v = \left(\frac{\partial m}{\partial x}(p) \cdot v\right) n$$

Example Reduction (Resource λ versus Differential λ)

$$D(\lambda x.xx, V, P)$$

$$(\lambda x.x[x^{\infty}])[V, P^{\infty}]$$

Example Reduction (Resource λ versus Differential λ)

$$D(\lambda x.xx, V, P)$$

$$(\lambda x.x[x^\infty])[V, P^\infty]$$

In the resource λ-calculus, reduction takes place one argument at a time. The argument of application is really a multiset, and is a choice of substitutions V and P. V is available once, and P is available forever.

Example Reduction (Resource λ versus Differential λ)

$$D(\lambda x.xx, V, P)$$

$$\begin{aligned}
&(\lambda x.x[x^\infty])[V, P^\infty] \\
&\to (\lambda x.V\,[x^\infty])[P^\infty] \\
&\quad + (\lambda x.P\,[x^\infty])[V, P^\infty]
\end{aligned}$$

In one world, V is chosen as the argument, and it is used depleting it from the choice of arguments. In the other world, P is used; however as it is infinite, it remains in the bag.

Example Reduction (Resource λ versus Differential λ)

$D(\lambda x.xx, V, P)$

$$(\lambda x.x[x^\infty])[V, P^\infty]$$
$$\to (\lambda x.V\,[x^\infty])[P^\infty]$$
$$+\,(\lambda x.P\,[x^\infty])[V, P^\infty]$$

Example Reduction (Resource λ versus Differential λ)

$$(\lambda x.x[x^\infty])[V, P^\infty]$$
$$\to (\lambda x.V\,[x^\infty])[P^\infty]$$
$$+\,(\lambda x.P\,[x^\infty])[V, P^\infty]$$

$$D(\lambda x.xx, V, P)$$
$$\to \frac{\partial xx}{\partial x}(P) \cdot V$$

On the differential side, there is just one rule, the reduction rule $D[\beta]$.

Example Reduction (Resource λ versus Differential λ)

$$
\begin{aligned}
&(\lambda x.x[x^\infty])[V, P^\infty] \\
&\quad \to (\lambda x.V\,[x^\infty])[P^\infty] \\
&\qquad + (\lambda x.P\,[x^\infty])[V, P^\infty]
\end{aligned}
$$

$$
\begin{aligned}
&D(\lambda x.xx, V, P) \\
&\quad \to \frac{\partial xx}{\partial x}(P) \cdot V \\
&\quad = \left(\frac{\partial x}{\partial x}(P) \cdot V\right) x\,[P/x] \\
&\qquad + D\left(P, V, \frac{\partial x}{\partial x}(P) \cdot P\right)
\end{aligned}
$$

Here we can see the derivative break up into two pieces. This is an application of the higher order chain rule at work. Note the component of the sum looks different than the second – this is because the derivative is linear in the first argument.

Example Reduction (Resource λ versus Differential λ)

$$
\begin{aligned}
&(\lambda x.x[x^\infty])[V, P^\infty] \\
&\quad \to (\lambda x.V\,[x^\infty])[P^\infty] \\
&\qquad + (\lambda x.P\,[x^\infty])[V, P^\infty]
\end{aligned}
$$

$$
\begin{aligned}
&D(\lambda x.xx, V, P) \\
&\quad \to \frac{\partial xx}{\partial x}(P) \cdot V \\
&\quad = \left(\frac{\partial x}{\partial x}(P) \cdot V\right) x\,[P/x] \\
&\qquad + D\left(P, V, \frac{\partial x}{\partial x}(P) \cdot P\right)
\end{aligned}
$$

This time we are substituting into an infinite variable.

Example Reduction (Resource λ versus Differential λ)

$$
\begin{aligned}
&(\lambda x.x[x^\infty])[V, P^\infty] \\
&\to (\lambda x.V\,[x^\infty])[P^\infty] \\
&\quad + (\lambda x.P\,[x^\infty])[V, P^\infty] \\
&\to V\,[P^\infty] + P\,[V, P^\infty]
\end{aligned}
$$

$$
\begin{aligned}
&D(\lambda x.xx, V, P) \\
&\to \frac{\partial xx}{\partial x}(P) \cdot V \\
&= \left(\frac{\partial x}{\partial x}(P) \cdot V\right) x\,[P/x] \\
&\quad + D\left(P, V, \frac{\partial x}{\partial x}(P) \cdot P\right)
\end{aligned}
$$

When substituting into an infinite variable, we get just a normal substitution.

Example Reduction (Resource λ versus Differential λ)

$$
\begin{aligned}
&(\lambda x.x[x^\infty])[V, P^\infty] \\
&\to (\lambda x.V\,[x^\infty])[P^\infty] \\
&\quad + (\lambda x.P\,[x^\infty])[V, P^\infty] \\
&\to V\,[P^\infty] + P\,[V, P^\infty]
\end{aligned}
$$

$$
\begin{aligned}
&D(\lambda x.xx, V, P) \\
&\to \frac{\partial xx}{\partial x}(P) \cdot V \\
&= \left(\frac{\partial x}{\partial x}(P) \cdot V\right) x\,[P/x] \\
&\quad + D\left(P, V, \frac{\partial x}{\partial x}(P) \cdot P\right)
\end{aligned}
$$

Example Reduction (Resource λ versus Differential λ)

$$
\begin{aligned}
&(\lambda x.x[x^\infty])[V, P^\infty] \\
&\to (\lambda x.V\,[x^\infty])[P^\infty] \\
&\quad + (\lambda x.P\,[x^\infty])[V, P^\infty] \\
&\to V\,[P^\infty] + P\,[V, P^\infty]
\end{aligned}
$$

$$
\begin{aligned}
&D(\lambda x.xx, V, P) \\
&\to \frac{\partial xx}{\partial x}(P) \cdot V \\
&= \left(\frac{\partial x}{\partial x}(P) \cdot V\right) x\,[P/x] \\
&\quad + D\left(P, V, \frac{\partial x}{\partial x}(P) \cdot P\right) \\
&= V\,P + D(P, V, P)
\end{aligned}
$$

The derivative of a variable with respect to itself is just the "direction vector."

Example Reduction (Resource λ versus Differential λ)

$$\rightarrow V\,[P^{\infty}] + P\,[V, P^{\infty}]$$

$$= V\,P + D(P, V, P)$$

Differential applicative systems (on board)

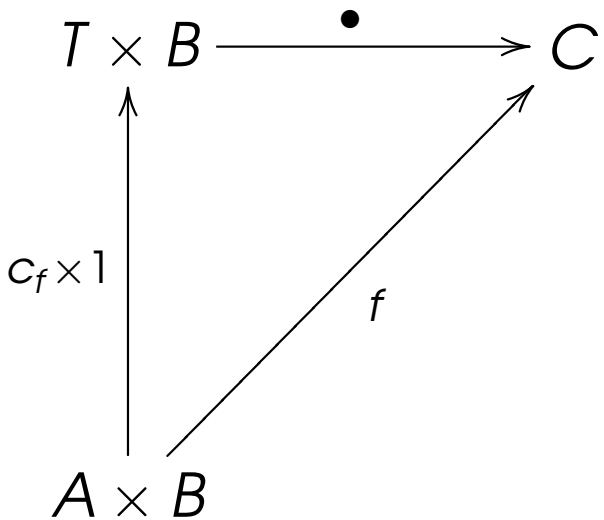
- The differential λ-calculus of Ehrhard and Regnier is the embodiment of the above analysis into a rewriting system, and is the first known example of a differential applicative system.
- The differential λ-calculus is confluent, and is a conservative extension of the λ-calculus.
  Hence, it contains a full model of computability.
- The structural story of differential applicative systems was worked out using the tools from the study of Turing categories.
- The only known models are total and have an idempotent sum (it even seemed this might be necessary).

Today's talk (on board)

It was unclear whether nice models could be found. Today we will exhibit a nice total model, and point to two ways to produce a partial model.

Turing Categories (Cockett and Hofstra)

$\underline{\mathbf{X}}$ is a Turing category when there is an object T such that:
There is a universal application $\bullet$, and for every f there is a c_f:



$$f(a,b) = c_f(a) \bullet b$$

Restriction categories

Definition

A **restriction category** *is a category in which, for each map* $f : A \to B$*, there is a map* $\overline{f} : A \to A$ *such that*

(R.1) $\overline{f}\, f = f$ **(R.3)** $\overline{f}\,\overline{g} = \overline{\overline{f}\, g}$

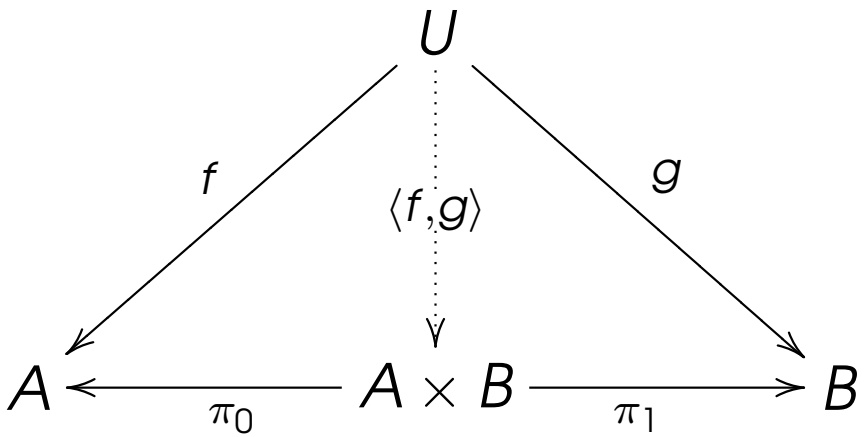
(R.2) $\overline{f}\,\overline{g} = \overline{g}\,\overline{f}$ **(R.4)** $f\overline{h} = \overline{fh}\, f$

The restriction partial order is defined as

$$f \leq g := \overline{f}\, g = f$$

Restriction products

The **binary restriction product** of A, B is $A \times B$ with total projections π_0, π_1 and a unique pairing such that in



$\langle f, g\rangle \, \pi_0 = \overline{g} \, f$ and $\langle f, g\rangle \, \pi_1 = \overline{f} \, g$.

Characterizing Turing Categories

Theorem

- $\underline{\mathbf{X}}$ *is a Turing category iff there is a universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a retract of* T*.*
- *Every Turing category contains a partial combinatory algebra (PCA) and every PCA generates a Turing category by taking the category of computable maps.*

Cartesian left additive restriction categories

Definition

- *A* **left additive restriction category** *is a restriction category in which each* $\underline{\mathbf{X}}(A, B)$ *is a commutative monoid such that* $\overline{f+g} = \bar{f}\,\bar{g}$ *and* $\bar{0} = 1$*; the sum is left additive:* $f(g+h) = fg + fh$ *and* $f0 = \bar{f}\,0$
- *An* **additive** *map is* h *such that* $(f+g)h \smile fh + gh$*. A* **strongly additive** *map is* h *such that* $(f+g)h \geq fh + gh$*.*
- *A* **Cartesian left additive restriction category** *has Cartesian and left additive restriction structure, and further,* $(f+g) \times (h+k) = (f \times h) + (g \times k)$ *and* $0 \times 0 = 0$*.*

Differential restriction categories

Definition

A **differential restriction category** *is a Cartesian left additive restriction category with a differential combinator*

$$\frac{f : X \to Y}{D[f] : X \times X \to Y}$$

Differential restriction categories

DR.1 $D[0] = 0$ and $D[f + g] = D[f] + D[g]$;

DR.2 $\langle 0, g\rangle\, D[f] = \overline{gf}\, 0$ and $\langle g + h, k\rangle\, D[f] = \langle g, k\rangle\, D[f] + \langle h, k\rangle\, D[f]$;

DR.3 $D[\pi_0] = \pi_0\pi_0$ and $D[\pi_1] = \pi_0\pi_1$;

DR.4 $D[\langle f, g\rangle] = \langle D[f], D[g]\rangle$;

DR.5 $D[fg] = \langle D[f], \pi_1 f\rangle\, D[g]$;

DR.6 $\langle\langle g, 0\rangle, \langle h, k\rangle\rangle\, D[D[f]] = \overline{h}\, \langle g, k\rangle\, D[f]$;

DR.7 $\langle\langle 0, h\rangle, \langle g, k\rangle\rangle\, D[D[f]] = \langle\langle 0, g\rangle, \langle h, k\rangle\rangle\, D[D[f]]$;

DR.8 $D[\bar{f}\,] = (1 \times \bar{f}\,)\pi_0$;

DR.9 $\overline{D[f]} = 1 \times \bar{f}$.

Differential Turing categories

- A **differential index** for $f : A \times B \longrightarrow C$ in $\bullet_{BC} : T \times B \longrightarrow C$ is a map $c_f : A \longrightarrow T$ such that (using the term logic):
  - $c_f(x) \bullet_{BC} y = f(x, y)$ (so it is a code);
  - $(v, a) \mapsto \frac{\partial c_f(x)}{\partial x}(a) \cdot v$ is a differential index for $((v, a), y) \mapsto \frac{\partial f(x,y)}{\partial x}(a) \cdot v$.
- A map $\bullet_{BC} : T \times B \longrightarrow C$ is said to be **differentially universal** in case it is both linear and strongly additive in its first argument and each map $f : A \times B \longrightarrow C$ has a differential index in $\bullet_{BC}$.
- $\underline{\mathbf{X}}$ is a **differential Turing category** if it has **differential Turing structure**, that is an object T with for each pair of objects B and C a differentially universal map $\bullet_{BC} : T \times B \longrightarrow C$.

Differential Turing categories

- A **differential index** for $f : A \times B \longrightarrow C$ in $\bullet_{BC} : T \times B \longrightarrow C$ is a map $c_f : A \longrightarrow T$ such that (using the term logic):
  - $c_f(x) \bullet_{BC} y = f(x, y)$ (so it is a code);
  - $(v, a) \mapsto \frac{\partial c_f(x)}{\partial x}(a) \cdot v$ is a differential index for $((v, a), y) \mapsto \frac{\partial f(x,y)}{\partial x}(a) \cdot v$.
- A map $\bullet_{BC} : T \times B \longrightarrow C$ is said to be **differentially universal** in case it is both linear and strongly additive in its first argument and each map $f : A \times B \longrightarrow C$ has a differential index in $\bullet_{BC}$.
- **X** is a **differential Turing category** if it has **differential Turing structure**, that is an object T with for each pair of objects B and C a differentially universal map $\bullet_{BC} : T \times B \longrightarrow C$.

$$x \notin m \Rightarrow \frac{\partial n \bullet m}{\partial x}(a) \cdot v \smile \left(\frac{\partial n}{\partial x}(a) \cdot v\right) \bullet m$$

Differential Turing categories

- A **differential index** for $f : A \times B \longrightarrow C$ in $\bullet_{BC} : T \times B \longrightarrow C$ is a map $c_f : A \longrightarrow T$ such that (using the term logic):
  - $c_f(x) \bullet_{BC} y = f(x, y)$ (so it is a code);
  - $(v, a) \mapsto \frac{\partial c_f(x)}{\partial x}(a) \cdot v$ is a differential index for $((v, a), y) \mapsto \frac{\partial f(x,y)}{\partial x}(a) \cdot v$.
- A map $\bullet_{BC} : T \times B \longrightarrow C$ is said to be **differentially universal** in case it is both linear and strongly additive in its first argument and each map $f : A \times B \longrightarrow C$ has a differential index in $\bullet_{BC}$.
- **X** is a **differential Turing category** if it has **differential Turing structure**, that is an object T with for each pair of objects B and C a differentially universal map $\bullet_{BC} : T \times B \longrightarrow C$.

$$(m + n) \bullet t \geq m \bullet t + n \bullet t$$

Characterizing Differential Turing Categories

Theorem

X *is a differential Turing category iff there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

Characterizing Differential Turing Categories

Theorem

X *is a differential Turing category iff there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

- r is linear:

$$\frac{\partial r(x)}{\partial x}(a) \cdot v \smile r(v)$$

- $D^n[s]r = \underbrace{\pi_0 \cdots \pi_0}_{\text{n-times}}$

Characterizing Differential Turing Categories

Theorem

$\underline{\mathbf{X}}$ *is a differential Turing category only if there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

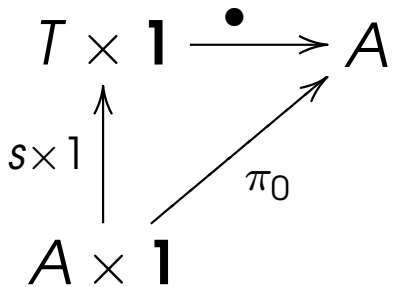
Define s to be the code for $\pi_0 : A \times \mathbf{1} \to A$, $r := \langle 1, ! \rangle \bullet$.

Characterizing Differential Turing Categories

Theorem

X *is a differential Turing category only if there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

Define s to be the code for $\pi_0 : A \times \mathbf{1} \to A$, $r := \langle 1, ! \rangle \bullet$. First, $sr = 1$.



$$s \langle 1, ! \rangle \bullet = \langle 1, ! \rangle (s \times 1) \bullet = \langle 1, ! \rangle \pi_0 = 1$$

Also, $r = 1 \bullet ()$ is linear as $\bullet$ is linear in its first argument.

Characterizing Differential Turing Categories

Theorem

$\underline{\mathbf{X}}$ *is a differential Turing category only if there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

Define s to be the code for $\pi_0 : A \times \mathbf{1} \to A$, $r := \langle 1, ! \rangle \bullet$. Second

$$(v, a) \mapsto \frac{\partial s(x)}{\partial x}(a) \cdot v \bullet () = \frac{\partial s(x) \bullet ()}{\partial x}(a) \cdot v = \frac{\partial x}{\partial x}(a) \cdot v = v$$

so that $D[s]r = \pi_0$ as required.

Characterizing Differential Turing Categories

Theorem

X *is a differential Turing category if there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

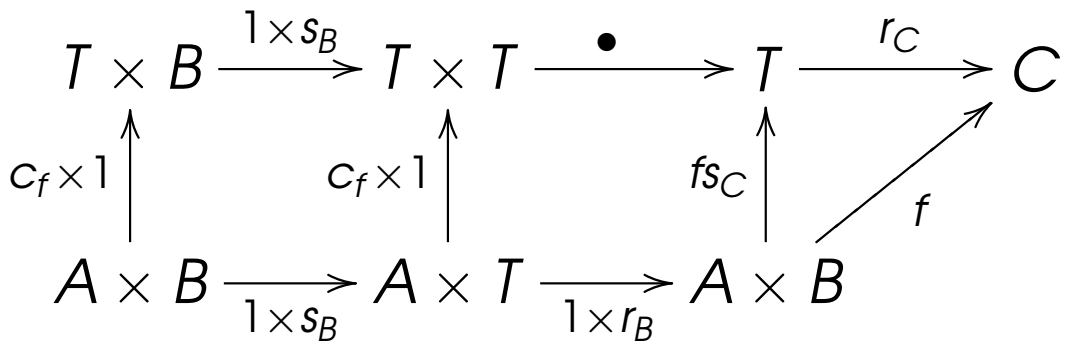
Define the derived application $\bullet_{BC} : T \times B \to C$ as $(1 \times s_B) \bullet r_C$. Given f define c_f to be the code for $(1 \times r_B) f s_C$.

Characterizing Differential Turing Categories

Theorem

X *is a differential Turing category if there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

Define the derived application $\bullet_{BC} : T \times B \to C$ as $(1 \times s_B) \bullet r_C$. Given f define c_f to be the code for $(1 \times r_B) f s_C$. To see that this gives a universal application:



Characterizing Differential Turing Categories

Theorem

X *is a differential Turing category if there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

Define the derived application $\bullet_{BC} : T \times B \to C$ as $(1 \times s_B) \bullet r_C$. Given f define c_f to be the code for $(1 \times r_B) f s_C$. That $\bullet_{BC}$ is also linear in its first variable follows mostly from Cartesian coherences and also that $\bullet r$ is linear in its first variable.

Characterizing Differential Turing Categories

Theorem

X *is a differential Turing category if there is a differentially universal application* $T \times T \xrightarrow{\bullet} T$ *and every object is a differential retract of* T.

Define the derived application $\bullet_{BC} : T \times B \to C$ as $(1 \times s_B) \bullet r_C$. Given f define c_f to be the code for $(1 \times r_B)fs_C$.

$$(1 \times s_B)\left(\frac{\partial c_f(x)}{\partial x}(a) \cdot v \bullet y\right) r_C = (1 \times s_B)\left(\frac{\partial s_C(f(x, r_B(y)))}{\partial x}(a) \cdot v\right) r_C$$
$$= \left(\frac{\partial s_C(f(x, r_B(s_B(y))))}{\partial x}(a) \cdot v\right) r_C$$
$$= \frac{\partial f(x, y)}{\partial x}(a) \cdot v$$

As generally $D[fs]r = D[f]$.

Differential PCAs

The above allows us to define *abstractly* what a differential PCA is *in a differential restriction category*.

Differential PCAs

The above allows us to define *abstractly* what a differential PCA is *in a differential restriction category*.

However: Do differential PCAs exist?

Differential PCAs

The above allows us to define *abstractly* what a differential PCA is *in a differential restriction category*.

However: Do differential PCAs exist?

- Last FMCS, a total model was presented with an idempotent sum (Manzonetto, G.).

Differential PCAs

The above allows us to define *abstractly* what a differential PCA is *in a differential restriction category*.

However: Do differential PCAs exist?

- Last FMCS, a total model was presented with an idempotent sum (Manzonetto, G.).
- This FMCS, idempotence will be removed, and we will point the way to partiality.

Quick Note on Total Differential Turing Categories

The differential indexing always holds for total differential Turing categories:

$$\begin{aligned}
&\frac{\partial f(x,y)}{\partial x}(a)\cdot v \\
&= \frac{\partial c_f(x)\bullet y}{\partial x}(a)\cdot v \\
&= \left(\frac{\partial c_f(x)}{\partial x}(a)\cdot v\right)\bullet y
\end{aligned}$$

Outline

- Differential Applicative Systems in **Sets**
- An Equational Completion
- Rewriting and Confluence Modulo

Differential combinatory algebra

Definition

A **mal-differential applicative system in Sets** is $\mathcal{A} = (A, \bullet, +, 0, \underline{\mathrm{d}})$ where $(A, +, 0)$ is a commutative monoid and

1. $(t_1 + t_2)\, m = t_1\, m + t_2\, m; \quad 0\, m = 0;$
2. $\underline{\mathrm{d}}\,(t_1 + t_2)\, v\, a = \underline{\mathrm{d}}\, t_1\, v\, a + \underline{\mathrm{d}}\, t_2\, v\, a; \quad \underline{\mathrm{d}}\, 0\, v\, a = 0;$
3. $\underline{\mathrm{d}}\, t\, (v_1 + v_2)\, a = \underline{\mathrm{d}}\, t\, v_1\, a + \underline{\mathrm{d}}\, t\, v_2\, a; \quad \underline{\mathrm{d}}\, t\, 0\, a = 0;$
4. $\underline{\mathrm{d}}\,(\underline{\mathrm{d}}\, x)\, y\, z = \underline{\mathrm{d}}\, x\, y;$
5. $\underline{\mathrm{d}}\,\underline{\mathrm{d}}\, x\, y = \underline{\mathrm{d}}\, x;$
6. $\underline{\mathrm{d}}\,(\underline{\mathrm{d}}\, x\, y)\, z = \underline{\mathrm{d}}\,(\underline{\mathrm{d}}\, x\, z)\, y.$ [a]

[a] The last rule is a permutation.

This is enough to define a derivative

The partial derivative in a DCA

1. $\frac{\partial t}{\partial y}(a) \cdot v := 0$ when $y \notin t$
2. $\frac{\partial x}{\partial x}(a) \cdot v := v$
3. $\frac{\partial t}{\partial x}(a) \cdot 0 := 0$
4. $\frac{\partial t}{\partial x}(a) \cdot (v_1 + v_2) := \frac{\partial t}{\partial x}(a) \cdot v_1 + \frac{\partial t}{\partial x}(a) \cdot v_2$
5. $\frac{\partial 0}{\partial x}(a) \cdot v := 0$
6. $\frac{\partial t_1 + t_2}{\partial x}(a) \cdot v := \frac{\partial t_1}{\partial x}(a) \cdot v + \frac{\partial t_2}{\partial x}(a) \cdot v$
7. $$\frac{\partial t_1\, t_2}{\partial x}(a) \cdot v := \left(\frac{\partial t_1}{\partial x}(a) \cdot v\right)(t_2\,[a/x]) + \underline{\mathrm{d}}\,(t_1\,[a/x])\left(\frac{\partial t_2}{\partial x}(a) \cdot v\right)(t_2\,[a/x])$$

The rules of differentiation (Blute *et al*)

DT.1 $\frac{\partial t_1+t_2}{\partial p}(s)\cdot u = \frac{\partial t_1}{\partial p}(s)\cdot u + \frac{\partial t_2}{\partial p}(s)\cdot u$ and $\frac{\partial 0}{\partial p}(s)\cdot u = 0$;

DT.2 $\frac{\partial t}{\partial p}(s)\cdot(u_1+u_2) = \frac{\partial t}{\partial p}(s)\cdot u_1 + \frac{\partial t}{\partial p}(s)\cdot u_2$ and $\frac{\partial t}{\partial p}(s)\cdot 0 = 0$;

DT.3

- $\frac{\partial x}{\partial x}(s)\cdot u = u$;
- $\frac{\partial t}{\partial (p,p')}((s,s'))\cdot(u,0) = \frac{\partial t[s'/p']}{\partial p}(s)\cdot u$, and
- $\frac{\partial t}{\partial (p,p')}((s,s'))\cdot(0,u') = \frac{\partial t[s/p]}{\partial p'}(s')\cdot u'$;

DT.4 $\frac{\partial (t_1,t_2)}{\partial p}(s)\cdot u = \left(\frac{\partial t_1}{\partial p}(s)\cdot u, \frac{\partial t_2}{\partial p}(s)\cdot u\right)$;

DT.5 $\frac{\partial t[t'/p']}{\partial p}(s)\cdot u = \frac{\partial t}{\partial p'}(t'\,[s/p])\cdot \frac{\partial t'}{\partial p}(s)\cdot u$;

DT.6 $\frac{\partial \frac{\partial t}{\partial p}(s)\cdot p'}{\partial p'}(r)\cdot u = \frac{\partial t}{\partial p}(s)\cdot u$;

DT.7 $\frac{\partial \frac{\partial t}{\partial p_1}(s_1)\cdot u_1}{\partial p_2}(s_2)\cdot u_2 = \frac{\partial \frac{\partial t}{\partial p_2}(s_2)\cdot u_2}{\partial p_1}(s_1)\cdot u_1$;

On the Form of the Equations

Given that one exactly needs an equation of the form

$$m\,a\,b = n\,a\,b$$

Do you take the equation in that form or

$$m\,a = n\,a$$

or even

$$m = n?$$

A second look at the equations

Recall,

$$\frac{\partial t\, x}{\partial x}(a) \cdot v = \underline{\mathrm{d}}\, t\, v\, a$$

$\underline{\mathrm{d}}$ is linear in its first two arguments. The equation

$$\underline{\mathrm{d}}\, \underline{\mathrm{d}}\, x\, y = \underline{\mathrm{d}}\, x$$

expresses exactly that

$$\partial[\underline{\mathrm{d}}\, t] = \underline{\mathrm{d}}\, \partial[t]$$

i.e. that $\underline{\mathrm{d}}$ is linear in its first argument. Similarly, the equation

$$\underline{\mathrm{d}}\, (\underline{\mathrm{d}}\, x)\, y\, z = \underline{\mathrm{d}}\, x\, y$$

expresses exactly that $\underline{\mathrm{d}}$ is linear in its second argument.

Well Definedness

The operation of differentiation is not well defined

$$\underline{\mathrm{d}}\,\underline{\mathrm{d}}\,x\,y = \underline{\mathrm{d}}\,x$$

$$\begin{aligned}
&\frac{\partial \underline{\mathrm{d}}\,\underline{\mathrm{d}}\,t_1\,t_2}{\partial x}(a)\cdot v \\
&= \text{Large complicated term} \\
&\neq \underline{\mathrm{d}}\,\underline{\mathrm{d}}\,\frac{\partial t_1}{\partial x}(a)\cdot v\,t_1\,[a/x] \\
&= \frac{\partial \underline{\mathrm{d}}\,t_1}{\partial x}(a)\cdot v
\end{aligned}$$

Also, the equation

$$\underline{\mathrm{d}}\,(\underline{\mathrm{d}}\,x)\,y\,z = \underline{\mathrm{d}}\,x\,y$$

is problematic.

Differential Applicative Systems

Definition

A **differential applicative system in Sets** is a mal-differential applicative system, $\mathcal{A}$, with two extra equations:

7. $\underline{\mathsf{d}}\,(\underline{\mathsf{d}}\,\underline{\mathsf{d}}\,x) = 0$;
8. $\underline{\mathsf{d}}\,(\underline{\mathsf{d}}\,(\underline{\mathsf{d}}\,x)\,y) = 0$.

Theorem

The generic category of a differential applicative system is a Cartesian differential category.

Differential Applicative Systems with $\underline{k}$

Definition

A **differential $\underline{k}$-applicative system in Sets** is a differential applicative system $\mathcal{A}$ with a distinguished point $\underline{k}$ that is additive in the first argument and such that

9. $\underline{k}\,x\,y = x$ (the $\underline{k}$ law);
10. $\underline{d}\,(\underline{k}\,x) = 0$ (well definedness for disappearing y);
11. $\underline{d}\,\underline{k}\,x\,y = \underline{k}\,x$ ($\underline{k}$ is linear in first argument);
12. $\underline{d}\,(\underline{d}\,\underline{k}\,x) = 0$ (well definedness for linear in first argument).

Theorem

The generic category of a $\underline{k}$-differential applicative system is a Cartesian differential category.

The equation for S is

$$\mathsf{S}\, x\, y\, z = x\, z\, (y\, z)$$

z is duplicated, and put into non-linear positions.

The immediately needed equations are

$$\underline{\mathsf{d}}\,\underline{\mathsf{s}}\,x\,y = \underline{\mathsf{s}}\,x$$

$$\underline{\mathsf{d}}\,(\underline{\mathsf{s}}\,x)\,y = \underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,x)\,y)$$

$$\underline{\mathsf{d}}\,(\underline{\mathsf{s}}\,x\,y)\,z = \underline{\mathsf{s}}\,(\underline{\mathsf{d}}\,x\,z)\,y + \underline{\mathsf{d}}\,(\underline{\mathsf{s}}\,x)\,(\underline{\mathsf{d}}\,y\,z)\,y$$

The first says $\underline{\mathsf{s}}$ is linear in its first argument; hence $\underline{\mathsf{d}}\,(\underline{\mathsf{d}}\,\underline{\mathsf{s}}\,x) = 0$ will be needed.

The others state what the derivative must do with each argument of $\underline{\mathsf{s}}$.

To get

$$\underline{\mathsf{d}}\,(\underline{\mathsf{s}}\,x)\,y = \underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,x)\,y)$$

to be well defined, we need three additional equations:

$$\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}}))x) = \underline{\mathsf{k}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})x)$$

$$\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,x))y) = \underline{\mathsf{k}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,x)y)$$

$$\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,0)\,x) = 0$$

Do not generalize these equations.

Victory

To get

$$\underline{\mathsf{d}}\,(\underline{\mathsf{s}}\,x\,y)\,z = \underline{\mathsf{s}}\,(\underline{\mathsf{d}}\,x\,z)\,y + \underline{\mathsf{d}}\,(\underline{\mathsf{s}}\,x)\,(\underline{\mathsf{d}}\,y\,z)\,y$$

to be well defined, all we need is:

$$\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,x)\,y))\,z) = \underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,(\underline{\mathsf{s}}\,(\underline{\mathsf{s}}\,(\underline{\mathsf{k}}\,\underline{\mathsf{d}})\,x)\,z))\,y)$$

The well definedness of differentiation has been machine checked.

Differential Combinatory Algebras

Definition

A **differential combinatory algebra in Sets** is a differential $\underline{k}$-applicative system with a distinguished point $\underline{s}$ that is additive in its first argument and where

13. $\underline{s}\,x\,y\,z = x\,z\,(y\,z)$;
14. $\underline{d}\,\underline{s}\,x\,y = \underline{s}\,x$;
15. $\underline{d}\,(\underline{d}\,\underline{s}\,x) = 0$;
16. $\underline{d}\,(\underline{s}\,x)\,y = \underline{s}\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,x)\,y)$;
17. $\underline{s}\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,(\underline{k}\,\underline{d}))x) = \underline{k}\,(\underline{s}\,(\underline{k}\,\underline{d})x)$;
18. $\underline{s}\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,(\underline{s}\,(\underline{k}\,\underline{d})\,x))y) = \underline{k}\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,x)y)$;
19. $\underline{s}\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,0)\,x) = 0$;
20. $\underline{d}\,(\underline{s}\,x\,y)\,z = \underline{s}\,(\underline{d}\,x\,z)\,y + \underline{d}\,(\underline{s}\,x)\,(\underline{d}\,y\,z)\,y$;
21. $\underline{s}\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,x)\,y))\,z) = \underline{s}\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,(\underline{s}\,(\underline{s}\,(\underline{k}\,\underline{d})\,x)\,z))\,y)$. [a]

[a] The last rule is a permutation

Differential Combinatory Algebras

Lemma

In a differential combinatory algebra, the simulation of abstraction satisfies:

$$(\lambda^* x.m)n = m[n/x] \qquad \text{and} \qquad \underline{\mathsf{d}}\,(\lambda^* x.m)\,v\,a = \frac{\partial m}{\partial x}(a) \cdot v$$

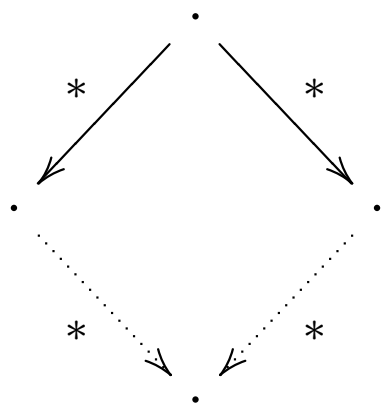
This lemma says that a differential combinatory can simulate the differential λ-calculus at a top level. Hence, we also get combinatory completeness for all polynomials, and their derivatives. Also, application can be shown to be linear in the first component; thus we have:

Theorem

The computable map category of a differential combinatory algebra in ***Sets*** *is a total differential Turing category.*

Confluence and Rewriting

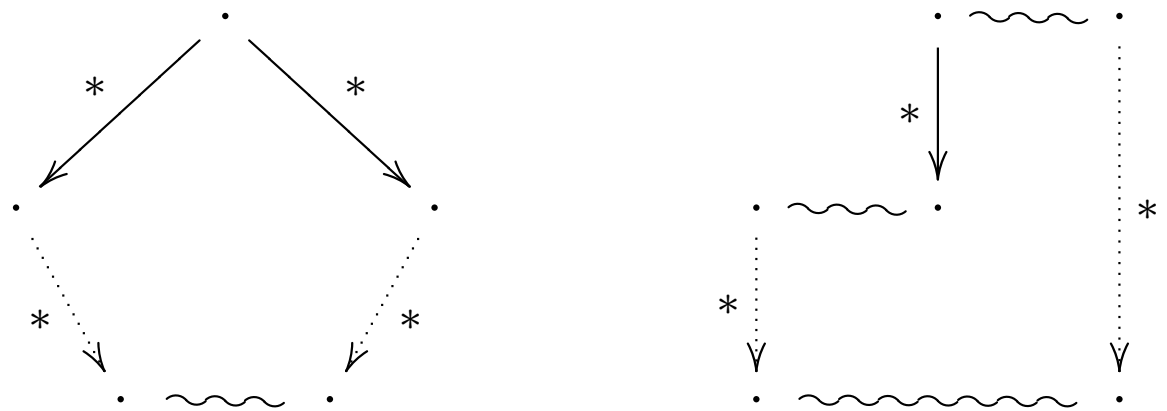
A term rewriting system is Confluent (Church-Rosser) if



Two terms are equal if and only if they have the same normal form.

Confluence Modulo Equivalence

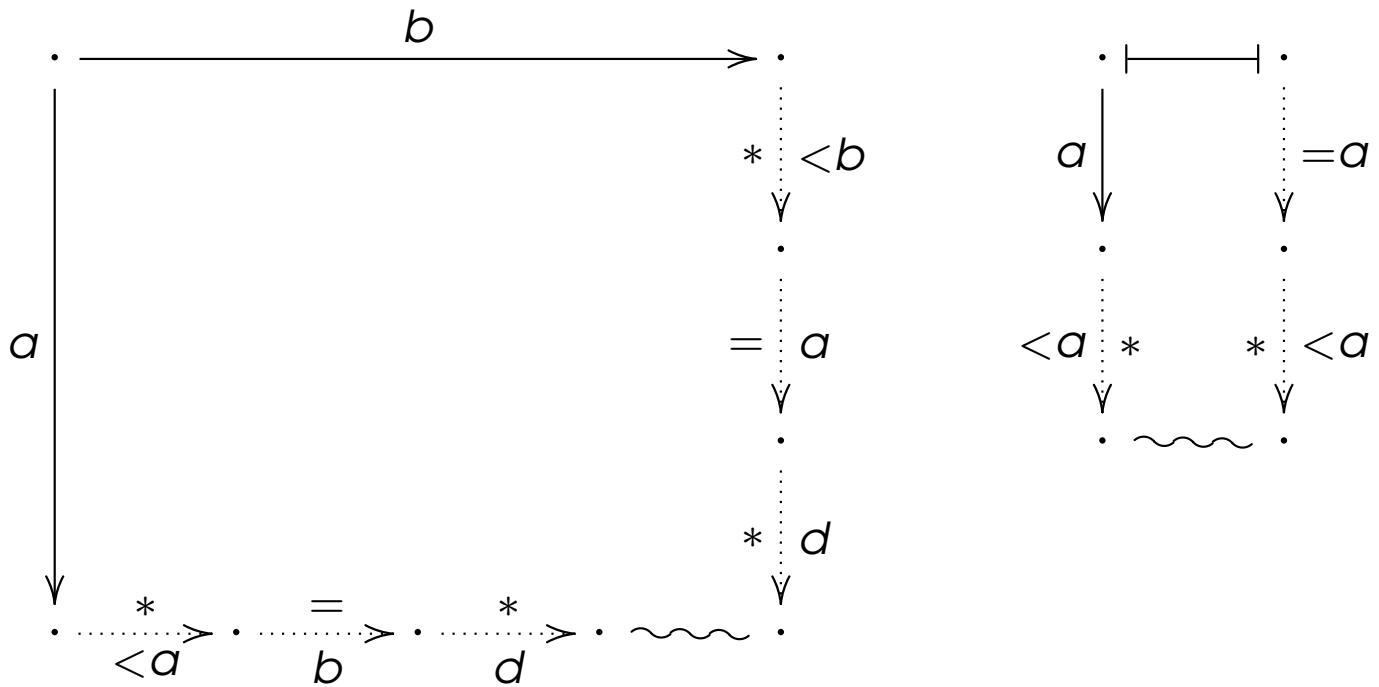
A term rewriting system is Church-Rosser modulo if



Two terms are equal if and only if they have the same normal form up to some simpler equivalence.

Confluence Modulo via Decreasing Diagrams (Ohlebusch)

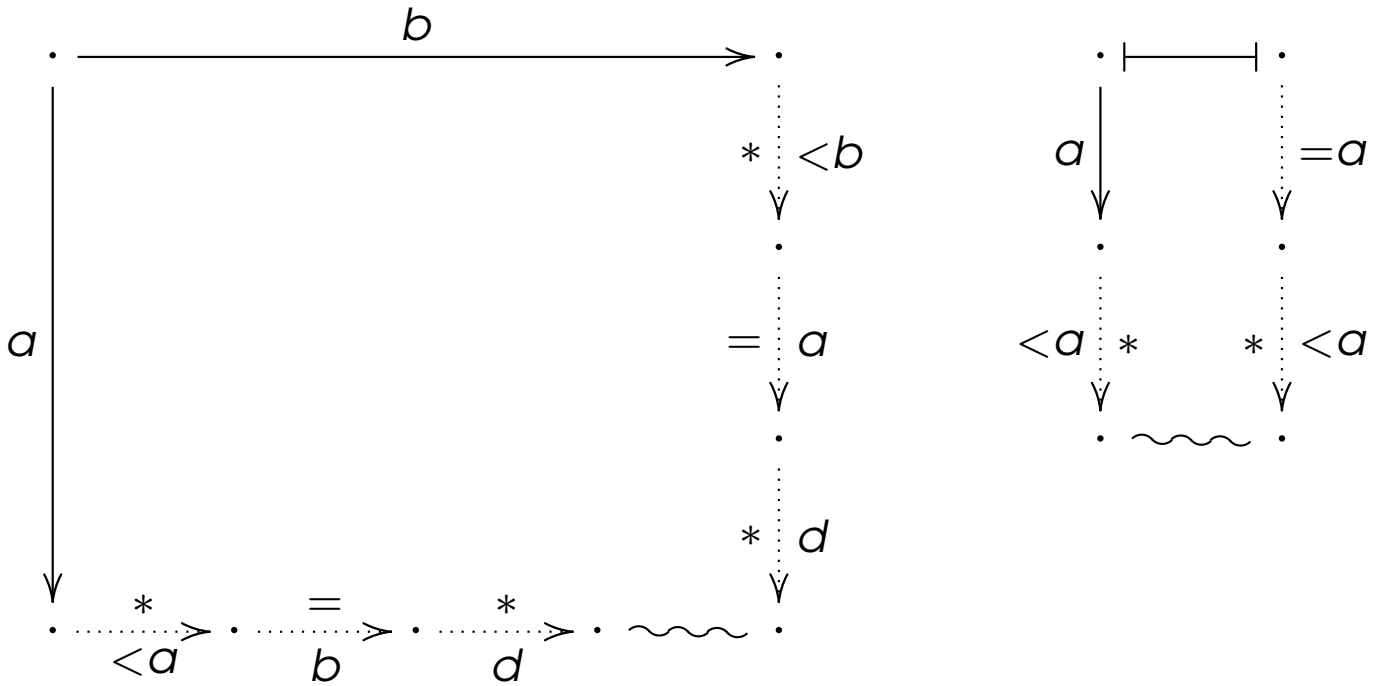
If $\rightarrow_\alpha$ is a labelled rewrite system where the labels are in a well ordered set W, and $\vdash$ is a symmetric relation where for every $a, b \in W$ we have



where $d < a$ or $d < b$.

Confluence Modulo via Decreasing Diagrams (Ohlebusch)

If $\rightarrow_\alpha$ is a labelled rewrite system where the labels are in a well ordered set W, and $\vdash$ is a symmetric relation where for every $a, b \in W$ we have



where $d < a$ or $d < b$. Locally decreasing modulo implies Church-Rosser modulo

Proving Confluence

To use this technique:

- A left linear labelling is needed [1]
- For all non-$\underline{s}$ steps label the rule by source. Termination gives well ordering.
- Treat $\underline{s}$ as a parallel move (ortogonality gives strict diamond property, so that decreasingness holds)

Then it just remains to show that the critical pairs are locally decreasing.

[1] See Felgenhauer, Middledorp, and Zankl "Labellings for Decreasing Diagrams." It turns out, that for left linear term rewriting systems, local decreasingness is not equivalent to local decreasingness of critical pairs; however, the latter with a left linear labelling does give a partial converse.

Confluence

Theorem

The rewriting system modulo associativity, commutativity, and permutation for differential combinatory algebras is Church-Rosser modulo.

Corollary

Differential combinatory algebras are conservative extensions of ordinary combinatory algebras.

Conclusion

- Differential combinatory algebras give a model of a total differential Turing category.
- The rewriting system for differential combinatory algebras is confluent.

Future

- Obtaining Partial Differential Combinatory Algebras
  - Use an extension to a partial term logic
  - Use non-termination of the rewriting system as the source of partiality
- Adding a coderliction to Abramsky *et al*'s linear combinatory algebras