Taylor Approximation and Infinitary λ-Calculi

Rémy Cerda, Aix-Marseille Université, I2M

PhD defence

Marseille, June 10, 2024

Contents

Introduction

What is it to deduce?

Why the λ-calculus?

What is it to deduce?

Given a sentence P, can we compute a (dis)proof of P?

Why the λ-calculus?

What is it to deduce?

What is it to compute?

Given a sentence P, can we compute a (dis)proof of P?

Why the λ-calculus?

What is it to deduce?

What is it to compute?

Given a sentence P, can we compute a (dis)proof of P?

Not always.

Church,Turing: there is a problem that no program can solve.

But also **Gödel:** there is a formula that no proof can (dis)prove.

Why the λ-calculus?

What is it to deduce? What is it to compute?

The same thing.

Church,Turing: there is a problem that no program can solve.

But also **Gödel:** there is a formula that no proof can (dis)prove.

What is it to deduce?

What is it to compute?

The same thing.

formulæ	$\leftrightarrow$	problems
proofs	$\leftrightarrow$	programs

This is the **Curry-Howard correspondence**.

Why the λ-calculus?

What is it to deduce? What is it to compute?

The same thing.

formulæ	$\leftrightarrow$	program specifications
proofs	$\leftrightarrow$	programs

This is the **Curry-Howard correspondence**.

Why the λ-calculus?

What is it to deduce? What is it to compute?

The same thing.

formulæ	$\leftrightarrow$	types
proofs	$\leftrightarrow$	programs

This is the **Curry-Howard correspondence**.

Why the λ-calculus?

What is it to deduce? What is it to compute?

The same thing.

formulæ	$\leftrightarrow$	types
proofs	$\leftrightarrow$	programs
cut-elimination	$\leftrightarrow$	execution

This is the **Curry-Howard correspondence**.

What is it to deduce? What is it to compute?

The same thing.

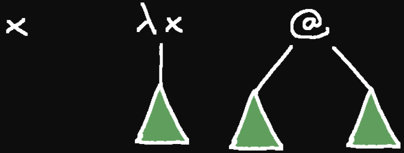
$$\begin{array}{rcl} \text{formulæ} & \leftrightarrow & \text{types} \\ \text{proofs} & \leftrightarrow & \text{programs} = \boldsymbol{\lambda}\textbf{-terms} \\ \text{cut-elimination} & \leftrightarrow & \text{execution} = \boldsymbol{\beta}\textbf{-reduction} \end{array}$$

This is the **Curry-Howard correspondence**.

The λ-calculus

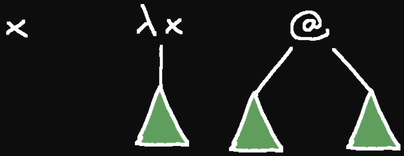
$$M, N, \ldots \quad := \quad x \quad | \quad \lambda x.M \quad | \quad MN$$

$$x \mapsto M \qquad M(N)$$

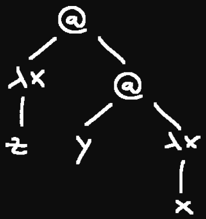


The λ-calculus

$$\begin{array}{ccccccc} M, N, \ldots & := & x & \mid & \lambda x.M & \mid & MN \\ & & & & x \mapsto M & & M(N) \end{array}$$



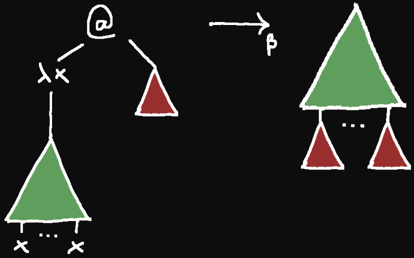
Example:
$(\lambda x.z)((y)\lambda x.x)$



The λ-calculus

$$(x \mapsto M)(N) \longrightarrow M \text{ where } x \text{ is replaced with } N$$

$$(\lambda x.M)N \longrightarrow_\beta M[N/x]$$



Example: $(\lambda x.\mathsf{add}\ (\mathsf{pow}\ 2\ x)\ (\mathsf{add}\ x\ x))\ 5$

$\longrightarrow_\beta \mathsf{add}\ (\mathsf{pow}\ 2\ 5)\ (\mathsf{add}\ 5\ 5)$

$\longrightarrow_\beta^* \mathsf{add}\ 32\ 10$

$\longrightarrow_\beta^* 42$

Böhm trees

A program may run forever!

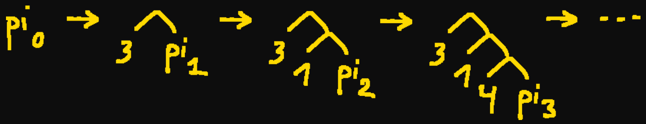
... and compute no meaningful result:

$$\Omega := (\lambda x.xx)(\lambda x.xx) \qquad \Omega \longrightarrow_\beta \Omega \longrightarrow_\beta \dots$$

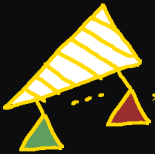
... or still produce a meaningful output:

$$\mathrm{Y}_f := (\lambda x.(f)(x)x)(\lambda x.(f)(x)x)$$

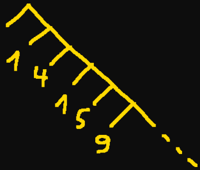
$$\mathrm{Y}_f \longrightarrow_\beta f(\mathrm{Y}_f) \longrightarrow_\beta f(f(\mathrm{Y}_f)) \longrightarrow_\beta \dots$$



The **Böhm tree** of a term is a (possibly infinite) description of all the meaningful information it can produce.

if $M \longrightarrow_\beta^*$ , then $\mathrm{BT}(M) :=$ 

otherwise, then $\mathrm{BT}(M) := \perp$.

$\mathrm{BT}(pi_0) =$ 

THE CONTINOUS APPROXIMATION OF $\longrightarrow_\beta$

For $M \in \Lambda$,

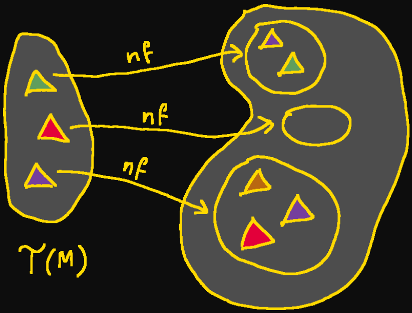
$\mathcal{A}(M) := \Big\{$  stable prefix $\Big|$ $M \longrightarrow_\beta^*$  $\Big\}$

is a directed set such that:

Syntactic approximation theorem (Wadsworth'76, Barendregt'84, ...):

$\mathrm{BT}(M) = \bigsqcup \mathcal{A}(M)$.

For $M \in \Lambda$, its **Taylor approximation** $\mathcal{T}(M)$ is a sum of "polynomial approximations"

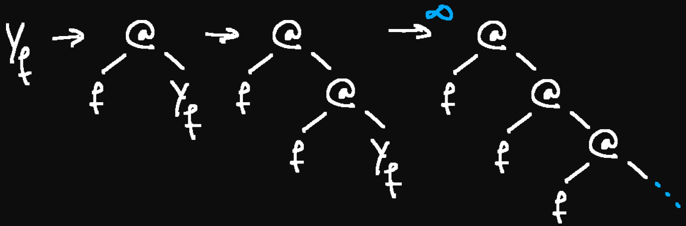


such that:

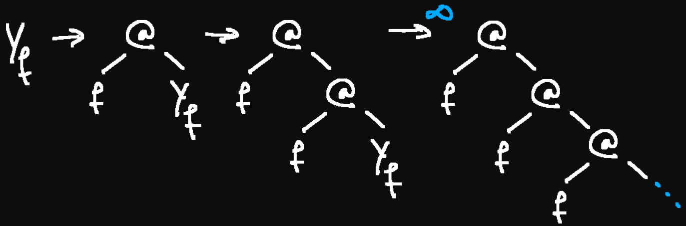
Commutation theorem (Ehrhard-Regnier'06):

$\mathcal{T}(\mathrm{BT}(M)) = \mathrm{nf}(\mathcal{T}(M))$.

Possibly infinite terms and reductions:



Possibly infinite terms and reductions:



such that

Theorem (Kennaway *et al.*'97):
The unique normal form of any $M \in \Lambda^{\infty}_{\perp}$ through $\longrightarrow^{\infty}_{\beta\perp}$ is $\mathrm{BT}(M)$.

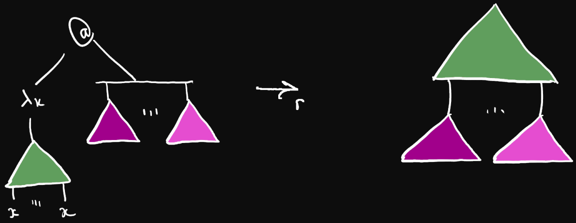
Conclusion

The **resource λ-calculus** is a (multi)linear λ-calculus.

Resource terms:

$$s, t, \ldots \quad := \quad x \quad | \quad \lambda x.s \quad | \quad (s)\,[t_1, \ldots, t_n].$$

Resource reduction, featuring a multilinear substitution:

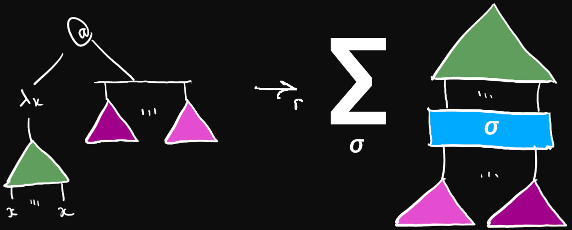


The **resource λ-calculus** is a (multi)linear λ-calculus.

Resource terms:

$$s, t, \ldots \quad := \quad x \quad | \quad \lambda x.s \quad | \quad (s)\,[t_1, \ldots, t_n].$$

Resource reduction, featuring a multilinear substitution:

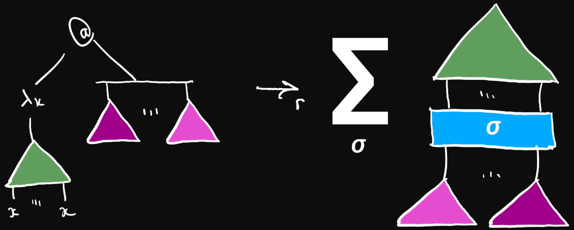


The **resource λ-calculus** is a (multi)linear λ-calculus.

Resource terms:

$$s, t, \ldots \quad := \quad x \quad | \quad \lambda x.s \quad | \quad (s)\,[t_1, \ldots, t_n].$$

Resource reduction, featuring a multilinear substitution:



... and $\mathbf{S} \longrightarrow\!\!\!\!\rightarrow_r \mathbf{T}$ denotes the pointwise reduction (through $\longrightarrow_r^*$) of possibly infinite sums of resource terms.

A maximal setting for the usual Taylor expansion

The **Taylor expansion** for finite λ-terms:

$$\begin{aligned}\mathcal{T}(x) &:= x\\ \mathcal{T}(\lambda x.M) &:= \lambda x.\mathcal{T}(M)\\ \mathcal{T}(MN) &:= \mathcal{T}(M)\left(\sum_{n\in\mathbb{N}}\frac{1}{n!}\mathcal{T}(N)^n\right)\end{aligned}$$

And for 001-infinitary λ-terms? **It's the same!**

- Same target calculus: the resource calculus.
- Same definition (kind of).

- The classical, continuous approximation can only speak about (infinitary) normalisation.
- The Taylor approximation can speak about reduction!

Theorem (commutation, Ehrhard-Regnier'06)

$$\begin{tikzcd} M \arrow[r, "\mathrm{nf}"] \arrow[d, "\mathcal{T}"'] & \mathrm{BT}(M) \arrow[d, "\mathcal{T}"] \\ \mathcal{T}(M) \arrow[r, "\mathrm{nf}"] & \mathcal{T}(\mathrm{BT}(M)) \end{tikzcd}$$

- The classical, continuous approximation can only speak about (infinitary) normalisation.
- The Taylor approximation can speak about reduction!

Theorem (simulation, C.-V.A.'23)

$$\begin{tikzcd} M \arrow[r, "\infty", "\beta\perp"'] \arrow[d, "\mathcal{T}"'] & N \arrow[d, "\mathcal{T}"] \\ \mathcal{T}(M) \arrow[r, twoheadrightarrow, "r"'] & \mathcal{T}(N) \end{tikzcd}$$

We retrieve what we knew about...

... the Taylor expansion:

Corollary: the commutation theorem.

... the 001-infinitary λ-calculus:

Corollary: $\longrightarrow^{\infty}_{\beta\bot}$ is confluent.

... the continuous approximation:

Corollary[1]**:** the syntactic approximation theorem.

[1]But the margin of the manuscript was too short for the second, direct proof...

And there's more!

Corollary

M has a HNF through $\longrightarrow_\beta^*$ or $\longrightarrow_\beta^\infty$
iff the head reduction strategy terminates on M
iff $\mathrm{nf}(\mathcal{T}(M)) \neq 0$.

Corollary

The Genericity lemma.

Corollary

$\mathrm{BT} : \Lambda_\perp^\infty \to \Lambda_\perp^\infty$ is Scott-continuous.

Perspectives

Is our method transferrable, for instance to...

- an extensional setting?
  - there is an extensional Taylor expansion (Blondeau-P. *et al.*'24)
  - infinitary η!-reductions are not well-behaved

PERSPECTIVES

Is our method transferrable, for instance to...

- an extensional setting?
  - there is an extensional Taylor expansion (Blondeau-P. *et al.*'24)
  - infinitary η!-reductions are not well-behaved
- richer languages?
  - use infinitary rewriting in languages having a Taylor approximation?
  - find a Taylor approximation for coinductive languages?

PERSPECTIVES

Is our method transferrable, for instance to...

- an extensional setting?
  - there is an extensional Taylor expansion (Blondeau-P. *et al.*'24)
  - infinitary $\eta!$-reductions are not well-behaved
- richer languages?
  - use infinitary rewriting in languages having a Taylor approximation?
  - find a Taylor approximation for coinductive languages?
- other proof systems?
  - non-wellfounded proof systems, *e.g.* infinets (De *et al.*'21)?
  - characterise good properties to prove validity criteria

BONUSES

Lazy normalisation:

- a meaningful prefix is a **weak** head normal form,

$$\lambda x.M \quad \text{or} \quad (y)M_1 \dots M_n,$$

- the normal forms are **Lévy-Longo trees** (LLT), *e.g.*

$$\mathtt{Y}_{\lambda y.\lambda x.y} \longrightarrow_\beta^* \lambda x.\mathtt{Y}_{\lambda y.\lambda x.y}$$
$$\mathrm{BT}(\mathtt{Y}_{\lambda y.\lambda x.y}) = \bot$$
$$\mathrm{LLT}(\mathtt{Y}_{\lambda y.\lambda x.y}) = \mathtt{O} = \lambda x_0.\lambda x_1.\lambda x_2.\dots$$

- the corresponding infinitary λ-calculus is $(\Lambda_\bot^{101}, \longrightarrow_{\beta\bot}^{101})$.

The **lazy resource λ-calculus**:

$$s, t, \dots \quad := \quad x \quad | \quad \lambda x.s \quad | \quad \mathbb{0} \quad | \quad (s)\,[t_1, \dots, t_n],$$

with $\quad (\mathbb{0})\,\bar{t} \longrightarrow_{\mathrm{r}} 0 \quad$ and $\quad \ell\mathcal{T}(\lambda x.M) := \lambda x.\ell\mathcal{T}(M) + \mathbb{0}.$

Theorem (simulation)

If $M \longrightarrow^{101}_{\beta\perp} N$ then $\ell\mathcal{T}(M) \twoheadrightarrow_{\ell\mathrm{r}} \ell\mathcal{T}(N)$.

Corollary (commutation)

$\mathrm{nf}(\ell\mathcal{T}(M)) = \ell\mathcal{T}(\mathrm{LLT}(M))$.

The **lazy resource λ-calculus**:

$$s, t, \ldots \quad ::= \quad x \quad | \quad \lambda x.s \quad | \quad \mathbb{0} \quad | \quad (s)\,[t_1, \ldots, t_n],$$

with $\quad (\mathbb{0})\,\bar{t} \longrightarrow_{\mathrm{r}} 0 \quad$ and $\quad \ell\mathcal{T}(\lambda x.M) := \lambda x.\ell\mathcal{T}(M) + \mathbb{0}$.

Theorem (simulation)

If $M \longrightarrow_{\beta\bot}^{101} N$ then $\ell\mathcal{T}(M) \twoheadrightarrow_{\ell\mathrm{r}} \ell\mathcal{T}(N)$.

Corollary (commutation)

$\mathrm{nf}(\ell\mathcal{T}(M)) = \ell\mathcal{T}(\mathrm{LLT}(M))$.

Theorem (Severi-de Vries'05)

Only BT and LLT are Scott-continuous.

Question: if $\mathcal{T}(M) \twoheadrightarrow_{\mathrm{r}} \mathcal{T}(N)$, is there a reduction $M \longrightarrow_{\beta}^{\infty} N$?

- In the finite λ-calculus, **yes.**
- In the 001-infinitary λ-calculus, **no.**

- In the finite λ-calculus, we “just” quotient by α-equivalence.

α-EQUIVALENCE FOR MIXED HIGHER-ORDER TERMS

- In the finite λ-calculus, we "just" quotient by α-equivalence.
- With infinitary λ-terms it's not that easy...

$$
\begin{tikzcd}
\begin{array}{c} U(\mu Z.\mathcal{F}_\Sigma(Z,Z)) \\ \mu Z.\mathcal{F}_\Sigma(Z,Z) \\ \mathcal{T}_\Sigma \end{array} \arrow[r, "\text{compl.}"] \arrow[dd, two heads] & \begin{array}{c} \nu Y.\mu X.\mathcal{F}_\Sigma(X,Y) \\ \mathcal{T}_\Sigma^\infty \end{array} \arrow[d, two heads] \\
 & \mathcal{T}_\Sigma^\infty/{=_\alpha} \arrow[d, hook, "\neq"] \\
\begin{array}{c} \mathcal{T}_\Sigma/{=_\alpha} \\ U(\mu Z.\mathcal{Q}_\Sigma(Z,Z)) \end{array} \arrow[r, "\text{compl.}"] & (\mathcal{T}_\Sigma/{=_\alpha})^\infty
\end{tikzcd}
$$

- In the finite λ-calculus, we “just” quotient by α-equivalence.
- With infinitary λ-terms it’s not that easy...
- ... but a solution can be found, adapting existing work (Kurz *et al.*’13).

$$
\begin{tikzcd}
U(\mu Z.\mathcal{F}_\Sigma(Z,Z)) & & & \\
\mu Z.\mathcal{F}_\Sigma(Z,Z) & U(\nu Y.\mu X.\mathcal{F}_\Sigma(X,Y)) & & \nu Y.\mu X.\mathcal{F}_\Sigma(X,Y) \\
\mathcal{T}_\Sigma \arrow[r, "\text{nom.}", "\text{compl.}"'] \arrow[d, two heads] \arrow[rrr, tail, bend left, "\text{compl.}"] & (\mathcal{T}^\infty_\Sigma)_{\mathsf{fs}} \arrow[r, hook] \arrow[d] & (\mathcal{T}^\infty_\Sigma)_{\mathsf{ffv}} \arrow[r, hook] \arrow[d, two heads] \arrow[dr, phantom, "\lrcorner", very near start] & \mathcal{T}^\infty_\Sigma \arrow[d] \\
\mathcal{T}_\Sigma/{=_\alpha} \arrow[r, "\text{nom.}", "\text{compl.}"'] \arrow[rrr, tail, bend right, "\text{compl.}"'] & (\mathcal{T}_\Sigma/{=_\alpha})^\infty_{\mathsf{fs}} \arrow[r, equal] & (\mathcal{T}^\infty_\Sigma)_{\mathsf{ffv}}/{=_\alpha} \arrow[r, hook] & (\mathcal{T}_\Sigma/{=_\alpha})^\infty \\
U(\mu Z.\mathcal{Q}_\Sigma(Z,Z)) & U(\nu Y.\mu X.\mathcal{Q}_\Sigma(X,Y)) & &
\end{tikzcd}
$$

Thanks for your attention!