Linear approximations of functional programs, revisited

Rémy Cerda, CNRS, IRIF

Séminaire APR (LIP6, Paris), 2 April 2025

Linear Approximation of Functional Programs

This is mostly joint work with Lionel Vaux Auclair.

program := instruction (program, ..., program)

program := instruction

program ... program

program := instruction ⇝ **result**

program … program

program := instruction ⇝ **result**

program … program

= program that cannot be further executed (*normal form*)

program := instruction ⇝ **result**

program … program

= program that cannot be further executed (*normal form*)

But the result might be *infinite* and *infinitely far*:

pi(0) ⇝ • ⇝ • ⇝ …

3 pi(1)

3 •

1 pi(2)

Linear Approximation of Functional Programs

What we can compute in finite time are *partial results*:

$$\bot \qquad\qquad \begin{array}{ccc} & \bullet & \\ / & & \backslash \\ 3 & & \bot \end{array} \qquad\qquad \begin{array}{ccccc} & \bullet & & & \\ / & & \backslash & & \\ 3 & & \bullet & & \\ & / & & \backslash & \\ & 1 & & \bot & \end{array} \qquad\qquad \ldots$$

What we can compute in finite time are *partial results*:

As a summary: [Wadsworth, Hyland, Barendregt, 1970s]

$$
\begin{tikzcd}
\text{program} \arrow[rrr, rightsquigarrow, "\text{(infinite) execution}"] \arrow[d, equal] & & & \text{result} \arrow[d, phantom, "\rotatebox{90}{$\sqsupseteq$}"] \\
\text{program} \arrow[rrr, rightsquigarrow, "\text{(finite) partial execution}"'] & & & \text{partial result}
\end{tikzcd}
$$

and the result is the *limit* of all partial results:
it's a *continuous* approximation.

Linear Approximation of Functional Programs

This continuous approximation:

$$\begin{tikzcd}
\text{program} \arrow[r, rightsquigarrow, "\text{(infinite) execution}"] \arrow[d, equal] & \text{result} \arrow[d, phantom, "\rotatebox{90}{$\sqsubseteq$}"] \\
\text{program} \arrow[r, rightsquigarrow, "\text{(finite) partial execution}"'] & \text{partial result}
\end{tikzcd}$$

can be refined into a *linear* one: [Ehrhard-Regnier, 2000s]

$$\begin{tikzcd}
\text{program} \arrow[r, rightsquigarrow, "\text{(infinite) execution}"] \arrow[d, phantom, "\rotatebox{90}{$\sqsubseteq$}"] & \text{result} \arrow[d, phantom, "\rotatebox{90}{$\sqsubseteq$}"] \\
\text{linear program} \arrow[r, rightsquigarrow, "\text{linear execution (finite!)}"'] & \text{linear partial result}
\end{tikzcd}$$

Linear programs: each argument of a function is used *exactly once*.

Linear programs: each argument of a function is used *exactly once*.

program := instruction

program … program

Linear Approximation of Functional Programs

Linear programs: each argument of a function is used *exactly once*.

program $:=$ instruction

program … program

linear program $:=$ instruction

$$\begin{bmatrix}\text{linear}\\ \text{program}\end{bmatrix}, \dots, \begin{bmatrix}\text{linear}\\ \text{program}\end{bmatrix} \quad \dots \quad \begin{bmatrix}\text{linear}\\ \text{program}\end{bmatrix}, \dots, \begin{bmatrix}\text{linear}\\ \text{program}\end{bmatrix}$$

LINEAR APPROXIMATION OF FUNCTIONAL PROGRAMS

λ-terms:

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

λ⊥-terms:

$$\begin{array}{ccccccccc} M, N, \ldots & \coloneqq & x & \mid & \lambda x.M & \mid & MN & \mid & \bot \\ & & & & x \mapsto M & & M(N) & & \end{array}$$

Let's go more technical

The **continuous** approximation:

$$
\begin{tikzcd}
\text{program} \arrow[r, rightsquigarrow, "\text{(infinite) execution}"] \arrow[d, equal] & \textcolor{cyan}{\text{result}} \arrow[d, phantom, sloped, "\sqsupseteq"] \\
\text{program} \arrow[r, rightsquigarrow, "\text{(finite) partial execution}"'] & \text{partial result}
\end{tikzcd}
$$

The **linear** approximation:

$$
\begin{tikzcd}
\text{program} \arrow[r, rightsquigarrow, "\text{(infinite) execution}"] \arrow[d, phantom, sloped, "\sqsupseteq"] & \textcolor{cyan}{\text{result}} \arrow[d, phantom, sloped, "\sqsupseteq"] \\
\text{linear program} \arrow[r, rightsquigarrow, "\text{linear execution (finite!)}"'] & \text{linear partial result}
\end{tikzcd}
$$

What is finite prefix of stable information?
A **head normal form** $\lambda x_1 \dots \lambda x_{.m}.(y)M_1 \dots M_n$.

What is the total information a term can output?
Its **Böhm tree**:

$$\mathrm{BT}(M) := \begin{cases} \lambda\vec{x}.(y)\,\mathrm{BT}(M_1)\ \dots\ \mathrm{BT}(M_n) & \text{if } M \longrightarrow_\beta^* \lambda\vec{x}.(y)M_1 \dots M_n \\ \perp & \text{otherwise.} \end{cases}$$

This is a coinductive definition: $\mathrm{BT}(Y) = \lambda f.f\,f\,f\,\dots$

The **continuous** approximation:

$$\begin{tikzcd}
M \arrow[r, rightsquigarrow, "\text{(infinite) execution}"] \arrow[d, equal] & \mathrm{BT}(M) \arrow[d, phantom, "\sqsupseteq"] \\
M \arrow[r, rightsquigarrow, "\text{(finite) partial execution}"'] & \text{partial result}
\end{tikzcd}$$

The **linear** approximation:

$$\begin{tikzcd}
M \arrow[r, rightsquigarrow, "\text{(infinite) execution}"] \arrow[d, phantom, "\sqsupseteq"] & \mathrm{BT}(M) \arrow[d, phantom, "\sqsupseteq"] \\
\text{linear program} \arrow[r, rightsquigarrow, "\text{linear execution (finite!)}"'] & \text{linear partial result}
\end{tikzcd}$$

- Infinitary terms
  (*via* coinduction, metric completion, ideal completion)
- Infinitary reductions
  (*via* coinduction, transfinite sequences of reductions)

Theorem [Kennaway *et al.* 1997]

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

Corollary

BT(M) is the unique β⊥-normal form of M through $\longrightarrow^{\infty}_{\beta\bot}$.

The **continuous** approximation:

$$\begin{tikzcd}[column sep=huge]
M \arrow[rr, "\infty", "\beta\perp"'] \arrow[d, equal] & & \mathrm{BT}(M) \arrow[d, phantom, sloped, "\sqsupseteq"] \\
M \arrow[rr, rightsquigarrow, "\text{(finite) partial execution}"'] & & \text{partial result}
\end{tikzcd}$$

The **linear** approximation:

$$\begin{tikzcd}[column sep=huge]
M \arrow[rr, "\infty", "\beta\perp"'] \arrow[d, phantom, sloped, "\sqsupseteq"] & & \mathrm{BT}(M) \arrow[d, phantom, sloped, "\sqsupseteq"] \\
\text{linear program} \arrow[rr, rightsquigarrow, "\text{linear execution (finite!)}"'] & & \text{linear partial result}
\end{tikzcd}$$

The **continuous** approximation:

$$\begin{tikzcd}
M \arrow[rr, "\infty", "\beta\perp"'] \arrow[d, equal] & & \mathrm{BT}(M) \arrow[d, phantom, sloped, "\sqsupseteq"] \\
M \arrow[r, "*", "\beta"'] & M' \arrow[r, phantom, "\sqsupseteq"] & P
\end{tikzcd}$$

Continous approximation theorem

- $\mathcal{A}(M) \coloneqq \left\{ P \text{ in } \beta\perp\text{-nf} \;\middle|\; \exists M',\ M \longrightarrow^*_\beta M' \sqsupseteq P \right\}$ is directed.
- $\bigsqcup \mathcal{A}(M) = \mathrm{BT}(M)$.

$$\begin{tikzcd}[column sep=huge]
M \arrow[r, "\infty", "\beta\bot"'] & \mathrm{BT}(M) \\
\text{linear program} \arrow[u, phantom, "\sqsubseteq" sloped] \arrow[r, squiggly, "\text{linear execution (finite!)}"'] & \text{linear partial result} \arrow[u, phantom, "\sqsubseteq" sloped]
\end{tikzcd}$$

Linear programs are **resource λ-terms**:

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

A λ-term is **Taylor expanded** into a formal sum of resource approximants:

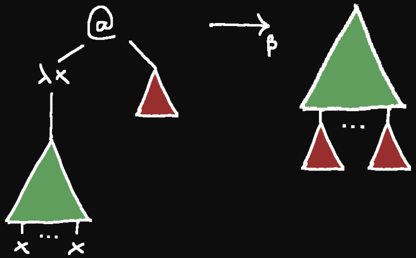
$$\begin{aligned}
\mathcal{T}(x) &:= x & \{x\} \\
\mathcal{T}(\lambda x.M) &:= \lambda x.\mathcal{T}(M) & \{\, \lambda x.s \mid s \in \mathcal{T}(M) \,\} \\
\mathcal{T}(MN) &:= \mathcal{T}(M) \sum_{n \in \mathbb{N}} \frac{1}{n!} \mathcal{T}(N)^n & \{\, s[t_1, \dots, t_n] \mid \dots \,\} \\
\mathcal{T}(\bot) &:= 0 & \varnothing
\end{aligned}$$

... and this also works for infinitary terms (kind of).

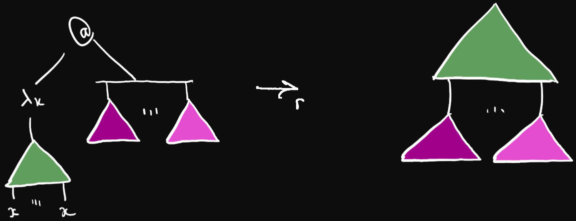
$$\begin{tikzcd}
M \arrow[rr, "\infty", "\beta\perp"'] \arrow[d, phantom, "\rotatebox{90}{$\simeq$}"] && \mathrm{BT}(M) \arrow[d, phantom, "\rotatebox{90}{$\simeq$}"] \\
\mathcal{T}(M) \arrow[rr, squiggly, "\text{linear execution (finite!)}"'] && \mathcal{T}(\mathrm{BT}(M))
\end{tikzcd}$$

Program execution is the **β-reduction** on λ-terms:

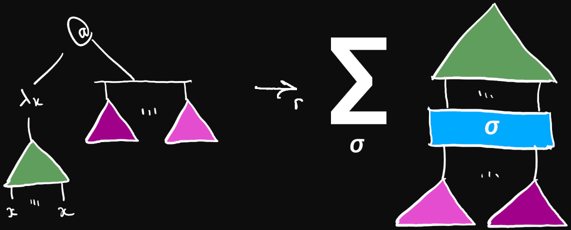
$$(\lambda x.M)N \longrightarrow_\beta M[N/x]$$



Linear execution is the **resource reduction** on resource λ-terms:

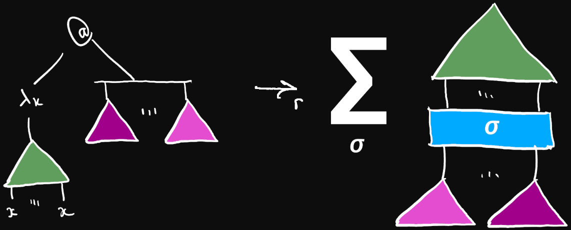


Linear execution is the **resource reduction** on resource λ-terms:



Linear Approximation of the λ-calculus

Linear execution is the **resource reduction** on resource λ-terms:



It is confluent and strongly normalising!

$$\begin{array}{ccc} M & \xrightarrow[\beta\perp]{\infty} & \mathrm{BT}(M) \\ \wr\!| & & \wr\!| \\ \mathcal{T}(M) & \underset{\mathrm{r}}{\twoheadrightarrow} & \mathcal{T}(\mathrm{BT}(M)) \end{array}$$

Theorem (simulation) [C. and Vaux 2023, C. 2024]

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

Corollary (commutation)

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

Other corollaries

All that we've seen before!

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 $\mathrm{nf}(\mathcal{T}(M)) \neq 0$
iff M is typable in "the" intersection type system.

Corollary

The Genericity lemma.

Corollary

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

What about other settings?

- **Lazy:** it works perfectly.
- **Extentional:** it should work (but there's an open problem to solve first).
- **Probabilistic:** complicated but certainly funny...

- WIP: refinement in order to give a semantic accound of terms "pushing to the infinity" different pieces of data

[C., Manzonetto, Saurin 2025]

Some satellite questions

Fact 1 (simulation, finite)

Let M, N be finite λ-terms.

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

Problem 1 (conservativity, finite)

Is the converse true?

Theorem 1 [C. and Vaux 2025]

Yes it is! If $\mathcal{T}(M) \twoheadrightarrow_{\mathrm{r}} \mathcal{T}(N)$ then $M \longrightarrow_\beta^* N$.

Fact 2 (simulation, infinitary)

Let M, N be infinitary λ-terms.

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

Problem 2 (conservativity, infinitary)

Is the converse true?

Theorem 2 [C. and Vaux 2025]

No, it isn't!

There are terms $\mathtt{A}$, $\bar{\mathtt{A}}$ such that $\mathcal{T}(\mathtt{A}) \twoheadrightarrow_{\mathrm{r}} \mathcal{T}(\bar{\mathtt{A}})$ but there is no reduction $\mathtt{A} \longrightarrow_\beta^\infty \bar{\mathtt{A}}$.

$\mathtt{A}$ is the *Accordion* λ-term.

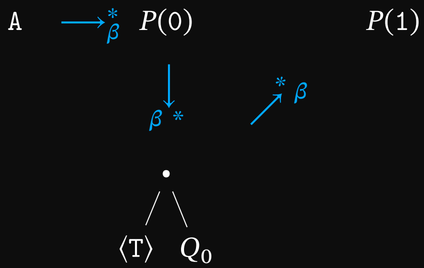
The Accordion

$$\mathtt{A} \longrightarrow^{*}_{\beta} P(0)$$

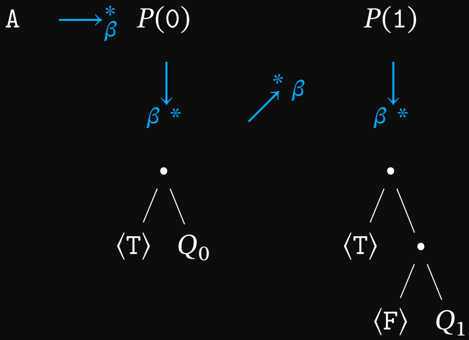
The Accordion

$$\begin{tikzcd} \mathtt{A} \arrow[r, "*", "\beta"'] & P(0) \arrow[d, "\beta\,*"] & \\ & \bullet \arrow[dl, dash] \arrow[dr, dash] & \\ \langle\mathtt{T}\rangle & & Q_0 \end{tikzcd}$$

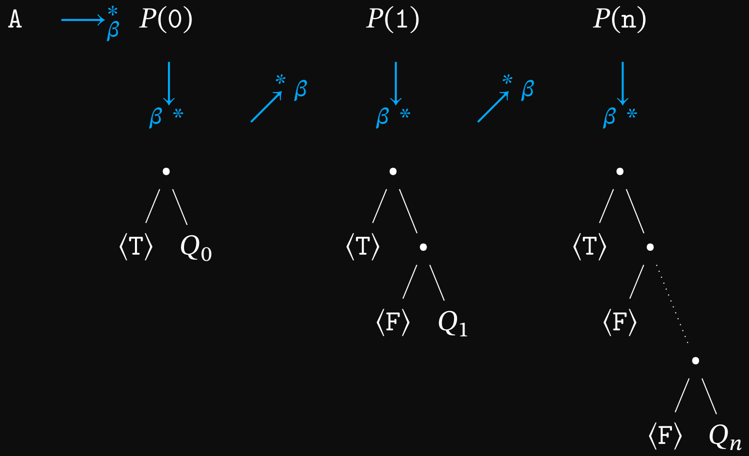
The Accordion



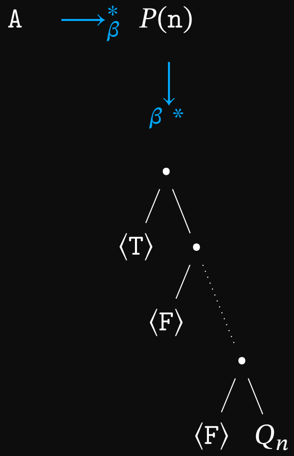
The Accordion



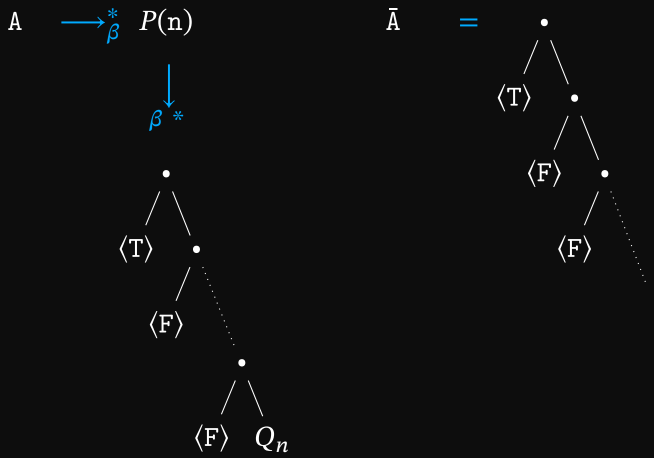
The Accordion



The Accordion



The Accordion



Problem 3 (restoring conservativity)

Can we restrict $\twoheadrightarrow_{\mathrm{r}}$ to obtain a conservative approximation?

Theorem 3 [C. and Vaux 2025]

Yes, thanks to the *uniform* lifting of the resource reduction $\stackrel{\frown}{\longrightarrow}{}^{\infty}_{\mathrm{r}}$!

If $\mathcal{T}(M) \stackrel{\frown}{\longrightarrow}{}^{\infty}_{\mathrm{r}} \mathcal{T}(N)$ then $M \longrightarrow^{\infty}_{\beta} N$.

In particular, there is no reduction $\mathcal{T}(\mathtt{A}) \stackrel{\frown}{\longrightarrow}{}^{\infty}_{\mathrm{r}} \mathcal{T}(\bar{\mathtt{A}})$.

α-EQUIVALENCE FOR MIXED HIGHER-ORDER TERMS

- 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...

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

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...
- ... but a solution can be found! [Kurz *et al.* 2013, C. 2025]

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

What's next?

What's next?

Towards real life!

- Stream calculi
  - Starting from non-wellfounded proofs
  - Starting for $\Lambda\mu$
- Concurrent programs
  - Looking at implicit complexity
- Non-wellfounded proofs for proof assistants
  - We need a compositional syntax
- Higer-order model checking

Thanks for your attention!