Böhm Trees and Taylor Expansion

Giulio Manzonetto

gmanzone@irif.fr

IRIF, Université Paris Cité

13 May 2024

Table of contents

Introduction

Böhm Trees

The Resource Calculus

Taylor Expansion and Applications

Conclusions

The Big Picture

Program Approximation

Calculi

Denotational Semantics

λ-Calculus

The Big Picture

Program Approximation

Calculi

Denotational Semantics

$$\begin{tikzcd} \lambda\text{-Calculus} \arrow[dr, bend left] & \\ & \text{Scott's Semantics} \end{tikzcd}$$

The Big Picture

Program Approximation	Calculi	Denotational Semantics

$$
\begin{tikzcd}
 & \lambda\text{-Calculus} \arrow[dl, bend right] \arrow[dr, bend left] & \\
\text{Böhm Trees} & & \text{Scott's Semantics}
\end{tikzcd}
$$

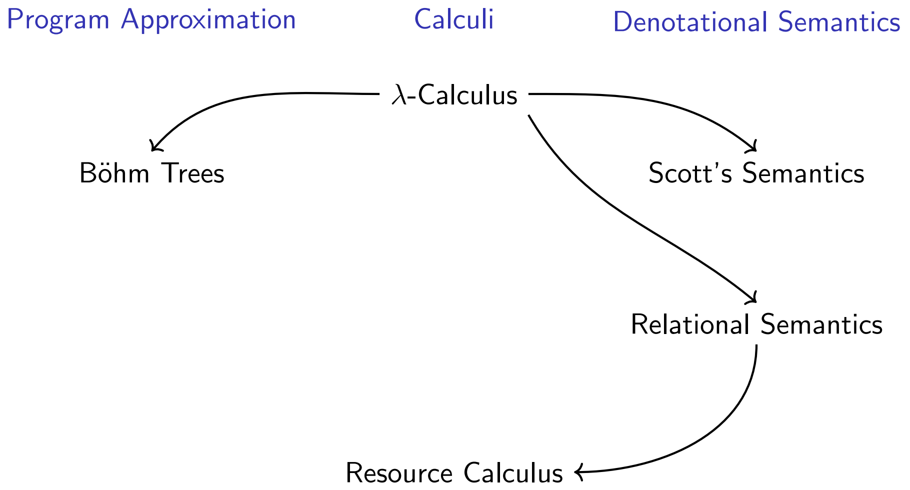
The Big Picture

Program Approximation | Calculi | Denotational Semantics

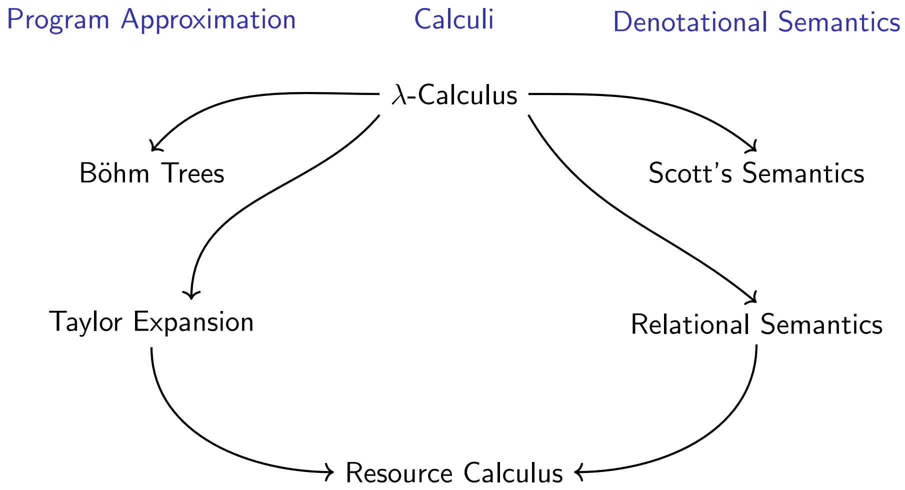
$$\begin{tikzcd}
 & \lambda\text{-Calculus} \arrow[dl] \arrow[dr] \arrow[ddr] & \\
\text{Böhm Trees} & & \text{Scott's Semantics} \\
 & & \text{Relational Semantics}
\end{tikzcd}$$

ELEPHANT IN THE ROOM

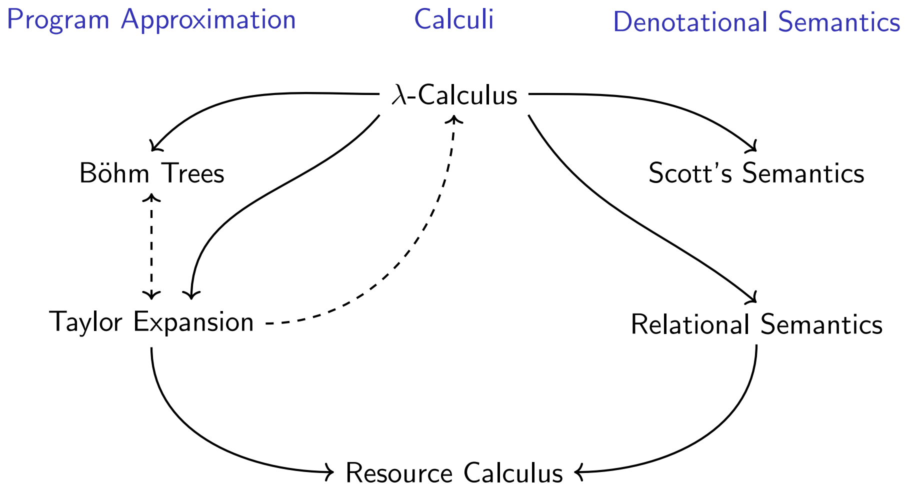
The Big Picture



The Big Picture



The Big Picture



The untyped λ-calculus (Church, 1932)

Based on a primitive notion of function.

$$M, N \ ::= \ x \mid \lambda x.M \mid MN$$

Computation becomes substitution

$$(\lambda x.M)N \to_\beta M\{N/x\}$$

A program is a closed term $M \in \Lambda^o$.

Some examples

1. The identity $\mathsf{I} = \lambda \mathsf{x}.\mathsf{x}$
$$\mathsf{IM} \to_\beta \mathsf{M}$$
2. The projections $\mathsf{K} = \lambda \mathsf{xy}.\mathsf{x}$ and $\mathsf{F} = \lambda \mathsf{xy}.\mathsf{y}$:
$$\mathsf{KMN} \to_\beta (\lambda \mathsf{y}.\mathsf{M})\mathsf{N} \to_\beta \mathsf{M}$$
3. The self-application $\Delta = \lambda x.xx$
$$\Delta M \to_\beta MM$$
4. The “looping” combinator $\Omega = \Delta\Delta$
$$\Omega = \Delta\Delta \to_\beta \Delta\Delta = \Omega \to_\beta \Omega \to_\beta \cdots$$

The Theory of Program Approximation

How to handle complex programs?

Denotational Semantics

- Model = abstract mathematical structure.
- Define a program interpretation satisfying compositionality.

$$[\![MN]\!] = [\![M]\!] \bullet [\![N]\!].$$

Systems of Approximants

- Decompose a program into elementary "bricks"
- Retrieve the whole program behaviour performing some "limit" of its (finite) approximants.

$$M = \bigvee \{A \mid A \text{ is an approximant of } M\}$$

Denotational vs Operational Models

Continuous Semantics (Scott, 1969)

$\mathcal{D}_\infty$: First denotational model of λ-calculus.

Böhm tree semantics (Barendregt, 1977)

Tree-like representation for program execution.

"Syntactic model" of λ-calculus.

Scott's topology

Let $\mathcal{D} = (D, \leq, \perp)$ be a complete partial ordering:

$$d \leq d' \iff \text{the datum } d \text{ is less defined than } d'$$

The Scott topology on $\mathcal{D}$ is defined as follows: O is Scott-open if

1. O is upward closed: $\forall d \in O, d' \in D \,.\, [d \leq d' \quad \Rightarrow \quad d' \in O]$
2. $\forall A \subseteq O \,.\, A \text{ directed and } \bigvee A \in O \quad \Rightarrow \quad A \cap O \neq \emptyset$

directed = non-empty, downward closed, closed under finite $\vee$.

Examples. Let $\mathcal{D}$ be a cpo, $d \in D$. The following sets are open:

- $\{x \in D \mid x \neq \perp\}$
- $\{x \in D \mid x \not\leq d\}$

The Scott topology is T_0: all points are topologically distinguishable.

Scott continuity

Proposition. Let $\mathcal{D}$ be a cpo. A function

$$f : D \to D$$

is Scott continuous if and only if, for every directed subset $\mathcal{I} \subseteq D$,

$$f\left(\bigvee \mathcal{I}\right) = \bigvee f(\mathcal{I}).$$

A function is Scott continuous means

"A finite portion of the output of a program must be generated by a finite portion of its input."

The Crucial Point — How to Handle Recursion?

Kleene Fixed Point Theorem

Let $\mathcal{D} = (D, \leq, \perp)$ be a domain. Every Scott-continuous function

$$f : \mathcal{D} \to \mathcal{D}$$

has a least fixed point $\mathrm{lfp}(f)$ that can be calculated as follows:

$$\mathrm{lfp}(f) = \bigvee_{n \in \mathbb{N}} f^n(\perp)$$

The Kleene Fixed Point Theorem *is used in denotational semantics, to give meaning to recursive function definitions in programming languages.*

Fixed Point Combinators

1. A λ-term X is a fixed point of a λ-term M if

$$MX =_\beta X$$

Open Problem. Define

$$\mathrm{Fix}(M) = \{[X]_\beta \mid X \in \Lambda^o \,.\, MX =_\beta X\}$$

Conjecture. For all $M \in \Lambda^o$ either $|\mathrm{Fix}(M)| = 1$ or $\mathrm{Fix}(M) = \aleph_0$.

Fixed Point Combinators

1. A λ-term X is a fixed point of a λ-term M if

$$MX =_\beta X$$

2. One can define fixed point combinators:

$$\mathsf{Y} = \lambda \mathsf{f}.(\lambda \mathsf{x}.\mathsf{f}(\mathsf{xx}))(\lambda \mathsf{x}.\mathsf{f}(\mathsf{xx}))$$

Lemma. For all $M \in \Lambda$, $\mathsf{Y}\mathsf{M}$ is a fixed point of M:

$$\begin{array}{rcl} \mathsf{YM} & = & (\lambda f.(\lambda x.f(xx))(\lambda x.f(xx)))M \\ & \to_\beta & (\lambda x.M(xx))(\lambda x.M(xx)) \\ & \to_\beta & M((\lambda x.M(xx))(\lambda x.M(xx)))\ {}_\beta\!\leftarrow M(\mathsf{YM}) \end{array}$$

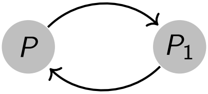
Possible behaviours of a program

Classification	Behaviour	Result
β-normalizable	$P \to P_1 \to P_2 \twoheadrightarrow_{97} P_{99} \to 42$	completely defined

$$P \longrightarrow P_1 \longrightarrow P_2 \cdots\cdots\twoheadrightarrow P_{99} \longrightarrow 42$$

Possible behaviours of a program

Classification	Behaviour	Result
β-normalizable	$P \to P_1 \to P_2 \twoheadrightarrow_{97} P_{99} \to 42$	completely defined
unsolvable	$P \to P_1 \to P \twoheadrightarrow_{97} P_1 \to \cdots$	undefined



Possible behaviours of a program

Classification	Behaviour	Result
β-normalizable	$P \to P_1 \to P_2 \twoheadrightarrow_{97} P_{99} \to 42$	completely defined
unsolvable	$P \to P_1 \to P \twoheadrightarrow_{97} P_1 \to \cdots$	undefined
solvable	$P \to o_1 P_1 \to o_1(o_2 P_2) \to \cdots$ $\twoheadrightarrow_\infty o_1(o_2(o_3(\cdots o_n))\cdots)$	stable parts (infinitary)

$$
\begin{array}{ccccccccc}
P & \longrightarrow & P_1 & \longrightarrow & P_2 & \longrightarrow & P_3 & \cdots\cdots\ggg & \pi \\
 & & 3, & & 1 & & 4 & & 15926535\cdots
\end{array}
$$

The Böhm Tree Semantics (Barendregt'77)

Given a λ-term M, its Böhm tree $\mathrm{BT}(M)$ is defined as follows:

- If M is unsolvable (completely undefined), then

$$\mathrm{BT}(M) = \perp$$

- Otherwise $M \twoheadrightarrow_\beta \lambda x_1 \dots x_n.y\, M_1 \cdots M_k$ and

$$\begin{array}{ccc} & \mathrm{BT}(M) = \lambda x_1 \dots x_n.y & \\ \swarrow & & \searrow \\ \mathrm{BT}(M_1) & \cdots & \mathrm{BT}(M_k) \end{array}$$

One of the first coinductive definition!

The Böhm Tree Semantics (Barendregt'77)

Given a λ-term M, its Böhm tree $\mathrm{BT}(M)$ is defined as follows:

- If M is unsolvable (completely undefined), then

$$\mathrm{BT}(M) = \bot$$

- Otherwise $M \twoheadrightarrow_\beta \lambda x_1 \dots x_n.y\, M_1 \cdots M_k$ and

$$\mathrm{BT}(M) = \begin{array}{c} \lambda x_1 \dots x_n.y \\ \diagup \qquad \diagdown \\ \mathrm{BT}(M_1) \quad \cdots \quad \mathrm{BT}(M_k) \end{array}$$

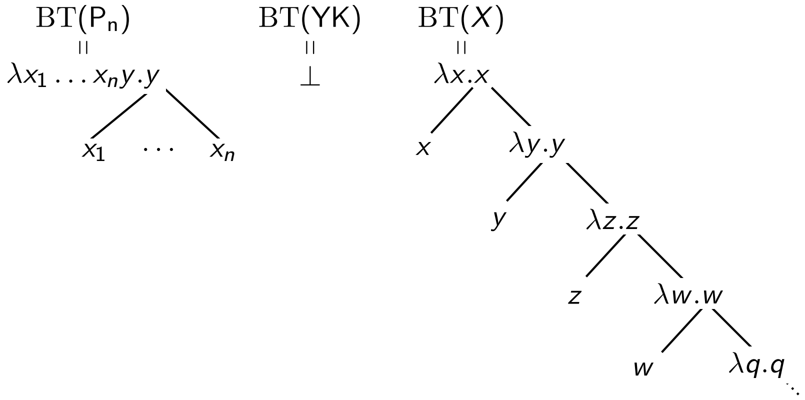
One of the first coinductive definition!

Example - Fixed point combinator Y

$$
\begin{aligned}
\mathsf{Y} &= \lambda f.(\lambda x.f(xx))(\lambda x.f(xx)) \\
&\to_\beta \lambda f.f((\lambda x.f(xx))(\lambda x.f(xx))) \\
&\to_\beta \lambda f.f(f((\lambda x.f(xx))(\lambda x.f(xx)))) \\
&\to_\beta \lambda f.f(f(f((\lambda x.f(xx))(\lambda x.f(xx))))) \\
&\twoheadrightarrow_\beta \lambda f.f^n((\lambda x.f(xx))(\lambda x.f(xx))) \\
&\twoheadrightarrow_\beta \cdots
\end{aligned}
$$

$$
\begin{array}{c}
\mathrm{BT}(\mathsf{Y}) \\
\shortparallel \\
\lambda f.f \\
| \\
f \\
| \\
f \\
| \\
f \\
| \\
\vdots
\end{array}
$$

Böhm trees - Examples



where

- $\mathsf{P_n} = \lambda \mathsf{x_1 \ldots x_n y.y x_1 \cdots x_n}$
- $X = \mathsf{Y}(\lambda \mathsf{yx.xxy})$

The Böhm tree semantics is "infinitary"

There are λ-terms M, N with the same Böhm tree, that cannot be equated by any "finite" reduction.

1. Take a λ-term M satisfying:

$$M \twoheadrightarrow_\beta \lambda zx.x(Mz)$$

2. Take a variable y. Then,

$$\mathrm{BT}(My) = \begin{array}{c} \lambda x.x \\ | \\ \mathrm{BT}(My) \end{array} = \begin{array}{c} \lambda x.x \\ | \\ \lambda x.x \\ | \\ \vdots \end{array}$$

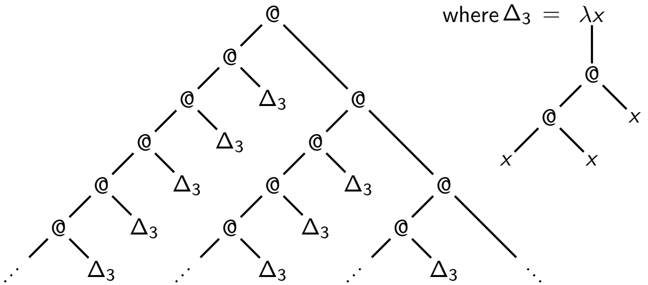
3. For $y \neq z$, we have $My \neq_\beta Mz$
   but $\mathrm{BT}(My) = \mathrm{BT}(Mz)$.

Digression - Böhm trees as infinitary normal forms

Generate Λ^∞ by taking the grammar of λ-calculus coinductively:

$$T, U ::=_{\text{co-ind}} x \mid \lambda x.T \mid TU$$

Example:



Digression - Böhm trees as infinitary normal forms

Take the infinitary λ-calculus $\Lambda^\infty_\perp$ with $\perp$:

$$T, U ::=_{\text{co-ind}} \perp \mid x \mid \lambda x.T \mid TU$$

Reduction is now coinductive:

$$\frac{T \twoheadrightarrow_\beta x}{T \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp} x} \qquad \frac{T \twoheadrightarrow_{\beta\perp} \lambda x.U' \quad U' \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp} U}{T \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp} \lambda x.U}$$

$$\frac{T \twoheadrightarrow_\beta U'V' \quad U' \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp} U \quad V' \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp} V}{T \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp} UV}$$

$$\frac{T \text{ has no hnf} \quad T \neq \perp}{T \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp} \perp}\ (\perp)$$

Theorem. $(\Lambda^\infty_\perp, \twoheadrightarrow\!\!\!\!\rightarrow_{\beta\perp})$ is confluent and strongly normalizing.
For all $M \in \Lambda$: $\mathrm{BT}(M) = \mathrm{nf}^\infty_{\beta\perp}(M)$.

That was scary. . . can we go back to induction?

- Finite trees are pieces of “output” that can be obtained in a finite amount of time.
- Böhm trees are naturally ordered, as follows:

$$\begin{array}{ccccccccc}
\bot & \sqsubseteq & \lambda f.f & \sqsubseteq & \lambda f.f & \sqsubseteq & \lambda f.f & \sqsubseteq & \lambda f.f \\
 & & | & & | & & | & & | \\
 & & \bot & & f & & f & & f \\
 & & & & | & & | & & | \\
 & & & & \bot & & f & & f \\
 & & & & & & | & & | \\
 & & & & & & \bot & & f \\
 & & & & & & & & | \\
 & & & & & & & & \vdots
\end{array}$$

Finite Approximants

The set $\mathcal{A}$ of finite approximants is defined as follows:

$$(\mathcal{A}) \qquad A, A_i \ ::= \ \bot \mid \lambda x_1 \ldots x_n . y\, A_1 \cdots A_k$$

The Scott-ordering $\sqsubseteq$ on $\mathcal{A}$ is defined by:

$$\frac{}{\bot \sqsubseteq A} \qquad \frac{A \sqsubseteq A'}{\lambda x.A \sqsubseteq \lambda x.A'} \qquad \frac{A_1 \sqsubseteq A_1' \cdots A_n \sqsubseteq A_n'}{xA_1 \cdots A_n \sqsubseteq xA_1 \cdots A_n}$$

The corresponding "sup" $A \sqcup A'$ is inductively given by:

$$\begin{aligned} \bot \sqcup A &= A \sqcup \bot = A \\ A_1 A_2 \sqcup A_1' A_2' &= (A_1 \sqcup A_1')(A_2 \sqcup A_2') \\ \lambda x.A \sqcup \lambda x.A' &= \lambda x.(A \sqcup A') \end{aligned}$$

Direct approximation

The direct approximant $\omega(M) \in \mathcal{A}$ of $M \in \Lambda$ is inductively defined:

$$\begin{aligned} \omega(\lambda x_1 \dots x_n . y M_1 \cdots M_k) = & \quad \lambda x_1 \dots x_n . y \omega(M_1) \cdots \omega(M_k), \\ \omega(\lambda x_1 \dots x_n . (\lambda y . P) Q M_1 \cdots M_k) = & \quad \perp . \end{aligned}$$

The set of finite approximants of a λ-term M is defined by:

$$\mathcal{A}(M) = \{\omega(N) \mid N =_\beta M\} \downarrow$$

Lemma.

1. $M \to_\beta N$ implies $\omega(M) \sqsubseteq \omega(N)$.
2. If $M =_\beta N$ then $\mathcal{A}(M) = \mathcal{A}(N)$.

Lemma. The set $\mathcal{A}(M)$ is an ideal w.r.t. $\sqsubseteq$:

1. $\perp \in \mathcal{A}(M)$;
2. if $A_1, A_2 \in \mathcal{A}(M)$ then $A_1 \sqcup A_2 \in \mathcal{A}(M)$;
3. downward closed: $A_1 \sqsubseteq A_2 \in \mathcal{A}(M) \quad \Rightarrow \quad A_1 \in \mathcal{A}(M)$.

Proof. (1) and (3) hold trivially.

Lemma. The set $\mathcal{A}(M)$ is an ideal w.r.t. $\sqsubseteq$:

1. $\bot \in \mathcal{A}(M)$;
2. if $A_1, A_2 \in \mathcal{A}(M)$ then $A_1 \sqcup A_2 \in \mathcal{A}(M)$;
3. downward closed: $A_1 \sqsubseteq A_2 \in \mathcal{A}(M) \quad \Rightarrow \quad A_1 \in \mathcal{A}(M)$.

Proof. (1) and (3) hold trivially.
2) If $A_1, A_2 \in \mathcal{A}(M)$ then $A_1 \sqsubseteq \omega(N_1)$ and $A_2 \sqsubseteq \omega(N_2)$ for some

$$N_1 \qquad =_\beta \qquad M \qquad =_\beta \qquad N_2$$

Lemma. The set $\mathcal{A}(M)$ is an ideal w.r.t. $\sqsubseteq$:

1. $\perp \in \mathcal{A}(M)$;
2. if $A_1, A_2 \in \mathcal{A}(M)$ then $A_1 \sqcup A_2 \in \mathcal{A}(M)$;
3. downward closed: $A_1 \sqsubseteq A_2 \in \mathcal{A}(M) \quad \Rightarrow \quad A_1 \in \mathcal{A}(M)$.

Proof. (1) and (3) hold trivially.
2) If $A_1, A_2 \in \mathcal{A}(M)$ then $A_1 \sqsubseteq \omega(N_1)$ and $A_2 \sqsubseteq \omega(N_2)$ for some

$$\begin{tikzcd}
N_1 \arrow[dr] & =_\beta & M \arrow[dl] \arrow[dr] & =_\beta & N_2 \arrow[dl] \\
& R_1 \arrow[dr] & & R_2 \arrow[dl] & \\
& & R & &
\end{tikzcd}$$

Conclude since direct approximants increase along reduction. □

Lemma. The set $\mathcal{A}(M)$ is an ideal w.r.t. $\sqsubseteq$:

1. $\perp \in \mathcal{A}(M)$;
2. if $A_1, A_2 \in \mathcal{A}(M)$ then $A_1 \sqcup A_2 \in \mathcal{A}(M)$;
3. downward closed: $A_1 \sqsubseteq A_2 \in \mathcal{A}(M) \quad \Rightarrow \quad A_1 \in \mathcal{A}(M)$.

Proof. (1) and (3) hold trivially.
2) If $A_1, A_2 \in \mathcal{A}(M)$ then $A_1 \sqsubseteq \omega(N_1)$ and $A_2 \sqsubseteq \omega(N_2)$ for some

$$\begin{tikzcd}
N_1 \arrow[dr, ->>] & =_\beta & M \arrow[dl, ->>] \arrow[dr, ->>] & =_\beta & N_2 \arrow[dl, ->>] \\
& R_1 \arrow[dr, ->>] & & R_2 \arrow[dl, ->>] & \\
& & R & &
\end{tikzcd}$$

Conclude since direct approximants increase along reduction. $\square$

The Syntactic Approximation Theorem

For all $M \in \Lambda$,

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

Examples:

- $\mathcal{A}(\Omega) = \{\perp\}$, for $\Omega = (\lambda x.xx)(\lambda x.xx)$,
- $\mathcal{A}(\mathrm{Y}) = \{ \perp, \lambda f.f\perp, \lambda f.f(f\perp), \lambda f.f(f(f\perp)), \ldots, \lambda f.f^n(\perp), \ldots \}$

Example

$$\begin{array}{c} \mathrm{BT}(\mathrm{Y}) \\ \shortparallel \\ \lambda f.f \\ | \\ f \\ | \\ f \\ | \\ f \\ | \\ \vdots \end{array}$$

The Syntactic Approximation Theorem

For all $M \in \Lambda$,

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

Examples:

- $\mathcal{A}(\Omega) = \{\perp\}$, for $\Omega = (\lambda x.xx)(\lambda x.xx)$,
- $\mathcal{A}(\mathsf{Y}) = \{ \perp, \lambda f.f\perp, \lambda f.f(f\perp), \lambda f.f(f(f\perp)), \dots, \lambda f.f^n(\perp), \dots \}$

Example

$$\mathrm{BT}(\mathsf{Y}) = \perp$$

The Syntactic Approximation Theorem

For all $M \in \Lambda$,

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

Examples:

- $\mathcal{A}(\Omega) = \{\perp\}$, for $\Omega = (\lambda x.xx)(\lambda x.xx)$,
- $\mathcal{A}(\mathsf{Y}) = \{ \perp, \lambda f.f\perp, \lambda f.f(f\perp), \lambda f.f(f(f\perp)), \ldots, \lambda f.f^n(\perp), \ldots \}$

Example

$$\begin{array}{c} \mathrm{BT}(\mathsf{Y}) \\ \shortparallel \\ \lambda f.f \\ | \\ \perp \end{array}$$

The Syntactic Approximation Theorem

For all $M \in \Lambda$,

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

Examples:

- $\mathcal{A}(\Omega) = \{\perp\}$, for $\Omega = (\lambda x.xx)(\lambda x.xx)$,
- $\mathcal{A}(\mathsf{Y}) = \{ \perp, \lambda f.f\perp, \lambda f.f(f\perp), \lambda f.f(f(f\perp)), \dots, \lambda f.f^n(\perp), \dots \}$

Example

$$\begin{array}{c} \mathrm{BT}(\mathsf{Y}) \\ \shortparallel \\ \lambda f.f \\ | \\ f \\ | \\ \perp \end{array}$$

The Syntactic Approximation Theorem

For all $M \in \Lambda$,

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

Examples:

- $\mathcal{A}(\Omega) = \{\perp\}$, for $\Omega = (\lambda x.xx)(\lambda x.xx)$,
- $\mathcal{A}(\mathsf{Y}) = \{\ \perp,\ \lambda f.f\perp,\ \lambda f.f(f\perp),\ \lambda f.f(f(f\perp)), \ldots,\ \lambda f.f^n(\perp), \ldots\}$

Example

$\mathrm{BT}(\mathsf{Y})$

$\shortparallel$

$\lambda f.f$

|

f

|

f

|

$\perp$

The Syntactic Approximation Theorem

For all $M \in \Lambda$,

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

Examples:

- $\mathcal{A}(\Omega) = \{\perp\}$, for $\Omega = (\lambda x.xx)(\lambda x.xx)$,
- $\mathcal{A}(\mathsf{Y}) = \begin{array}{ll} \{ & \perp, \\ & \lambda f.f\perp, \\ & \lambda f.f(f\perp), \\ & \lambda f.f(f(f\perp)), \ldots, \\ & \lambda f.f^n(\perp), \ldots \} \end{array}$

Example

$$\begin{array}{c} \mathrm{BT}(\mathsf{Y}) \\ \shortparallel \\ \lambda f.f \\ | \\ f \\ | \\ f \\ | \\ f \\ | \\ \perp \end{array}$$

The Syntactic Approximation Theorem

For all $M \in \Lambda$,

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

Examples:

- $\mathcal{A}(\Omega) = \{\perp\}$, for $\Omega = (\lambda x.xx)(\lambda x.xx)$,
- $\mathcal{A}(\mathsf{Y}) = \begin{array}{l} \{ \quad \perp, \\ \qquad \lambda f.f\perp, \\ \qquad \lambda f.f(f\perp), \\ \qquad \lambda f.f(f(f\perp)), \ldots, \\ \qquad \lambda f.f^n(\perp), \ldots \} \end{array}$

Example

$$\begin{array}{c} \mathrm{BT}(\mathsf{Y}) \\ \shortparallel \\ \lambda f.f \\ | \\ f \\ | \\ f \\ | \\ f \\ | \\ \vdots \end{array}$$

There is a 1-to-1 correspondence between $\mathcal{A}(M)$ and $\mathrm{BT}(M)$.

The Böhm-tree semantics

How to prove that the equivalence

$$M =_{\mathcal{B}} N \iff \mathrm{BT}(M) = \mathrm{BT}(N)$$

is an equational theory of λ-calculus? We need to check:

- $=_{\mathcal{B}}$ contains $=_{\beta}$. (done)
- for all contexts $C[]$, $M =_{\mathcal{B}} N \;\Rightarrow\; C[M] =_{\mathcal{B}} C[N]$. (difficult)

Problem. If I give you $\mathrm{BT}(M)$ and $\mathrm{BT}(N)$, what can you tell me of

$$\mathrm{BT}(MN)?$$

The topological approach

Consider the map

$$\mathrm{BT}(-) : \Lambda \to \text{Böhm trees}.$$

Böhm trees are an algebraic cpo endowed with Scott topology.

The inverse image of Scott topology defines a topology on Λ.

1. $\{M \mid M \text{ solvable}\}$ is open.
2. $\{M \mid M \text{ unsolvable}\}$ is closed.
3. Unsolvables are compactification point:
   The only open set containing all unsolvables is the whole set Λ.
4. Every β-normalizable M is an isolated point:
   $$\{N \mid M =_\beta N\} \text{ is an open set.}$$
5. The application and λ-abstraction on Λ are Scott continuous.

See [§14.2,Barendregt84.]

Böhm trees contextuality - Method 1.

Theorem. $=_{\mathcal{B}}$ is compatible with the application.

Proof. Let $N =_{\mathcal{B}} N'$, we want to prove $MN =_{\mathcal{B}} MN'$. Assume that $MN \neq_{\mathcal{B}} MN'$, i.e., $\mathrm{BT}(MN) \neq \mathrm{BT}(MN')$, towards a contradiction.

- Since Scott's topology is T_0, there is a Scott open O containing, say, MN but not MN'.
- By continuity of application there exists an open set U containing N' such that $\{MX \mid X \in U\} \subseteq O$.

In conclusion:

$$\mathrm{BT}(N) = \mathrm{BT}(N') \quad \Rightarrow \quad N' \in \{MX \mid X \in U\} \subseteq O \text{ (absurd)}.$$

Böhm trees contextuality

Method 2. Use $\Lambda^\infty_\perp$ with infinitary ordinal reduction $\twoheadrightarrow_\infty$.

Assume $\mathrm{BT}(M) = \mathrm{BT}(M')$ and $\mathrm{BT}(N) = \mathrm{BT}(N')$.

$$\begin{tikzcd}
MN \arrow[r, twoheadrightarrow, "\infty"] \arrow[d, twoheadrightarrow, "\infty"] & \mathrm{BT}(M)\cdot\mathrm{BT}(N) \arrow[dl, twoheadrightarrow, "\infty"] & = & \mathrm{BT}(M')\cdot\mathrm{BT}(N') \arrow[d, twoheadrightarrow, "\infty"] \\
\mathrm{BT}(MN) & & & \mathrm{BT}(M'N')
\end{tikzcd}$$

Conclude $\mathrm{BT}(MN) = \mathrm{BT}(M'N')$ by the unicity of normal forms.

The Genericity Lemma

Genericity. Let M be a λ-term, U an unsolvable and N a β-nf.

$$MU =_\beta N \quad \Rightarrow \quad \forall L \in \Lambda \,.\, ML =_\beta N.$$

Proof.

1. We have seen that $\{P \mid P =_\beta N\}$ is Scott open.
2. By continuity of $M \mapsto MN$, the set $O = \{P \mid MP =_\beta N\}$ is also a Scott open containing U.
3. The leftmost strategy is normalizable, whence

$$MU =_\beta N \iff MU =_\mathcal{B} N \iff MV =_\mathcal{B} N, \forall V \text{ unsolvable}$$

4. So O contains all unsolvables.
5. But the unique open set including all unsolvables is Λ.

The Big Picture

Program Approximation | Calculi | Denotational Semantics

$$\begin{tikzcd} & \lambda\text{-Calculus} \arrow[dl, bend right] \arrow[dr, bend left] & \\ \text{Böhm Trees} & & \text{Scott's Semantics} \end{tikzcd}$$

(Relational Semantics)

Resource Calculus

The Resource Calculus

Lambda Calculus is not resource conscious

In one step of β-reduction

$$(\lambda x.M)N \to_\beta M\{N/x\}$$

the argument N can be:

- Erased: $(\lambda xy.y)N \to_\beta \lambda y.y$
- Duplicated: $(\lambda x.xx)N \to_\beta NN$
- Copied an arbitrary number of times:

$$(\lambda fz.f(f(\cdots f(z))))N \to_\beta \lambda z.N(N(\cdots N(z)))$$

Linear Logic is resource sensitive

Linear Logic decomposes the intuitionistic arrow

$$A \to B \qquad \text{as} \qquad !A \multimap B$$

and this suggests that one step of β-reduction

$$(\lambda x.M)N \to_\beta M\{N/x\}$$

should be decomposable into more elementary steps.

Linear Lambda Calculus is ~~stupid~~ extremely basic

The naïve calculus arising from linearity

$$\lambda x.M \quad \Rightarrow \quad x \text{ occurs exactly once in } M$$

is not very interesting from the operational point of view.

$$\lambda xyz.xyz,\ \lambda xyz.yzx,\ \lambda xyz.zxy,\ \lambda xyz.x(\lambda f.yf)(\lambda g.gz),\ldots$$

It can be interesting from a combinatorial perspective:

Noam Zeilberger. Counting isomorphism classes of β-normal linear lambda terms. arXiv:1509.07596 (2015)

Noam Zeilberger: Linear lambda terms as invariants of rooted trivalent maps. J. Funct. Program. 26: e21 (2016)

The Resource Calculus

It is a resource sensitive version of λ-calculus where

- variables can occur multiple times in its programs,
- resources cannot be erased nor copied during the reduction.

Introduced in

T. Ehrhard, L. Regnier: The differential lambda-calculus. Theor. Comput. Sci. 309(1-3): 1-41 (2003)

More understandable syntax in

M. Pagani, P. Tranquilli: Parallel Reduction in Resource Lambda-Calculus. APLAS 2009: 226-242

Ancestor

G. Boudol: The Lambda-Calculus with Multiplicities. CONCUR 1993: 1-6

Its syntax

Syntactic categories:

Terms	s, t, u	$::=$	$x \mid \lambda x.t \mid tb$	Λ^r
Bags	b	$::=$	$[t_1, \ldots, t_n], \text{ for } n \geq 0,$	Λ^b
Formal sums	$\mathbb{S}, \mathbb{T}, \mathbb{U}$	$::=$	$0 \mid t + \mathbb{T}$	$\mathbb{N}\langle\Lambda^r\rangle$

Intuitively

- Terms are the protagonists of our calculus.
- Bags are multisets of linear resources.
- Sums represent non-deterministic choice between terms

$$s + t \to s \qquad\qquad s + t \to t$$

but the choice is never actually made.

Assumptions

On formal sums

- The operator $+$ is associative and commutative.
- As usual, we write $\sum_{i=1}^{k} t_i = t_1 + \cdots + t_k$

Bags are multisets represented in multiplicative notation.

- 1 is the empty bag.
- $b_1 \cdot b_2$ represents the multiset union of b_1 and b_2.
- Structural induction on bags, becomes:
  - 1, base case.
  - $[t] \cdot b$, induction step.

Sums of bags $\mathbb{B} \in \mathbb{N}\langle \Lambda^b \rangle$ are useful but not important.

All constructors are linear

In this context

"linearity" $\simeq$ commutation with sums

Notation.
For sums in $\mathbb{N}\langle\Lambda^r\rangle$, we introduce a syntactic sugar:

$$\begin{array}{rcl} \lambda x.\sum_{i=1}^n t_i & := & \sum_{i=1}^n \lambda x.t_i \\ (\sum_{i=1}^n t_i)b & := & \sum_{i=1}^n t_i b \\ t(\sum_{i=1}^n b_i) & := & \sum_{i=1}^n t b_i \end{array}$$

In other words, sums can always be pushed to surface.

Remark. A subterm 0 annihilates the whole term:

$$\lambda x.0 = t0 = 0b = 0 \qquad \text{and} \qquad [0]\cdot b = 0$$

All constructors are linear

In this context

"linearity" $\simeq$ commutation with sums

Notation.
For sums in $\mathbb{N}\langle\Lambda^r\rangle$, we introduce a syntactic sugar:

$$\begin{array}{rcl} \lambda x. \sum_{i=1}^n t_i & := & \sum_{i=1}^n \lambda x.t_i \\ (\sum_{i=1}^n t_i)b & := & \sum_{i=1}^n t_i b \\ t(\sum_{i=1}^n b_i) & := & \sum_{i=1}^n t b_i \end{array}$$

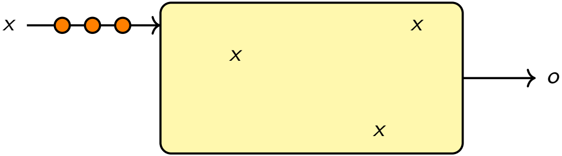
In other words, sums can always be pushed to surface.

Remark. A subterm 0 annihilates the whole term:

$$\lambda x.0 = t0 = 0b = 0 \qquad \text{and} \qquad [0] \cdot b = 0$$

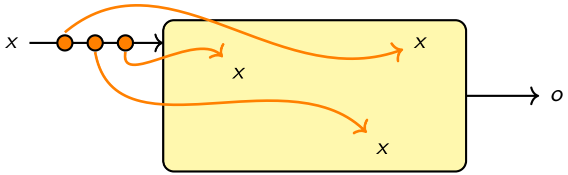
Its operational semantics – the idea

$$(\lambda x.t)[s_1, s_2, s_3] \rightarrow \ ?$$



Its operational semantics – the idea

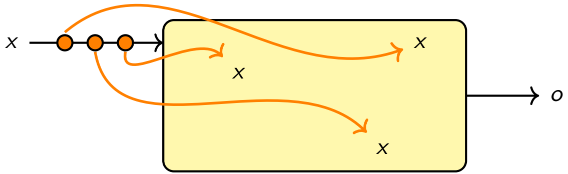
$$(\lambda x.t)[s_1, s_2, s_3] \to t\langle s_1/x_1, s_2/x_2, s_3/x_3\rangle$$



$\deg_x(x)$	$=$	1	$\deg_x(y)$	$=$	0
$\deg_x(\lambda y.t)$	$=$	$\deg_x(t)$	$\deg_x(tb)$	$=$	$\deg_x(t) + \deg_x(b)$
$\deg_x(1)$	$=$	0	$\deg_x([t] \cdot b)$	$=$	$\deg_x(t) + \deg_x(b)$

Its operational semantics – the idea

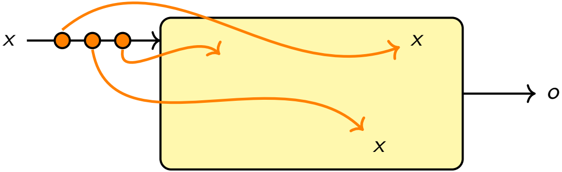
$$(\lambda x.t)[s_1, s_2, s_3] \to \sum_{\sigma \in \mathfrak{S}_3} t\langle s_1/x_{\sigma(1)}, s_2/x_{\sigma(2)}, s_3/x_{\sigma(3)}\rangle$$



$\deg_x(x)$	$=$	1	$\deg_x(y)$	$=$	0
$\deg_x(\lambda y.t)$	$=$	$\deg_x(t)$	$\deg_x(tb)$	$=$	$\deg_x(t) + \deg_x(b)$
$\deg_x(1)$	$=$	0	$\deg_x([t] \cdot b)$	$=$	$\deg_x(t) + \deg_x(b)$

Its operational semantics – the idea

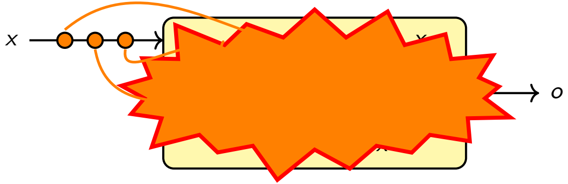
$$(\lambda x.t)[s_1, s_2, s_3] \to \ ?$$



$\deg_x(x)$	$=$	1	$\deg_x(y)$	$=$	0
$\deg_x(\lambda y.t)$	$=$	$\deg_x(t)$	$\deg_x(tb)$	$=$	$\deg_x(t) + \deg_x(b)$
$\deg_x(1)$	$=$	0	$\deg_x([t] \cdot b)$	$=$	$\deg_x(t) + \deg_x(b)$

Its operational semantics – the idea

$$(\lambda x.t)[s_1, s_2, s_3] \to 0$$



$\deg_x(x)$	$=$	1	$\deg_x(y)$	$=$	0
$\deg_x(\lambda y.t)$	$=$	$\deg_x(t)$	$\deg_x(tb)$	$=$	$\deg_x(t) + \deg_x(b)$
$\deg_x(1)$	$=$	0	$\deg_x([t] \cdot b)$	$=$	$\deg_x(t) + \deg_x(b)$

Linear substitution

For $s, t \in \Lambda^r$, define $t\langle s/x\rangle \in \mathbb{N}\langle\Lambda^r\rangle$ aka the

linear substitution of s for one occurrence of x in t

$$
\begin{aligned}
y\langle s/x\rangle &= \begin{cases} s, & \text{if } x = y, \\ 0, & \text{otherwise,} \end{cases} \\
(\lambda y.t)\langle s/x\rangle &= \lambda y.t\langle s/x\rangle && (\text{wlog. } x \neq y) \\
(tb)\langle s/x\rangle &= t\langle s/x\rangle b + t(b\langle s/x\rangle)
\end{aligned}
$$

on bags:

$$
\begin{aligned}
1\langle s/x\rangle &= 0 \\
[t]\langle s/x\rangle &= [t\langle s/x\rangle] \\
(b_1 \cdot b_2)\langle s/x\rangle &= b_1\langle s/x\rangle \cdot b_2 + b_1 \cdot (b_2\langle s/x\rangle)
\end{aligned}
$$

Linear substitution

For $s, t \in \Lambda^r$, define $t\langle s/x\rangle \in \mathbb{N}\langle\Lambda^r\rangle$ aka the

linear substitution of s for one occurrence of x in t

$$
\begin{aligned}
y\langle s/x\rangle &= \begin{cases} s, & \text{if } x = y, \\ 0, & \text{otherwise,} \end{cases} \\
(\lambda y.t)\langle s/x\rangle &= \lambda y.t\langle s/x\rangle && (\text{wlog. } x \neq y) \\
(tb)\langle s/x\rangle &= t\langle s/x\rangle b + t(b\langle s/x\rangle)
\end{aligned}
$$

on bags:

$$
\begin{aligned}
1\langle s/x\rangle &= 0 \\
[t]\langle s/x\rangle &= [t\langle s/x\rangle] \\
(b_1 \cdot b_2)\langle s/x\rangle &= b_1\langle s/x\rangle \cdot b_2 + b_1 \cdot (b_2\langle s/x\rangle)
\end{aligned}
$$

Its operational semantics

Baby-step. Define $\to_{\mathsf{b}} \subseteq \Lambda^r \times \mathbb{N}\langle\Lambda^r\rangle$ by

$$\begin{array}{lcl} (\lambda x.t)([s]\cdot b) & \to_{\mathsf{b}} & (\lambda x.t\langle s/x\rangle)b \\ (\lambda x.t)1 & \to_{\mathsf{b}} & \begin{cases} t, & \text{if } x \notin \mathsf{fv}(t), \\ 0, & \text{otherwise.} \end{cases} \end{array}$$

Normal step. Define $\to_{\mathsf{r}} \subseteq \Lambda^r \times \mathbb{N}\langle\Lambda^r\rangle$ by

$$(\lambda x.t)[s_1,\dots,s_n] \to_{\mathsf{r}} \begin{cases} \sum_{\sigma\in\mathfrak{S}_n} t\{s_1/x_{\sigma(1)},\dots,s_n/x_{\sigma(n)}\}, & \text{if } \mathsf{deg}_x(t) = n, \\ 0, & \text{otherwise.} \end{cases}$$

Examples of reductions

Both notions of reduction extend to $\to \subseteq \mathbb{N}\langle\Lambda^r\rangle \times \mathbb{N}\langle\Lambda^r\rangle$ by

$$t \to t' \quad \Rightarrow \quad t + \mathbb{T} \to t' + \mathbb{T}$$

- Baby-steps:

$$\begin{array}{ll}
 & (\lambda xy.x[y,y])[a][b,b] \\
\to_{\mathsf{b}} & (\lambda xy.a[y,y])1[b,b] \\
\to_{\mathsf{b}} & (\lambda y.a[y,y])[b,b] \\
\to_{\mathsf{b}} & (\lambda y.a[b,y])[b] + (\lambda y.a[y,b])[b] = 2.(\lambda y.a[b,y])[b] \\
\to_{\mathsf{b}} & (\lambda y.a[b,b])1 + (\lambda y.a[b,y])[b] \\
\to_{\mathsf{b}} & 2.(\lambda y.a[b,b])1 \\
\overset{2}{\to}_{\mathsf{b}} & 2.a[b,b] = a[b,b] + a[b,b]
\end{array}$$

Examples of reductions

Both notions of reduction extend to $\to \subseteq \mathbb{N}\langle\Lambda^r\rangle \times \mathbb{N}\langle\Lambda^r\rangle$ by

$$t \to t' \quad \Rightarrow \quad t + \mathbb{T} \to t' + \mathbb{T}$$

- Baby-steps:

$$\begin{array}{rl}
& (\lambda xy.x[y,y])[a][b,b] \\
\to_{\mathsf{b}} & (\lambda xy.a[y,y])1[b,b] \\
\to_{\mathsf{b}} & (\lambda y.a[y,y])[b,b] \\
\to_{\mathsf{b}} & (\lambda y.a[b,y])[b] + (\lambda y.a[y,b])[b] = 2.(\lambda y.a[b,y])[b] \\
\to_{\mathsf{b}} & (\lambda y.a[b,b])1 + (\lambda y.a[b,y])[b] \\
\to_{\mathsf{b}} & 2.(\lambda y.a[b,b])1 \\
\overset{2}{\to}_{\mathsf{b}} & 2.a[b,b] = a[b,b] + a[b,b]
\end{array}$$

Examples of reductions

Both notions of reduction extend to $\to \subseteq \mathbb{N}\langle\Lambda^r\rangle \times \mathbb{N}\langle\Lambda^r\rangle$ by

$$t \to t' \quad \Rightarrow \quad t + \mathbb{T} \to t' + \mathbb{T}$$

- Normal-steps:

$$\begin{aligned}(\lambda xy.x[y,y])[a][b,b] \quad &\to_{\mathsf{r}} \quad (\lambda y.a[y,y])[b,b] \\ &\to_{\mathsf{r}} \quad 2.a[b,b]\end{aligned}$$

Theorem (Equivalence baby $\leftrightarrow$ normal reductions)

- *If* $t \to_{\mathsf{r}} \mathbb{T}$ *then* $t \twoheadrightarrow_{\mathsf{b}} \mathbb{T}$.
- *For* $\mathbb{T} = t_1 + \cdots + t_n$ *with each* t_i *in normal form, we have:*

$$t \twoheadrightarrow_{\mathsf{b}} \mathbb{T} \iff t \twoheadrightarrow_{\mathsf{r}} \mathbb{T}$$

Main Properties – Strong Normalization

Theorem. The resource calculus is strongly normalizing (SN).

Proof Define:

- the size $\#t \in \mathbb{N}$ of a term t, as you imagine.
- the size of a sum $\#(t_1 + \cdots + t_n) = [\#t_1, \ldots, \#t_n] \in \mathcal{M}_f(\mathbb{N})$

Check that $\#$ decreases along a reduction

$$(\lambda x.t)[s_1, \ldots, s_n] \to_{\mathrm{b}} \begin{cases} t\langle s_1/x_1, \ldots, s_n/x_n\rangle \ {\color{red}?} & \deg_x(t) = n \\ 0 \ {\color{green}\checkmark} & \textit{otherwise} \end{cases}$$

w.r.t. the multiset ordering $<_m$.

Main Properties – Confluence

Theorem. The resource calculus is confluent.

Proof. The resource calculus is *locally confluent*, i.e., $t \to_{\mathsf{r}} \mathbb{T}_1$ and $t \to_{\mathsf{r}} \mathbb{T}_2$ imply $\exists \mathbb{S}$ such that $\mathbb{T}_1 \twoheadrightarrow_{\mathsf{r}} \mathbb{S}$ and $\mathbb{T}_2 \twoheadrightarrow_{\mathsf{r}} \mathbb{S}$.

$$\begin{tikzcd}
 & t \arrow[dl] \arrow[dr] & \\
\mathbb{T}_1 \arrow[dr, dashed, two heads] & & \mathbb{T}_2 \arrow[dl, dashed, two heads] \\
 & \mathbb{S} &
\end{tikzcd}$$

Conclude by Newmann's Lemma. A term rewriting system is confluent if it is SN and locally confluent. $\square$

Main Properties – Linearity

- Starvation:

$$(\lambda x.x[x])[\lambda x.x[x], \lambda x.x[x]] \to_{\mathsf{r}} 2.(\lambda x.x[x])[\lambda x.x[x]] \to_{\mathsf{r}} 0$$

- Surfeit:

$$(\lambda fgx.f[g[x]])[h][b, c] \to_{\mathsf{r}} (\lambda gx.h[g[x]])[b, c] \to_{\mathsf{r}} 0$$

- Non-determinism:

$$(\lambda x.x[x])[f, g] \to_{\mathsf{r}} f[g] + g[f]$$

Main Properties – Summary

The Resource Calculus

$$
\begin{array}{rcll}
t & ::= & x \mid \lambda x.t \mid t\,b & \\
b & ::= & [t_1, \ldots, t_n] & \text{where } n \geq 0 \\
\mathbb{T} & ::= & t_1 + \cdots + t_n &
\end{array}
$$

Reduction:

$(\lambda x.t)[s_1, \ldots, s_n] \twoheadrightarrow_{\mathsf{r}} \mathbb{T} \neq 0 \quad \Rightarrow \quad t$ must use each s_i exactly once in the reduction to a value.

$t \twoheadrightarrow_{\mathsf{r}} c(\!|0|\!) = 0 \quad \Leftarrow \quad$ otherwise, the whole program t becomes an empty program 0.

Main Properties

Strong Normalization:	Trivial, because there is no duplication.	✓
Confluence:	Locally confluent + strongly normalizing.	✓
Linearity:	Nothing gets erased in a non-zero reduction.	✓

Its expressive power

The Big Picture

Program Approximation

Calculi

Denotational Semantics

$$
\begin{array}{ccccc}
 & & \lambda\text{-Calculus} & & \\
 & \swarrow & & \searrow & \\
\text{Böhm Trees} & & & & \text{Scott's Semantics}
\end{array}
$$

Taylor Expansion

(Relational Semantics)

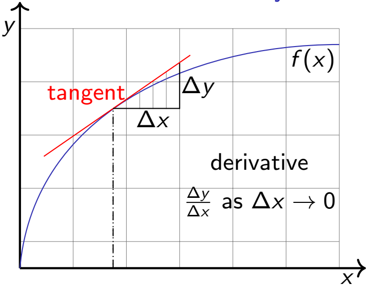
Resource Calculus

Is the Resource Calculus of any interest?

$$t\langle s/x\rangle \qquad \cong \qquad \frac{\partial t}{\partial x} \cdot s$$

Towards a differential theory of program approximations

Mathematical Analysis

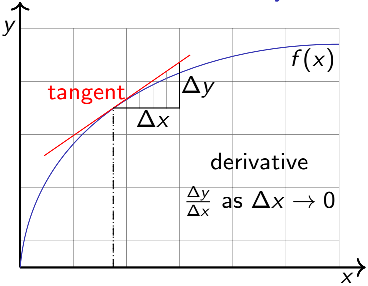


Taylor expansion

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Towards a differential theory of program approximations

Mathematical Analysis



Taylor expansion

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Theory of Programming Languages

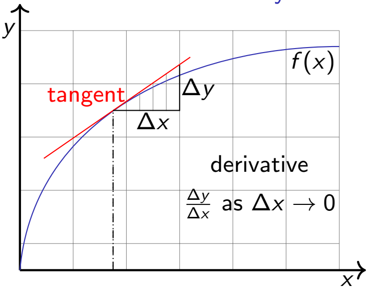
The differential λ-calculus

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

linear substitution of N
for one occurrence of x in M

Towards a differential theory of program approximations

Mathematical Analysis



Taylor expansion

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Theory of Programming Languages

The differential λ-calculus

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

linear substitution of N
for one occurrence of x in M

Taylor expansion $\mathcal{T}(-)$

$$P\,x = \sum_{n=0}^{\infty} \frac{1}{n!}\big(\mathrm{D}^n(P) \cdot (x, \ldots, x)\big)0$$

The ambitious goal

Replace the theory of program approximation based on

Scott-continuity and Böhm trees

with the theory of

resource consumption based on Taylor expansion.

Resource calculus = Target language of Taylor expansion

Taylor Expansion

$$\mathcal{T}(-): \quad \Lambda \quad \to \quad \text{power series of resource approximants}$$

$$\begin{array}{rcl} \mathcal{T}(x) & = & x \\ \mathcal{T}(\lambda x.M) & = & \lambda x.\mathcal{T}(M) \\ \mathcal{T}(MN) & = & \displaystyle\sum_{k\in\mathbb{N}} \frac{1}{k!}\mathcal{T}(M)[\underbrace{\mathcal{T}(N),\ldots,\mathcal{T}(N)}_{k \text{ times}}] \end{array}$$

Taylor Expansion

$$\mathcal{T}(-) : \quad \Lambda \quad \to \quad \text{sets of resource approximants}$$

$$\begin{aligned}
\mathcal{T}(x) &= \{x\} \\
\mathcal{T}(\lambda x.M) &= \{\lambda x.t \mid t \in \mathcal{T}(M)\} \\
\mathcal{T}(MN) &= \bigcup_{k \in \mathbb{N}} \{t[s_1, \ldots, s_k] \mid t \in \mathcal{T}(M), s_1, \ldots, s_k \in \mathcal{T}(N)\}
\end{aligned}$$

Examples

- $\mathcal{T}(\mathrm{I}) = \{\lambda \mathrm{x.x}\}$,
- $\mathcal{T}(\Delta) = \{\lambda \mathrm{x.x1}, \lambda \mathrm{x.x[x]}, \lambda \mathrm{x.x[x,x]}, \lambda \mathrm{x.x[x,x,x]}, \ldots\}$,
  $= \{\lambda x.x[n.x] \mid n \geq 0\}$,
- $\mathcal{T}(\Omega) = \{(\lambda \mathrm{x.x[n.x]})[\lambda \mathrm{x.x[n_1.x]}, \ldots, \lambda \mathrm{x.x[n_k.x]}] \mid \mathrm{n, k, n_i} \geq 0\}$.

The Dynamics of Taylor Expansion

The Taylor expansion is a “static” operation

- $M = x \quad\Rightarrow\quad t \in \mathcal{T}(M)$ has the shape of a variable,
- $M = \lambda x.N \quad\Rightarrow\quad t \in \mathcal{T}(M)$ has the shape of an abstraction,
- $M = PQ \quad\Rightarrow\quad t \in \mathcal{T}(M)$ has the shape of an application,
- $M = (\lambda x.P)Q \quad\Rightarrow\quad t \in \mathcal{T}(M)$ has the shape of a redex.

As the Resource Calculus enjoys SN, we can define:

$$\mathrm{NF}(\mathcal{T}(M)) = \bigcup \{\mathrm{nf}(t) \mid t \in \mathcal{T}(M)\}$$

Normalizing the Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(\Omega)) = \emptyset$$

- For every $t \in \mathcal{T}(\Omega)$, check $t \twoheadrightarrow_{\mathsf{r}} 0$.
- Conclude $\mathcal{T}(\Omega) = \emptyset$.

More generally:

$$M \text{ is unsolvable} \quad \Rightarrow \quad \mathrm{NF}(\mathcal{T}(M)) = \emptyset$$

(We'll prove it later)

Normalizing the Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(\Omega)) = \emptyset$$

- For every $t \in \mathcal{T}(\Omega)$, check $t \twoheadrightarrow_{\mathsf{r}} 0$.
- Conclude $\mathcal{T}(\Omega) = \emptyset$.

More generally:

$$M \text{ is unsolvable} \quad \Rightarrow \quad \mathrm{NF}(\mathcal{T}(M)) = \emptyset$$

(We'll prove it later)

Normalizing the Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(\mathsf{Y})) = ?$$

Normalizing the Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(\mathsf{Y})) = ?$$

- Mmm...

Let us look at its shape:

$$\mathsf{Y} = \lambda \mathsf{f}.(\lambda \mathsf{x}.\mathsf{f}(\mathsf{xx}))(\lambda \mathsf{x}.\mathsf{f}(\mathsf{xx}))$$

Normalizing the Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(\mathsf{Y})) = ?$$

- Mmm…

Let us look at its shape:

$$\mathsf{Y} = \lambda \mathsf{f}.(\lambda \mathsf{x}.\mathsf{f}(\mathsf{xx}))(\lambda \mathsf{x}.\mathsf{f}(\mathsf{xx}))$$

- Mmm…

Problem! That's a toughy…

We know how to compute its Böhm tree

$$\mathrm{BT}(\mathsf{Y}) = \lambda \mathsf{f}.\mathsf{f}(\mathsf{f}(\mathsf{f}(\mathsf{f}(\cdots))))$$

since

$$\mathcal{A}(\mathsf{Y}) = \{\lambda \mathsf{f}.\mathsf{f}\bot, \lambda \mathsf{f}.\mathsf{f}(\mathsf{f}\bot), \lambda \mathsf{f}.\mathsf{f}(\mathsf{f}(\mathsf{f}\bot)), \lambda \mathsf{f}.\mathsf{f}(\mathsf{f}(\mathsf{f}(\mathsf{f}\bot))), \dots\}$$

Can we Taylor expand a Böhm tree?

For an approximant A, define:

$$\begin{array}{rcl} \mathcal{T}(\perp) & = & \emptyset, \\ \mathcal{T}(x) & = & \{x\}, \\ \mathcal{T}(\lambda x.A) & = & \{\lambda x.t \mid t \in \mathcal{T}(A)\}, \\ \mathcal{T}(A_1A_2) & = & \bigcup_{k\in\mathbb{N}} \left\{ t[s_1,\dots,s_k] \mid t \in \mathcal{T}(A_1), s_1,\dots,s_k \in \mathcal{T}(A_2) \right\}. \end{array}$$

Then, we can simply define

$$\mathcal{T}(\mathrm{BT}(M)) = \bigcup_{A\in\mathcal{A}(M)} \mathcal{T}(A)$$

Commutation Taylor / Böhm

Theorem (Ehrhard & Regnier 2003)

For every λ-term M, we have:

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

Thanks! $\mathcal{T}(\mathrm{BT}(\mathsf{Y})) = \{\lambda \mathsf{f}.\mathsf{f}1, \lambda \mathsf{f}.\mathsf{f}[\lambda \mathsf{f}.\mathsf{f}1], \lambda \mathsf{f}.\mathsf{f}[\lambda \mathsf{f}.\mathsf{f}1, \lambda \mathsf{f}.\mathsf{f}1], \dots\}$

Commutation Taylor / Böhm

Theorem (Ehrhard & Regnier 2003)

For every λ-term M, we have:

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

Corollary 1

$$M \text{ is unsolvable} \iff \mathrm{NF}(\mathcal{T}(M)) = \emptyset$$

Corollary 2

$$\mathrm{BT}(M) = \mathrm{BT}(N) \iff \mathrm{NF}(\mathcal{T}(M)) = \mathrm{NF}(\mathcal{T}(N))$$

Taylor Expansion vs Böhm Trees

$$\begin{tikzcd}
\Lambda \arrow[r, "\mathcal{T}"] \arrow[d, "\mathrm{BT}"'] & \Lambda^r \arrow[d, "\mathrm{NF}"] \\
\mathcal{B} \arrow[r, "\mathcal{T}"'] & \mathrm{NF}(\Lambda^r)
\end{tikzcd}$$

Advantages:

1. Approximants are closed under application.
2. Enjoy Strong Normalization + Linearity.
3. Generalizable to the mainstream languages.

Disadvantage:

1. lots of indices arise from the linearization.

Classic results

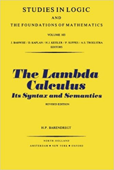
Scott's Continuity

topological argument

Contextuality of BTs

Genericity Lemma

Chapter 14



Berry's Stability

$\nexists$ parallel or

Perpendicular Lines Lemma

Khan & Plotkin's Sequentiality

D. Barbarossa and G. Manzonetto. Taylor Subsumes Scott, Berry, Kahn and Plotkin. PACMPL Vol. 4, pp. 1:1-1:23, 2020.

Classic results with simpler inductive proofs

Scott's Continuity

Berry's Stability

Commutation Theorem
$\mathrm{NF}(\mathcal{T}(P)) = \mathcal{T}(\mathrm{BT}(P))$

$\nexists$ parallel or

Perpendicular
Lines Lemma

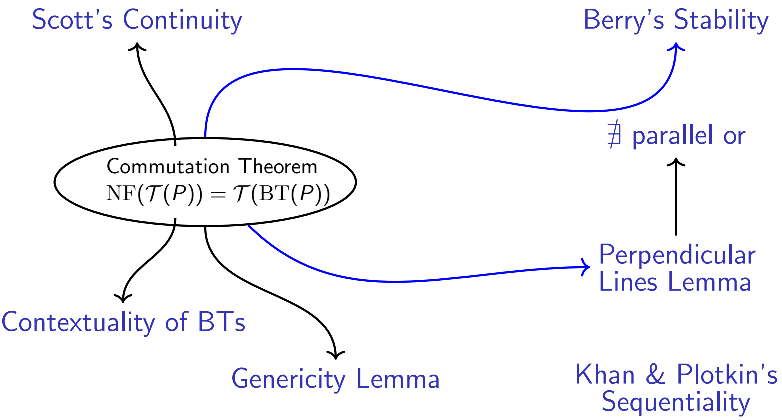
Contextuality of BTs

Genericity Lemma

Khan & Plotkin's
Sequentiality

D. Barbarossa and G. Manzonetto. Taylor Subsumes Scott, Berry, Kahn and Plotkin. PACMPL Vol. 4, pp. 1:1-1:23, 2020.

Classic results with simpler inductive proofs



D. Barbarossa and G. Manzonetto. Taylor Subsumes Scott, Berry, Kahn and Plotkin. PACMPL Vol. 4, pp. 1:1-1:23, 2020.

Contextuality of $=_{\mathcal{B}}$ via Taylor Expansion

$$\mathrm{BT}(N) = \mathrm{BT}(N') \quad \Rightarrow \quad \forall M \,.\, \mathrm{BT}(MN) = \mathrm{BT}(MN')$$

Proof. More precisely:

$$\mathrm{BT}(N) \sqsubseteq \mathrm{BT}(N') \quad \Rightarrow \quad \forall M \,.\, \mathrm{BT}(MN) \sqsubseteq \mathrm{BT}(MN')$$

Equivalently, by Corollary 2: let

$$\mathrm{NF}(\mathcal{T}(N)) \subseteq \mathrm{NF}(\mathcal{T}(N'))$$

we have to prove:

$$\mathrm{NF}(\mathcal{T}(MN)) \subseteq \mathrm{NF}(\mathcal{T}(MN'))$$

A proof of context closure via Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(N)) \subseteq \mathrm{NF}(\mathcal{T}(N')) \quad \Rightarrow \quad \mathrm{NF}(\mathcal{T}(MN)) \subseteq \mathrm{NF}(\mathcal{T}(MN'))$$

Proof. Take $t \in \mathrm{NF}(\mathcal{T}(MN))$, then $\exists t' \in \mathcal{T}(MN)$ such that

$$t' = \qquad s_1[u_1, \dots, u_k] \twoheadrightarrow t + \mathbb{T}$$

with $s_1 \in \mathcal{T}(M)$
and $u_1, \dots, u_k \in \mathcal{T}(N)$.

A proof of context closure via Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(N)) \subseteq \mathrm{NF}(\mathcal{T}(N')) \quad \Rightarrow \quad \mathrm{NF}(\mathcal{T}(MN)) \subseteq \mathrm{NF}(\mathcal{T}(MN'))$$

Proof. Take $t \in \mathrm{NF}(\mathcal{T}(MN))$, then $\exists t' \in \mathcal{T}(MN)$ such that

$$t' = \qquad s_1[u_1, \ldots, u_k] \longrightarrow\!\!\!\!\!\rightarrow t + \mathbb{T}$$

with $\mathrm{nf}(s_1) \in \mathrm{NF}(\mathcal{T}(M))$
and $\mathrm{nf}(u_1), \ldots, \mathrm{nf}(u_k) \in \mathrm{NF}(\mathcal{T}(N))$

A proof of context closure via Taylor Expansion

$$\mathrm{NF}(\mathcal{T}(N)) \subseteq \mathrm{NF}(\mathcal{T}(N')) \quad \Rightarrow \quad \mathrm{NF}(\mathcal{T}(MN)) \subseteq \mathrm{NF}(\mathcal{T}(MN'))$$

Proof. Take $t \in \mathrm{NF}(\mathcal{T}(MN))$, then $\exists t' \in \mathcal{T}(MN)$ such that

$$t' = \begin{tikzcd} s_1[u_1, \ldots, u_k] \arrow[r, ->>] \arrow[d] & t + \mathbb{T} \\ \mathrm{nf}(s_1)[\mathrm{nf}(u_1), \ldots, \mathrm{nf}(u_k)] \arrow[ur, ->>] & \end{tikzcd}$$

with $\mathrm{nf}(s_1) \in \mathrm{NF}(\mathcal{T}(M))$
and $\mathrm{nf}(u_1), \ldots, \mathrm{nf}(u_k) \in \mathrm{NF}(\mathcal{T}(N)) = \mathrm{NF}(\mathcal{T}(N'))$.
We conclude that $t \in \mathrm{NF}(\mathcal{T}(MN'))$. $\square$

Some Taylor approximants are “just like” Böhm’s

A resource term t is called

- linearized if every bag in t has cardinality 1.
- affined if every bag in t has cardinality at most 1.

Every affined normal $t \in \Lambda^r$ can be sent to an approximant $|t| \in \mathcal{A}$:

$$
\begin{array}{lcl}
|x| & = & x, \\
|\lambda x.t| & = & \lambda x.|t|, \\
|s[t]| & = & |s|\,|t|, \\
|s[]| & = & |s|\,\bot.
\end{array}
$$

Some Taylor approximants are “just like” Böhm's

A resource term t is called

- linearized if every bag in t has cardinality 1.
- affined if every bag in t has cardinality at most 1.

Every approximant $A \neq \bot$, can be sent to an affined $A^\circ \in \Lambda^r$:

$$\begin{array}{rcl} x^\circ & = & x, \\ (\lambda x.A)^\circ & = & \lambda x.A^\circ, \\ (A_1 A_2)^\circ & = & A_1^\circ[A_2^\circ], \\ (A\bot)^\circ & = & A^\circ[]. \end{array}$$

Properties

- For all $A \in \mathcal{A} - \{\perp\}$ and $t \in \Lambda^r$, we have:

$$|P^\circ| = P$$

  and

$$|t|^\circ = t.$$

- For all M there exists a unique linearized t such that

$$t \in \mathrm{NF}(\mathcal{T}(M)) \iff M \text{ is } \beta\text{-normalizable.}$$

  In this case, we have $\mathrm{nf}_\beta(M) = |t|$.

The Genericity Lemma

Let M unsolvable. $C[M]$ has a β-nf $\quad\Rightarrow\quad \forall N \,.\, C[M] =_\beta C[N]$.

Standard proof: Topological method.

Compactification points in the tree topology are precisely the unsolvables.

Several proofs in the literature:

Masako Takahashi: A Simple Proof of the Genericity Lemma. Logic, Language and Computation 1994: 117-118

Jan Kuper: Proving the Genericity Lemma by Leftmost Reduction is Simple. RTA 1995: 271-278

The Genericity Lemma

Let M unsolvable. $C[M]$ has a β-nf $\quad\Rightarrow\quad \forall N\,.\,C[M] =_\beta C[N]$.

Proof. As $C[M]$ normalizable, $\exists ! t \in \mathrm{NF}(\mathcal{T}(C[M]))$ linearized s.t.

$$|t| = \mathrm{nf}(C[M])$$

So, there exist $t' \in \mathcal{T}(C[M])$ such that:

$$t' = \qquad c\llparenthesis s_1, \ldots, s_k \rrparenthesis \twoheadrightarrow t + \mathbb{T}$$

for some $c\llparenthesis - \rrparenthesis \in \mathcal{T}(C[-])$ and $s_1, \ldots, s_k \in \mathcal{T}(M)$.

The Genericity Lemma

Let M unsolvable. $C[M]$ has a β-nf $\quad \Rightarrow \quad \forall N \,.\, C[M] =_\beta C[N]$.

Proof. As $C[M]$ normalizable, $\exists ! t \in \mathrm{NF}(\mathcal{T}(C[M]))$ linearized s.t.

$$|t| = \mathrm{nf}(C[M])$$

So, there exist $t' \in \mathcal{T}(C[M])$ such that:

$$\begin{tikzcd}
t' = & c\llparenthesis s_1, \ldots, s_k \rrparenthesis \arrow[r, ->>] \arrow[d, ->>] & t + \mathbb{T} \\
& c\llparenthesis \mathrm{nf}(s_1), \ldots, \mathrm{nf}(s_k) \rrparenthesis \arrow[ur, ->>] &
\end{tikzcd}$$

for some $c\llparenthesis - \rrparenthesis \in \mathcal{T}(C[-])$ and $s_1, \ldots, s_k \in \mathcal{T}(M)$. (By Confluence and Strong Normalization.)

The Genericity Lemma

Let M unsolvable. $C[M]$ has a β-nf $\quad \Rightarrow \quad \forall N \,.\, C[M] =_\beta C[N]$.

Proof. As $C[M]$ normalizable, $\exists ! t \in \mathrm{NF}(\mathcal{T}(C[M]))$ linearized s.t.

$$|t| = \mathrm{nf}(C[M])$$

So, there exist $t' \in \mathcal{T}(C[M])$ such that:

$$\begin{tikzcd} t' = & c\llparenthesis s_1, \dots, s_k \rrparenthesis \arrow[r, twoheadrightarrow] \arrow[d, twoheadrightarrow] & t + \mathbb{T} \\ & c\llparenthesis 0, \dots, 0 \rrparenthesis \arrow[ur, twoheadrightarrow] & \end{tikzcd}$$

for some $c\llparenthesis - \rrparenthesis \in \mathcal{T}(C[-])$ and $s_1, \dots, s_k \in \mathcal{T}(M)$. Now, M unsolvable entails $\mathrm{nf}(s_i) = 0$.

The Genericity Lemma

Let M unsolvable. $C[M]$ has a β-nf $\quad\Rightarrow\quad \forall N\,.\, C[M] =_\beta C[N]$.

Proof. As $C[M]$ normalizable, $\exists! t \in \mathrm{NF}(\mathcal{T}(C[M]))$ linearized s.t.

$$|t| = \mathrm{nf}(C[M])$$

So, there exist $t' \in \mathcal{T}(C[M])$ such that:

$$t' = \begin{tikzcd} c(\!| s_1, \ldots, s_k |\!) \arrow[rr, two heads] \arrow[d] & & t + \mathbb{T} \\ c(\!| 0, \ldots, 0 |\!) \arrow[urr, two heads] & & \end{tikzcd}$$

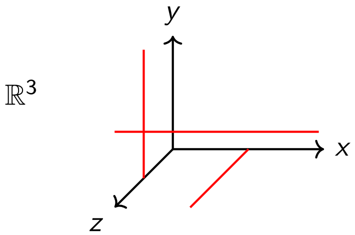
for some $c(\!|-|\!) \in \mathcal{T}(C[-])$ and $s_1, \ldots, s_k \in \mathcal{T}(M)$. Now, M unsolvable entails $\mathrm{nf}(s_i) = 0$. Thus, $(\!|-|\!)$ cannot occur in $c(\!|-|\!)$ so we get:

$$c(\!| s_1, \ldots, s_k |\!) \in \mathcal{T}(C[N]) \quad\Rightarrow\quad t \in \mathrm{NF}(\mathcal{T}(C[N]))$$

and since t is linearized we obtain $\mathrm{nf}_\beta(C[N]) = |t|$. $\square$

Perpendicular Lines Lemma

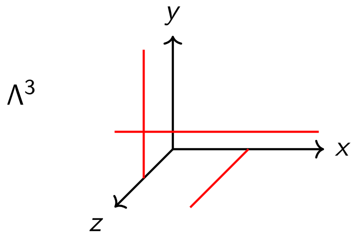
PLL: If a context $C[-_1, \ldots, -_n] : \Lambda^n \to \Lambda$ is constant on n perpendicular lines, then it must be constant everywhere.



$$\ell_1 = \{(x, 1, 2) \mid x \in \mathbb{R}\},$$
$$\ell_2 = \{(0, y, 1) \mid y \in \mathbb{R}\},$$
$$\ell_3 = \{(1, 0, z) \mid z \in \mathbb{R}\}.$$

Perpendicular Lines Lemma

PLL: If a context $C[-_1, \dots, -_n] : \Lambda^n \to \Lambda$ is constant on n perpendicular lines, then it must be constant everywhere.



$$\ell_1 = \{(X, \lambda x.x, \lambda xy.x) \mid X \in \Lambda\},$$
$$\ell_2 = \{(\lambda xy.y, Y, \lambda x.x) \mid Y \in \Lambda\},$$
$$\ell_3 = \{(\lambda x.x, \lambda xy.x, Z) \mid Z \in \Lambda\}.$$

Known results

Perpendicular Lines Lemma	β	$\mathcal{B}$
open term model	✓	✓
closed term model	✗	?

- $\mathcal{M}(\mathcal{B}) \models$ PLL, Barendregt's Book 1982,
  Proof technique: Sequentiality.
- $\mathcal{M}^o(\mathcal{B}) \models$ PLL?
- $\mathcal{M}^o(\beta) \not\models$ PLL, by Barendregt & Statman 1999.
  Proof: Counterexample via Plotkin's terms.
- $\mathcal{M}(\beta) \models$ PLL, by De Vrijer & Endrullis 2008.
  Proof: via Reduction under Substitution.

Perpendicular Lines Lemma

$$\forall Z \left\{ \begin{array}{lcl} C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\ C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\ \qquad\ddots & \vdots & \vdots \\ C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n \end{array} \right.$$

$$\Downarrow$$

$$\forall \vec{Z} \,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

Perpendicular Lines Lemma

$$\forall Z \left\{ \begin{array}{lcl} C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\ C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\ \qquad\quad \ddots & \vdots & \vdots \\ C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n \end{array} \right.$$

$$\Downarrow$$

$$\forall \vec{Z}\,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

In $\mathcal{B}$ a context $C[-]$ can be constant for several reasons:

1. C does not contain the hole in the first place (the trivial case);
2. the hole is erased during its reduction;
3. the hole is “hidden” behind an unsolvable;
4. the hole is never erased but “pushed into infinity”.

Perpendicular Lines Lemma

$$\forall Z \left\{ \begin{array}{lcl} C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\ C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\ \qquad\quad \ddots & \vdots & \vdots \\ C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n \end{array} \right.$$

$$\Downarrow$$

$$\forall \vec{Z} \,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

An approximant $c \in \mathcal{T}(C[-])$ such that $\mathrm{nf}(c) \neq 0$ can be constant for only one reason:

1. c does not contain the hole in the first place (the trivial case);
2. the hole is erased during its reduction ;
3. the hole is “hidden” behind an unsolvable;
4. the hole is never erased but “pushed into infinity”.

Perpendicular Lines Lemma

$$\forall Z \left\{ \begin{array}{lcl} C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\ C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\ \ddots & \vdots & \vdots \\ C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n \end{array} \right.$$

$$\Downarrow$$

$$\forall \vec{Z} \,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

An approximant $c \in \mathcal{T}(C[-])$ such that $\mathrm{nf}(c) \neq 0$ can be constant for only one reason:

1. c does not contain the hole in the first place (the trivial case);
2. ~~the hole is erased during its reduction~~ (linearity);
3. the hole is “hidden” behind an unsolvable;
4. the hole is never erased but “pushed into infinity”.

Perpendicular Lines Lemma

$$\forall Z \begin{cases} C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\ C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\ \qquad \ddots & \vdots & \vdots \\ C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n \end{cases}$$

$$\Downarrow$$

$$\forall \vec{Z} \,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

An approximant $c \in \mathcal{T}(C[-])$ such that $\mathrm{nf}(c) \neq 0$ can be constant for only one reason:

1. c does not contain the hole in the first place (the trivial case);
2. ~~the hole is erased during its reduction~~ (linearity);
3. ~~the hole is "hidden" behind an unsolvable~~ (SN);
4. the hole is never erased but "pushed into infinity".

Perpendicular Lines Lemma

$$\forall Z \left\{ \begin{array}{lcl} C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\ C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\ \qquad\ddots & \vdots & \vdots \\ C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n \end{array} \right.$$

$$\Downarrow$$

$$\forall \vec{Z} \,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

An approximant $c \in \mathcal{T}(C[-])$ such that $\mathrm{nf}(c) \neq 0$ can be constant for only one reason:

1. c does not contain the hole in the first place (the trivial case);
2. ~~the hole is erased during its reduction~~ (linearity);
3. ~~the hole is “hidden” behind an unsolvable~~ (SN);
4. ~~the hole is never erased but “pushed into infinity”~~ (finiteness).

Perpendicular Lines Lemma

$$
\forall Z \left\{ \begin{array}{lcl}
C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\
C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\
\qquad\quad \ddots & \vdots & \vdots \\
C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n
\end{array} \right.
$$

$$\Downarrow$$

$$\forall \vec{Z}\,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

Claim.

$\forall c \in \mathcal{T}(C[-_1, \ldots, -_n])$, $\mathrm{nf}(c) \neq 0 \Rightarrow c$ cannot contain any hole.

By induction on the size of c, using all the properties above.

Perpendicular Lines Lemma

$$\forall Z \begin{cases} C[Z, M_{12}, \ldots\ldots, M_{1n}] & =_{\mathcal{B}} & N_1 \\ C[M_{21}, Z, \ldots\ldots, M_{2n}] & =_{\mathcal{B}} & N_2 \\ \quad\ddots & \vdots & \vdots \\ C[M_{n1}, \ldots, M_{n(n-1)}, Z] & =_{\mathcal{B}} & N_n \end{cases}$$

$$\Downarrow$$

$$\forall \vec{Z} \,.\, C[Z_1, \ldots, Z_n] =_{\mathcal{B}} N_1 =_{\mathcal{B}} \cdots =_{\mathcal{B}} N_n$$

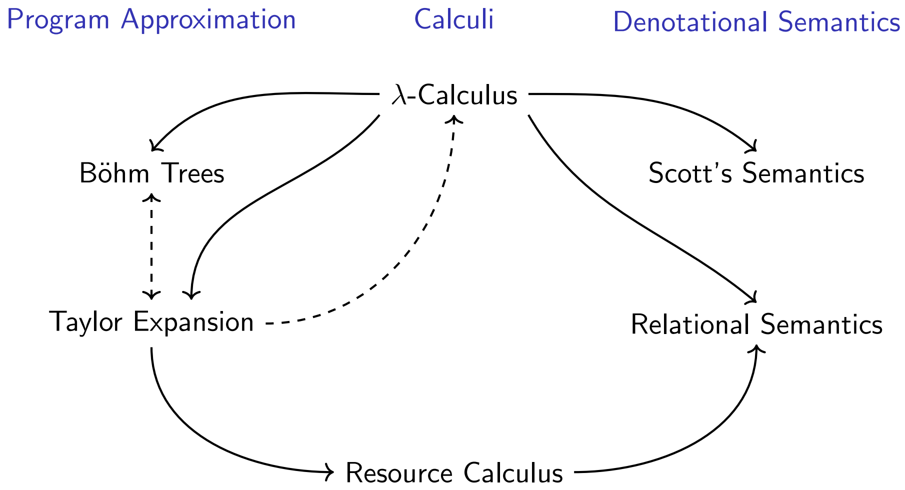
Our proof does not need open terms!

$\mathcal{M}^o(\mathcal{B}) \models \mathrm{PLL}$ ✓

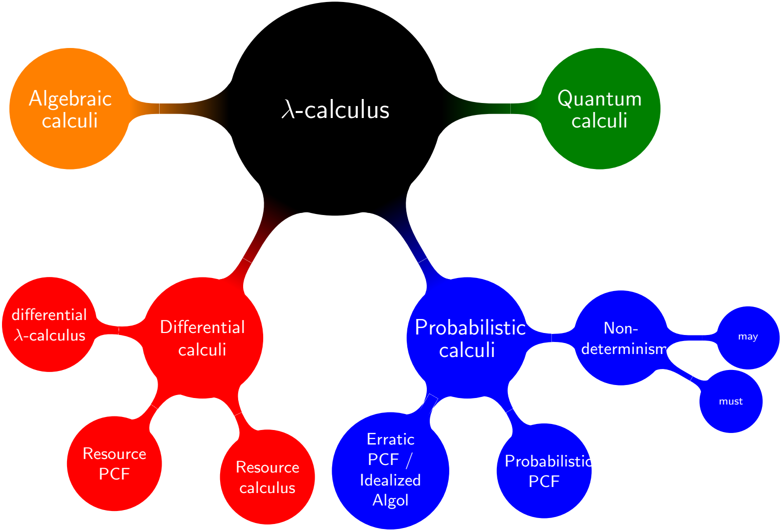
This completes the picture!

Perpendicular Lines Lemma	β	$\mathcal{B}$
open term model	✓	✓
closed term model	✗	✓

The Big Picture



Advantage: These techniques scale to many languages



Extensionality

- Extensional BT's.
  Nakajima, Hyland, Wadsworth, Lévy.
- Degrees of extensionality in BT.
  Intrigila, Manzonetto and Polonsky.
- Extensional Taylor Expansion.
  Blondeau-Patissier, Clairambault, Vaux Auclair.

Meaningfulness

- Berarducci Trees
  Berarducci.
- CbV solvability
  Accattoli, Guerrieri.
- Magnificent Böhm approximant
  Arrial, Kesner, Guerrieri.

Call-by-value

- CbV Böhm Tree.
  Kerinec, Manzonetto, Pagani.
  Accattoli, Lancelot, Faggian.
- CbV Taylor Expansion.
  Ehrhard, Guerrieri.

Approximations for Λ^∞

- Infinitary Linear Logic.
  Baelde, Doumane, Kuperberg, Saurin.
  Ehrhard.
- Taylor Expansion for Λ^∞.
  Cerda, Vaux Auclair.


ANY
QUESTIONS
?