Taylor expansion for Call-By-Push-Value

Jules Chouquet    Christine Tasson

\textit{IF}, Université de Paris

CSL, January 2020
1 Introduction

2 Quantitative semantics

3 Quantitative syntax

4 Taylor expansion

5 Call-By-Push-Value

6 Conclusions
Programs as transformations

$P : A \rightarrow B$ transforms elements of $A$ into elements of $B$.

$A$ and $B$ can be ordinary types, or more complex objects as power series, probability distributions, . . .

But $P$ is still a transformation, $P$ does something.
A program $P : A \rightarrow B$, as every natural process, needs energy to run. It consumes the elements of $A$. 
Resource consumption

A program \( P : A \rightarrow B \), as every natural process, needs energy to run. It consumes the elements of \( A \).

\[
\begin{array}{cccc}
\lambda xx & \lambda x(xxxxxx) & \lambda xy \\
\end{array}
\]
A program $P : A \to B$, as every natural process, needs energy to run. It consumes the elements of $A$.

| $\lambda xx$ | $\lambda x(xxxxx)$ | $\lambda xy$ |
A program $P : A \rightarrow B$, as every natural process, needs energy to run. It *consumes* the elements of $A$.

<table>
<thead>
<tr>
<th>$\lambda xx$</th>
<th>$\lambda x(xxxxxx)$</th>
<th>$\lambda xy$</th>
</tr>
</thead>
</table>

[Images of a scooter and a monster truck]
**Resource consumption**

A program $P : A \rightarrow B$, as every natural process, needs energy to run. It *consumes* the elements of $A$.

<table>
<thead>
<tr>
<th>$\lambda xx$</th>
<th>$\lambda x(\ldots)$</th>
<th>$\lambda xy$</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Scooter" /></td>
<td><img src="image2.png" alt="Monster Truck" /></td>
<td><img src="image3.png" alt="Skateboard" /></td>
</tr>
</tbody>
</table>
A program $P : A \rightarrow B$, as every natural process, needs energy to run. It *consumes* the elements of $A$.

<table>
<thead>
<tr>
<th>$\lambda x x$</th>
<th>$\lambda x(\ldots)$</th>
<th>$\lambda x y$</th>
</tr>
</thead>
</table>

Quantitative semantics

Interpret precisely duplication and erasing in the semantics.
1 Introduction

2 Quantitative semantics

3 Quantitative syntax

4 Taylor expansion

5 Call-By-Push-Value

6 Conclusions
Denotational semantics of $\Lambda$

<table>
<thead>
<tr>
<th>Calculus</th>
<th>$M, N ::= x \mid \lambda xM \mid MN$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operations</td>
<td>$(\lambda xM)N \rightarrow_{\beta} M[N/x]$</td>
</tr>
<tr>
<td>Interpretation</td>
<td>$⟦M⟧<em>M$ (something in $M$ invariant under $\rightarrow</em>{\beta}$)</td>
</tr>
</tbody>
</table>

For $M : A \rightarrow B$, $⟦M⟧$ can be e.g. a function/relation from $⟦A⟧$ to $⟦B⟧$. 
Quantitative approach
Think about resources

- Status of $[M]$ in $(\lambda x(xx))M$ ? wrt $(\lambda xx)M$ ?
- Interpret probabilistic reduction ?
- ...

Girard (Normal functors, 1988)

Uses of arguments $\rightsquigarrow$ degree of a monomial in a power series.

Types: $[[A]] \subseteq S^{|A|}$ where $S$ is a semiring
Programs : power series
Quantitative Semantics

Example: multirelations

S Types Programs Invariance
⇝ ⇝ ⇝ ⇝

Boolean semiring

$[[A \rightarrow B]] = \mathcal{M}_{\text{fin}}(|A|) \times |B|$

$P : A \rightarrow B \Rightarrow [[P]] \subseteq \mathcal{M}_{\text{fin}}(|A|) \times |B|$

Composition of multirelations.

Key idea

let $M : A \rightarrow B$, $N : A$.

$([a_1, \ldots, a_k], b) \in [[M]]$ will match with $k$ uses of the argument $N$ in the application $(MN)$. 
Quantitative Semantics
Models

- Probabilistic coherence spaces (Danos, Ehrhard). (Fully abstract for probabilistic PCF (Ehrhard, Pagani, Tasson 2015))
- Weighted relational model (Laird, McCusker, Manzonetto, Pagani)
- Finiteness spaces (Ehrhard)
- Convenient vector spaces (Blute, Ehrhard, Tasson) (Kerjean).
Quantitative syntax

Extract a calculus from the multiset theoretic-semantics in order to provide a step-by-step analysis of duplication.
1 Introduction

2 Quantitative semantics

3 Quantitative syntax

4 Taylor expansion

5 Call-By-Push-Value

6 Conclusions
Resource calculus
A quantitative syntax

\[ m, n ::= x \mid \lambda x m \mid \langle m \rangle[n_1, \ldots, n_k] \text{ (k-linear application)} \]

\[ \langle \lambda x m \rangle[n_1, \ldots, n_k] \rightarrow_{\partial} \sum_{\sigma \in \mathcal{S}_n} m \left[ n_{\sigma(1)}/x_1, \ldots, n_{\sigma(k)}/x_k \right] \]

\[ \langle \lambda x \langle x \rangle \rangle[x, x]\rangle[z] \rightarrow_{\partial} 0 \quad \partial \leftarrow \langle \lambda x x \rangle[z, z, z] \]

<table>
<thead>
<tr>
<th>\lambda\text{-calculus}</th>
<th>resource calculus</th>
</tr>
</thead>
<tbody>
<tr>
<td>( MN ) ( \leadsto ) ( \langle m \rangle[n_1, \ldots, n_k] )</td>
<td></td>
</tr>
</tbody>
</table>
| \( (\lambda x(x)x)z \) \( \leadsto \) \( \langle \lambda x \langle x \rangle[x]\rangle[z, z] \)
| \( \downarrow \beta \) \( \downarrow \partial \) |
| \( zz \) \( \leadsto \) \( \langle z \rangle[z] + \langle z \rangle[z] \) |
Multilinear Approximations

We define the approximation relation $m \triangleleft M$:

- $x \triangleleft x$
- $\lambda x n \triangleleft \lambda x N$ if $n \triangleleft N$.
- $\langle m \rangle[n_1, \ldots, n_k] \triangleleft MN$ for all $k \in \mathbb{N}$ if $m \triangleleft M$ and $n_i \triangleleft N$.

(Resource terms can be seen as polynomials that approximate power series.)
We define the approximation relation $m \triangleleft M$ :

- $x \triangleleft x$
- $\lambda x n \triangleleft \lambda x N$ if $n \triangleleft N$.
- $\langle m \rangle [n_1, \ldots, n_k] \triangleleft MN$ for all $k \in \mathbb{N}$ if $m \triangleleft M$ and $n_i \triangleleft N$.

(Resource terms can be seen as polynomials that approximate power series.)

**Property:** simulation of $\rightarrow_\beta$ with approximants

If $m \triangleleft M$ and $M \rightarrow M'$, $m \rightarrow 0$ or $\exists m'$ s.t. $m \Rightarrow_\emptyset m'$ and $m' \triangleleft M'$. 
Parallel reduction

**Definition**

We extend \( \rightarrow_{\partial} \) to a parallel reduction \( \equiv_{\partial} \).

**Example**

- \( MN \rightarrow MN' \)
- Let \( \langle m \rangle[n_1, \ldots, n_k] \triangleleft MN \).
- \( \langle m \rangle[n_1, \ldots, n_k] \equiv_{\partial} \langle m \rangle[n'_1, \ldots, n'_k] \triangleleft MN' \) if \( n_i \equiv_{\partial} n'_i \) for all \( i \).

In the resource setting, Taylor expansion consists in taking infinite sums of resource terms.

The idea is that taken together, the combination of all \( m \triangleleft M \), behave exactly as \( M \).
1 Introduction

2 Quantitative semantics

3 Quantitative syntax

4 Taylor expansion

5 Call-By-Push-Value

6 Conclusions
Taylor expansion - Combining approximants
A bridge between syntax and semantics

Semantic approach: Interpret a term/function as an infinite series of approximants.

Syntactic Taylor expansion:

\[ T(MN) = \sum_{k \in \mathbb{N}} \frac{1}{k!} \langle T(M) \rangle [T(N), \ldots, T(N)]_k \]

\[ T(\lambda xM) = \lambda xT(M) \quad T(x) = x. \]

Remark

\( T(M) \) is a weighted sum of all resource nets \( m \) s.t. \( m \triangleleft M \)
Simulation
A convergence problem

**Wanted result: correction**

Extend $\Rightarrow_{\partial}$ to infinite sums of terms ($\Rightarrow_{\partial}$), in order to have $M \rightarrow_{\beta} N \Rightarrow T(M) \Rightarrow_{\partial} T(N)$. 

Problem
Can $\Rightarrow_{\partial}$ be always well-defined?

No $\sum_{k \in N} \langle \lambda_{xx} \rangle \left[ \langle \lambda_{xx} \rangle \ldots \right] \Rightarrow_{\partial} \infty \cdot y$

If $S$ is not a complete semiring, the reduction is not defined for any series.
**Simulation**

A convergence problem

---

**Wanted result: correction**

Extend $\Rightarrow_\partial$ to infinite sums of terms ($\Rightarrow_\partial$), in order to have $M \rightarrow_\beta N \Rightarrow T(M) \Rightarrow_\partial T(N)$.

---

**Problem**

Can $\Rightarrow_\partial$ be always well-defined?
Simulation
A convergence problem

Wanted result: correction

Extend $\Rightarrow_{\partial}$ to infinite sums of terms $(\Rightarrow_{\partial})$, in order to have $M \rightarrow_{\beta} N \Rightarrow T(M) \Rightarrow_{\partial} T(N)$.

Problem

Can $\Rightarrow_{\partial}$ be always well-defined?

No

$$\sum_{k \in \mathbb{N}} \langle \lambda x x \rangle [\langle \lambda x x \rangle \ldots [y]] \ldots ] \Rightarrow_{\partial} \infty \cdot y$$

If $S$ is not a complete semiring, the reduction is not defined for any series.
Some correction results

→₀ is well defined and simulates →β in Taylor expansion:

- Linear Logic proof nets: Chouquet, Vaux-Auclair – CSL 2018
- Call-By-Value, PCF, Call-By-Need: Chouquet – CSL MFPS 2019
Some correction results

⇒_∂ is well defined and simulates →_β in Taylor expansion:

- Linear Logic proof nets: Chouquet, Vaux-Auclair – CSL 2018
- Call-By-Value, PCF, Call-By-Need: Chouquet – CSL MFPS 2019
- Call-By-Push-Value: Chouquet, Tasson – CSL 2020
1 Introduction

2 Quantitative semantics

3 Quantitative syntax

4 Taylor expansion

5 Call-By-Push-Value

6 Conclusions
Call-By-Push-Value
A Linear Logic inspired-presentation (Ehrhard)

\[ M, N ::= \begin{align*}
  x & \mid \lambda x M & \mid \langle M \rangle N & \mid (M, N) & \mid \pi_i(M) & \mid \nu_i(M) \\
  \text{case}(M, y \cdot N_1, z \cdot N_2) & \mid M! & \mid \text{der}(M) & \mid \text{fix}_x(M)
\end{align*} \]

\[ V, U ::= \begin{align*}
  x & \mid \lambda x M & \mid M! & \mid (V, U) & \mid \nu_i(V)
\end{align*} \]

\[ A, B ::= \begin{align*}
  I & \mid A \otimes B & \mid A \oplus B & \mid \text{(positives)} \\
  I, J ::= A & \mid A \twoheadrightarrow B & \mid \top & \mid \text{(general)}
\end{align*} \]

- Subsumes Call-By-Name and Call-By-Value at the operational and denotational level
- Results in quantitative semantics (Ehrhard-Tasson)
- Coinductive datatypes
\[
\Gamma, x : A \vdash x : A \\
\Gamma \vdash M : A \rightarrow I \\
\Delta \vdash N : A \\
\Gamma \vdash M : A \\
\Delta \vdash N : B \\
\Gamma, \Delta \vdash (M, N) : A \otimes B \\
\Gamma \vdash \pi_i(M) : A_i \quad i \in \{1, 2\} \\
\Gamma \vdash \nu_i(M) : A_i \quad i \in \{1, 2\} \\
\Gamma \vdash m : !A \\
\Gamma \vdash \text{der}(m) : A \\
\Gamma, x : !I \vdash M : I \\
\Gamma \vdash \text{fix}_x(M) : I \\
\Gamma \vdash M : I \\
\Gamma \vdash M' : !I \\
\Gamma \vdash M_1 : A \oplus B \\
\Delta \vdash M_2 : I \\
\Theta \vdash M_3 : I \\
\Gamma, \Delta, \Theta \vdash \text{case}(M_1, y \cdot M_2, z \cdot M_3) : I \\
\langle \lambda xM \rangle V \rightarrow_{pv} M[V/x] \\
\pi_i(V_1, V_2) \rightarrow_{pv} V_i \\
\text{case}(\nu_i(V), x_1 \cdot M_1, x_2 \cdot M_2) \rightarrow_{pv} M_i[V/x_i] \\
\text{der}(M') \rightarrow_{pv} M \\
\text{fix}_x(M) \rightarrow_{pv} M[(\text{fix}_x(M))^! / x] \
\]
## Duplication : Exponentials VS coalgebras morphisms

<table>
<thead>
<tr>
<th></th>
<th>$\Lambda$</th>
<th>$\Lambda_{pv}$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Application</strong></td>
<td>$(\lambda x M) N$</td>
<td>$\lambda x M (V_1, V_2)$</td>
</tr>
<tr>
<td><strong>Approximation</strong></td>
<td>$\langle \lambda x m \rangle [n_1, \ldots, n_k]$</td>
<td>$\langle \lambda x m \rangle (v_1, v_k)$</td>
</tr>
<tr>
<td><strong>Interpretation</strong></td>
<td>$N : ! A$</td>
<td>$(V_1, V_1) : P_1 \otimes P_2$</td>
</tr>
</tbody>
</table>
### Duplication: Exponentials VS coalgebras morphisms

<table>
<thead>
<tr>
<th></th>
<th>$\Lambda$</th>
<th>$\Lambda_{pv}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>$(\lambda x M) N$</td>
<td>$\lambda x M (V_1, V_2)$</td>
</tr>
<tr>
<td>Approximation</td>
<td>$\langle \lambda x m \rangle [n_1, \ldots, n_k]$</td>
<td>$\langle \lambda x m \rangle (v_1, v_k)$</td>
</tr>
<tr>
<td>Interpretation</td>
<td>$N : ! A$</td>
<td>$(V_1, V_1) : P_1 \otimes P_2$</td>
</tr>
</tbody>
</table>

\[
P_1 \otimes P_2 \xrightarrow{h} (P_1 \otimes P_2) \otimes \ldots \ldots \otimes (P_1 \otimes P_2)
\]

$h$ is a morphism coming from the coalgebra structure in the interpretation of positive types (the duplicable ones).

If duplication exists in the semantics, what about syntactic Taylor expansion?
Resource reduction with splitting

Splitting operator « \texttt{split} »

\[
\text{split}(([m, m], [m, m])) = ([m], [m]), ([m], [m])
\]

In general: \(\text{split}^k(v) = (v_1, \ldots, v_k)\) where the \(v_i\) have the same syntactic tree than \(v\).

\[
\begin{align*}
&\text{(m, (m', m''))} &\xrightarrow{\text{split}^k} &\text{((m_1, (m'_1, m''_1)), \ldots, (m_k, (m'_k, m''_k)))}
\end{align*}
\]

Call-By-Push-Value resource reduction

\[
\langle \lambda x \text{m} \rangle v \rightarrow \sum_{(v_1, \ldots, v_k) \in \text{split}^k(v)} m[v_1/x_1, \ldots, v_k/x_k]
\]
Then we have a resource calculus for Call-By-Push-Value and we can define Taylor expansion.

**Theorem**

For any Call-By-Push-Value term $M$, if $M \rightarrow N$, then $T(M) \Rightarrow T(N)$
Taylor expansion is a bridge between syntax and semantics

Its definition and consistence \(w.r.t\) the models may be tricky and depends on the calculus

Following the semantics, we can build a convenient resource calculus for Call-By-Push-Value giving a syntactic account to coalgebras morphisms, and prove the correction of Taylor expansion

Some perspectives:

- Link these results to Linear Logic proof nets
- Extend Taylor expansion’s correction to non uniform settings of Call-By-Push-Value
- Go to lunch