Reversing Taylor expansion

Christine TASSON

Laboratoire Preuves Programmes Systèmes - Université Paris Diderot
Common work with Michele Pagani

March 16, 2009
Table of contents

1. From linear logic to differential linear logic
   - Linear Logic
   - Differential linear logic
   - Taylor expansion

2. Reversing Taylor expansion
   - Problems
   - Solutions
Multiplicative Exponential Linear Logic

Formulæ

\[ A, B : = 1 | \bot | A^\perp | A \otimes B | A \bowtie B | !A | ?A \]

- Duality: \( A^{\perp \perp} = A \), \( (A \bowtie B)^\perp = A^\perp \otimes B^\perp \), \( !A^\perp = ?A^\perp \).
- Linear implication: \( A \rightarrow B = A^\perp \bowtie B \).
- Intuitionist implication: \( A \Rightarrow B = !A \rightarrow B \).

Proof system

- Sequent calculus
- Nets (geometrical presentation)
An example
Linear logic nets

Units

\[ \ast = \bot / 1 \]

Functions

\[ A \otimes B \]

\[ A \otimes B \]

Resources

Boxes around subprograms

The most complex rule:

- Can be erased or duplicated during evaluation.
- No symmetry with resources.
Differential nets

Units

\( * = \perp / 1 \)

Functions

\( A \otimes B \)

\( A \otimes B \)

Resources

\( ?A \)

\( ?A \)

\( ?A \)

Boxes

\( !A \)

\( ?B_1 \)

\( \cdots \)

\( ?B_k \)

\( ?B_1 \)

\( \cdots \)

\( ?B_k \)
Differential linear logic

Formulæ

\[ A, B : = 1 \mid \bot \mid A\perp \mid A \otimes B \mid A \boxtimes B \mid !A \mid ?A \]

Some differences?

- Boxes are replaced by three new cells.
- Symmetry between \( ! \) and \(?\) modalities.
- Erasing and duplication not on will but explicit.
Differential nets

Units

\[ * = \perp/1 \]

Functions

\[ A \otimes B \]

\[ A \bowtie B \]

Resources

\[ ?A \]

\[ A \]

\[ !A \]

Co-resources

\[ !A \]

\[ !A \]

\[ !A \]
Taylor expansion origins

Linear logic seminal idea [Girard87]
Decompose program with respect to their resource handling:

\[ \text{Prog}(x) = \sum_{n} \text{Prog}_n(x, \ldots, x). \]

Semantic realization [Ehrhard05]
Taylor expansion in the finiteness spaces semantics.

Lambda-calculus implementation [Ehrhard-Regnier05]

- \( \lambda \)-terms are decomposed into sums of resource \( \lambda \)-terms.
- Uniform relation between resource \( \lambda \)-terms in the sum.
- Evaluation in \( \lambda \)-calculus is simulated by evaluation in resource \( \lambda \)-calculus.
Taylor expansion

Recursive principle

Each linear logic box is mapped to a sum of differential nets.

\[ N \rightsquigarrow \sum_{n} \frac{1}{n!} \]

where \( N_k^* \)s appear in the Taylor expansion of \( N \).
Taylor expansion, an example
Taylor expansion, an example

Take 2 copies.
Taylor expansion, an example

\[ \alpha_1 \] \quad \alpha_2

\[ \alpha_i \in \mathcal{T} \]
Taylor expansion, an example

Take 0/1 copy.
Taylor expansion, an example

\[ \alpha \in T \]
Taylor expansion, an example
Taylor expansion, an example

A linear logic net and a differential net in its Taylor expansion
Table of contents

1 From linear logic to differential linear logic
   Linear Logic
   Differential linear logic
   Taylor expansion

2 Reversing Taylor expansion
   Problems
   Solutions
How to get back boxes?

Example

Notice that in lambda-calculus, we don’t have this problem.
How to get back boxes?

Example

Notice that in lambda-calculus, we don’t have this problem.
Definition
A collection \((\alpha_i)_{i \in J}\) of differential nets is uniform iff there is a linear logic net \(\pi\) such that \(\forall i \in J, \alpha_i \in T(\pi)\).

Hypercoherence
The uniformity relation is not a binary relation:

\[
\begin{align*}
\alpha_1 := & \quad \begin{array}{cccc}
1 & 1 & \perp & \\
! & ! & ? & ? \\
\end{array} \\
\alpha_2 := & \quad \begin{array}{cccc}
1 & 1 & \perp & \\
! & ! & ? & ? \\
\end{array} \\
\alpha_3 := & \quad \begin{array}{cccc}
1 & 1 & \perp & \\
! & ! & ? & ? \\
\end{array}
\end{align*}
\]
Injectivity

Relational semantics injectivity [Falco03]

In the relational semantics of linear logic, types are interpreted by sets and proofs by relations.

Question: Can we always differentiate two proofs of linear logic with their relational semantics?

\[ \forall \pi \neq \pi', \exists [\cdot]_{\text{Rel}} ; [\pi]_{\text{Rel}} \neq [\pi']_{\text{Rel}} \]

Separation Roughly speaking [Mazza-Pagani07]

- \( \forall \mu \neq \mu' \) differential nets with same conclusion, there is \( \nu \) such that: \( \langle \mu, \nu \rangle \rightarrow^{*}_\beta 0 \) and \( \langle \mu', \nu \rangle \rightarrow^{*}_\beta 1 \).
- Every denotational semantics of differential nets separates every two nets with different interpretations.
- Through Taylor expansion, every semantics of differential nets generates a semantics of linear logic.
Taylor inverse

We are looking for an algorithm with:

**Input:** A finite collection of differential nets: \((\alpha_i)_{i \in J}\).

**Output:** \((\alpha_i)_{i \in J}\) is

- Either Dead-lock.
- Or uniform and a linear logic net \(\pi\) such that \(\forall i \in J, \alpha_i \in T(\pi)\).

Naive solution

Try every possible subgraphs and check wanted properties.

Locality

We wish to rebuild boxes locally.
Labelling
Labelling

\[ A \leq B \leq C \]

\[ A \leq B \leq C \]

\[ B \leq A \leq C \]

\[ C \leq B \leq A \]
Merging on an example

\[ \alpha_1 \]

\[ \alpha_2 \]
Merging on an example
Merging on an example

\[ \alpha_1 \]

\[ \alpha_2 \]

\[ \beta \]

\[ \gamma \]

\[ \delta \]

\[ \epsilon \]

\[ \zeta \]

\[ \eta \]

\[ \theta \]

\[ \iota \]

\[ \kappa \]

\[ \lambda \]

\[ \mu \]

\[ \nu \]

\[ \xi \]

\[ \omicron \]

\[ \pi \]

\[ \rho \]

\[ \sigma \]

\[ \upsilon \]

\[ \phi \]

\[ \chi \]

\[ \psi \]

\[ \omega \]

\[ \alpha \]

\[ \beta \]

\[ \gamma \]

\[ \delta \]

\[ \epsilon \]

\[ \zeta \]

\[ \eta \]

\[ \theta \]

\[ \iota \]

\[ \kappa \]

\[ \lambda \]

\[ \mu \]

\[ \nu \]

\[ \xi \]

\[ \omicron \]

\[ \pi \]

\[ \rho \]

\[ \sigma \]

\[ \upsilon \]

\[ \phi \]

\[ \chi \]

\[ \psi \]

\[ \omega \]

\[ \alpha \]

\[ \beta \]

\[ \gamma \]

\[ \delta \]

\[ \epsilon \]

\[ \zeta \]

\[ \eta \]

\[ \theta \]

\[ \iota \]

\[ \kappa \]

\[ \lambda \]

\[ \mu \]

\[ \nu \]

\[ \xi \]

\[ \omicron \]

\[ \pi \]

\[ \rho \]

\[ \sigma \]

\[ \upsilon \]

\[ \phi \]

\[ \chi \]

\[ \psi \]

\[ \omega \]
Merging on an example
Merging on an example
Merging on an example
Merging on an example
Merging on an example
Merging on an example
Merging on an example

\begin{align*}
&\mathbf{A}, \{\mathbf{A}, \mathbf{B}\}, \{\mathbf{A}, \mathbf{B}, \mathbf{C}\} \\
&\begin{array}{c}
1_{1,1}^{1,1} \\
1_{2,1}^{1,1} \\
1_{2,2}^{2,2}
\end{array}
\end{align*}
Merging on an example

\[ \{A, B, C\} \]
Merging on an example

\[ \alpha_1 \]

\[ \alpha_2 \]

\[ A \leq B \leq C \]
**Algorithm**

**Principle**
Counters go through nets, merge compatible cells, and label with respect to boxes order.

**Properties**
- If succeeds give a witness.
- Not completely local because of the alphabet.
- Efficiency: more efficient than the naive try.
- Order: describe boxes order and Taylor expansion order.
Conclusion

Summary
We tackled the inverse Taylor expansion problem in the setting of nets and described an algorithm taking benefits from both boxes order and Taylor order.

And after
- Injectivity?
- Reduction, cut elimination.
- Coefficients and simulation.

Questions?
Bibliographie