Towards a Curry-Howard Correspondence for Quantum Computation

TLLA 2021

Kostia Chardonnet[1,2] Alexis Saurin[2] Benoît Valiron[1]

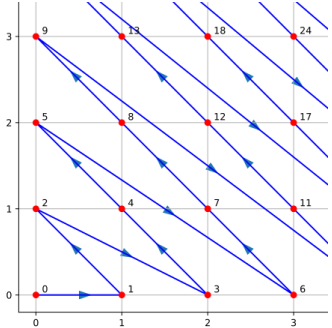
[1]LMF [2]IRIF

Origins

- Landauer and Bennet, 1961 : Reversible Computation and Energy Dissipation
- Functions f are reversible: $\exists f^{-1}$, st $f \circ f^{-1} = id = f^{-1} \circ f$
- Multiple approach : Turing Machines, imperative / functional programming languages, ...

Example

$\mathbb{N} \times \mathbb{N} \to \mathbb{N}$



Reversible Programming Languages

Programming languages:

- RFun : Untyped language
- Theseus : Typed, Linear language, based on Pattern-Matching

Uses

- Reversible Computation
- Quantum Computation

Our Case

Work by [Sabry, Valiron, Vizzotto]:

- Subset of Theseus
- Linear, Reversible language with Lists
- Require terminating recursion
- Extension to the quantum case
- No general recursive type
- No expressivity theorem
- No relation with logic

} What we are trying to extend

Previous Work - Logic

- Logic $\mu\mathsf{MALL}^{\infty}$: Linear Logic with Least and Greatest Fixpoint
- Allow inductive and co-inductive statement
- We only look at the inductive fragment.
- Formulas $A, B ::= 1 \mid A \otimes B \mid A \oplus B \mid \mu X.A \mid X$

Examples:

- $nat = \mu X.\mathbb{1} \oplus X$
- $lists(A) = [A] = \mu X.\ \mathbb{1} \oplus (A \otimes X)$
- BinaryTree$(A) = \mu X.1 \oplus (A \otimes X \otimes X)$

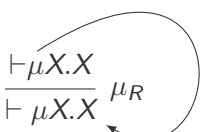
Linear Logic with Induction ($\mu X.A$)

$$\frac{A \vdash B[X \leftarrow \mu X.B]}{A \vdash \mu X.B}\ \mu_R \qquad \frac{A[X \leftarrow \mu X.A] \vdash B}{\mu X.A \vdash B}\ \mu_L$$

Linear Logic with Induction ($\mu X.A$)

$$\frac{A \vdash B[X \leftarrow \mu X.B]}{A \vdash \mu X.B}\ \mu_R \qquad \frac{A[X \leftarrow \mu X.A] \vdash B}{\mu X.A \vdash B}\ \mu_L$$

$$\cfrac{\cfrac{\vdots}{\vdash \mu X.X}\ \mu_R}{\vdash \mu X.X}\ \mu_R \qquad \cfrac{\vdash \mu X.X}{\vdash \mu X.X}\ \mu_R$$



Infinite derivations represented as graphs

Beware : can derive any formula F

$$\dfrac{\dfrac{\dfrac{\vdots}{\mu X.X \vdash \mathtt{F}}\,\mu_L \qquad \dfrac{\vdots}{\vdash \mu X.X}\,\mu_R}{\vdash \mathtt{F}}\,\mathtt{cut}}$$

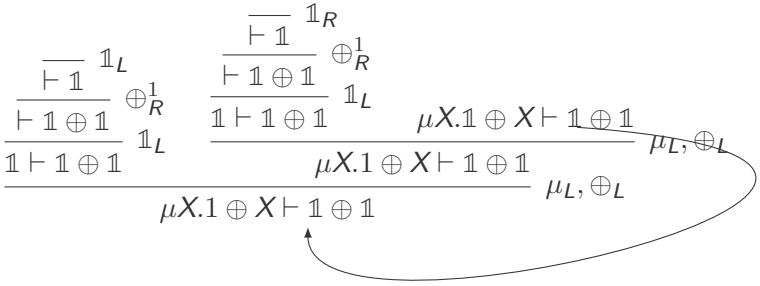
Beware : can derive any formula F

$$\dfrac{\dfrac{\vdots}{\mu X.X \vdash \mathtt{F}}\,\mu_L \qquad \dfrac{\vdots}{\vdash \mu X.X}\,\mu_R}{\vdash \mathtt{F}}\,\mathtt{cut}$$

A derivation is valid if in every infinite branch:

- Infinity of rules μ_L

Example : Parity function



Our Work

Based on [Sabry, Valiron, Vizzotto] and [Baelde, Doumane, Saurin]

	Sabry et al.	Baelde et al.	This Work
Linear	✓	✓	✓
Reversible	✓	✗	✓
Induction	✗	✓	✓
Curry-Howard	✗	✗	WIP

Syntax

(Base types) $A, B ::= \mathbb{1} \mid X \mid A \oplus B \mid A \otimes B \mid \mu X.A$

(Isos, first-order) $T ::= A \leftrightarrow B$

Syntax

(Base types) $A, B ::= \mathbb{1} \mid X \mid A \oplus B \mid A \otimes B \mid \mu X.A$

(Isos, first-order) $T ::= A \leftrightarrow B$

(Isos) $\omega ::= \{e_1 \leftrightarrow e_1' \mid \ldots \mid e_n \leftrightarrow e_n'\} \mid \mu f.\omega \mid f \mid \mathtt{inv}\ \omega$

$$\mu f. \left\{ \begin{array}{lll} [\,] & \leftrightarrow & [\,] \\ h :: t & \leftrightarrow & \mathtt{let}\ x = \omega\ h\ \mathtt{in} \\ & & \mathtt{let}\ y = f\ t\ \mathtt{in}\ x :: y \end{array} \right\} : [A] \leftrightarrow [B]$$

Syntax

(Base types) $A, B ::= \mathbb{1} \mid X \mid A \oplus B \mid A \otimes B \mid \mu X.A$

(Isos, first-order) $T ::= A \leftrightarrow B$

(Isos) $\omega ::= \{e_1 \leftrightarrow e_1' \mid \ldots \mid e_n \leftrightarrow e_n'\} \mid \mu f.\omega \mid f \mid \mathtt{inv}\ \omega$

$$\mu f. \left\{ \begin{array}{lll} [\,] & \leftrightarrow & [\,] \\ h :: t & \leftrightarrow & \mathtt{let}\ x = \omega\ h\ \mathtt{in} \\ & & \mathtt{let}\ y = f\ t\ \mathtt{in}\ x :: y \end{array} \right\} : [A] \leftrightarrow [B]$$

Properties

Syntax

- Language comes with a rewriting system and a type system.
  $\Rightarrow$ Confluence, Subject Reduction
- Ensuring exhaustivity and non-overlapping of clauses.
- Ensuring productivity.
  $\Rightarrow$ Similar to Coq, recursive argument have to be smaller.
- Can encode any Primitive Recursive Function

Semantic

- Isos denote computations from $A \to B$ and $B \to A$.

Syntax - Example 2

$$\mathtt{map}(\omega) = \mu \mathtt{f}. \left\{ \begin{array}{lcl} [\,] & \leftrightarrow & [\,] \\ h :: t & \leftrightarrow & \mathtt{let}\, x = \omega\ h\ \mathtt{in} \\ & & \mathtt{let}\, y = f\ t\ \mathtt{in}\ x :: y \end{array} \right\} : [\mathtt{A}] \leftrightarrow [\mathtt{B}]$$

Syntax - Example 2

$$\mathtt{map}(\omega) = \mu \mathtt{f}. \left\{ \begin{array}{lcl} [\,] & \leftrightarrow & [\,] \\ h :: t & \leftrightarrow & \mathtt{let}\, x = \omega\ h\ \mathtt{in} \\ & & \mathtt{let}\, y = f\ t\ \mathtt{in}\ x :: y \end{array} \right\} : [\mathtt{A}] \leftrightarrow [\mathtt{B}]$$

$$\mathtt{map}(\omega)^{\perp} = \mu \mathtt{f}. \left\{ \begin{array}{lcl} [\,] & \leftrightarrow & [\,] \\ \mathtt{let}\, x = \omega\ h\ \mathtt{in} & \leftrightarrow & h :: t \\ \mathtt{let}\, y = f\ t\ \mathtt{in}\ x :: y & & \end{array} \right\} : [\mathtt{B}] \leftrightarrow [\mathtt{A}]$$

Syntax - Example 2

$$\texttt{map}(\omega) = \mu \texttt{f}. \left\{ \begin{array}{lcl} [\,] & \leftrightarrow & [\,] \\ h :: t & \leftrightarrow & \texttt{let}\, x = \omega\ h\ \texttt{in} \\ & & \texttt{let}\, y = f\ t\ \texttt{in}\ x :: y \end{array} \right\} : [\texttt{A}] \leftrightarrow [\texttt{B}]$$

$$\texttt{map}(\omega)^{\perp} = \mu \texttt{f}. \left\{ \begin{array}{lcl} [\,] & & \leftrightarrow\ [\,] \\ \texttt{let}\, x = \omega\ h\ \texttt{in} & & \leftrightarrow\ h :: t \\ \texttt{let}\, y = f\ t\ \texttt{in}\ x :: y & & \end{array} \right\} : [\texttt{B}] \leftrightarrow [\texttt{A}]$$

$$\texttt{map}(\omega)^{\perp} = \mu \texttt{f}. \left\{ \begin{array}{lcl} [\,] & \leftrightarrow & [\,] \\ x :: y & \leftrightarrow & \texttt{let}\, h = (\texttt{inv}\ (\omega))\ x\ \texttt{in} \\ & & \texttt{let}\, t = f\ y\ \texttt{in}\ h :: t \end{array} \right\} : [\texttt{B}] \leftrightarrow [\texttt{A}]$$

From Type Derivation To Proofs

Typed Terms $\rightsquigarrow$ Proofs

$$\omega : A \leftrightarrow B \rightsquigarrow \pi : A \vdash B$$

$$\omega^{\perp} : B \leftrightarrow A \rightsquigarrow \pi^{\perp} : B \vdash A$$

From Derivations To Proofs - Example

Let us take the recursive identity on lists

$$\mu f. \left\{ \begin{array}{ll} [\,] & \leftrightarrow [\,] \\ h :: t & \leftrightarrow \texttt{let}\ t' = f\, t\ \texttt{in} \\ & \quad\ h :: t' \end{array} \right\} : [A] \leftrightarrow [A]$$

From Derivations To Proofs - Example

$$\dfrac{\dfrac{\vdash [A]}{\mathbb{1} \vdash [A]}\,\mathbb{1}_L \qquad \dfrac{\dfrac{A, [A] \vdash [A]}{A \otimes [A] \vdash [A]}\,\otimes_L}{\mathbb{1} \oplus (A \otimes [A]) \vdash [A]}\,\oplus_L}{[A] \vdash [A]}\,\mu_L$$

$$\mu f. \left\{ \begin{array}{rl} [\,] & \leftrightarrow [\,] \\ h :: t & \leftrightarrow \texttt{let}\ t' = f\, t\ \texttt{in} \\ & \quad\ h :: t' \end{array} \right\}$$

From Derivations To Proofs - Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathbb{1}}\,\mathbb{1}_R}{\vdash \mathbb{1} \oplus (A \otimes [A])}\,\oplus_R^1}{\vdash [A]}\,\mu_R \qquad \dfrac{\dfrac{}{\mathbb{1} \vdash [A]}\,\mathbb{1}_L \qquad \dfrac{A, [A] \vdash [A]}{A \otimes [A] \vdash [A]}\,\otimes_L}{\mathbb{1} \oplus (A \otimes [A]) \vdash [A]}\,\oplus_L}{[A] \vdash [A]}\,\mu_L$$

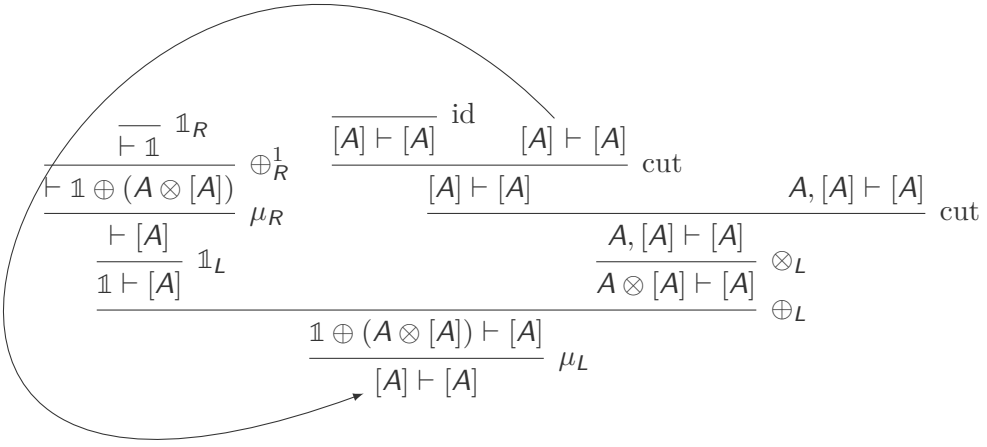
$$\mu f.\left\{\begin{array}{lll} [\,] & \leftrightarrow [\,] \\ h :: t & \leftrightarrow \texttt{let}\ t' = f\,t\ \texttt{in} \\ & \qquad h :: t' \end{array}\right\}$$

From Derivations To Proofs - Example

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\vdash \mathbb{1}}\,\mathbb{1}_R}{\vdash \mathbb{1} \oplus (A \otimes [A])}\,\oplus_R^1}{\vdash [A]}\,\mu_R \qquad \dfrac{\dfrac{[A] \vdash [A] \qquad A, [A] \vdash [A]}{A, [A] \vdash [A]}\,\otimes_L}{A \otimes [A] \vdash [A]}\,\oplus_L\ \text{cut}}{\mathbb{1} \oplus (A \otimes [A]) \vdash [A]}\,\mathbb{1}_L}{[A] \vdash [A]}\,\mu_L$$

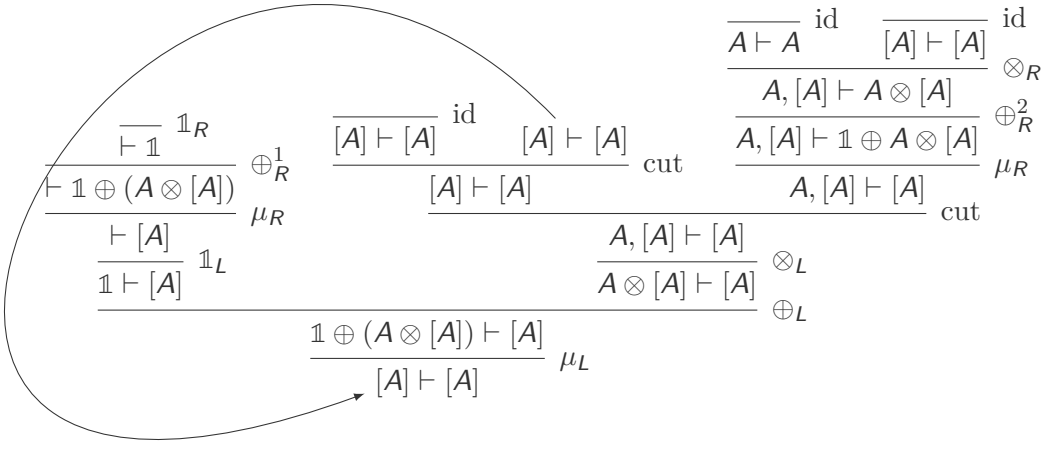
$$\mu f.\left\{\begin{array}{lll} [\,] & \leftrightarrow & [\,] \\ h :: t & \leftrightarrow & \texttt{let}\ t' = f t\ \texttt{in} \\ & & h :: t' \end{array}\right\}$$

From Derivations To Proofs - Example



$$\mu f.\left\{\begin{array}{ll} [\,] & \leftrightarrow [\,] \\ h :: t & \leftrightarrow \texttt{let } t' = f\,t \texttt{ in} \\ & \quad h :: t' \end{array}\right\}$$

From Derivations To Proofs - Example



$$\mu f. \left\{ \begin{array}{ll} [\,] & \leftrightarrow [\,] \\ h :: t & \leftrightarrow \texttt{let}\ t' = f\, t\ \texttt{in} \\ & \quad h :: t' \end{array} \right\}$$

Conclusion

Results

- Linear and Reversible language
- Curry-Howard Correspondence

In Progress

- Finish some proofs
- Consider the quantum case

Future Work

- Consider co-induction
- Categorical Semantic
- Term language for ZX-Calculus
- What makes a proof reversible