

# Profinite $\lambda$ -terms and parametricity

Vincent Moreau, joint work with Sam van Gool and Paul-André Melliès

---

LHC days 2023

June the 7<sup>th</sup>, 2023

IRIF, Université Paris Cité, Inria Paris

## Who am I?

PhD student since September 2021. This is joint work with my two advisors.



Paul-André Mellès



Sam van Gool



at IRIF, Paris

## Context of the talk

Regular languages have a central place in theoretical computer science. Profinite methods are well established for words using finite monoids.

Salvati proposed a notion of regular language of  $\lambda$ -terms using semantic tools.

Contribution: definition of profinite  $\lambda$ -terms using the CCC **FinSet** such that

**profinite words are in bijection with profinite  $\lambda$ -terms**

and living in harmony with Stone duality and the principles of Reynolds parametricity.



# Languages

---

## Regular languages of words

Let  $\Sigma$  be a finite alphabet,  $M$  be a finite monoid and  $p : \Sigma \rightarrow M$  a set-theoretic function. We write  $\bar{p}$  for the associated monoid homomorphism  $\Sigma^* \rightarrow M$ .

For each subset  $F \subseteq M$ , the set

$$L_F := \{w \in \Sigma^* \mid \bar{p}(w) \in F\}$$

is a regular language. These sets assemble into the Boolean algebra

$$\text{Reg}_M\langle\Sigma\rangle := \{L_F : F \subseteq M\}.$$

When  $M$  ranges over all finite monoids, we get in this way all regular languages:

$$\text{Reg}\langle\Sigma\rangle = \bigcup_M \text{Reg}_M\langle\Sigma\rangle.$$

## The Church encoding for words

Any natural number  $n$  can be encoded in the simply typed  $\lambda$ -calculus as

$$s : \circ \Rightarrow \circ, z : \circ \vdash \underbrace{s(\dots(s z))}_{n \text{ applications}} : \circ .$$

A natural number is just a word over a one-letter alphabet.

For example, the word *abba* over the two-letter alphabet  $\{a, b\}$

$$a : \circ \Rightarrow \circ, b : \circ \Rightarrow \circ, c : \circ \vdash a(b(b(a c))) : \circ .$$

is encoded as the closed  $\lambda$ -term

$$\lambda a. \lambda b. \lambda c. a(b(b(a c))) : \underbrace{(\circ \Rightarrow \circ)}_{\text{letter } a} \Rightarrow \underbrace{(\circ \Rightarrow \circ)}_{\text{letter } b} \Rightarrow \underbrace{\circ}_{\text{input}} \Rightarrow \underbrace{\circ}_{\text{output}} .$$

For any alphabet  $\Sigma$ , we define  $\text{Church}_\Sigma$  as  $\underbrace{(\circ \Rightarrow \circ) \Rightarrow \dots \Rightarrow (\circ \Rightarrow \circ)}_{|\Sigma| \text{ times}} \Rightarrow \circ \Rightarrow \circ .$

## Categorical interpretation

Let  $\mathbf{C}$  be a cartesian closed category and  $Q$  be one of its objects.

For any simple type  $A$  built from  $\circ$ , we define the object  $\llbracket A \rrbracket_Q$  by induction as

$$\llbracket \circ \rrbracket_Q := Q \quad \text{and} \quad \llbracket A \Rightarrow B \rrbracket_Q := \llbracket A \rrbracket_Q \Rightarrow \llbracket B \rrbracket_Q .$$

Using the cartesian closed structure, one defines an interpretation function

$$\llbracket - \rrbracket_Q : \Lambda_{\beta\eta}\langle A \rangle \longrightarrow \mathbf{C}(1, \llbracket A \rrbracket_Q) .$$

In **FinSet** which is cartesian closed, given a finite set  $Q$  used to interpret  $\circ$ , every word  $w$  over the alphabet  $\Sigma = \{a, b\}$ , seen as a  $\lambda$ -term, is interpreted as a point

$$\llbracket w \rrbracket_Q \in (Q \Rightarrow Q) \Rightarrow (Q \Rightarrow Q) \Rightarrow Q \Rightarrow Q$$

which describes how the word will interact with a deterministic automaton.

## Regular languages of $\lambda$ -terms

The notion of regular language of  $\lambda$ -terms has been introduced by Salvati.

For any finite set  $Q$  and any subset  $F \subseteq \llbracket A \rrbracket_Q$ , we define the language

$$L_F := \{ M \in \Lambda_{\beta\eta}\langle A \rangle \mid \llbracket M \rrbracket_Q \in F \} .$$

All the languages recognized by  $Q$  assemble into a Boolean algebra

$$\text{Reg}_Q\langle A \rangle := \{ L_F \mid F \subseteq \llbracket A \rrbracket_Q \} .$$

We can then make  $Q$  range over all finite sets, and we get the definition

$$\text{Reg}\langle A \rangle := \bigcup_Q \text{Reg}_Q\langle A \rangle .$$

Notice that  $\text{Reg}\langle A \rangle$  has no reason to be a Boolean algebra for the moment.



## Salvati generalizes Kleene

The Church encoding induces an isomorphism of Boolean algebras

$$\text{Reg}\langle \text{Church}_\Sigma \rangle \cong \text{Reg}\langle \Sigma \rangle .$$

Indeed, every automaton  $(Q, \delta, q_0, \text{Acc})$  induces a subset

$$F := \{q \in \llbracket \text{Church}_\Sigma \rrbracket_Q \mid q(\delta, q_0) \in \text{Acc}\}$$

On the other hand, every  $q \in \llbracket \text{Church}_\Sigma \rrbracket_Q$  induces a finite family of automata

$$(Q, \delta, q_0, \{q(\delta, q_0)\}) \quad \text{for all } \delta : \Sigma \times Q \rightarrow Q \text{ and } q_0 \in Q$$

which determines the behavior of  $q$ , and from which one gets finite monoids.

## A first observation using logical relations

If  $Q$  and  $Q'$  are two finite sets and  $R \subseteq Q \times Q'$ , for any simple type  $A$  we have

$$\llbracket A \rrbracket_R \subseteq \llbracket A \rrbracket_Q \times \llbracket A \rrbracket_{Q'}$$

In particular, if  $f : Q \twoheadrightarrow Q'$  is a partial surjection, then so is  $\llbracket A \rrbracket_f : \llbracket A \rrbracket_Q \twoheadrightarrow \llbracket A \rrbracket_{Q'}$ .

Using the fundamental lemma of logical relations, one can deduce that

$$\text{if } |Q| \geq |Q'|, \quad \text{then } \text{Reg}_{Q'}\langle A \rangle \subseteq \text{Reg}_Q\langle A \rangle.$$

This shows that the diagram

$$\left( \text{Reg}_{Q'}\langle A \rangle \hookrightarrow \text{Reg}_Q\langle A \rangle \right)_{f:Q \twoheadrightarrow Q'}$$

is directed so we have

$$\text{Reg}\langle A \rangle = \text{colim}_Q \text{Reg}_Q\langle A \rangle.$$



# Entering the profinite world

---

## The monoid of profinite words

A **profinite word**  $u$  is a family  $(u_p)$  of elements

$$u_p \in M \quad \text{where} \quad \begin{array}{l} M \text{ ranges over all finite monoids} \\ p : \Sigma \rightarrow M \text{ ranges over all functions} \end{array}$$

such that for every function  $p : \Sigma \rightarrow M$  and homomorphism  $\varphi : M \rightarrow N$ , with  $M$  and  $N$  finite monoids, we have  $u_{\varphi \circ p} = \varphi(u_p)$ .

The monoid  $\widehat{\Sigma}^*$  of profinite words contains  $\Sigma^*$  as a submonoid, since any word  $w = w_1 \dots w_n$ , where each  $w_i \in \Sigma$ , induces a profinite word with components

$$p(w_1) \dots p(w_n) \quad \text{for all } p : \Sigma \rightarrow M.$$

## A profinite word which is not a word

For any finite monoid  $M$  there exists  $n(M) \geq 1$  such that for all elements  $m$  of  $M$ , the element  $m^{n(M)}$  is the idempotent power of  $m$ , which is unique.

Let  $a$  be any letter in  $\Sigma$ . The family of elements

$$u_p := p(a)^{n(M)} \quad \text{for all } p : \Sigma \rightarrow M$$

is an idempotent profinite word written  $a^\omega$  which is not a finite word.

There is a more general construction: if  $u$  is a profinite word, then one can build another profinite word  $u^\omega$  which is idempotent.

## Duality: words

Stone spaces, i.e. compact and totally separated spaces, and continuous maps form a category **Stone**. Boolean algebras and their homomorphisms form a category **BA**.

There is an equivalence of categories

$$\mathbf{Stone} \cong \mathbf{BA}^{\text{op}}$$

which associates to every Stone space its algebra of clopens and to every Boolean algebra its space of ultrafilters.

In particular, the monoid of profinite words  $\widehat{\Sigma}^*$  has a natural topology such that

$$\widehat{\Sigma}^* \text{ is the Stone dual of } \text{Reg}\langle \Sigma \rangle .$$

## Duality: $\lambda$ -terms

For any simple type  $A$  and finite set  $Q$ , we consider the subset

$$[[A]]_Q^\bullet := \{[[M]]_Q \mid M \in \Lambda_{\beta\eta}\langle A \rangle\} \subseteq [[A]]_Q$$

of definable elements of  $[[A]]_Q$ . Equivalently, it is the quotient

$$\Lambda_{\beta\eta}\langle A \rangle / \approx_Q \quad \text{with } M \approx_Q N \text{ if and only if } [[M]]_Q = [[N]]_Q .$$

The finite set of definable elements is related to regular languages as

$$[[A]]_Q^\bullet \quad \text{is the Stone dual of} \quad \text{Reg}_Q\langle A \rangle$$

and the inclusion  $\text{Reg}_{Q'}\langle A \rangle \hookrightarrow \text{Reg}_Q\langle A \rangle$  induced by a partial surjection  $f : Q \twoheadrightarrow Q'$  dualizes to the surjection  $[[A]]_f^\bullet : [[A]]_Q^\bullet \rightarrow [[A]]_{Q'}^\bullet$ , which is the restriction of  $[[A]]_f$ .

## Definition of profinite $\lambda$ -terms

By dualizing the diagram defining  $\text{Reg}\langle A \rangle$ , we obtain a codirected diagram

$$\left( [[A]_f^\bullet : [[A]_Q^\bullet \longrightarrow [[A]_{Q'}^\bullet \right)_{f:Q \twoheadrightarrow Q'}$$

and we define  $\widehat{\Lambda}_{\beta\eta}\langle A \rangle$  as its limit. As expected,

$$\widehat{\Lambda}_{\beta\eta}\langle A \rangle \quad \text{is the Stone dual of} \quad \text{Reg}\langle A \rangle .$$

Concretely: a **profinite  $\lambda$ -term**  $\theta$  of type  $A$  is a family of elements  $\theta_Q \in [[A]_Q^\bullet$  s.t.

$$[[A]_f^\bullet(\theta_Q) = \theta_{Q'} \quad \text{for every partial surjection } f : Q \twoheadrightarrow Q'.$$



## The CCC of profinite $\lambda$ -terms

**Theorem.** The profinite  $\lambda$ -terms assemble into a CCC **ProLam** such that

$$\mathbf{ProLam}(A, B) \quad := \quad \widehat{\Lambda}_{\beta\eta}\langle A \Rightarrow B \rangle .$$

This means that we have a compositional notion of profinite  $\lambda$ -calculus.

The interpretation of the simply typed  $\lambda$ -calculus into **ProLam** yields a functor

$$\mathbf{Lam} \longrightarrow \mathbf{ProLam}$$

which sends a simply typed  $\lambda$ -term  $M$  of type  $A$  on the profinite  $\lambda$ -term

$$\llbracket M \rrbracket_Q \quad \text{where } Q \text{ ranges over all finite sets.}$$

This assignment is injective thanks to Statman's finite completeness theorem.

## Profinite $\lambda$ -terms of Church type are profinite words

The Church encoding gives a bijection

$$\Lambda_{\beta\eta}\langle\text{Church}_{\Sigma}\rangle \cong \Sigma^* .$$

This extends to the profinite setting. Indeed, profinite  $\lambda$ -terms of simple type  $\text{Church}_{\Sigma}$  are exactly profinite words as we have a homeomorphism

$$\widehat{\Lambda}_{\beta\eta}\langle\text{Church}_{\Sigma}\rangle \cong \widehat{\Sigma}^* .$$

## The profinite $\lambda$ -term $\Omega$

We consider the profinite  $\lambda$ -term  $\Omega$  of type  $(\circ \Rightarrow \circ) \Rightarrow \circ \Rightarrow \circ$  such that

$$\Omega_Q \quad : \quad f \longmapsto \underbrace{f \circ \dots \circ f}_{n \text{ times}}$$

where  $f^n$  is the idempotent power of the element  $f$  of the finite monoid  $Q \Rightarrow Q$ .


Using  $\Omega$ , for any  $\Sigma$  of cardinal  $n$ , one gets the profinite  $\lambda$ -term

$$\lambda u \lambda a_1 \dots \lambda a_n . \Omega(u a_1 \dots a_n) \quad : \quad \text{Church}_\Sigma \Rightarrow \text{Church}_\Sigma$$

which is the representation in the profinite  $\lambda$ -calculus of the operator

$$(-)^\omega \quad : \quad \widehat{\Sigma}^* \longrightarrow \widehat{\Sigma}^*$$

on profinite words.



# Profinite $\lambda$ -terms and Reynolds parametricity

---

## Parametric families

Let  $A$  be a simple type. A **parametric family**  $\theta$  is a family of elements  $\theta_Q \in \llbracket A \rrbracket_Q$  s.t.

$$(\theta_Q, \theta_{Q'}) \in \llbracket A \rrbracket_R \quad \text{for all relations } R \subseteq Q \times Q'.$$

Two differences with profinite  $\lambda$ -terms:

- the element  $\theta_Q$  is not asked to be definable...
- ...but the family is parametric with respect to all relations.

## A theorem and its partial converse

We first have a general theorem at every type.

**Theorem.** Every profinite  $\lambda$ -term is a parametric family.

This theorem admits the following converse at Church types.

**Theorem.** Every parametric family of type  $\text{Church}_\Sigma$  is a profinite  $\lambda$ -term.

The proof of the converse uses the Yoneda terms, which generalize the constructors

$$\lambda s \lambda z. z : \text{Nat} \quad \text{and} \quad \lambda n \lambda s \lambda z. s(n s z) : \text{Nat} \Rightarrow \text{Nat}$$

of the simple type  $\text{Nat} := \text{Church}_1$  to any Church type.

## Conclusion

Future work:

- study the situation in other locally finite CCCs;
- generalize the notion of Yoneda term to any simple type;
- investigate a generalization of logic on words, which uses monadic second-order logic (MSO), to a logic on  $\lambda$ -terms.

## Conclusion

Future work:

- study the situation in other locally finite CCCs;
- generalize the notion of Yoneda term to any simple type;
- investigate a generalization of logic on words, which uses monadic second-order logic (MSO), to a logic on  $\lambda$ -terms.

Thank you for your attention!

Any questions?



## Bibliography

- [Geh16] Mai Gehrke. **“Stone duality, topological algebra, and recognition”**. In: *Journal of Pure and Applied Algebra* 220.7 (2016), pp. 2711–2747. ISSN: 0022-4049. DOI: <https://doi.org/10.1016/j.jpaa.2015.12.007>.
- [Mel17] Paul-André Melliès. **“Higher-order parity automata”**. In: *Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2017, Reykjavik, Iceland, 2017*. 2017, pp. 1–12.
- [Pin] Jean-Eric Pin. **“Profinite Methods in Automata Theory”**. In: *26th International Symposium on Theoretical Aspects of Computer Science STACS 2009*. IBFI Schloss Dagstuhl. URL: <https://hal.inria.fr/inria-00359677>.
- [Sal09] Sylvain Salvati. **“Recognizability in the Simply Typed Lambda-Calculus”**. In: *16th Workshop on Logic, Language, Information and Computation*. Vol. 5514. Lecture Notes in Computer Science. Tokyo Japan: Springer, 2009, pp. 48–60.