A fibrational approach to regular languages of λ-terms

Vincent Moreau, joint work with Paul-André Melliès

LHC days 2024

Wednesday the 5th of June, 2024

IRIF, Université Paris Cité, Inria Paris

The Church encoding

The type of words over a two-letter alphabet $\{a, b\}$ is

$$\mathsf{Church}_{\{a,b\}} \quad := \quad \underbrace{(\circ \Rightarrow \circ)}_{a} \Rightarrow \underbrace{(\circ \Rightarrow \circ)}_{b} \Rightarrow \underbrace{\circ}_{\text{input}} \Rightarrow \underbrace{\circ}_{\text{output}} .$$

Any finite word can be encoded as a λ-term through the Church encoding:

$$abb \in \{a, b\}^* \quad \rightsquigarrow \quad \lambda(a : \circ \Rightarrow \circ).\lambda(b : \circ \Rightarrow \circ).\lambda(e : \circ).\, b\,(b\,(a\,e)) \ .$$

$\rightarrow$ **The simply typed λ-calculus generalizes finite words.**

Languages of λ-terms: the semantic side

If Q is a finite set, then any $M : \mathsf{Church}_{\{a,b\}}$ can be interpreted as

$$[\![M]\!]_Q \quad \in \quad (Q \Rightarrow Q) \Rightarrow (Q \Rightarrow Q) \Rightarrow Q \Rightarrow Q \;=\; [\![\mathsf{Church}_{\{a,b\}}]\!]_Q \,.$$

For all $\delta_a : Q \to Q$, $\delta_b : Q \to Q$ and $q_0 \in Q$, then

$$[\![\lambda a.\lambda b.\lambda c.\, b\,(b\,(a\,c))]\!]_Q(\delta_a, \delta_b, q_0) \quad = \quad \delta_b(\delta_b(\delta_a(q_0)))\,,$$

$\to$ **Semantics of λ-calculus in finite sets generalize the interpretation in DFAs.**

The logic of regular languages of λ-terms

is a fibration of CCCs.

– through 3 ingredients –

Quantifiers in bifibrations

The idea that quantifiers are adjoints has been developed by Lawvere:

$$\exists y.\psi \ \vdash_{\vec{x}} \ \varphi \qquad \text{if and only if} \qquad \psi \ \vdash_{\vec{x},y} \ \varphi[\vec{x}, \_]$$

Following the work by Melliès and Zeilberger on type refinement systems, we use bifibrations $p : \mathbf{E} \to \mathbf{B}$ and we write $e \sqsubset b$ to mean that $p(e) = b$.

The quantifiers are modelled as liftings of f from $\mathbf{B}$ to $\mathbf{E}$

$$\begin{array}{ccc} e & \dashrightarrow & \mathbf{push}_f(e) \\ \sqcap & & \sqcap \\ b & \xrightarrow{\ f\ } & b' \end{array} \qquad\qquad \begin{array}{ccc} \mathbf{pull}_f(e') & \dashrightarrow & e' \\ \sqcap & & \sqcap \\ b & \xrightarrow{\ f\ } & b' \end{array}$$

and adjointness is encoded in their universal properties.

Alternatively, a (cloven) bifibration on $\mathbf{B}$ can be seen as a pseudofunctor

$$\mathbf{B} \quad \longrightarrow \quad \mathbf{Adj}$$

The subset bifibration

Let **SubSet** be the category of pairs (X, S) where X is a set and $S \subseteq X$, and with morphisms $f : (X, S) \to (Y, T)$ the functions $X \to Y$ such that

$$\text{if } x \in S\ , \qquad \text{then } f(x) \in T$$

We have a functor **SubSet** $\to$ **Set** which is a bifibration, with liftings

$$\begin{array}{ccc} S & \dashrightarrow & f(S) \\ \cap\!| & & \cap\!| \\ X & \xrightarrow{f} & Y \end{array} \qquad \text{and} \qquad \begin{array}{ccc} f^{-1}(T) & \dashrightarrow & T \\ \cap\!| & & \cap\!| \\ X & \xrightarrow{f} & Y \end{array}$$

The subset CCC

The category **SubSet** is moreover a CCC, with products computed as

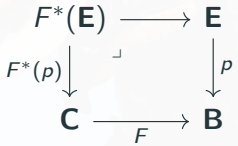
$$(X, S) \times (Y, T) \quad = \quad (X \times Y, \{(x, y) \in X \times Y \mid x \in S \text{ and } y \in T\})$$

and internal hom-objects computed as

$$(X, S) \Rightarrow (Y, T) \quad = \quad (X \Rightarrow Y, \{f : X \to Y \mid \text{for all } x \in S, \text{we have } f(x) \in T\})$$

The bifibration $p : \mathbf{SubSet} \to \mathbf{Set}$ is a CCC functor, i.e. preserves cartesian products and internal homs.

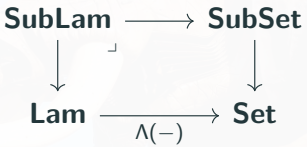
Ingredient 1 (Mitchell, Scedrov and Hasegawa)



The pullback of a CCC bifibration along a cartesian functor is still a CCC bifibration.

A CCC bifibration of all languages

Using the first ingredient, we have a CCC bifibration



More concretely, its objects are pairs

$$(A, L) \qquad \text{where } A \text{ is a simple type and } L \subseteq \Lambda(A)$$

and a morphism $(A, L) \to (B, K)$ is a λ-term $M : A \Rightarrow B$ such that

$$\text{if } N \in L\,, \qquad \text{then } M\,N \in K\,.$$

The internal hom-object is computed as

$$(A, L) \Rightarrow (B, K) \quad = \quad (A \Rightarrow B, \{M : A \Rightarrow B \mid M(L) \subseteq K\})$$

A CCC bifibration of Q-languages, for a fixed finite set Q

Let Q be a finite set. We are interested in recognizers which are definable:

$$[\![A]\!]^{\bullet}_{Q} \quad := \quad \{q \in [\![A]\!]_{Q} \mid \exists M \in \Lambda(A) \text{ s.t. } q = [\![M]\!]_{Q}\}$$

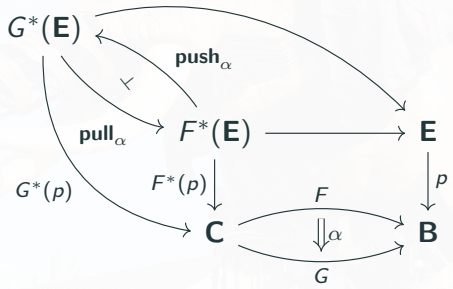
The notion of definable elements yields a functor

$$[\![-]\!]^{\bullet}_{Q} \quad : \quad \mathbf{Lam} \longrightarrow \mathbf{Set}$$

which preserves cartesian products. Therefore, it defines a CCC bifibration

$$\begin{array}{ccc} \mathbf{Reg}_{Q} & \longrightarrow & \mathbf{SubSet} \\ \downarrow & \lrcorner & \downarrow \\ \mathbf{Lam} & \xrightarrow[[\![-]\!]^{\bullet}_{Q}]{} & \mathbf{Set} \end{array}$$

Ingredient 2



A natural transformation between cartesian functors induces an adjunction between the respective pullbacks.

Relating models through partial surjections

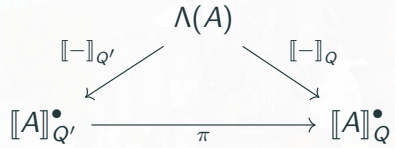
For every finite sets Q and Q' and partial surjection $f \subseteq Q' \times Q$, the logical relation

$$[\![A]\!]_f \quad \subseteq \quad [\![A]\!]_{Q'} \times [\![A]\!]_Q$$

is a partial surjection for every simple type A. All of them coincide on definable elements with a unique (total) surjection

$$\pi \quad : \quad [\![A]\!]^\bullet_{Q'} \longrightarrow [\![A]\!]^\bullet_Q$$

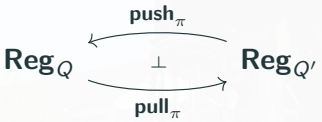
which commutes with the semantic interpretation:



Relating the bifibrations

If $|Q| \leq |Q'|$, we therefore have a natural transformation

which induces the adjunction over **Lam**

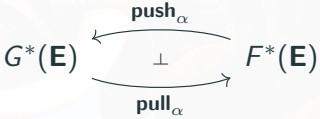


More concretely,

- the pullback $\mathbf{pull}_\pi(R) := \{q \in [\![A]\!]^\bullet_Q \mid \pi(q) \in R\}$ includes each Q-language,
- the pushforward $\mathbf{push}_\pi(S) := \{\pi(q) : q \in S\}$ saturates Q'-languages

A CCC functor?

In the adjunction



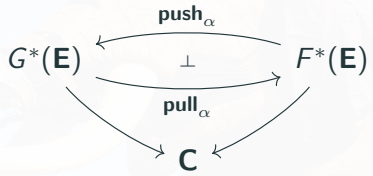
the right adjoint $\mathbf{pull}_\alpha$ preserves cartesian products and is a fibration morphism.

Is it a CCC functor? We have a canonical map

$$\mathrm{ap} \quad : \quad \mathbf{pull}_\alpha(e \Rightarrow e') \longrightarrow \mathbf{pull}_\alpha(e) \Rightarrow \mathbf{pull}_\alpha(e')$$

obtained by currying the image of the evaluation map.

Ingredient 3



The right adjoint induced by a natural transformation is a CCC functor if and only if the following Frobenius reciprocity condition is verified:

The canonical map

$$\mathbf{push}_{\mathsf{ev}}(\mathbf{push}_{\mathsf{ap}} f \times \mathbf{pull}_\alpha e) \quad \longrightarrow \quad \mathbf{push}_{\mathsf{ev}}(\mathbf{push}_{\mathsf{ap}}(\mathbf{push}_\alpha f) \times e)$$

is cocartesian above $\alpha_{c'} : F(c') \to G(c')$ for all $f \overset{F}{\sqsubset} c \Rightarrow c'$ and $e \overset{G}{\sqsubset} c$.

Bifibrational calculus, first half

For all $f \overset{F}{\sqsubset} c \Rightarrow c'$ and $e \overset{G}{\sqsubset} c$ and $e' \overset{G}{\sqsubset} c'$,

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\mathbf{push}_{\alpha_{c'}}(\mathbf{push}_{\mathsf{ev}}(\mathbf{push}_{\mathsf{ap}_F} f \times \mathbf{pull}_{\alpha_c} e)) \xrightarrow[\mathsf{Id}_{G(c')}]{} e'}{\mathbf{push}_{\mathsf{ev}}(\mathbf{push}_{\mathsf{ap}_F} f \times \mathbf{pull}_{\alpha_c} e) \xrightarrow[\mathsf{Id}_{F(c')}]{} \mathbf{pull}_{\alpha_{c'}} e'}}{\mathbf{push}_{\mathsf{ap}_F} f \times \mathbf{pull}_{\alpha_c} e \xrightarrow[\mathsf{ev}_{F(c),F(c')}]{} \mathbf{pull}_{\alpha_{c'}} e'}}{\mathbf{push}_{\mathsf{ap}_F} f \xrightarrow[\mathsf{Id}_{F(c)\Rightarrow F(c')}]{} \mathbf{pull}_{\alpha_c} e \Rightarrow \mathbf{pull}_{\alpha_{c'}} e'}}{f \xrightarrow[\mathsf{Id}_{F(c\Rightarrow c')}]{} \mathbf{pull}_{\mathsf{ap}_F}(\mathbf{pull}_{\alpha} e \Rightarrow \mathbf{pull}_{\alpha} e')}$$

Bifibrational calculus, second half

For all $f \overset{F}{\sqsubset} c \Rightarrow c'$ and $e \overset{G}{\sqsubset} c$ and $e' \overset{G}{\sqsubset} c'$,

$$\dfrac{\dfrac{\dfrac{\dfrac{\mathbf{push}_{\mathsf{ev}}(\mathbf{push}_{\mathsf{ap}_G}(\mathbf{push}_{\alpha_{c\Rightarrow c'}} f) \times e) \xrightarrow[\mathsf{Id}_{G(c')}]{} e'}{\mathbf{push}_{\mathsf{ap}_G}(\mathbf{push}_{\alpha_{c\Rightarrow c'}} f) \times e \xrightarrow[\mathsf{ev}_{G(c),G(c')}]{} e'}}{\mathbf{push}_{\mathsf{ap}_G}(\mathbf{push}_{\alpha_{c\Rightarrow c'}} f) \xrightarrow[\mathsf{Id}_{G(c)\Rightarrow G(c')}]{} e \Rightarrow e'}}{\mathbf{push}_{\alpha_{c\Rightarrow c'}} f \xrightarrow[\mathsf{Id}_{G(c\Rightarrow c')}]{} \mathbf{pull}_{\mathsf{ap}_G}(e \Rightarrow e')}}{f \xrightarrow[\mathsf{Id}_{F(c\Rightarrow c')}]{} \mathbf{pull}_{\alpha_{c\Rightarrow c'}}(\mathbf{pull}_{\mathsf{ap}_G}(e \Rightarrow e'))}$$

A CCC of regular languages

For $|Q| \leq |Q'|$, π verifies the Frobenius condition. Therefore, the inclusion functor

$$\mathbf{pull}_\pi \quad : \quad \mathbf{Reg}_Q \longrightarrow \mathbf{Reg}_{Q'}$$

is a CCC functor. As a consequence, we have a chain of CCC functors

$$\mathbf{Reg}_1 \longrightarrow \mathbf{Reg}_2 \longrightarrow \mathbf{Reg}_3 \longrightarrow \dots$$

and we let **Reg** be its colimit. More concretely, an object of **Reg** is a pair

$$(A, F) \qquad \text{where } F \subseteq [\![A]\!]_Q \text{ for some finite set } Q$$

As a filtered colimit of CCC fibrations, we get that **Reg** $\to$ **Lam** is a CCC fibration.

Residuation = CCC + fibrations

The Brzozowski derivative of a language L of words by a word u is the regular language

$$u^{-1}L \quad := \quad \{v \in \Sigma^* \mid uv \in L\}.$$

This construction generalizes to any simple type A equipped with a binary operation

$$O : A \Rightarrow A \Rightarrow A$$

and we get back the case of words when A is a Church type.

For every $M : A$ and regular language L, we have a derivative regular language

$$M^{-1}(L) \quad := \quad \{N : A \mid O\,M\,N \in L\} \quad = \quad \mathbf{pull}_O(\{M\} \Rightarrow L)$$

where $\{M\}$ is a regular language by Statman's theorem.

Towards MSO on λ-terms

Each functor $\mathbf{Reg}_Q \to \mathbf{Reg}_{Q'}$ is a fibration morphism, but not an opfibration morphism. As a consequence, $\mathbf{Reg} \to \mathbf{Lam}$ is a fibration but not an opfibration.

We consider the simple types

$$\begin{aligned} A &:= (o \Rightarrow o) \Rightarrow (o \Rightarrow o) \Rightarrow o \Rightarrow o \\ B &:= (o \Rightarrow o \Rightarrow o) \Rightarrow (o \Rightarrow o) \Rightarrow o \Rightarrow o \end{aligned}$$

whose λ-terms are ranked trees. We then have that the pushforward

$$\mathbf{push}_{a:=\lambda x.a'\,x\,x}(\{ab^n e : n \in \mathbb{N}\}) \quad = \quad \{a'(b^n e, b^n e) : n \in \mathbb{N}\}$$

is not a regular language of trees.

Conclusion

Future work:

- Characterize which λ-terms admit pushforwards.
- What is the rôle of non-linearity?

Conclusion

Future work:

- Characterize which λ-terms admit pushforwards.
- What is the rôle of non-linearity?

Thank you for your attention!

Any questions?

Bibliography

[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.

[MZ15] Paul-André Melliès and Noam Zeilberger. **"Functors are Type Refinement Systems".** In: *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*. POPL '15. Mumbai, India: Association for Computing Machinery, 2015, 3–16. ISBN: 9781450333009. DOI: 10.1145/2676726.2676970. URL: https://doi.org/10.1145/2676726.2676970.

[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.

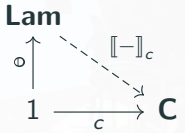
Interpretation: the universal property of λ-terms

The category **Lam**, whose objects are types and where

$$\text{a morphism } A \to B \qquad \text{is} \qquad \text{a } \lambda\text{-terms } t \in \Lambda(A \Rightarrow B)$$

is the free CCC on one object $\mathbb{O}$.

This means that, for every object c of $\mathbf{C}$, there exists a unique CCC functor



.

Its action on morphisms restricts to a function on closed λ-terms

$$[\![-]\!]_c \quad : \quad \underbrace{\mathbf{Lam}(1, A)}_{\cong\, \Lambda(A)} \longrightarrow \mathbf{C}(1, [\![A]\!]_c) \; .$$

Semantic languages of λ-terms

For **words**, a morphism $\varphi : \Sigma^* \to M$ into a finite monoid and $F \subseteq M$ induce

$$L_F \quad := \quad \{w \in \Sigma^* \mid \varphi(w) \in F\} \ .$$

The notion of regular language of λ-terms has been introduced by Salvati.

For λ-**terms** of type A, any object c of $\mathbf{C}$ and $F \subseteq \mathbf{C}(1, [\![A]\!]_c)$ induce

$$L_F \quad := \quad \{M \in \Lambda(A) \mid [\![M]\!]_c \in F\} \ .$$

Definition. A language of λ-terms is **recognized by C** if it is of the form L_F.

Interpreting in the CCC of finite sets yields the deterministic automata semantics.