Finitary semantics and languages of λ-terms

V. Moreau

Vincent Moreau (moreau@irif.fr)
IRIF, Université Paris Cité

Abstract.

This is joint work with Sam van Gool, Paul-André Melliès and Tito Nguyễn.

There is a growing connection between automata theory and the theory of λ-calculus. Indeed, the Church encoding shows that finite words and ranked trees are simply typed λ-terms. For instance, words over the alphabet $\Sigma = \{a, b\}$ correspond to λ-terms of type

$$\mathsf{Church}_\Sigma \quad := \quad \underbrace{(\mathtt{o} \Rightarrow \mathtt{o})}_{a \text{ transition}} \Rightarrow \underbrace{(\mathtt{o} \Rightarrow \mathtt{o})}_{b \text{ transition}} \Rightarrow \underbrace{\mathtt{o}}_{\text{initial state}} \Rightarrow \underbrace{\mathtt{o}}_{\text{output state}}$$

Moreover, their semantic interpretations in the cartesian closed category **FinSet** coincides with their behavior in finite deterministic automata. This semantic observation led Salvati to define the notion of **recognizable language** in [7] as any set of λ-terms of a given type A of the form

$$\{M \in \Lambda(A) \mid [\![M]\!]_Q \in F\} \qquad \text{for some finite set } Q \text{ and subset } F \subseteq [\![A]\!]_Q.$$

The recognizable languages of type Church_Σ are then exactly the regular languages of words, seen through the Church encoding. Moreover, Salvati has shown that, for any type A, languages of λ-terms of that type assemble into a Boolean algebra. This definition, using finite sets, extends to any cartesian closed category.

There is another, more syntactic link between automata theory and λ-calculus. A seminal result by Hillberand and Kanellakis [3] states that a set of finite words is a regular language if and only if its characteristic function is λ-definable, modulo a type-casting operation sending any $M \in \Lambda(A)$ to $M[B] \in \Lambda(A[B])$. This observation is at the heart of the implicit automata program started in [5], which shows an analogous correspondence between star-free languages and planar λ-terms.

This line of work yields another, more syntactic notion of regular language of λ-terms of type A, implicit in the work of Hillebrand and Kanellakis. A **syntactically regular language** of λ-terms of a given type A is any set of the form

$$\{M \in \Lambda(A) \mid R\ M[B] =_{\beta\eta} \mathsf{true}\} \qquad \text{for some type } B \text{ and } \lambda\text{-term } R \in \Lambda(A[B] \Rightarrow \mathsf{Bool})$$

where Bool is the type $\mathtt{o} \Rightarrow \mathtt{o} \Rightarrow \mathtt{o}$ and true is the first projection.

In [4], we show that, for a large class of sufficiently well-behaved cartesian closed categories, the associated recognizable languages are exactly the syntactically regular ones. More precisely:

Theorem 1 (§7 of [4])**.** *A language of λ-terms of type A is recognizable by a non-thin well-pointed locally finite cartesian closed category if and only if it is syntactically regular.*

Theorem 1 provides evidence that the notion of recognizable language of λ-terms is robust, and does not depend on the category of finite sets. Its proof relies on a new construction on cartesian closed categories called **squeezing**, which is inspired by normalization by evaluation.

In [2], we have introduced profinite λ-terms, using semantic interpretation in finite sets, which assemble into a cartesian closed category **ProLam**. Profinite λ-terms of type Church_Σ are exactly the profinite words, and they extend the correspondance coming from Stone duality with regular languages [6, 1] in the following way:

Theorem 2 (Proposition 3.4 of [2]). *The space of profinite λ-terms of type A is the Stone dual of the Boolean algebra of regular languages of λ-terms of type A.*

Dually, the combination of Theorem 1 with Theorem 2 shows that the space of profinite λ-terms, initially defined in the setting of semantic interpretation in finite sets, does not depent on that construction.

References

[1] Mai Gehrke, Serge Grigorieff, and Jean-Éric Pin. Duality and equational theory of regular lan guages. In Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, volume 5126 of Lecture Notes in Computer Science, pages 246–257. Springer, 2008. doi:10.1007/978-3-540-70583-3_21.

[2] Sam van Gool, Paul-André Melliès, and Vincent Moreau. Profinite lambda-terms and parametricity. Electronic Notes in Theoretical Informatics and Computer Science, Volume 3 - Proceedings of MFPS XXXIX , November 2023. URL: https://entics.episciences.org/12280, doi:10.46298/entics.12280.

[3] Gerd G. Hillebrand and Paris C. Kanellakis. On the expressive power of simply typed and let-polymorphic lambda calculi. In Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA, July 27-30, 1996, pages 253–263. IEEE Computer Society, 1996. doi:10.1109/LICS.1996.561337.

[4] Vincent Moreau and Lê Thành Dũng (Tito) Nguyễn. Syntactically and semantically regular languages of lambda-terms coincide through logical relations. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. URL: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CSL.2024.40, doi:10.4230/LIPICS.CSL.2024.40.

[5] Lê Thành Dũng Nguyễn and Cécilia Pradic. Implicit automata in typed λ-calculi I: aperiodicity in a non-commutative logic. In 47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany, volume 168 of LIPIcs, pages 135:1–135:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPIcs.ICALP.2020.135.

[6] Nicholas Pippenger. Regular languages and stone duality. Theory of Computing Systems, 30(2):121–134, April 1997. URL: http://dx.doi.org/10.1007/BF02679444, doi:10.1007/bf02679444.

[7] Sylvain Salvati. Recognizability in the simply typed lambda-calculus. In Logic, Language, Information and Computation, 16th International Workshop, WoLLIC 2009, Tokyo, Japan, June 21-24, 2009. Proceedings, volume 5514 of Lecture Notes in Computer Science, pages 48–60. Springer, 2009. doi:10.1007/978-3-642-02261-6_5