Call-by-value and call-by-name calculi: syntax, semantics, and logics

Internship proposal

December 2015

Keywords
Lambda-calculus, Curry-Howard correspondence, linear logic, proof-nets, denotational semantics.

Thematics
Analysis of the properties of programming languages. Study of the call-by-value mechanism for the evaluation of functional programs. Design and analysis of programming constructs based on the logical analysis of computation.

Audience
M2/M1 students in logic or computer science.

Hosting institution
LIPN, Laboratoire d'Informatique de Paris Nord (UMR 7030 CNRS). Institut Galilée, Université Paris 13.
99, avenue Jean-Baptiste Clément. 93430 Villetaneuse.
`http://www.lipn.univ-paris13.fr`

Supervisors
Stefano Guerrini (email: `stefano.guerrini@univ-paris13.fr`)
Giulio Manzonetto (email: `giulio.manzonetto@univ-paris13.fr`)

PhD Thesis
Possible.

Duration
From 3 to 6 months.

Period
From February to July.

Required competencies
The candidate is supposed to have some basic knowledge of λ-calculus and first order logic.

Context

In Computer Science, an evaluation strategy is a set of rules for evaluating expressions in a programming language. An evaluation strategy determines in which order programs are evaluated; emphasis is typically placed on functions and arguments.

In a *call-by-name* language, the arguments of a function application are not evaluated before the function is called, as arguments are passed to the function using a copy mechanism. On the contrary, in a *call-by-value* programming language the arguments of a function application must be evaluated before the evaluation of the function. For efficiency reasons most practical functional programming languages are call-by-value.

Because of its very simple syntax, the λ-calculus [Bar84] is an excellent language where to start analyzing differences and analogies between these two parameter passing methods, abstracting away from implementation details [Plo75]. The techniques developed in this context can be generalized to extensions of λ-calculus like, e.g., Parigot's $\lambda\mu$-calculus [Par92, Bie98], Ehrhard and Regnier's differential λ-calculus [ER03], and others.

Linear logic [Gir87] plays a key role in the fine analysis of computational properties of λ-calculus. Its decomposition of logical connectives into linear connectives and exponential modalities allows to have a fine use of resources, and in particular on how and when to use values or to evaluate functions. Moreover, the graph representation of proofs introduced by Girard, the so-called proof nets, give a particularly useful framework for studying properties and implementations of term evaluations (for a simple introduction on the correspondence between proof nets and λ-calculus, see [Gue04]).

Objectives

During the stage, the student is supposed to write a survey on the state of the art in the subject, developing one (or more) of the following aspects depending from his/hers main interests. In general, we expect the outcome of his/her work to be a clearer road map of the area, and a deeper understanding of the call-by-value parameter passing methods. The survey should be a good starting point for a possible PhD thesis.

Logical Approach. The call-by-value passing mechanism can be explained from a logical point of view as a particular translation of intuitionistic implication into Linear Logic. This translation is based on the mapping $A \to B = !(A \multimap B)$ and corresponds to a call-by-value encoding of λ-calculus (see also [MOTW99]). In the case of λ-calculus this translation is well-known[1] and corresponds to a suitable decoration of the *boxes* in the corresponding proof nets. Boxes control evaluations and play a key role in the aforementioned call-by-value translation of λ-calculus into linear logic. In [AG09], Accattoli and Guerrini have shown an efficient way to represent boxes by means of so-called *jumps*, and have shown a direct correspondence between proof nets with jumps and λ-calculus with explicit substitutions [Kes07]. A possible topic of the stage might be to analyze the call-by-value translation in the framework proposed by Accattoli and Guerrini and to study the corresponding calculi with explicit substitutions.

Semantic Approach. Denotational semantics aims to formalize the 'meanings' of programming languages by constructing suitable objects called *models*, which describe the meanings of expressions from the languages. Using this approach it is possible to capture operational properties of programs by means of more abstract models, on which a broader range of tools and proof techniques are available. For instance, in an *adequate* model a term has a non-empty interpretation if and only if it is normalizing; if the model is moreover *fully abstract* two programs are equivalent in the model whenever one can be replaced by the other in every context without noticing any difference with respect to a given observational equivalence (typically termination).

The definitions of a model of the call-by-name λ-calculus are well known and understood. From the point of view of universal algebra a model is a *combinatory algebra* satisfying suitable properties. From the point of view of category theory a model is a *reflexive objects* in cartesian closed category. Concrete examples of models can be built by solving domain equations as $D \cong (D \Rightarrow D)$. The problem of defining models of the call-by-value λ-calculus is much more open since there are no algebraic definitions, and only a few papers discussing the categorical notion [PRR99]. We propose to start from recent work in the domain [Ehr12, DMP13, CG14] where the authors describe concrete models for call-by-value non-deterministic/resource sensitive calculi, in order to propose abstract notions based on universal algebra or category theory and build and study new kind of models.

Practical aspects and scientific framework

The Logic, Computation and Reasoning (LCR) team of LIPN has a well-established expertise in Linear Logic, and on its applications to several domains of foundations of logic and computation: proof theory, λ-calculus and functional programming, denotational semantics, computational complexity. At the moment (December 2015), 3 full professors, 5 assistant professors, 2 CNRS assistant researchers, and 1 postdoctoral researchers actively work on these thematics; moreover, 5 PhD thesis are currently under development on logic and computation thematics. The researchers of the LCR team lead or contribute to national and international research projects, and have research collaborations with Italy, United Kingdom, The Netherlands, Japan, and United States. Each year, the LCR team hosts leading international researchers for short and long periods.

[1] It can be found in Girard's original paper on linear logic [Gir87].

References

[AG09] Beniamino Accattoli and Stefano Guerrini. Jumping boxes: Representing lambda-calculus boxes by jumps. In *Computer science logic*, volume 5771 of *Lecture Notes in Comput. Sci.*, pages 55–70. Springer, Berlin, 2009.

[Bar84] Hendrik Pieter Barendregt. *The Lambda Calculus – Its Syntax and Semantics*, volume 103 of *Studies in Logic and the Foundations of Mathematics*. North-Holland, 1984.

[Bie98] G.M. Bierman. A computational interpretation of the $\lambda\mu$-calculus. In L. Brim, J. Gruska, and J. Zlatuška, editors, *Proceedings of Symposium on Mathematical Foundations of Computer Science*, volume 1450 of *Lecture Notes in Computer Science*, pages 336–345, 1998.

[CG14] Alberto Carraro and Giulio Guerrieri. A semantical and operational account of call-by-value solvability. In Anca Muscholl, editor, *Foundations of Software Science and Computation Structures - 17th International Conference, FOSSACS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings*, volume 8412 of *Lecture Notes in Computer Science*, pages 103–118. Springer, 2014.

[DMP13] Alejandro Díaz-Caro, Giulio Manzonetto, and Michele Pagani. Call-by-value non-determinism in a linear logic type discipline. In Sergei N. Artëmov and Anil Nerode, editors, *Logical Foundations of Computer Science, International Symposium, LFCS 2013, San Diego, CA, USA, January 6-8, 2013. Proceedings*, volume 7734 of *Lecture Notes in Computer Science*, pages 164–178. Springer, 2013.

[Ehr12] Thomas Ehrhard. Collapsing non-idempotent intersection types. In Patrick Cégielski and Arnaud Durand, editors, *Computer Science Logic (CSL'12) - 26th International Workshop/21st Annual Conference of the EACSL, CSL 2012, September 3-6, 2012, Fontainebleau, France*, volume 16 of *LIPIcs*, pages 259–273. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2012.

[ER03] Thomas Ehrhard and Laurent Regnier. The differential lambda-calculus. *Theor. Comput. Sci.*, 309(1-3):1–41, 2003.

[Gir87] Jean-Yves Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987.

[Gue04] Stefano Guerrini. Proof nets and the lambda-calculus. In Thomas Ehrhard, Paul Ruet, Jean-Yves Girard, and Phil Scott, editors, *Linear Logic in Computer Science*, volume 316 of *London Mathematical Society Lecture Notes Series*, pages 65–118. Cambridge University Press, 2004.

[Kes07] Delia Kesner. The theory of calculi with explicit substitutions revisited. In Jacques Duparc and Thomas A. Henzinger, editors, *CSL*, volume 4646 of *Lecture Notes in Computer Science*, pages 238–252. Springer, 2007.

[MOTW99] J. Maraist, M. Odersky, D. N. Turner, and P. Wadler. Call-by-name, call-by-value, call-by-need and the linear lambda calculus. *Theoretical Computer Science*, 228:175–210, 1999.

[Par92] Michel Parigot. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In Andrei Voronkov, editor, *Logic Programming and Automated Reasoning,International Conference LPAR'92, St. Petersburg, Russia, July 15-20, 1992, Proceedings*, volume 624 of *Lecture Notes in Computer Science*, pages 190–201. Springer, 1992.

[Plo75] Gordon Plotkin. Call-by-name, call-by-value and the λ-calculus. *Theoretical Computer Science*, 1(2):125–159, December 1975.

[PRR99] Alberto Pravato, Simona Ronchi Della Rocca, and Luca Roversi. The call-by-value [lambda]-calculus: a semantic investigation. *Mathematical Structures in Computer Science*, 9(5):617–650, 1999.