Information and Computation 248 (2016) 2–21

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Logarithmic space and permutations ☆

Clément Aubert [a,1], Thomas Seiller [b,2]

[a] LIPN – UMR CNRS 7030 Institut Galilée - Université Paris-Nord, 99 avenue J.-B. Clément, 93430 Villetaneuse, France
[b] Institut des Hautes Études Scientifiques, Le Bois-Marie, 35 Route de Chartres, 91440 Bures-sur-Yvette, France

ARTICLE INFO

Article history:
Received 15 November 2013
Available online 6 January 2016

Keywords:
Linear logic
Complexity
Geometry of interaction
Pointer machine
Finite automata
Logarithmic space

ABSTRACT

In a recent work, Girard proposed a new and innovative approach to computational complexity based on the proofs-as-programs correspondence. In a previous paper, the authors showed how Girard's proposal succeeds in obtaining a new characterization of co-NL languages as a set of operators acting on a Hilbert Space. In this paper, we extend this work by showing that it is also possible to define a set of operators characterizing the class L of logarithmic space languages.

1. Introduction

1.1. Linear logic and implicit computational complexity

Logic, and more precisely proof theory – the domain whose purpose is the formalization and study of mathematical proofs – recently yielded numerous developments in theoretical computer science. These developments are founded on a correspondence, often called Curry–Howard correspondence, between mathematical proofs and computer programs (usually formalized in lambda-calculus). The main interest of this correspondence lies in its dynamic nature: program execution corresponds to a procedure on mathematical proofs known as the cut-elimination procedure.

In the eighties, Jean-Yves Girard discovered linear logic through a study of mathematical models of the lambda-calculus. This logical system, as a direct consequence of this correspondence between proofs and programs, is particularly interesting from the point of view of the mathematical foundations of computer science for its resource-awareness. In particular, it gave birth to a number of developments in the field of implicit computational complexity, for instance through the definition and study of restricted logical systems (sub-systems of linear logic) in which the set of representable functions captures a complexity class. For instance, elementary linear logic (ELL) restricts the rules governing the use of exponential connectives – the connectives dealing with the duplication of the arguments of a function – and the set of representable functions in ELL is exactly the set of elementary time functions [3]. It was also shown [4] that a characterization of logarithmic space computation can be obtained if one restricts both the rules of exponential connectives and the use of universal quantifiers. Finally, a variation on the notion of linear logic *proof nets* succeeded in characterizing the classes **NC** of problems that can be efficiently parallelized [5].

☆ Work partially supported by the ANR projects ANR-10-BLAN-0213 LOGOI and ANR-08-BLAN-0211 COMPLICE.
E-mail addresses: aubert@lipn.fr (C. Aubert), seiller@ihes.fr (T. Seiller).
[1] Current address: Appalachian State University, Boone, NC, USA.
[2] Current address: Department of Computer Science, University of Copenhagen, Njalsgade 128 bygning 24, 2300 København S, Denmark.

http://dx.doi.org/10.1016/j.ic.2014.01.018
0890-5401/

1.2. Geometry of interaction

A deep study of the formalization of proofs in linear logic, in particular their formalization as proof nets, led Jean-Yves Girard to initiate a program entitled *geometry of interaction* (GoI) [6]. This program, in a first approximation, intends to define a semantics of proofs that accounts for the dynamics of the cut-elimination procedure. Through the correspondence between proofs and programs, this would define a semantics of programs that accounts for the dynamics of their execution. However, the geometry of interaction program is more ambitious: beyond the mere interpretation of proofs, its purpose is to completely reconstruct logic around the dynamics of the cut-elimination procedure. This means reconstructing the logic of programs, where the notion of formula – or type – accounts for the behavior of algorithms.

Informally, a geometry of interaction is defined by a set of *paraproofs* together with a notion of interaction, in the same way one defines strategies and their composition in game semantics. An important tool in the construction is a binary function that measures the interaction between two paraproofs. With this function one defines a notion of orthogonality that corresponds to the negation of logic and reconstructs the formulas as sets of paraproofs equal to the orthogonal of a given set of paraproofs: a formula is therefore a set of "programs" that interact in a similar way to a given set of *tests*.

Since the introduction of this program Jean-Yves Girard proposed different constructions to realize it. These constructions share the notion of paraproofs: operators in a von Neumann algebra. They however differ on the notion of orthogonality they use: in the first constructions, this notion was founded on the nilpotency of the product of two operators, while the more recent construction [7] uses Fuglede–Kadison determinant – a generalization of the usual determinant of matrices that can be defined in type II_1 factors.

Since the reconstruction of logic is based on the notion of execution, geometry of interaction constructions are particularly interesting for the study of computational complexity. It is worth noting that the first construction of GoI [8] allowed Abadi, Gonthier, and Lévy [9] to explain the optimal reduction of λ-calculus defined by Lamping [10]. This first GoI construction was also used to obtain results in the field of implicit computational complexity [11].

1.3. A new approach to complexity

Recently Jean-Yves Girard proposed a new approach for the study of complexity classes that was inspired by his latest construction of a geometry of interaction. Using the crossed product construction between a von Neumann algebra and a group acting on it, he proposed to characterize complexity classes as sets of operators obtained through the internalization of outer automorphisms of the type II_1 hyperfinite factor. The authors showed in a recent paper [2] that this approach succeeds in defining a characterization of the set of **co**-**NL** languages as a set of operators in the type II_1 hyperfinite factor. The proof of this result was obtained through the introduction of *non-deterministic pointer machines*, which are abstract machines designed to mimic the computational behavior of operators. The result was obtained by showing that a **co**-**NL** complete problem could be solved by these machines.

In this paper, we extend these results in two ways. The first important contribution is that we give an alternative proof of the fact that **co**-**NL** is indeed characterized by non-deterministic pointer machines. This new proof consists in showing that pointer machines can simulate the well-known and studied two-way multi-head finite automata [12,13]. The second contribution of this paper consists in obtaining a characterization of the class **L** as a set of operators in the hyperfinite factor of type II_1. By studying the set of operators that characterize the class **co**-**NL** and in particular the encoding of non-deterministic pointer machines as operators, we are able to show that the operators encoding a deterministic machine satisfy a condition expressed in terms of norm. We then manage to show that the language decided by an operator satisfying this norm condition is in the class **L**, showing that the set of all such operators characterizes **L**.

2. The basic picture

The construction uses an operator-theoretic construction known as the crossed product of an algebra by a group acting on it. The interested reader can find a quick overview of the theory of von Neumann algebras in the appendix of the second author's work on geometry of interaction [14], and a brief presentation of the crossed product construction in the authors' previous work [2] on the characterization of **co**-**NL**. For a more complete presentation of the theory of operators and the crossed product construction, we refer to the well-known series of Takesaki [15–17].

In a nutshell, the crossed product construction $\mathfrak{A} \rtimes_\alpha G$ of a von Neumann algebra $\mathfrak{A}$ and a group action $\alpha : G \to \mathrm{Aut}(\mathfrak{A})$ defines a von Neumann algebra containing $\mathfrak{A}$ and unitaries that internalize the automorphisms $\alpha(g)$ for $g \in G$. For this, one considers the Hilbert space[3] $\mathbb{K} = L^2(G, \mathbb{H})$ where $\mathbb{H}$ is the Hilbert space $\mathfrak{A}$ is acting on, and one defines two families of unitary operators[4] in $\mathcal{L}(\mathbb{K})$:

[3] The construction $L^2(G, \mathbb{H})$ is a generalization of the well-known construction of the Hilbert space of square-summable functions: in case G is considered with the discrete topology, the elements are functions $f : G \to \mathbb{H}$ such that $\sum_{g \in G} \|f(g)\|^2 < \infty$.

[4] Recall that in the algebra $\mathcal{L}(\mathbb{H})$ of bounded linear operators on the Hilbert space $\mathbb{H}$ (we denote by $\langle \cdot, \cdot \rangle$ its inner product), there exists an anti-linear involution $(\cdot)^*$ such that for any $\xi, \eta \in \mathbb{H}$ and $A \in \mathcal{L}(\mathbb{H})$, $\langle A\xi, \eta \rangle = \langle \xi, A^*\eta \rangle$. This *adjoint operator* coincides with the conjugate-transpose in the algebras of square matrices. A *unitary operator* u is an operator such that $uu^* = u^*u = 1$.