Journal of Computer and System Sciences 81 (2015) 1592–1613

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Normality and automata

Verónica Becher [a,b], Olivier Carton [c], Pablo Ariel Heiber [a,*]

[a] Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Pabellón I, Ciudad Universitaria, (1428) Buenos Aires, Argentina

[b] CONICET, Argentina

[c] Laboratoire d'Informatique Algorithmique: Fondements et Applications, CNRS UMR 7089, Université Paris Diderot – Paris 7, Case 7014, 75205 Paris Cedex 13, France

ARTICLE INFO

Article history:
Received 5 May 2014
Received in revised form 5 March 2015
Accepted 30 April 2015
Available online 10 June 2015

Keywords:
Normal numbers
Finite automata
Non-deterministic automata

ABSTRACT

We prove that finite-state transducers with injective behavior, deterministic or not, real-time or not, with no extra memory or a single counter, cannot compress any normal word. We exhaust all combinations of determinism, real-time, and additional memory in the form of counters or stacks, identifying which models can compress normal words. The case of deterministic push-down transducers is the only one still open. We also present results on the preservation of normality by selection with finite automata. Complementing Agafonov's theorem for prefix selection, we show that suffix selection preserves normality. However, there are simple two-sided selection rules that do not.

1. Introduction

Let us recall the definition of *normality* for real numbers, given by Émile Borel [4] more than one hundred years ago. A real number is normal to an integer base if, in its infinite expansion expressed in that base, all blocks of digits of the same length have the same limiting frequency. Borel proved that almost all real numbers are normal to all integer bases. However, very little is known on how to prove that a given number has the property. The problem of proving normality to just one base does not seem easier, see Bugeaud's book [6]. This is a motivation to investigate different equivalent definitions of normality and different operations that preserve it.

The present work deals with the characterization and the preservation of normality by means of operations done by finite automata. For this, given an integer base b, greater than or equal to 2, we consider the alphabet consisting of the digits $0, 1, \dots, b-1$. We regard the expansion of a real number expressed in base b as a sequence of symbols in this alphabet, or, as we shall call it, an infinite word. This determines the notion of normality for infinite words.

A fundamental theorem relates normality and finite automata: an infinite word is normal to a given alphabet if and only if it cannot be compressed by lossless finite transducers. These are deterministic finite automata with injective input–output behavior. This result was first obtained by joining a theorem by Schnorr and Stimm [17] with a theorem by Dai, Lathrop, Lutz and Mayordomo [10]. Becher and Heiber gave a direct proof [3].

What is the true computational power needed to compress normal words? Of course, each computable normal word is compressible by some Turing machine. For instance, consider Champernowne's construction [9]. Since automata with

* Corresponding author.
E-mail addresses: vbecher@dc.uba.ar (V. Becher), Olivier.Carton@liafa.univ-paris-diderot.fr (O. Carton), pheiber@dc.uba.ar (P.A. Heiber).

http://dx.doi.org/10.1016/j.jcss.2015.04.007
0022-0000/

Table 1
Compressibility of normal infinite words by different kinds of transducers.

Finite-state transducer	Deterministic	Non-deterministic	Non-real-time
No extra memory	Not compress (Theorem 2.3)	Not compress (Theorem 4.1)	Not compress (Corollary 4.4)
One counter	Not compress	Not compress	Not compress (Theorem 5.2)
More than one counter	Not compress (Theorem 3.1)	Not compress (Corollary 5.1)	Compress (Turing complete)
One stack	**?**	Compress (Theorem 6.1)	Compress
One stack and one counter	Compress (Theorem 6.2)	Compress	Compress (Turing complete)

enough computational power are equivalent to a Turing machine, to answer the question we ought to start with the most elementary type of automata, which are finite-state automata, and consider different enhancements. Here we analyze deterministic, non-deterministic real-time and non-real-time transition functions; having zero, one, or more counters; with or without a stack.

Whether deterministic or non-deterministic machines have the same computational power is a fundamental question in theoretical computer science. Here we address it with respect to the ability of compressing normal words. Recall that although deterministic and non-deterministic finite automata recognize the same rational sets, deterministic Büchi automata are strictly less expressive than the non-deterministic ones (see Perrin and Pin's book [15]). It is not always possible to determinize a Büchi automaton to recognize the same set of infinite words (it is only possible as a Muller automaton). Furthermore, functions and relations realized by deterministic transducers are proper subclasses of rational functions and relations realized by non-deterministic ones [2].

Here we prove that finite transducers with injective behavior, even non-deterministic non-real-time ones, but with no extra memory or just a single counter, cannot compress any normal infinite word. Adding memory yields compressibility results: there are non-deterministic non-real-time transducers with more than one counter that compress some normal infinite words. Also there are non-deterministic real-time transducers with a stack that can do it.

Table 1 summarizes the results we obtain about compressibility of normal infinite words by different kinds of transducers. The columns represent different levels of restrictions on the transitions. The first column represents determinism, that is, there is exactly one transition leaving a given state by reading a given symbol. The second column represents non-determinism, there are several transitions leaving a given state by reading the same symbol. The restriction represented in the third column adds the possibility of also having transitions that do not read any symbol (usually called λ-transitions). The rows of the table represent different memory models. In all cases there is bounded memory represented by states. Each row details possible additions of counters or stacks. The realized relation is assumed to be bounded-to-one. The case of a deterministic transducers with a single stack remains open:

Open question. Can any deterministic push-down transducer compress a normal word?

In the present paper, we consider transducers that process the input word from left to right without coming back. These are called one-way transducers. In contrast, two-way transducers can move their reading head back and forth. They have been investigated with regards to normality by Carton and Heiber in [7] where it is shown that these more powerful transducers still cannot compress normal words.

We also obtain new results on the preservation of normality by selection. A celebrated theorem by Agafonov describes an operation by a finite automaton that selects symbols from an infinite word by looking at its prefixes and by recognizing those that belong to a rational set. In case the word is normal, the word obtained by the selected symbols is normal as well. Agafonov published it in 1968 [1], but unfortunately the proof there depends on work only available in the Russian literature. Schnorr and Stimm [17] proved a generalization of the theorem. M. O'Connor [14] gave another proof of Agafonov's result using automata predictors, and Broglio and Liardet [5] generalized it to arbitrary alphabets. Becher and Heiber [3] wrote an alternative proof using the characterization of normality in terms of incompressibility. It is known that Agafonov's theorem fails for slightly more powerful selection. Merkle and Reimann [12] showed that normality is preserved neither by deterministic one-counter sets (recognized by deterministic one-counter automata) nor by linear sets (recognized by one-turn pushdown automata).

Here we complement Agafonov's theorem and we show that selection based on suffixes, as opposed to prefixes, also preserves normality. However, there are simple two-sided selection rules that do not preserve normality and we exhibit one. These results are proved in Theorems 7.2 and 7.3.