Information and Computation 247 (2016) 23–36

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Finite state incompressible infinite sequences [☆], [☆☆]

Cristian S. Calude [a,*], Ludwig Staiger [b], Frank Stephan [c]

[a] Dept. of Computer Science, University of Auckland, Auckland, New Zealand
[b] Martin-Luther-Universität Halle-Wittenberg, Institut für Informatik, D-06099 Halle, Germany
[c] Department of Mathematics and Department of Computer Science, National University of Singapore, Singapore 119076, Republic of Singapore

ARTICLE INFO

Article history:
Received 4 September 2014
Received in revised form 21 August 2015
Available online 1 December 2015

Keywords:
AIT
Finite state complexity and incompressibility
Normality

ABSTRACT

In this paper we define and study finite state complexity of finite strings and infinite sequences as well as connections between these complexity notions to randomness and normality. We show that the finite state complexity does not only depend on the codes for finite transducers, but also on how the codes are mapped to transducers. As a consequence we relate the finite state complexity to the plain (Kolmogorov) complexity, to the process complexity and to prefix-free complexity. Working with prefix-free sets of codes we characterise Martin-Löf random sequences in terms of finite state complexity: the weak power of finite transducers is compensated by the high complexity of enumeration of finite transducers. We also prove that every finite state incompressible sequence is normal, but the converse implication is not true. These results also show that our definition of finite state incompressibility is stronger than all other known forms of finite automata based incompressibility, in particular the notion related to finite automaton based betting systems introduced by Schnorr and Stimm. The paper concludes with a discussion of open questions.

1. Introduction

Algorithmic Information Theory (AIT) [8,20,30,27] uses various measures of descriptional complexity to define and study various classes of "algorithmically random" finite strings or infinite sequences. The theory, based on the existence of a universal Turing machine (of various types), is very elegant and has produced many important results.

The incomputability of all descriptional complexities is an obstacle towards more "down-to-earth" applications of AIT (e.g. for practical compression). One possibility to avoid incomputability is to restrict the resources available to the universal Turing machine and the result is resource-bounded descriptional complexity [7]. Another approach is to restrict the computational power of the machines used, for example, using context-free grammars or straight-line programs instead of Turing machines [15,24,25,34].

☆ This work was done in part during C.S. Calude's visits to the Martin-Luther-Universität Halle-Wittenberg in October 2012 and the National University of Singapore in November 2013 and June 2014, and L. Staiger's visits to the CDMTCS, University of Auckland and the National University of Singapore in March 2013. The work was supported in part by NUS grant R146-000-181-112 (PI F. Stephan).

☆☆ A preliminary version of this paper has appeared in T.V. Gopal, M. Agrawal, A. Li, B.S. Cooper (eds.), *Proceedings of the 11th Annual Conference on Theory and Applications of Models of Computation*, Lecture Notes in Computer Science, vol. 8402, Springer, 2014, pp. 50–66.

* Corresponding author.
E-mail addresses: cristian@cs.auckland.ac.nz (C.S. Calude), staiger@informatik.uni-halle.de (L. Staiger), fstephan@comp.nus.edu.sg (F. Stephan).

http://dx.doi.org/10.1016/j.ic.2015.11.003
0890-5401/

The first connections between finite state machine computations and randomness have been obtained for infinite sequences. Agafonov [1] proved that every subsequence selected from a (Borel) normal sequence by a regular language is also normal. Characterisations of normal infinite sequences have been obtained in terms of finite state gamblers, information lossless finite state compressors and finite state dimension: (a) a sequence is normal iff there is no finite state gambler that succeeds on it [35] (see also [6,17]) and (b) a sequence is normal iff it is incompressible by any information lossless finite state compressor [46]. Doty and Moser [18,19] used computations with finite transducers for the definition of finite state dimension of infinite sequences. The NFA-complexity of a string [15] can be defined in terms of finite transducers that are called in [15] "NFAs with advice"; the main problem with this approach is that NFAs used for compression can always be assumed to have only one state.

The definition of *finite state complexity of a finite string x* in terms of a computable enumeration of finite transducers and the input strings used by transducers which output x proposed in [10,11] is utilised to define *finite state incompressible sequences*. In Theorem 9 we prove that the finite state complexity lies properly between the plain complexity, as a lower bound, and the prefix-free complexity, as an upper bound, in the case that the enumeration of transducers considered is a universal one. Furthermore, while finite state incompressibility depends on the enumeration of finite transducers, many results presented here are *independent* of the chosen enumeration. For example, we prove that for every enumeration S every C_S-incompressible sequence is normal, Theorem 22. Furthermore, we can show that a sequence is Martin-Löf random iff it satisfies a strong incompressibility condition (parallel to the one for prefix-free Kolmogorov complexity) for every measure C_S based on some perfect enumeration S. One can furthermore transfer this characterisation to the measure C_S for universal enumerations S.

Finally, we illustrate the dependence of finite state complexity on the enumeration of finite transducers. We prove that in every sequence there are infinitely many finite state complexity dips when the complexity is based on some exotic enumerations.

2. Notation

In this section we introduce the notation used throughout the paper. By $\mathbb{N} = \{0, 1, 2, \ldots\}$ we denote the set of natural numbers. Its elements will be usually denoted by letters $i, \ldots, n$. By $\{0,1\}^*$ we denote the set of all binary strings (words) with ε denoting the empty string; $\{0,1\}^\omega$ is the set of all (infinite) binary sequences. The length of a finite string $x \in \{0,1\}^*$ is denoted by $|x|$. Sequences (infinite strings) are usually denoted by $\mathbf{x}, \mathbf{y}$; the prefix of length n of the sequence $\mathbf{x}$ is denoted by $\mathbf{x} \upharpoonright n$; the nth element of $\mathbf{x}$ is denoted by $\mathbf{x}(n)$.

For $w \in \{0,1\}^*$ and $\eta \in \{0,1\}^* \cup \{0,1\}^\omega$ let $w \cdot \eta$ be their *concatenation*. This concatenation product extends in an obvious way to subsets $L \subseteq \{0,1\}^*$ and $B \subseteq \{0,1\}^* \cup \{0,1\}^\omega$.

By $w \sqsubseteq u$ and $w \sqsubset \mathbf{y}$ we denote that w is a prefix of u and $\mathbf{y}$, respectively, and a prefix-free set $L \subset \{0,1\}^*$ is a set with the property that for all strings $p, q \in \{0,1\}^*$, if $p, pq \in L$ then $p = pq$.

3. Admissible transducers and their enumerations

We consider transducers which try to generate prefixes of infinite binary sequences from shorter binary strings and consider hence the following transducers: An *admissible transducer* is a deterministic transducer given by a finite set of states Q with starting state q_0 and transition functions δ, μ with domain $Q \times \{0,1\}$, and say that the transducer on state q and current input bit a transitions to $q' = \delta(q,a)$ and appends $w = \mu(q,a)$ to the output produced so far.

One can generalise inductively the functions μ and δ by stating that $\mu(q, \varepsilon) = \varepsilon$ and $\mu(q, av) = \mu(q,a) \cdot \mu(\delta(q,a), v)$ for states q and input strings av with a being one bit; similarly, $\delta(q, \varepsilon) = q$ and $\delta(q, av) = \delta(\delta(q,a), v)$. The output $T(v)$ of a transducer T on input-string v is then $\mu(q_0, v)$.

Definition 1. A partially computable function S mapping binary strings to admissible transducers is called an *enumeration* provided every admissible transducer T has a string $\sigma \in \mathrm{dom}(S)$; for a string $\sigma \in \mathrm{dom}(S)$, the admissible transducer assigned by S to σ is denoted as $S(\sigma) = T^S_\sigma$.

If the domain $\mathrm{dom}(S)$ is a prefix-free subset of $\{0,1\}^*$ then we call S a *prefix-free enumeration*.

Next we introduce two subclasses of prefix-free enumerations, that is, enumerations S having a prefix-free domain $\mathrm{dom}(S)$.

Definition 2. (See Calude, Salomaa and Roblot [10,11].) A *perfect enumeration* S of all admissible transducers is a partially computable function with a prefix-free and computable domain mapping each binary string $\sigma \in \mathrm{dom}(S)$ to an admissible transducer T^S_σ in an onto way.

Note that partially computable functions with a computable range (as considered here) have a computable inverse, that is, for each input y from the range, an algorithm finds, by searching in parallel over all possible inputs, an x which is mapped to y. It is known that there are perfect enumerations with a regular domain and that every perfect enumeration S