Theoretical Computer Science 702 (2017) 60–64

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

On the rate of decrease in logical depth

L.F. Antunes [a,1], A. Souto [b,c,2], P.M.B. Vitányi [d,e,*,3]

[a] University of Porto & CRACS/INESC-TEC, Rua do Campo Alegre, 1021, 4169-007 Porto, Portugal
[b] LaSIGE, Faculdade de Ciências, Universidade de Lisboa, Portugal Campo Grande, 1749-016 Lisboa, Portugal
[c] Departamento de Informática, Faculdade de Ciências, Universidade de Lisboa, Campo Grande, 1749-016 Lisboa, Portugal
[d] CWI, Science Park 123, 1098 XG Amsterdam, The Netherlands
[e] University of Amsterdam, The Netherlands

ARTICLE INFO

Article history:
Received 15 March 2017
Received in revised form 26 July 2017
Accepted 19 August 2017
Available online 25 August 2017
Communicated by L.M. Kirousis

Keywords:
Logical depth
Kolmogorov complexity
Compression

ABSTRACT

The logical depth with significance b of a string x is the shortest running time of a program for x that can be compressed by at most b bits. Another definition is based on algorithmic probability. We give a simple new proof for the known relation between the two definitions. We also prove the following: Given a string we can consider the maximal decrease in logical depth when the significance parameter increases by 1. There exists a sequence of strings of lengths $n = 1, 2, \ldots,$ such that this maximal decrease as a function of n rises faster than any computable function but not as fast as the Busy Beaver function. This holds also for the computation times of the shortest programs of these strings.

1. Introduction

The logical depth is related to complexity with bounded resources and measures the tradeoff between program sizes and running times. Computing a string x from one of its shortest programs may take a very long time, but computing the same string from a simple "`print(x)`" program of length about $|x|$ bits takes very little time.

A program p for x, i.e. such that $U(p) = x$ where U is a universal Turing machine, of larger length than a given program q for x may require less computation time than q does. This need not always be the case, as a longer program might simply perform some pointless redundant steps.

In general we associate longer computation times with shorter programs for x. As a consequence, one may raise the question of how much time can be saved by computing a given string from a longer program.

* Corresponding author.
E-mail addresses: antunes.lfa@gmail.com (L.F. Antunes), andrenunosouto@gmail.com (A. Souto), Paul.Vitanyi@cwi.nl (P.M.B. Vitányi).

[1] This work was supported by "Project *NanoSTIMA: Macro-to-Nano Human Sensing: Towards Integrated Multimodal Health Monitoring and Analytics*/NORTE-01-0145-FEDER-000016" which is financed by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF).
[2] This work was supported by LaSIGE Research Unit, ref. UID/CEC/00408/2013 and the FCT project Confident PTDC/EEI-CTP/4503/2014, tSQIG – Security and Quantum Information Group, the Instituto de Telecomunicações (IT) Research Unit, ref. UID/EEA/50008/2013, and the FCT Post-doc scholarship SFRH/BPD/76231/2011 during which the major part of the work was done.
[3] CWI is the National Research Institute for Mathematics and Computer Science in the Netherlands.

http://dx.doi.org/10.1016/j.tcs.2017.08.012
0304-3975/

1.1. Previous work

The above question was first considered by C. Bennett in [1]. The minimum time to compute a string by a b-incompressible program was called the *logical depth at significance* b of the string considered. Bennett also provided variations of this definition and studied their basic properties and relations. A more formal treatment, as well as an intuitive approach, was given in [4], Section 7.7.

1.2. Results

Section 2 introduces notations, definitions and results needed in the remainder. Section 3 presents two versions of logical depth and gives a simple new proof of quantitative relations between them. Section 4 shows that slight variations on the significance level may cause drastic variations of logical depth. Section 5 presents conclusions.

2. Preliminaries

We use *string* for a finite binary string. Strings are denoted by lower case letters, such as x, y and z. The *length* of a string x (the number of occurrences of bits in it) is denoted by $|x|$, and the *empty* string by ϵ. Thus, $|\epsilon| = 0$. The nth string in the lexicographic length-increasing order is viewed also as the natural number n and vice versa.

Computability, resource-bounded computation time, self-delimiting strings, big-O notation, and (prefix) Kolmogorov complexity are well-known and the properties, notations, are treated in [4].[4] Originally Kolmogorov complexity was introduced in 1965 in [2] and the prefix version in 1974 in [3]. The length of a self-delimiting version of a string of length n will be $n + 2\log n + 1$ where log denotes the logarithm base 2. That is, the self-delimiting version of x is $x' = 1^{||x||}0|x|x$ where $||x||$ denotes the length of $|x|$. Restricting the computation time resource is indicated by a superscript giving the allowed number of steps, usually denoted by d.

We choose a *reference optimal universal prefix Turing machine* and call it U. Let x, y be strings. The *prefix Kolmogorov complexity* $K(x|y)$ of x with auxiliary y is defined by

$$K(x|y) = \min_p\{|p| : U(p, y) = x\}.$$

If x is a string of length n then $K(x|y) \le n + O(\log n)$. The notation $U^d(p, y) = x$ means that $U(p, y) = x$ within d steps. The *d-time-bounded prefix Kolmogorov complexity* $K^d(x|y)$ is defined by[5]

$$K^d(x|y) = \min_p\{|p| : U^d(p, y) = x\}.$$

If the auxiliary string y is the empty string ϵ, then we usually drop it. Similarly, we write $U(p)$ for $U(p, \epsilon)$. The string x^* is a *shortest program* for x if $U(x^*) = x$ and $K(x) = |x^*|$. A string x is *c-incompressible* if $|x^*| \ge |x| - c$ and it is *c-compressible* if $|x^*| \le |x| - c$.

Define $Q(x) = \sum_{p:U(p)=x} 2^{-|p|}$. In [3] L.A. Levin proved in the Coding Theorem that (see [4] Theorem 4.3.3 for details):

$$-\log Q(x) = K(x) + O(1)$$

3. Different versions of logical depth

The logical depth as defined in [1] for a string (the finite case) comes in two versions: one based on the compressibility of programs of prefix Turing machines and the other using the ratio between algorithmic probabilities with and without time limits.

The algorithmic probability version is based on the so-called *a priori* probability [4] and its time-bounded version:

$$Q(x) = \sum_{U(p)=x} 2^{-|p|}, \quad Q^d(x) = \sum_{U^d(p)=x} 2^{-|p|}.$$

[4] The reader may be less familiar with the *prefix Turing machine*. It is a Turing machine with a one-way read-only program tape, an auxiliary tape, one or more work tapes and an output tape. All tapes are linear and divided in cells capable of containing one out of a finite set of symbols. Initially the program tape is inscribed with an infinite sequence of 0's and 1's and the head is scanning the leftmost cell (the program tape is semi-infinite). When the computation terminates the sequence of bits scanned is the *program*. For every fixed contents of the auxiliary tape the set of programs for such a machine is a prefix code (no program is a proper prefix of another program). Another consequence is that if the computation of a prefix Turing machine takes d steps then for its program p holds that $d \ge |p|$. The prefix Kolmogorov complexity is based on the prefix Turing machine similar to the (plain) Kolmogorov complexity based on the (plain) Turing machine.

[5] Since we deal with running times of computations the following can happen. Two different reference optimal universal prefix Turing machines may have different computation times for the same combination of input, auxiliary string, and output. It can also be the case that they have different sets of programs. Let U and U' be two optimal universal prefix Turing machines in the standard enumeration of prefix Turing machines. For every auxiliary y and every program q there is a program p with $|q| = |p| + O(1)$ such that $U^d(q, y) = U'^{d'}(p, y)$ for integers d, d' and $d' = g(d)$ with g a computable function. Therefore, although U and U' have the same length of shortest programs for a string x (with $O(1)$ precision, that is, up to a constant additive term), the time-limited prefix Kolmogorov complexity of a string x may differ by a non-constant additive term.