On time-bounded incompressibility of compressible strings
Daylight, E.G.; Koolen Wijkstra, W.; Vitanyi, P.M.B.

Citation for published version (APA):

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: http://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.
On Time-Bounded Incompressibility of Compressible Strings

Edgar G. Daylight\textsuperscript{a,1} Wouter M. Koolen\textsuperscript{b} Paul M.B. Vitányi\textsuperscript{b,c}

\textsuperscript{a}University of Amsterdam, Institute of Logic, Language, and Computation, Amsterdam, The Netherlands
\textsuperscript{b}Centrum voor Wiskunde en Informatica, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands
\textsuperscript{c}University of Amsterdam, Department of Computer Science, Amsterdam, The Netherlands

Abstract

We prove that there exist finite strings with very low Kolmogorov complexity that have very high time-bounded Kolmogorov complexity. Such strings are compressible but time-bounded incompressible. For every total recursive time bound \( t \), a constant fraction of all compressible strings is \( t \)-bounded incompressible.

Key words: Kolmogorov complexity, compressible strings, time-bound ed incompressible strings, Barzdins’s Lemma, computational complexity

1 Introduction

Informally, the Kolmogorov complexity of a finite binary string is the length of the shortest string from which the original can be losslessly reconstructed by an effective general-purpose computer such as a particular universal Turing machine \( U \). Hence it constitutes a lower bound on how far a lossless compression program can compress. Formally, the \textit{conditional Kolmogorov complexity} \( C(x|y) \) is the length of the shortest input \( z \) such that the universal Turing machine \( U \) on input \( z \) with auxiliary information \( y \) outputs \( x \). The \textit{unconditional Kolmogorov complexity} \( C(x) \) is defined by \( C(x|\epsilon) \) where \( \epsilon \) is the empty
string (of length 0). Let \( t \) be a total recursive function. Then, the time-bounded conditional Kolmogorov complexity \( C^t(x|y) \) is the length of the shortest input \( z \) such that the universal Turing machine \( U \) on input \( z \) with auxiliary information \( y \) outputs \( x \) within \( t(n) \) steps where \( n \) is the length in bits of \( x \). The time-bounded unconditional Kolmogorov complexity \( C^t(x) \) is defined by \( C^t(x|\epsilon) \). For an introduction to the definitions and notions of Kolmogorov complexity (algorithmic information theory) see [2].

1.1 Related Work

Already in 1968 J. Barzdins [1] obtained a result known as Barzdins’s Lemma, probably the first result in resource-bounded Kolmogorov complexity, of which the lemma below quotes the items that are relevant here. Let \( \chi \) denote the characteristic sequence of an arbitrary recursively enumerable (r.e.) set \( A \). That is, \( \chi \) is an infinite sequence \( \chi_1\chi_2\ldots \) where bit \( \chi_i \) equals 1 if and only if \( i \in A \). Let \( \chi_{1:n} \) denote the first \( n \) bits of \( \chi \), and let \( C(\chi_{1:n}|n) \) denote the conditional Kolmogorov complexity of \( \chi_{1:n} \), given the number \( n \).

**Lemma 1** (i) For every characteristic sequence \( \chi \) of a r.e. set \( A \) there exists a constant \( c \) such that for all \( n \) we have \( C(\chi_{1:n}|n) \leq \log n + c \).

(ii) There exists a r.e. set \( A \) with characteristic sequence \( \chi \) such that for every total recursive function \( t \) there is a constant \( c_t \) with \( 0 < c_t < 1 \) such that for all \( n \) we have \( C^t(\chi_{1:n}|n) \geq c_t n \).

Barzdins actually proved this statement in terms of D.W. Loveland’s version of Kolmogorov complexity [3], which is a slightly different setting. The converse of Item (i) does not hold. To see this, consider a sequence \( \chi = \chi_1\chi_2\ldots \), such that for some constant \( c' \geq 2 \), for every \( n \) we have \( C(\chi_{1:n}|n) \geq n - c' \log n \). By Item (i) \( \chi \) cannot be the characteristic sequence of a r.e. set. Transform \( \chi \) into a new sequence \( \zeta = \chi_1\alpha_1\chi_2\alpha_2\ldots \) with \( \alpha_i = 0^{2^i} \), a string of 0s of length \( 2^i \). While obviously \( \zeta \) cannot be the characteristic sequence of a r.e. set there is a constant \( c \) such that for every \( n \) we have that \( C(\zeta_{1:n}|n) \leq \log n + c \).

Item (i) is easy to prove and Item (ii) is hard to prove. Putting Items (i) and (ii) together, there is a characteristic sequence \( \chi \) of a r.e. set \( A \) whose initial segments are both logarithmic compressible and time-bounded linearly incompressible, for every total recursive time bound.
1.2 Present Results

In contrast to Barzdins, we investigate finite objects (i.e. strings), as opposed to infinite sequences that have to be characteristic sequences of r.e. sets. We are interested in compressible, yet time-bounded incompressible, strings.

**Definition 1** Let $k_0, k_1$ be nonnegative constants and $t$ be a total recursive function. For every $n$, a string $x$ of length $n$ is *logarithmically compressible* if $C(x|n) < k_0 \log n$ and it is *$t$-bounded incompressible* if $C^t(x|n) \geq n - k_1$.

**Theorem 1** (i) Let $t$ be a total recursive function (a time bound). A constant fraction of all logarithmically compressible strings $x$ is $t$-bounded incompressible (Lemma 2).

(ii) Let $t$ be a total recursive function such that $t(n) \geq cn$ for $c > 1$ large enough. For every $n$, a constant fraction of all logarithmically compressible strings $x$ of length $n$ is also $t$-bounded compressible in the sense that $C^t(x|n) < k_0 \log n$ (Lemma 3).

We generalize Item (i) in the corollaries. Section 2 presents preliminaries, Section 3 presents and proves the main results of this paper with an application in Section 4. Finally, conclusions are presented in Section 5.

2 Preliminaries

A (binary) program is a concatenation of instructions, and an instruction is merely a string. Hence, we may view a program as a string. A program and a Turing machine (or machine for short) are used synonymously. The length in bits of a string $x$ is denoted by $|x|$. If $m$ is a natural number, then $|m|$ is the length in bits of the $m$th binary string in length-increasing lexicographic order, starting with the empty string $\epsilon$. We also use the notation $|S|$ to denote the cardinality of a set $S$; context will disambiguate.

Consider a standard enumeration of all Turing machines $T_1, T_2, \ldots$. Let $U$ denote a universal Turing machine such that for every $y \in \{0, 1\}^*$ and $i \geq 1$ we have $U(i, y) = T_i(y)$. That is, for all finite binary strings $y$ and every machine index $i \geq 1$, we have that $U$’s execution on inputs $i$ and $y$ results in the same output as that obtained by executing $T_i$ on input $y$. Fix $U$ and define that $C(x|y)$ equals $\min\{|p| : p \in \{0, 1\}^* \text{ and } U(p, y) = x\}$. For the same fixed $U$, define that $C^t(x|y)$ equals $\min\{|p| : p \in \{0, 1\}^* \text{ and } U(p, y) = x \text{ in } t(|x|) \text{ steps}\}$.

**Definition 2** Let $k_0, k_1$ be nonnegative constants, $TREC$ be the set of total
recursive functions and \( t \in TREC \).

The set \( L_n \) of logarithmically compressible strings of length \( n \) is defined by
\[
L_n = \{ x : C(x|n) \leq k_0 \log n \}.
\]

The set \( LI^t_n \) of logarithmically compressible yet \( t \)-bounded incompressible strings of length \( n \) is defined by
\[
LI^t_n = L_n \cap \{ x : C^t(x|n) \geq n - k_1 \}.
\]

The set \( LL^t_n \) of logarithmically compressible and simultaneously \( t \)-bounded logarithically compressible strings of length \( n \) is defined by
\[
LL^t_n = L_n \cap \{ x : C^t(x|n) \leq k_0 \log n \}.
\]

**Remark 1** By definition, \( LL^t_n \subseteq L_n \). For sufficiently large \( n \) we have \( LI^t_n \subseteq (L_n - LL^t_n) \). We call \( L_n - LL^t_n \) the set logarithmically compressible, yet \( t \)-bounded logarithically incompressible strings of length \( n \).

3 Results

**Lemma 2** For every \( t \in TREC \) there is a positive constant \( c_t \) such that for large enough \( n \) we have \( |LI^t_n| \geq c_t |L_n| \).

**Proof.** The proof is by diagonalization. Let \( k_0, k_1 \) be positive constants and \( t \in TREC \). We use the following algorithm with input \( t, n, k_0, k_1 \) and a natural number \( m \geq 0 \).

**Algorithm** \( A(t, n, k_0, k_1, m) \)

**Step 1.** Using the universal reference Turing machine \( U \), recursively enumerate a finite list of all binary programs \( p_i \) of length \( |p_i| < n - k_1 \). There are at most \( 2^n/2^{k_1} - 1 \) such programs. Execute each of these programs on input \( n \) and discard all programs that do not halt both within \( t(n) \) steps and with output precisely \( n \) bits. This takes time \( O(2^n t(n)/2^{k_1}) \).

**Step 2.** If \( |m| \leq (k_0 - 1) \log n \) then output the \((m + 1)\)th string of length \( n \), say \( x \), in the lexicographic order of all strings of length \( n \) that are not output by any nondiscarded program in Step 1, and halt. If there is no such string or \( |m| > (k_0 - 1) \log n \) then halt with output \( \perp \). **End of Algorithm**

By Step 1, every string \( x \) of length \( n \) that is output by the algorithm has time-bounded complexity
\[
C^t(x|n) \geq n - k_1.
\]

By Step 2, every string \( x \) of length \( n \) that is output by the Algorithm \( A(t, n, k_0, k_1, m) \)
has complexity

\[ C(x|t, n, k_0, k_1, A) \leq |m| \leq (k_0 - 1) \log n. \]

Since \( C(x|t, n, k_0, k_1, A) \geq C(x|n) - O(C(t, k_0, k_1, A)) \), and \( C(t, k_0, k_1, A) = O(1) \) for fixed \( t, k_0, k_1 \), and \( A \), we obtain \( C(x|n) \leq (k_0 - 1) \log n + O(1) \). Since \( (k_0 - 1) \log n + O(1) \leq k_0 \log n \) for large enough \( n \) we have

\[ C(x|n) \leq k_0 \log n, \]

for \( n \) large enough. There are \( n^{k_0} - 1 \) possibilities for \( m \) and at least \( 2^n - 2^n/2^{k_1} + 1 \) strings of length \( n \) to choose from, which implies the requirement

\[ n^{k_0} - 1 \leq 2^n \left( 1 - \frac{1}{2^{k_1}} \right) + 1. \]

For given \( k_0 \) and \( k_1 \), the displayed inequality holds for every large enough \( n \). Hence, \( c_t = 2^{-O(1)} \) suffices.

**Corollary 1** For every \( t \in TREC \) and large enough natural number \( n \) there is a constant \( c_t \) such that \( |L_n - LL^t_n| \geq c_t |L_n| \).

**Proof.** By Lemma 2 (another constant \( c_t \)) and the fact that \( LI^t_n \subseteq (L_n - LL^t_n) \subseteq L \) for \( n \) sufficiently large. \( \square \)

**Corollary 2** We can generalize Lemma 2. Let \( t \in TREC \), and \( f, g, \delta \) be computable positive integer functions such that (1) below is satisfied. Let \( L_n(f) \) denote the set of strings \( x \) such that \( C(x|n) < f(n) \) \( (n = |x|) \) and \( LL^t_n(g) \) be the set of strings \( x \) such that \( C^t(x|n) > g(n) \) \( (n = |x|) \). (So \( L_n = L_n(k_0 \log n) \)) and \( LL^t_n = LL^t_n(n - k_1) \) where \( n = |x| \). For every large enough natural number \( n \) there is a constant \( c_t \) such that \( |LL^t_n(g)| \geq c_t |L_n(f)| \).

**Proof.** Use a similar algorithm \( A(t, n, f, g, m) \) with \( |p_i| < g(n) \) in Step 1, and \( |m| \leq f(n) - \delta(n) \) with \( \delta(n) \to \infty \) as \( n \to \infty \) in Step 2. Require

\[ 2^{f(n) - \delta(n) + 1} - 1 \leq 2^n - 2^{g(n)} + 1 \quad (1) \]

assuming that the description of the function \( \delta \) is a part of algorithm \( A \). \( \square \)

**Lemma 3** For every \( t \in TREC \) with \( t(n) \geq cn \) for some \( c > 1 \) there is a positive constant \( c_t \) such that \( c_t |LL^t_n| \geq |L_n| \).

**Proof.** We use the following algorithm with input \( n, k_0 \) and a natural number \( m \geq 0 \).
Algorithm $B(n, k_0, m)$

For $m$ with $|m| \leq k_0 \log n - c$, with $c$ a large enough constant to make the reasoning below true, output the $(m+1)$th string, say $x$, in the lexicographic order of all strings of length $n$ and halt. **End of Algorithm**

The running time of algorithm $B$ is $t(n) = O(n)$, since the output strings are length $n$ and to output the $m$th string ($m \leq 2^{k_0 \log n - c}$) we simply take the binary representation of $m$ and pad it with nonsignificant 0s to length $n$. Every string $x$ that is output by algorithm $B$ satisfies $C^t(x|n) \leq k_0 \log n - c$. Hence $C^t(x|n) \leq k_0 \log n$ with $c$ the number of bits to contain a description of $B$ and $k_0$ and the means to tell them apart. There are at least $n^{k_0 - 1/2^c}$ such strings while there are at most $n^{k_0}$ strings $x$ with $C(x|n) \leq k_0 \log n$. Setting $c_t = 2^{-c}$ finishes the proof. 

4 Application

It is well known that if we flip a fair coin $n$ times, that is, given $n$ random bits, then we obtain a string $x$ of length $n$ with Kolmogorov complexity $C(x|n) \geq n - c$ with probability at least $1 - 1/2^{n-c}$. Such a string $x$ is algorithmically random. We can also get by with less random bits to obtain resource-bounded algorithmic randomness, as follows.

**Lemma 4** Given the set $L_n$, a function $t \in TREC$, the constant $k_1$ as before, and $O(b \log n)$ fair coin flips, we obtain a set of $O(b)$ strings of length $n$ such that with probability at least $1 - 1/2^b$ one string in this set, say $x$, satisfies $C^t(x|n) \geq n - k_1$.

**Proof.** By Lemma [2] a $c_t$th fraction of the strings in $L_n$ belong to $LI_n$. Therefore, by choosing, uniformly at random, a constant number $a$ of strings from $L_n$ we increase (e.g. by means of a Chernoff bound [2]) the probability that (at least) one of those strings is in $LI_n$ to at least $\frac{1}{2}$. To choose one string from $L_n$ (it is easy to see that $|L_n| = n^{O(1)}$) requires $\tilde{O}(\log n)$ random bits by dividing $L_n$ in two equal size parts and repeating this with the chosen half, and so on. Applying the previous step $b$ times, the probability that at least one of the $ab$ chosen strings is in $LI_n$ is at least $1 - 1/2^b$. 

6
5 Conclusions

We have proved that a constant fraction of compressible strings is time-bounded incompressible, for every recursive time bound, and, another constant fraction of such strings is time-bounded compressible. Alternatively, we could have studied space-bounded incompressibility. It is easily verified that both Lemma 2 and Lemma 3 also hold when the time-bound $t$ is replaced by a space bound $s$ and the time-bounded Kolmogorov complexity, used in Definition 2 is replaced by space-bounded Kolmogorov complexity.

References

