Electronic Colloquium on Computational Complexity, Report No. 89 (2001)

# Logical operations and Kolmogorov complexity II

Andrei A. Muchnik
Institute of New Technologies,
10 Nizhnyaya Radischewskaya, Moscow, Russia 109004

Nikolai K. Vereshchagin*
Dept. of Mathematical Logic and Theory of Algorithms,
Moscow State University, Vorobjevy Gory,
Moscow 119899, Russia; e-mail: ver@mccme.ru

## Abstract

*We investigate Kolmogorov complexity of the problem $(a \to c) \wedge (b \to d)$, defined as the minimum length of a program that given $a$ outputs $c$ and given $b$ outputs $d$. We prove that unlike all known problems of this kind its complexity is not expressible in terms of Kolmogorov complexity of $a$, $b$, $c$, and $d$, their pairs, triples etc. This solves the problem posed in [9].*

*In the second part we consider the following theorem: there are two strings, whose mutual information is large but which have no common information in a strong sense. This theorem was proven in [7] via a non-constructive argument. We present a constructive proof, thus solving a problem posed in [7].*

## 1. Introduction

Kolmogorov complexity $K(x)$ of a binary string $x$ is defined as minimal length of a program that generates this string. This definition can be extended to sets of strings. Let $A$ be a (finite or infinite) set of strings. We define the complexity $K(A)$ as the length of a shortest program that generates some string $x \in A$. Informally, we consider $A$ as a problem "Generate any element of $A$"; $K(A)$ is complexity of this problem. Evidently, $K(A) = \min\{K(x) \mid x \in A\}$, so this generalization gives nothing really new.

However, it can be combined with the definition of logical operations on sets of strings that goes back to Kolmogorov's paper [5] and Kleene's notion of realizability [4]. Let $A$ and $B$ be two sets of strings. We define sets $A \wedge B$, $A \vee B$ and $A \to B$ as follows:

- $A \wedge B = \{\langle a, b\rangle \mid a \in A, b \in B\}$
- $A \vee B = \{\langle 0, a\rangle \mid a \in A\} \cup \{\langle 1, b\rangle \mid b \in B\}$
- $A \to B = \{p \mid [p](x) \in B \text{ for all } x \in A\}$

Here $\langle \cdot, \cdot \rangle$ is a computable encoding of pair of strings; $[p](x)$ stands for the output of $p$ (considered as a program) when applied to input $x$; $[p](x)$ may be undefined for some $x$.

**Example.** Let $a$ and $b$ be two strings. Consider the set $a \to b$ (to simplify notation we identify a string $s$ and the singleton $\{s\}$). This set contains all programs that map $a$ to $b$. It is easy to see that $K(a \to b) = K(b|a) + O(1)$ where $K(b|a)$ denotes complexity of $b$ conditional to $a$. Here are other well known examples.

- $K(a \wedge b)$ is the complexity of $\langle a, b\rangle$, the usual notation for this is $K(a, b)$.
- $K(a \vee b) = \min\{K(a), K(b)\} + O(1)$.
- $K((a \to b) \wedge (b \to a)) = \max\{K(b|a), K(a|b)\} + O(\log K(a, b))$.
  The lower bound $K((a \to b) \wedge (b \to a)) \geq \max\{K(b|a), K(a|b)\} - O(1)$ is obvious, the upper bound was proven in [2].
- $K((a \to c) \wedge (b \to c)) = \max\{K(c|a), K(c|b)\} + O(\log K(a, b, c))$,
  where $K(a, b, c)$ denotes the complexity of $\langle\langle a, b\rangle, c\rangle$. The lower bound here is also evident: $K((a \to c) \wedge (b \to c)) \geq \max\{K(c|a), K(c|b)\} - O(1)$, the upper bound was established in the paper [8].

The complexity of all other problems will be computed also up to an additive $O(\log K(a, b, \dots))$ term.

*The authors were partially supported by Russian Foundation for Basic Research grants 01-01-01028 and 01-01-00505

ISSN 1433-8092

Note that the complexity of any set obtained from singletons $\{a\}$, $\{b\}$, ... does not exceed $K(a,b,\dots)+O(1)$. So this is a sound level of precision. To simplify notation we will neglect additive terms of the order $O(\log K(a,b,\dots))$.

- $K(c\wedge(a\to b))=K(c)+K(b|a,c)$ where $K(b|a,c)$ stands for the conditional complexity $K(b|\langle a,c\rangle)$. A solution to $c\wedge(a\to b)$ is any string of the type $\langle c,p\rangle$, where $p$ is a program mapping $a$ to $b$. The upper bound: Let $q$ denote a minimum length program mapping $\langle a,c\rangle$ to $b$. Then the string $\langle c$, the program pairing its input with $c$ and applying $q$ to the result$\rangle$ belongs to $c\wedge(a\to b)$ and has complexity at most $K(c)+K(q)=K(c)+K(b|a,c)$. The lower bound: Let $p$ be a program mapping $a$ to $b$ and having minimum $K(p,c)$. We need to prove that $K(p,c)\geq K(c)+K(b|a,c)$. Obviosly, $K(b|a,c)\leq K(p|c)$ (any program $q$ mapping $c$ to $p$ can be transformed to a program mapping $\langle a,c\rangle$ to $b$: apply $q$ to $c$ to find $p$, then apply $p$ to $a$ to find $b$). It remains to use the equality $K(p|c)=K(p,c)-K(c)$.

- $K((a\to b)\wedge(b\to c))=\max\{K(b,c|a),K(c|b)\}$. Any solution $\langle p,q\rangle$ to this problem may be transformed to a solution of the problem $a\to(b\wedge c)$, as $c=[q]([p](a))$. Therefore this problem is equivalent to the problem $(a\to(b\wedge c))\wedge(b\to c)$. (Two problems are called equivalent if there is an algorithm that for all $a,b,c$ given any solution to the first problem computes a solution to the second one, and vice versa. Obviosly, equivalent problems have complexity differing by $O(1)$.) The latter problem is equivalent to the problem $(a\to(b\wedge c))\wedge(b\to(b\wedge c))$. Using the previous item we see that its complexity is equal to $\max\{K(b,c|a),K(b,c|b)\}$.

- $K(a\to(b\to c))=K(c|a,b)$.

- $K((a\to b)\to c)=\min\{K(c),K(a)+K(c|a,b)\}$. The upper bound is obvious. To prove the lower bound we will use the method of the paper [9] (where the weaker assertion $K((a\to b)\to c)\geq\min\{K(c),K(a)\}$ was proven). Let $n$ denote $\max\{K(a),K(b),K(c)\}$ and $P$ the set of all strings of length at most $n$. Given $n$ and any string $x$ of complexity $n$ or less search for a program in $P$ to print $x$ and let $\bar{x}$ denote the first found program.

  Let $s$ be a program in $(a\to b)\to c$. For any function $\tau\colon P\to P$ fix some program $l_\tau$ that given a string $x$ of complexity at most $n$ finds $\bar{x}$ and then applies the program $\tau(\bar{x})$ to the empty input. Thus $l_\tau(u)=v$ whenever $\tau(\bar{u})=\bar{v}$. A triple of strings $(u,v,w)$ is called *$s$-coherent* if $K(u)\leq n$, $K(v)\leq n$ and $[s](l_\tau)=w$ for any $\tau$ such that $\tau(\bar{u})=\bar{v}$. By definition the triple $(a,b,c)$ is $s$-coherent and given $n$ and $s$ we can enumerate all $s$-coherent triples. Let $(u,v,w)$ be the first enumerated $s$-coherent pair. Assume first that $u\neq a$. Choose any function $\tau\colon P\to P$ such that $\tau(\bar{a})=\bar{b}$ and $\tau(\bar{u})=\bar{v}$. As both triples $(a,b,c)$ and $(u,v,w)$ are $s$-coherent we have $c=w=s[l_\tau]$ and $K(c)\leq K(u,v,w)\leq K(s)+O(\log n)$.

  Assume that $u=a$. In this case we will prove that $K(a)+K(c|a,b)\leq K(s)+O(\log n)$. Given $s,n$ we can find $a$ as the first component of the first enumerated $s$-coherent triple. Moreover given $s,b,n$ we can find $c$: find $a$ and apply $s$ to any program mapping $a$ to $b$. Thus the pair $(s,n)$ gives a solution to the problem $a\wedge(b\to c)$ and we know that the complexity of this problem is $K(a)+K(c|a,b)+O(\log n)$.

In all above examples, it possible to express the complexity of the problem under consideration in terms of the complexity of involved strings, their pairs, triples etc. up to an additive logarithmic term. (In some of them we used conditional complexity, but it is possible to avoid it by means of the equality $K(x|y)=K(x,y)-K(y)$.) In this context the following notion is useful. The *complexity vector* of a tuple of stings $\langle a_1,\dots,a_k\rangle$ is the vector of length $2^k-1$ consisting of complexity of $a_1,\dots,a_k$, their pairs, triples etc. For instance, the complexity vector of the triple $\langle a,b,c\rangle$ is defined as $\langle K(a),K(b),K(c),K(a,b),K(a,c),K(b,c),K(a,b,c)\rangle$.

The following natural question arises. Substitute singletons $\{a_1\},\dots,\{a_k\}$ for variables in a propositional formula $A(p_1,\dots,p_k)$ with connectivities $\vee,\wedge,\to$ and denote the resulting problem by $A(a_1,\dots,a_k)$. Is it true that for any formula $A(p_1,\dots,p_k)$ the complexity of $A(a_1,\dots,a_k)$ is determined up to an additive $O(\log K(a_1,\dots,a_k))$ term by the complexity vector of the tuple $\langle a_1,\dots,a_k\rangle$? (We realize that this formulation is rather vague, as the notion "determined up to an additive term" is not rigorously defined. This notion can be made rigorous in several ways, but we will not do this being sure that the intuitive meaning suffices for the results presented below.) This question was posed by A. Shen at the Workshop on Algorithmic Information Theory in Nancy in 1999 (see [9]). In the present paper, we prove that the answer is negative already for the formula $(u\to v)\wedge(r\to s)$. A solution to problem $(a\to c)\wedge(b\to d)$ is a pair $\langle p,q\rangle$ of programs such that $p$ maps $a$ to $c$, and $q$ maps $b$ to $d$. We will construct two sequences of quadruples of strings $\tilde{a}_n, \tilde{b}_n, \tilde{c}_n, \tilde{d}_n$ and $\bar{a}_n, \bar{b}_n, \bar{c}_n, \bar{d}_n$ having complexity linear in $n$, whose complexity vectors differ by only $O(1)$ (in the corresponding components), but the difference $K((\tilde{a}_n \to \tilde{c}_n) \wedge (\tilde{b}_n \to \tilde{d}_n)) - K((\bar{a}_n \to \bar{c}_n) \wedge (\bar{b}_n \to \bar{d}_n))$ has linear growth. This implies that the complexity of the problem $(a \to c) \wedge (b \to d)$ is not determined by the complexity vector of $\langle a, b, c, d\rangle$ even up to an additive $o(K(a, b, c, d))$ term.

**Acknowledgements.** The authors are sincerely grateful to A. Shen for bringing their attention to the topic.

## 2. Preliminaries

A *programming language* is a partial computable function $F$ from $\{0,1\}^* \times \{0,1\}^*$ to $\{0,1\}^*$. The first argument of $F$ is a program, the second argument is called the input, and $F(p, x)$ is called the output of program $p$ on input $x$. A programming language $U$ is called *universal* if for any other programming language $F$ there exists a constant $t_F$ such that $U(t_F p, x) = F(p, x)$ for any $p, x$. By Solomonoff — Kolmogorov theorem (see e.g. [6]) universal programming languages exist. We fix some universal programming language $U$ and define $K(x|y)$ as the minimum length of $p$ such that $U(p, y) = x$ (Kolmogorov complexity of $x$ conditional to $y$), $K(x) = K(x|\text{empty string})$ (Kolmogorov complexity of $x$). We use $[p](x)$ to abbreviate $U(p, x)$.

We will use the following two well known facts that are easy to prove: For any algorithm $A$ there is a constant $c$ such that $K(A(x)) \le K(x) + c$ for all $x$ in the domain of $A$. For any $n$ there are less than $2^n$ strings $x$ with $K(x) < n$.

We will use also the equality $|K(x, y) - K(x) - K(y|x)| = O(\log K(x, y))$ proven by Kolmogorov and Levin (see [6]).

By $\log n$ we denote $\log_2 n$. When we talk about a "random string" in a set $S$ we mean the following. We assume that some non-negative integer constant $\alpha$ is fixed. A *random string in a set* $S$ is any string $x \in S$ such that $K(x) \ge \lfloor \log |S| \rfloor - \alpha$. As there are less than $2^k$ strings whose complexity is less than $k$, any non-empty set has a random string (for any $\alpha$). As $\alpha$ increases the weaker is the notion of randomness.

The following well known principle allows to prove that a string has low complexity. Assume that we have an enumerable family $V_n$ of sets of strings. (The enumerability of $V_n$ means that the set $\{\langle n, x\rangle \mid x \in V_n\}$ is enumerable.) Assume that $|V_n| < 2^n$ for all $n$. Then there is a constant $\beta$ such that $K(x) < n + \beta$ for all $n$ and all $x \in V_n$.

Assume that we are given a sequence $S_n$ of sets and $|S_n| = 2^{\beta n + O(1)}$ where $\beta$ is a rational constant. To prove that all random strings in $S_n$ satisfy some property we will use the following argument. Assume that we have an enumerable property $P_n$ of elements of $S_n$ such that the fraction of $s \in S_n$ satisfying $P_n(s)$ is $O(1/n)$. Then for all but finitely many $n$ all random elements in $S$ satisfy $P_n$. This is an easy consequence of the above principle.

## 3. The complexity of the problem $(a \to c) \wedge (b \to d)$

The best bounds of the complexity of the problem $(a \to c) \wedge (b \to d)$ known to the authors are presented in the following theorem.

**Theorem 1.**

$$K((a \to c) \wedge (b \to d)) \le \min\{K(c|a)+K(d|b), K(c)+K(d|b, c), K(d)+K(c|a, d)\},$$
$$K((a \to c) \wedge (b \to d)) \ge \max\{K(b, c, d|a)-K(b|a, c), K(a, c, d|b)-K(a|b, d)\}.$$

*Proof.* By definition a solution to this problem is a pair $\langle p, q\rangle$ of programs such that $p$ maps $a$ to $c$, and $q$ maps $b$ to $d$. Taking as $p$ and $q$ the shortest such programs we get the upper bound $K(c|a)+K(d|b)$ for its complexity. Letting $p$ be equal to the program mapping all inputs to $c$ and $q$ to the program pairing its input with $c$ and applying to the resulting pair a minimum length program mapping $\langle b, c\rangle$ to $d$, we obtain the upper bound $K(c) + K(d|b, c)$. The upper bound $K(d) + K(c|a, d)$ is proven in a similar way.

To prove the lower bound let $\langle p, q\rangle$ be any solution to $(a \to c) \wedge (b \to d)$. Consider the triple $\langle p, q, r\rangle$, where $r$ is a shortest program mapping $\langle a, c\rangle$ to $b$. Given this triple and $a$, we can find $\langle b, c, d\rangle$: apply $p$ to $a$ to get $c$, then apply $r$ to $\langle a, c\rangle$ to get $b$, finally apply $q$ to $b$ to get $d$. Therefore

$$K(b, c, d|a) \le K(p, q) + K(r) = K(p, q) + K(b|a, c),$$

hence $K(b, c, d|a) - K(b|a, c) \le K(p, q)$. The bound $K(a, c, d|b) - K(a|b, d) \le K(p, q)$ is proven in a similar way. □

If, for instance, $a, b, c, d$ are obtained by cutting a random binary string of length $4n$ in four blocks of length $n$ then both the lower and upper bounds in the above theorem are equal to $2n$, hence the complexity of our problem is $2n$. It is easy to find strings $a, b, c, d$ for which the lower bound is less then the upper bound and the complexity of our problem is equal to the lower bound. Take, for example, a random string of length $2n$, divide it in two blocks $x, y$ of length $n$ and let $a = d = x$, $b = c = y$. For these $a, b, c, d$ the lower bound is equal to $n$, and the upper bound to $2n$. The complexity of the problem $(a \to c) \wedge (b \to d)$ is equal to $n$, as we can take $p = q = x \oplus y$ (the sign $\oplus$ refers to bitwise addition modulo 2).

It is much harder to find strings $a, b, c, d$ for which $K((a \to c) \wedge (b \to d))$ is greater than the lower bound of theorem 1. We start by giving such example. Then we will find another quadruple of strings having the same complexity vector for which the complexity of the problem $(a \to c) \wedge (b \to d)$ is equal to the lower bound of theorem 1. This will prove that the complexity of the problem $(a \to c) \wedge (b \to d)$ is not determined by the complexity vector of $a, b, c, d$.

It is difficult to find such $a, b, c, d$ because the method applied in the former example does not work. Indeed one can prove that whenever $a, b, c, d$ are built form the constant number of blocks of the same random string then $K((a \to c) \wedge (b \to d))$ is equal to the lower bound of theorem 1. We will use the method from [7], based on linear algebra over finite fields.

Let $F_n$ denote the field of cardinality $2^n$. We will consider points, lines and planes in the three-dimensional affine space over $F_n$. There are $2^{3n}$ points, $2^{4n+o(1)}$ lines (the exact number of lines is $\frac{2^{3n}(2^{3n}-1)}{2^n(2^n-1)}$) and $2^{3n+o(1)}$ planes in this space. As $\langle \tilde{a}, \tilde{b} \rangle$ we take any random pair of different intersecting lines, $\tilde{c}$ will be its common point and $\tilde{d}$ its common plane. It is easy to see that the lower bound of theorem 1 is equal to $n$ in this case.

**Theorem 2.** $K((\tilde{a} \to \tilde{c}) \wedge (\tilde{b} \to \tilde{d})) \geq 1.5n - O(\log n)$.

*Proof.* Let $r$ be a solution to $(\tilde{a} \to \tilde{c}) \wedge (\tilde{b} \to \tilde{d})$, that is, $r = \langle p, q \rangle$ where $p$ maps $\tilde{a}$ to $\tilde{c}$ and $q$ maps $\tilde{b}$ to $\tilde{d}$. Consider the set $S$, consisting of all pairs of different intersecting lines $\langle a, b \rangle$ such that $p$ maps $a$ to the common point of $a$ and $b$, and $q$ maps $b$ to the common plane of $a$ and $b$. Given $p$, $q$, and $n$ we can generate all elements of $S$. As the pair $\langle \tilde{a}, \tilde{b} \rangle$ belongs to $S$, we conclude that

$$7n - O(1) \leq K(\tilde{a}, \tilde{b}) \leq K(r) + \log |S| + 2 \log \log |S| + O(1).$$

Thus it suffices to prove that the cardinality of $S$ does not exceed $2^{5.5n+O(1)}$. This is a direct corollary of the next lemma.

**Lemma 1.** *Let $f$ be a function mapping every line to a point on that line, and $g$ a function that maps every line to a plane having this line. Let $S$ consist of all pairs $\langle a, b \rangle$ such that $f(a) \in b$ and $a \subset g(b)$. Then $|S| \leq 2^{5.5n+O(1)}$.*

*Proof.* Let us see first what bound can be proven by easy arguments. For any line $b$ there are at most $2^{2n+o(1)}$ lines $a$ in the plane $g(b)$, hence the cardinality of $S$ is at most $2^{2n+o(1)}$ times bigger than the number of lines ($2^{4n+o(1)}$); this gives the bound $|S| \leq 2^{6n+o(1)}$. The same bound can be proven by counting for every $a$ the number of lines $b$ passing through $f(a)$. Note that in the first argument we did not use the fact that the line $b$ must pass through $f(a)$, and in the second one that the line $a$ must lie in the plane $g(b)$. Our plan now is as follows: we modify the first argument to show that the average number of pairs $\langle a, b \rangle$ in $S$ having the same second component $b$ is at most $2^{1.5n+O(1)}$. In that argument we will take into account the condition $f(a) \in b$. (We could argue in a symmetrical way to show that the average number of pairs $\langle a, b \rangle$ in $S$ having the same first component $a$ is at most $2^{1.5n+O(1)}$.)

Partition $S$ into slices, every slice consists of all pair $\langle a, b \rangle \in S$ with the same value $d = g(b)$. We will upperbound the number of pairs in every slice and then we will sum the obtained bounds. So, fix a plane $d$ and upperbound the number of $\langle a, b \rangle$ such that $f(a) \in b$ and $a \subset g(b) = d$. To this end consider any point $c \in d$ and denote by $A_c$ the set of all lines $a$ on the plane $d$ for which $f(a) = c$ and by $B_c$ the set of all lines $b$ passing through $c$ for which $g(b) = d$ (thus $a$ passes through $c$ and $b$ lies on $d$). The number of pairs $\langle a, b \rangle$ satisfying $f(a) \in b$ and $a \subset g(b) = d$ does not exceed

$$\sum_{c \in d} |A_c||B_c| \leq \sqrt{\sum_{c \in d} |A_c|^2 \sum_{c \in d} |B_c|^2}.$$

Both sums in the right hand side have a clear interpretation. Indeed, $\sum_c |A_c|^2$ determines the probability of the following event: two lines $a', a''$ in the plane $d$ chosen at random satisfy $f(a') = f(a'')$. More specifically, let $N$ denote the total number of lines on the plane $d$. Then

$$\begin{aligned} \mathrm{Prob}[f(a') = f(a'')] &= \sum_c \mathrm{Prob}[f(a') = f(a'') = c] \\ &= \sum_c \mathrm{Prob}[f(a') = c]\, \mathrm{Prob}[f(a'') = c] \\ &= \sum_c |A_c|^2 / N^2. \end{aligned}$$

For any fixed $a'$ the probability of event $f(a') = f(a'')$ does not exceed the probability that the line $a''$ passes through the point $f(a')$. The latter probability is equal to $2^{-n+o(1)}$, hence

$$\operatorname{Prob}[f(a') = f(a'')] \le 2^{-n+o(1)} \implies \sum_c |A_c|^2 \le N^2 2^{-n+o(1)} = 2^{3n+o(1)}.$$

The other sum $\sum_c |B_c|^2$ determines the average number of common points between two lines $b', b''$ chosen at random in the set $M_d$, consisting of those lines $b$ for which $g(b) = d$ (all they lie in the plane $d$). More specifically,

$$\operatorname{E}|b' \cap b''| = \sum_c \operatorname{Prob}[c \in b' \cap b''] = \sum_c \operatorname{Prob}[c \in b'] \operatorname{Prob}[c \in b''] = \sum_c |B_c|^2/|M_d|^2.$$

Any two distinct lines have at most 1 common point, and equal lines have $2^n$ common points, therefore

$$\operatorname{E}|b' \cap b''| \le 1 + 2^n \operatorname{Prob}[b' = b''] = 1 + 2^n/|M_d|,$$

hence

$$\sum_c |B_c|^2 = \operatorname{E}|b' \cap b''||M_d|^2 \le |M_d|^2 + |M_d|2^n.$$

Recall that the number of those $\langle a, b\rangle$ for which $f(a) \in b$ and $a \subset g(b) = d$ does not exceed $\sqrt{\sum_c |A_c|^2 \sum_c |B_c|^2}$, therefore it does not exceed

$$\sqrt{2^{3n+o(1)}(|M_d|^2 + |M_d|2^n)} \le 2^{1.5n+o(1)}(|M_d| + 2^n)$$

(the latter inequality is proven by mere squaring). It remains to to sum the obtained bounds over $d$:

$$|S| \le 2^{1.5n+o(1)} \sum_d (|M_d| + 2^n).$$

The families of lines $M_d$ form a partition of the set of all lines, therefore the sum of their cardinalities is equal to $2^{4n+o(1)}$. The number of different $d$ is equal to $2^{3n+o(1)}$, thus the sum over all $d$ of $2^n$ is also equal to $2^{4n+o(1)}$. Hence

$$|S| \le 2^{1.5n+o(1)}(2^{4n+o(1)} + 2^{4n+o(1)}) = 2^{5.5n+1+o(1)}. \quad \square$$

Let us define now another quadruple $\langle \bar a, \bar b, \bar c, \bar d\rangle$ having the same complexity vector as $\langle \tilde a, \tilde b, \tilde c, \tilde d\rangle$. To this end take a random binary string of length $7n$ and cut it into 7 blocks $u, v, w, p, q, r, s$ of length $n$. Let $\bar a = uvws$, $\bar b = pqrs$, $\bar c = ups$, $\bar d = vqs$. It is easy to see that both quadruples have the same complexity vector:

$$K(a) = K(b) = 4n, \quad K(c) = K(d) = 3n,$$
$$K(a,b) = 7n,$$
$$K(a,c) = K(a,d) = K(b,c) = K(b,d) = K(c,d) = 5n,$$
$$K(a,c,d) = K(b,c,d) = 6n,$$
$$K(a,b,c) = K(a,b,d) = K(a,b,c,d) = 7n.$$

And $K((\bar a \to \bar c) \wedge (\bar b \to \bar d)) = n$, as given $p \oplus v$ one can map $\bar a$ to $\bar c$, and $\bar b$ to $\bar d$. Recall that the above theorem states that $K((\tilde a \to \tilde c) \wedge (\tilde b \to \tilde d)) \ge 1.5n$.

Note that the formula $(p \to q) \wedge (r \to s)$, like all the formulas in the above examples, has depth 2. There is another problem of depth 2 whose complexity is not expressible in terms of the complexity vector of involved strings, namely $(p \vee q) \to (r \vee s)$. A solution to this problem is a program that given any of two strings $p, q$ computes any of strings $r, s$. Make in this formula the following substitution $p = a$, $q = b$, $r = \langle a, c\rangle$, $s = \langle b, d\rangle$ where $\langle a, b, c, d\rangle$ is one of the two quadruples constructed above. As given $a$ it is much easier to compute $\langle a, c\rangle$ than $\langle b, d\rangle$, and given $b$ it is much easier to compute $\langle b, d\rangle$ than $\langle a, c\rangle$, it is easy to show that the complexity of the resulting problem is the same as the that of $(a \to \langle a, c\rangle) \wedge (b \to \langle b, d\rangle)$. The latter problem is equivalent to the problem $(a \to c) \wedge (b \to d)$. And we know that the complexity of this problem is different for two above constructed quadruples.

One may prove by exhaustion that the complexity of all other problems of depth 2 is determined by the complexity vector of involved strings.

We have a gap of $0.5n$ between obtained lower bound of $K((a \to c) \wedge (b \to d))$ and upper bound of theorem 1. Is it possible to find $a, b, c, d$ for which $K((a \to c) \wedge (b \to d))$ is equal to the upper bound of theorem 1 and this is bigger than the lower bound of that theorem? The following theorem answers this question in positive.

**Theorem 3.** *For all $n$ there are strings $\tilde a, \tilde b, \tilde c, \tilde d$ of complexity $n$ such that the complexity of all pairs $\langle \tilde a, \tilde b\rangle$, $\langle \tilde a, \tilde c\rangle$, ..., $\langle \tilde c, \tilde d\rangle$ is equal to $2n$, the complexity of all triples $\langle \tilde a, \tilde b, \tilde c\rangle$, $\langle \tilde a, \tilde b, \tilde d\rangle$, ... is equal to $3n$, the complexity of the quadruple $\langle \tilde a, \tilde b, \tilde c, \tilde d\rangle$ is also equal to $3n$, and the complexity of the problem $(a \to c) \wedge (b \to d)$ is equal to $2n$ (all the equalities hold up an additive $O(\log n)$ term).*

It is easy to verify that for such quadruple the lower and upper bounds of theorem 1 are equal to $n$ and $2n$, respectively. Thus we indeed get an example we looked for.

*Proof.* Fix $n$. Call a set $S$ of strings of length $n$ *uniform* if for every triple $\langle a, b, c\rangle$ of strings of length $n$ there is unique $d$ such that $\langle a, b, c, d\rangle \in S$. We will define a uniform set $S$ that can be effectively found given $n$. As $\langle \tilde a, \tilde b, \tilde c, \tilde d\rangle$ we will take a random quadruple in $S$. This will imply $K(\tilde a, \tilde b, \tilde c, \tilde d) = K(\tilde a, \tilde b, \tilde c) = 3n$, and, as a corollary, $K(\tilde a) = K(\tilde b) = K(\tilde c) = n$, $K(\tilde a, \tilde b) = K(\tilde a, \tilde c) = K(\tilde b, \tilde c) = 2n$. To obtain the inequality $K((\tilde a \to \tilde c) \wedge (\tilde b \to \tilde d)) \ge 2n$, we define $S$ so that this inequality be true for most quadruples $\langle a, b, c, d\rangle$

in $S$ (more specifically, the fraction of quadruples that do not satisfy the inequality will be $O(1/n)$). Then this inequality will be true for any random quadruple in $S$ (for large enough $n$). To satisfy the remaining requirements on the complexity vector of $\langle\tilde a,\tilde b,\tilde c,\tilde d\rangle$ it suffices to ensure that both triples $\langle\tilde a,\tilde b,\tilde d\rangle$ and $\langle\tilde a,\tilde c,\tilde d\rangle$ have also complexity $3n$. Note that we need not to care that $K(\langle\tilde b,\tilde c,\tilde d\rangle)=3n$, since this is implied by the inequality $K((\tilde a\to\tilde c)\wedge(\tilde b\to\tilde d))\ge 2n$. Indeed, we have

$$\begin{aligned}2n &\le K((\tilde a\to\tilde c)\wedge(\tilde b\to\tilde d))\\ &\le K(\tilde d|\tilde b,\tilde c)+K(\tilde c)=K(\tilde d|\tilde b,\tilde c)+n\le 2n,\end{aligned}$$

hence

$$\begin{aligned}&K(\tilde d|\tilde b,\tilde c)=n\Longrightarrow\\ &\quad K(\tilde b,\tilde c,\tilde d)=K(\tilde b,\tilde c)+K(\tilde d|\tilde b,\tilde c)=2n+n=3n.\end{aligned}$$

In the same way we will ensure that $K(\langle\tilde a,\tilde b,\tilde d\rangle)=K(\langle\tilde a,\tilde c,\tilde d\rangle)=3n$. Namely, we will construct $S$ so that the complexity of both problems $(c\to b)\wedge(a\to d)$ and $(b\to a)\wedge(c\to d)$ be also at least $2n$ for the majority of quadruples in $S$. This implies via a symmetrical argument that both triples $\langle\tilde a,\tilde b,\tilde d\rangle$ and $\langle\tilde a,\tilde c,\tilde d\rangle$ are random for any random quadruple $\langle\tilde a,\tilde b,\tilde c,\tilde d\rangle$ in $S$ (for large enough $n$).

So let $k=2n-2\log n-\alpha$ where the constant $\alpha$ is to be specified later. It suffices given $n$ to find a uniform set $S$ such that neither of the three inequalities

$$\begin{aligned}K((a\to c)\wedge(b\to d))&\ge k, \qquad (1)\\ K((c\to b)\wedge(a\to d))&\ge k,\\ K((b\to a)\wedge(c\to d))&\ge k\end{aligned}$$

holds for more than $O(2^{3n}/n)$ quadruples in $S$.

The inequality (1) means that there is no $\langle p,q\rangle$ of complexity less than $k$ such that $[p](a)=c$, $[q](b)=d$. There are less than $2^k$ such pairs. Thus it suffices to prove the following

**Lemma 2.** *There are constant $\alpha,\beta$ such that for all $n$ there is a uniform set $S$ such that the following holds. For any set $M$ consisting of at most $2^k$ functions from $\{0,1\}^n$ to $\{0,1\}^n$ neither of the sets*

$$\begin{aligned}&\{\langle a,b,c,d\rangle\in S\mid(\exists\langle f,g\rangle\in M)\ f(a)=c,\ g(b)=d\},\\ &\{\langle a,b,c,d\rangle\in S\mid(\exists\langle f,g\rangle\in M)\ f(c)=b,\ g(a)=d\},\\ &\{\langle a,b,c,d\rangle\in S\mid(\exists\langle f,g\rangle\in M)\ f(b)=a,\ g(c)=d\}\end{aligned}$$

*has more than $(\beta/n)2^{3n}$ quadruples from $S$.*

*Proof.* We will prove that with high probability a uniform set $S$ chosen at random satisfies the statement of the lemma. As the three above sets are symmetrical, it suffices to show that with probability close to 1 the first of them has at most $O(2^{3n}/n)$ elements of $S$ for every $M$.

Say that $M$ *serves* $\langle a,b,c,d\rangle$ if $f(a)=c$ and $g(b)=d$ for some pair $\langle f,g\rangle\in M$. Fix $M$ and upperbound the probability that $M$ serves more than $O(2^{3n}/n)$ of quadruples in $S$. Call a triple $\langle a,b,c\rangle$ *bad* if $f(a)=c$ for more than $n2^{-n}|M|$ pairs $\langle f,g\rangle\in M$ (this property does not depend on $b$). Otherwise call the triple *good*. The probability of event $f(a)=c$ when $\langle a,b,c\rangle$ and $\langle f,g\rangle\in M$ are chosen at random is equal to $2^{-n}$. Hence the fraction of bad triples is less than $1/n$, and the fraction of good triples is greater than $(1-1/n)$.

For any good triple $\langle a,b,c\rangle$ the probability that $M$ serves $\langle a,b,c,d\rangle$ for $d$ chosen at random is at most $O(1/n)$. Indeed, if $M$ serves $\langle a,b,c,d\rangle$ then $d$ belongs to the set $\{g(b)\mid\langle f,g\rangle\in M,\ f(a)=c\}$. As $\langle a,b,c\rangle$ is good, this set has less than $n2^{-n}|M|=2^{n-\log n+O(1)}$ elements.

We will use now the well known Chernoff bound [3]: if in each single trial the probability of success is $p$ (or less), then for every $0<\varepsilon\le p(1-p)$ the probability that the number of successful trials in a sequence of $N$ independent trials is greater than $(p+\varepsilon)N$ is less than $2^{-\varepsilon^2N/(2p)}$. We will use a simple corollary of this bound (obtained by letting $\varepsilon=p/2$): the probability that the number of successful trials is greater than $3pN/2$ is less than $2^{-pN/8}$. In our case trials correspond to good triples, thus $2^{3n-O(1)}\le N\le 2^{3n}$. In trial $\langle a,b,c\rangle$ we choose $d$ at random and the trial is successful if $M$ serves $\langle a,b,c,d\rangle$. So we have $p=O(1/n)$. Therefore, with probability at least $1-2^{-\Omega(N/n)}$ for at most $O(N/n)$ good $\langle a,b,c\rangle$ the triple $\langle a,b,c,d\rangle$ in $S$ is served by $M$. Hence the number of served quadruples in $S$ is less than the number of served good quadruples $O(N/n)=O(2^{3n}/n)$ plus the number of bad quadruples $2^{3n}/n$ with probability at least $1-2^{-\Omega(2^{3n}/n)}$ (the quadruple $\langle a,b,c,d\rangle$ is called good if the triple $\langle a,b,c\rangle$ is good).

The number of different $M$'s is at most

$$((2^n\cdot 2^n)^{2^n})^{2^{2n-2\log n-\alpha}}=2^{2^{3n-\alpha+1}/n}.$$

Hence with probability at least

$$1-2^{2^{3n-\alpha+1}/n}.2^{-2^{3n-O(1)}/n}=1-2^{-(2^{3n}/n)(2^{-O(1)}-2^{-\alpha+1})}$$

every $M$ serves at most $O(2^{3n}/n)$ of quadruples in $S$. For $\alpha$ large enough this probability tends to 1 as $n$ tends to infinity. ☐

The requirements on $S$ in the lemma are decidable. Therefore given $n$ we can find by brute force the first set $S$ satisfying the lemma. As explained above any random quadruple from $S$ satisfies the statement of the theorem. $\square$

There is another quadruple with the same complexity vector as in the theorem but for which $K((a \to c) \wedge (b \to d)) = n$. Namely, consider a random binary string of length $3n$ and cut it into three parts of length $n$ to obtain $a$, $b$ and $c$; then let $d = a \oplus b \oplus c$. If we know $a \oplus c$ then given $a$ we can find $c$, and given $b$ we can find $d$. Hence $K((a \to c) \wedge (b \to d)) = n$. Thus we have a new proof of the fact that the complexity of the problem $(a \to c) \wedge (b \to d)$ is not determined by complexity vector of $a, b, c, d$.

It is instructive to compare the new proof with the old one. In both proofs we show that there is a quadruple having some specific properties, namely, a quadruple having a given complexity vector and a given lower bound for $(a \to b) \wedge (c \to d)$. The lower bound is stronger in the second proof, however the proof itself is less constructive. What do we mean by that? In both proofs given $n$ we effectively find a set $S$, and then take an arbitrary random element in $S$. The important difference is that in the first proof the set $S$ is explicitly presented, and in the second one not. The proofs of the first type are called *effective*, the ones of the second type *quasi-effective* (we consider only proofs of the existence of an object with specific properties). In both cases there is a probabilistic algorithm that given $n$ with probability close to 1 computes an object having the desired property. (Note that such algorithm cannot be deterministic, as in that case the complexity of outputted object would be $\log n + O(1)$.) But in the first case the algorithm runs in polynomial time in $n$ (addition, multiplication and division in the field $F_n$ can be performed in polynomial time), and in the second case we do not know any efficient algorithm. It would be interesting to find out whether a set $S$ satisfying lemma 2 exists such that there is a polynomial time algorithm that given $a, b, c$ finds $d$ for which $\langle a, b, c, d\rangle \in S$.

Effective and quasi-effective proofs are opposed to non-effective ones—those in which we do not construct any algorithm to find an object with desired properties. Usually such proof is easier to find than a quasi-effective one. The present paper is an exception: we do not know an easier proof of the existence of $a, b, c, d$ for which the complexity of the problem $(a \to c) \wedge (b \to d)$ is greater than the lower bound of theorem 1.

In conclusion we present another example of a theorem for which there is a quasi-effective proof that is at least as easy as the known non-effective ones.

## 4. Constructing strings having large amount of mutual information but having no common information

It is well known that there are strings $a, b$ such that the mutual information $I(a : b) \rightleftharpoons K(a) + K(b) - K(a, b) = K(a) - K(a|b) = K(b) - K(b|a)$ of $a, b$ is large, but $a, b$ have no common information in the following sense: any $c$ with small $K(c|a)$ and $K(c|b)$ has small complexity. (A string $c$ may be regarded as a piece of common information of $a$ and $b$ if it can be computed both from $a$ and $b$.) More specifically, there are sequences $a_n, b_n$ such that $K(a_n) = K(b_n) = O(n)$, $I(a_n : b_n) = \Omega(n)$, but for any sequence $c_n$ such that $K(c_n|a_n) = K(c_n|b_n) = o(n)$ we have $K(c_n) = o(n)$. Such $a_n, b_n$ may be obtained as follows. The string $a_n$ has length $n$ and is a result of $n$ independent trials of a random variable $\xi$ in $\{0, 1\}$; the string $b_n$ is obtained in the same way from another binary variable $\eta$. As $\xi, eta$ on may choose any dependent random variables such that the outcome of $\xi$ does not determine the outcome of $\eta$ and vice versa. This is an easy corollary of a theorem on Shannon entropy proven by Ahlswede and Körner in [1].

Another effective proof of the existence of strings having large mutual information but having no common information was given in [7]. In the latter paper it was also pointed out that the words "$a, b$ have no common information" may be understood in a stronger sense. Let us present the relevant definition from [7].

The main idea is that one could consider a string $c$ to have a piece of common information of $a$ and $b$ also if $K(c)$ is smaller than $I(c : a) + I(c : b)$. In other words, if $K(c) + K(a|c) + K(b|c) < K(a) + K(b)$. However, this definition makes no sense, as this is true for $c = \langle a, b\rangle$ provided $I(a : b) > 0$. So we need to modify it (at least requiring that $K(c) < K(a, b)$). To be more precise consider two sequences $a_n, b_n$ whose complexities have linear growth. Consider the set of all those triples $\langle u, v, w\rangle$ of reals for which for all $n$ there is a string $c_n$ such that $K(c_n) \le un + O(\log n)$, $K(a_n|c_n) \le vn + O(\log n)$ and $K(b_n|c_n) \le wn + O(\log n)$. This set is called the *complexity profile* of $a$ and $b$. For instance, if $a$ and $b$ are random strings of length $2n$ having a common substring of length $n$ then this substring may be taken as $c$ hence the profile will contain the triple $\langle 1, 1, 1\rangle$.

As the profile of $a, b$ decreases the less common common information have $a$ and $b$. To simplify the presentation let us restrict ourselves (as it is done in [7]) to the case $K(a_n) = K(b_n) = 2n$, $K(a_n, b_n) = 3n$. What is the minimum complexity profile? It consists of all triples $\langle u, v, w\rangle$ satisfying the inequalities

$$u+v \geq 2, \quad u+w \geq 2, \quad u+v+w \geq 3$$

and at least one of the inequalities

$$u+v \geq 3, \quad u+w \geq 3, \quad u+v+w \geq 4.$$

Let $M_{\min}$ denote this set. It is easy to verify that the profile of any $a, b$ such that $K(a) = K(b) = 2n$, $K(a, b) = 3n$ includes the set $M_{\min}$. To show this it suffices to consider only strings $c$ consisting of some substrings of the shortest programs to print $a$ and $b$ and of the shortest programs mapping $a$ to $b$ and $b$ to $a$. (For detailed proof see [7].)

**Theorem 4** ( [7]). *There are sequences $a_n, b_n$ such that $K(a_n) = K(b_n) = 2n$, $K(a_n, b_n) = 3n$ whose profile is equal to $M_{\min}$ (all equalities are valid up to an additive $O(\log n)$ term).*

The proof of this theorem presented in the paper [7] is non-effective and its authors wonder whether there is a quasi-effective one. Note that the profile of neither of constructive examples from [1] and [7] is known.

*Quasi-effective proof of theorem 4.* The profile of $a, b$ is equal to $M_{\min}$ if the following holds. Any triple $u, v, w$ such that $u+v < 2$, or $u+w < 2$, or $u+v+w < 3$, or simultaneously

$$u+v < 3, \quad u+w < 3, \quad u+v+w < 4$$

does not belong to the profile of $a, b$. If at least one of the inequalities $u+v < 2$, $u+w < 2$, $u+v+w < 3$ holds then the profile of all $a, b$ with $K(a_n) = K(b_n) = 2n$, $K(a_n, b_n) = 3n$ does not contain $\langle u, v, w\rangle$. This is a direct consequence of the inequalities $K(a) \leq K(c) + K(a|c)$, $K(b) \leq K(c) + K(b|c)$, $K(a, b) \leq K(c) + K(a|c) + K(b|c)$, respectively. So it suffices to construct strings $a_n, b_n$ for which $K(a_n, b_n) = 3n + O(\log n)$, $K(a_n) = K(b_n) = 2n + O(\log n)$ and that have the following property. For any $U, V, W$ satisfying the inequalities

$$\begin{aligned} U + \max\{V, W\} &< 3n - k, \\ U + V + W &< 4n - k, \end{aligned} \tag{2}$$

there is no $c$ such that

$$K(c) < U, \quad K(a_n|c) < V, \quad K(b_n|c) < W.$$

Here $k$ is a linear function of $\log n$ to be specified later. We will construct for any $n$ a directed graph whose vertices are strings of length $2n$ and the number of edges is $2^{3n+O(1)}$. As $\langle a, b\rangle$ we will take any random edge in the graph. Then $K(a, b)$ will be about $3n$. What are other properties of the graph we need? The first one is as follows. For any set $N$ of cardinality $2^{2n-k}$, only a small fraction of edges is incident to a node in $N$. Applying this property to the set of strings of complexity less than $2n-k$, we will prove that for any random edge $\langle a, b\rangle$ of the graph the complexity of both $K(a)$ and $K(b)$ is greater than $2n - k = 2n - O(\log n)$. The other property is as follows. Most edges of the graph should not belong to the set

$$\{\langle a, b\rangle \mid \text{there are } U, W, W \text{ satisfying inequalities (2), and a string } c \text{ such that } K(c) < U, \quad K(a_n|c) < V, \quad K(b_n|c) < W\}.$$

Since we are not able to find this set, we have to define some its decidable property and to ensure that for any set $M$ having this property most edges of the graph do not belong to $M$. Here is this property. Our set is a union over all $U, V, W$, satisfying inequalities (2), of some sets $M_{UVW}$, each of those is a union of $2^U$ sets of the form $A \times B$ where $|A| < 2^V$, $|B| < 2^W$. So let us call any set of pairs of strings of length $2n$ *special* if it has this property. Thus it suffices to prove the following lemma.

**Lemma 3.** *For some function $k = O(\log n)$ and some constant $\beta$ for all $n$ there is a graph whose nodes are strings of length $2n$ that satisfies the following properties. 1. The number of edges is between $2^{3n-2}$ and $2^{3n}$. 2. For any set $N$ of cardinality less than $2^{2n-k}$ at most $\beta \cdot 2^{3n}/n$ edges are incident to a node in $N$. 3. Any special set has at most $\beta \cdot 2^{3n}/n$ edges.*

*Proof.* Let us make $2^{3n}$ independent trials choosing in each trial an edge at random (we allow loops). Let us prove that with probability close to 1 all the three properties hold.

1. For any fixed set $E$ of pairs of nodes of cardinality $2^{3n-2}$ the probability of that a random edge gets into $E$ is $2^{-n-2}$. The probability of that all $2^{3n}$ edges get into $E$ is $2^{-(n+2)2^{3n}}$. The number of such $E$'s is less than $2^{4n \cdot 2^{3n-2}} = 2^{n \cdot 2^{3n}}$. Thus the probability of that all edges get into a set of cardinality $2^{3n-2}$ does not exceed $2^{-(n+2)2^{3n}+n \cdot 2^{3n}} = 2^{-2 \cdot 2^{3n}} \ll 1$.

2. The probability that a random edge in incident to a node in a fixed set $N$ of cardinality $2^{2n-k}$ is less than $1/n$ (provided $k > \log n$). By Chernoff bound the probability of that this happens for a fraction $O(1/n)$ of $2^{3n}$ random edges does not exceed $2^{-\Omega(2^{3n}/n)} = 2^{-2^{3n-\log n - O(1)}}$. The number of such $N$ is less than $2^{2n \cdot 2^{2n-k}} = 2^{2^{2n+\log n+O(1)-k}}$. Hence with probability at least

$$1 - 2^{2^{2n+\log n+O(1)-k}} \cdot 2^{-2^{3n-\log n-O(1)}}$$

for every $N$ at most $O(2^{3n}/n)$ edges of the graph are incident to a node in $N$. This probability tends to 1.

3. Let us prove that any special set has at most $O(2^{4n}/n)$ edges. The number of $U, V, W$ does not exceed $O(n^3)$. For fixed $U, V, W$ any union of $2^U$ sets of the form $A \times B$ where $|A| < 2^V$, $|B| < 2^W$ has at most $2^{U+V+W} \le 2^{4n-k}$ pairs. Multiplying the latter number by $O(n^3)$ we get $2^{4n-k+3\log n+O(1)} = O(2^{4n}/n)$ (provided $k > 3\log n$). Therefore for any special $M$ with probability at least $1 - O(1/n)$ a random edge does not get into $M$. By Chernoff bound with probability $1 - 2^{-\Omega(2^{3n}/n)}$ at most $O(2^{3n}/n)$ edges of the graph get into $M$. Let us estimate now the number of special sets. For any fixed $U, V, W$ there are at most

$$(2^{2n\cdot 2^V} \cdot 2^{2n\cdot 2^W})^{2^U} = 2^{2n(2^V+2^W)2^U} < 2^{2^{3n-k+\log n+O(1)}}$$

special sets. Rising this number to the power of the number of different $U, V, W$ we obtain an upper bound $2^{2^{3n-k+4\log n+O(1)}}$ for the number of special sets. Hence with probability at least

$$1 - 2^{2^{3n-k+4\log n+O(1)}} \cdot 2^{-2^{3n-\log n-O(1)}} = 1 - 2^{-2^{3n-\log n}(2^{-O(1)}-2^{5\log n-k+O(1)})}$$

all special sets have at most $O(2^{3n}/n)$ edges of the graph. So it suffices to let $k = 5\log n + O(1)$ in order to make this probability close to 1. $\square$

## References


[1] R. Ahlswede and J. Körner. On common information and related characteristics of correlated information sources. Preprint. Presented at the *7th Prague Conference on Information Theory,* 1974.

[2] C.H. Bennett, P. Gács, M. Li, P. Vitanyi and W. Zurek. *Information Distance, IEEE transactions on Information Theory*, Vol. 44, No. 4, 1407–1423.

[3] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–509, 1952.

[4] S.C. Kleene. *Introduction to metamathematics.* New York, Toronto, 1952.

[5] A. Kolmogoroff. Zur Deutung der Intuitionistischen Logik. *Mathematische Zeitschrift*, Bd. 35 (1932), H. 1, S. 57–65.

[6] M. Li, P. Vitányi. An Introduction to Kolmogorov Complexity and its Applications. Springer Verlag, 1997.

[7] An. Muchnik, A. Romashchenko, A. Shen, and N. Vereshchagin. Upper semi-lattice of binary strings with the relation "x is simple conditional to y". *Proc. of 14th Annual IEEE Conference on Computational Complexity*, Atlanta, May 4-6, 1999, 114–122.

[8] An. Muchnik. Conditional Kolmogorov complexity and codes. To be published in *Theoretical Computer Science.*

[9] A. Shen, N. Vereshchagin. Logical operations and Kolmogorov complexity. To be published in *Theoretical Computer Science.*



ECCC ISSN 1433-8092
http://www.eccc.uni-trier.de/eccc
ftp://ftp.eccc.uni-trier.de/pub/eccc
ftpmail@ftp.eccc.uni-trier.de, subject 'help eccc'