ISSN 0081-5438, Proceedings of the Steklov Institute of Mathematics, 2011, Vol. 274, pp. 193–203. © Pleiades Publishing, Ltd., 2011.
Published in Russian in Trudy Matematicheskogo Instituta imeni V.A. Steklova, 2011, Vol. 274, pp. 210–221.

Kolmogorov Complexity and Cryptography[1]

Andrej A. Muchnik[†]

Received March 2011

Abstract—We consider (in the framework of algorithmic information theory) questions of the following type: construct a message that contains different amounts of information for recipients that have (or do not have) certain a priori information. Assume, for example, that a recipient knows some string a and we want to send him some information that allows him to reconstruct some string b (using a). On the other hand, this information alone should not allow the eavesdropper (who does not know a) to reconstruct b. This is indeed possible (if the strings a and b are not too simple). Then we consider more complicated versions of this question. What if the eavesdropper knows some string c? How long should our message be? We provide some conditions that guarantee the existence of a polynomial-size message; we show then that without these conditions this is not always possible.

DOI: 10.1134/S0081543811060125

1. NON-INFORMATIVE CONDITIONAL DESCRIPTION

In this section we construct, for given strings a and b that satisfy some conditions, a string f that contains enough information to obtain b from a, but does not contain any information about b in itself (without a), and discuss some generalizations of this problem.

Uniform and nonuniform complexity. Let us start with some general remarks about conditional descriptions and their complexity. Let X be a set of binary strings, and let y be a string. Then $C(X \to y)$ can be defined as the minimal length of a program that maps every element of X to y. (As usual, we fix some optimal programming language. We can also replace minimal length by minimal complexity.) Evidently,

$$C(X \to y) \geq \max_{x \in X} C(y|x)$$

(if a program p works for all $x \in X$, it works for every x), but the reverse inequality is not always true. It may happen that the "uniform" complexity of the problem $X \to y$ (the left-hand side) is significantly greater than the "nonuniform" complexity of the same problem (the right-hand side).

To prove this, let us consider an incompressible string y of length n and let X be the set of all strings x such that $C(y|x) < n/2$. Then the right-hand side is bounded by $n/2$ by construction. Let us show that the left-hand side is greater than $n - O(\log n)$. Indeed, let p be a program that outputs y for every input x such that $C(y|x) < n/2$. Among those x there are strings of complexity $n/2 + O(\log n)$, and together with p they are enough to obtain y; hence $C(y|p) \leq n/2 + O(\log n)$. Therefore, there exists a string e of length $O(\log n)$ such that $C(y|\langle p, e\rangle) < n/2$. Then, by our assumption, $p(\langle p, e\rangle) = y$ and therefore the complexity of p is at least $n - O(\log n)$.

Remark. In this example the set X can be made finite if we restrict ourselves to strings of bounded length, say, of length at most $2n$.

[1]This paper contains some results of An.A. Muchnik (1958–2007) reported in his talks at the Kolmogorov seminar (Moscow State University, Faculty of Mechanics and Mathematics, Department of Mathematical Logic and Theory of Algorithms, March 11, 2003, and April 8, 2003) but not published at that time. These results were stated (without proofs) in the joint talk of Andrej Muchnik and Alexei Semenov at Dagstuhl Seminar 03181, April 27–May 03, 2003. This text was prepared by Alexey Chernov and Alexander Shen in 2008–2009.

[†]Deceased.

Complexity of the problem $(a \to b) \to b$**.** The example above shows that the uniform and nonuniform complexities could differ significantly. In the next example they coincide, but some work is needed to show that they coincide.

Let a and b be binary strings. By $(a \to b)$ we denote the set of all programs that transform the input a into the output b. It is known [2] that

$$C((a \to b) \to b) = \min\{C(a), C(b)\} + O(\log N)$$

for any two strings a and b of length at most N. It turns out that a stronger version of this statement (when the uniform complexity is replaced by a nonuniform one) is also true:

Theorem 1. *For any two strings a and b of length at most N, there exists a program f that maps a to b such that*

$$C(b|f) = \min\{C(a), C(b)\} + O(\log N).$$

Proof. Note that the $\leq$-inequality is obviously true for any program f that maps a to b. Indeed, having such a function and any of the strings a and b, we can reconstruct b.

Let us prove that the reverse inequality is true for some function f that maps a to b. We restrict ourselves to total functions that are defined on the set of all strings of length at most N and take values in the same set, so such a function is a finite object and the conditional complexity with respect to f is defined in a natural way. Note also that (up to $O(\log N)$ precision) it does not matter whether we consider f as an explicitly given finite object or as a program, since (for known N) both representations can be transformed into each other.

Let m be the maximum value of $C(b|f)$ for all functions (of the type described) that map a to b. We need to show that one of the strings a and b has complexity at most $m + O(\log N)$. This can be done as follows.

Consider the set S of all pairs $\langle a', b' \rangle$ where a' and b' are strings of length at most N that have the following property: *$C(b'|f) \leq m$ for every total function f whose arguments and values are strings of length at most N and $f(a') = b'$.* By the definition of m, the pair $\langle a, b \rangle$ belongs to S.

Given m and N, one can effectively enumerate the set S. Let us perform this enumeration and delete the pairs whose first or second coordinate was already encountered (as the first/second coordinate of some other undeleted pair during the enumeration); only "original" pairs with two "fresh" components are placed in $\widetilde{S}$. This guarantees that $\widetilde{S}$ is a graph of a bijection. The pair $\langle a, b \rangle$ is not necessarily in $\widetilde{S}$; however, some other pair with the first component a or with the second component b is in $\widetilde{S}$ (otherwise nothing prevents $\langle a, b \rangle$ from appearing in $\widetilde{S}$).

Since $\widetilde{S}$ can also be effectively enumerated (given m and N), it is enough to show that it contains $O(2^m)$ elements (then the ordinal number of the above-mentioned pair describes either a or b).

To show this, let us extend $\widetilde{S}$ to the graph of some bijection g. If some $\langle a', b' \rangle \in \widetilde{S}$, then $g(a') = b'$ and therefore $C(b'|g) \leq m$ by construction (recall that $\widetilde{S}$ is a subset of S). Therefore, $\widetilde{S}$ contains at most $O(2^m)$ different values of b', but $\widetilde{S}$ is a bijection graph. $\square$

Cryptographic interpretation. Theorem 1 has the following "cryptographic" interpretation. We want to transmit some information (string b) to an agent who already knows some "background" string a by sending some message f. Together with a this message should allow the agent to reconstruct b. At the same time we want f to carry minimal information about b for a "non-initiated" listener; i.e., the complexity $C(b|f)$ should be maximal. This complexity cannot exceed $C(b)$ for evident reasons and cannot exceed $C(a)$ since a and f together determine b. Theorem 1 shows that this upper bound can be reached for an appropriate f.

Let us consider a relativized version of this result that also has a natural cryptographic interpretation. Assume that a non-initiated listener knows some string c. Our construction (properly relativized) proves the existence of a function f that maps a to b such that

$$C(b|f,c) \approx \min\{C(a|c), C(b|c)\}.$$

This function has minimal possible amount of information about b for people who know c. More formally, the following statement is true (and its proof is a straightforward relativization of the previous argument):

Theorem 2. *Let a, b, and c be strings of length at most N. Then there exists a string f such that*

(1) $C(b|a,f) = O(\log N)$;

(2) $C(b|c,f) = \min\{C(a|c), C(b|c)\} + O(\log N)$.

Claim (1) says that for recipients who know a the message f is enough to reconstruct b; claim (2) says that for the recipients who know only c the message f contains minimal possible information about b.

Remark. One may try to prove Theorem 1 as follows: let f be the shortest description of b when a is known; we may hope that it does not contain "redundant" information. However, this approach does not work: if a and b are independent random strings of length n, then b is such a shortest description, but it cannot be used as f in Theorem 1. In this case one can let $f = a \oplus b$ (the bitwise sum modulo 2) instead: knowing f and a, we reconstruct $b = a \oplus f$, but $C(b|f) \approx n$.

This trick can be generalized to provide an alternative proof of Theorem 1. To this end we use the conditional description theorem from [1]. It says that for any two strings a and b of length at most N there exist a string b' such that $C(b|a,b') = O(\log N)$ (b' is a description of b when a is known), $C(b'|b) = O(\log N)$ (b' is simple relative to b), and the length of b' is $C(b|a) + O(\log N)$ (b' has the minimal possible length for descriptions of b when a is known).

To prove Theorem 1, take this b' and also a' defined in the symmetric way (a short description of a when b is known that is simple relative to a). Let us add trailing zeros or truncate a' to get a string a'' that has the same length as b'. (Adding zeros is needed when $C(a) < C(b)$; the truncation is needed when $C(a) > C(b)$.) Then let $f = a'' \oplus b'$.

A person who knows a and gets f can compute (with logarithmic additional advice) first a', then a'', then b', and then b. It is also easy to check that $C(b|f) = \min\{C(a), C(b)\}$ with logarithmic precision.

Indeed,

$$\begin{aligned} C(b|f) &= C(b,f|f) = C(b,b',f|f) = C(b,a''|f) \\ &\geq C(b,a'') - C(f) \geq C(b,a'') - |f| = C(b,a'') - C(b|a) \end{aligned}$$

with logarithmic precision. The strings a' and b are independent (have logarithmic mutual information), so b and a'' (which is a simple function of a') are independent too. Then we get the lower bound $C(b) - C(b|a) + C(a'')$, which is equal to $\min\{C(a), C(b)\}$. (End of the alternative proof of Theorem 1.)

The advantage of this proof is that it provides a message f of polynomial (in N) length (unlike our original proof, where the message is some function that has domain of exponential size), and, moreover, f has the minimal possible length $C(b|a)$. The result it gives can be stated as follows:

Theorem 3. *For any two strings a and b of length at most N there exists a string f of length $C(b|a)$ such that*

$$C(b|f,a) = O(\log N)$$

and

$$C(b|f) = \min\{C(a), C(b)\} + O(\log N).$$

The disadvantage is that this proof does not work in the relativized case (Theorem 2), at least literally. For example, let a and b be independent strings of length $2n$ and let $a = a_1a_2$ and $b = b_1b_2$ be their divisions in two halves. Then let $c = (a_1 \oplus a_2 \oplus b_1)(a_2 \oplus b_1 \oplus b_2)$. Then $C(a|c) = C(a, c|c) = C(a, b|c) = 2n$, $C(b|c) = 2n$, but $C(b|c, a \oplus b) = 0$.

In the next section we provide a different construction of a short message f that has the required properties (contains information about b only for those who know a but not for those who know c).

2. A COMBINATORIAL CONSTRUCTION OF A LOW COMPLEXITY DESCRIPTION

We will prove that if a contains enough information (more precisely, if $C(a|c) \geq C(b|c)+C(b|a)+O(\log N)$), then there exists a message f that satisfies the claim of Theorem 2 and has complexity $C(b|a) + O(\log N)$. We need the following combinatorial statement. (By $\mathbb{B}^k$ we denote the set of k-bit binary strings.)

Combinatorial lemma. *Let $n \geq m$ be two positive integers. There exists a family $\mathcal{F}$ consisting of $2^m \operatorname{poly}(n)$ functions of type $\mathbb{B}^n \to \mathbb{B}^m$ with the following property: for every string $b \in \mathbb{B}^m$ and for every subfamily $\mathcal{F}'$ that contains at least half of the elements of $\mathcal{F}$, there are at most $O(2^m)$ points with the second coordinate b and not covered by the graphs of the functions in $\mathcal{F}'$.*

Formally the property of $\mathcal{F}$ claimed by the lemma (see the figure) can be written as follows:

$$\forall b \,\forall \mathcal{F}' \subset \mathcal{F}\left[\#\mathcal{F}' \geq \frac{1}{2}\#\mathcal{F} \;\Rightarrow\; \#\bigl\{a \in \mathbb{B}^n \mid f(a) \neq b \text{ for all } f \in \mathcal{F}'\bigr\} = O(2^m)\right].$$

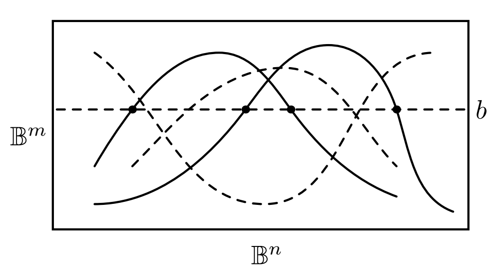
(Note that the condition $n \geq m$ is in fact redundant: if $n < m$, the claim is trivial since the number of all a is $O(2^m)$.)

Before proving the lemma, let us try to explain informally why it could be relevant. The family $\mathcal{F}$ is a reservoir for messages (f will be a number of some function from $\mathcal{F}$). Most functions in $\mathcal{F}$ (as in any other simple family) have almost no information about b; they form $\mathcal{F}'$. If the pair $\langle a, b\rangle$ is covered by the graph of some function $f \in \mathcal{F}'$, then f (i.e., its number) is the required message. If not, a belongs to a small set of exceptions, and its complexity is small, so the condition of the theorem is not satisfied. (See the detailed argument below.)

Proof of lemma. We use a probabilistic method and show that for a random family of 2^t independent random functions the required property holds with positive probability. (The exact value of the parameter t will be chosen later.)

Let us estimate from above the probability of the event "a random family $\varphi_1, \ldots, \varphi_{2^t}$ does not satisfy the required property." This happens if there exist

- an element $b \in \mathbb{B}^m$,
- a set $S \subset \mathbb{B}^n$ containing $s \cdot 2^m$ elements (the exact value of the constant s will be chosen later),
- a set $I \subset \{1, 2, \ldots, 2^t\}$ that contains half of all indices



Some functions (up to 50%) are deleted from $\mathcal{F}$; nevertheless the graphs of the remaining ones cover every horizontal line almost everywhere (except for $O(2^m)$ points).

such that

$$\varphi_i(a) \neq b \qquad \text{for every } a \in S \text{ and every } i \in I. \qquad (*)$$

To get an upper bound for the probability of this event, note that there are 2^m different values of b, at most 2^{2^t} different values of I, and at most $(2^n)^{s\cdot 2^m}$ different values of S. For fixed b, I, and S the probability of $(*)$ is

$$\left(1-\frac{1}{2^m}\right)^{2^{t-1}s\cdot 2^m}$$

(each of the 2^{t-1} functions with indices belonging to I has a value different from b at each point $a \in S$ with probability $1-1/2^m$). In total we get an upper bound

$$2^m \cdot 2^{2^t} \cdot 2^{ns\cdot 2^m}\left(1-\frac{1}{2^m}\right)^{2^{t-1}s\cdot 2^m},$$

and we have to show that this product is less than 1 if the values of the parameters are chosen properly. We can replace $(1-1/2^m)^{2^m}$ by $1/e$ (the difference is negligible within our precision) and rewrite the expression as

$$2^{m+2^t} \cdot 2^{ns\cdot 2^m}(1/e)^{s\cdot 2^{t-1}}.$$

The most important terms are those containing 2^t and 2^m in the exponents (since $2^t, 2^m \gg m, n, s$). We want the last small term to outweigh the first two. Let us split it into two parts $(1/e)^{s\cdot 2^{t-2}}$ and use these parts to compensate for the first and the second term. It is enough that

$$2^{m+2^t}(1/e)^{s\cdot 2^{t-2}} < 1 \qquad \text{and} \qquad 2^{ns\cdot 2^m}(1/e)^{s\cdot 2^{t-2}} < 1$$

at the same time. The first inequality can be made true if the constant s is large enough (note that $m \ll 2^t$). The second inequality (where both exponents can be divided by s) is achievable with $2^t = 2^m \operatorname{poly}(n)$. □

Main result. Now we are ready to give the formal statement and proof:

Theorem 4. *There exists a constant C such that for any strings a, b, and c of length at most N satisfying the inequality*

$$C(a|c) \geq C(b|c) + C(b|a) + C\log N$$

there exists a string f of length at most $C(b|a) + C\log N$ such that $C(b|a,f) \leq C\log N$ and $C(b|c,f) \geq C(b|c) - C\log N$.

Recall the idea behind this result. The condition of the theorem guarantees that the agent's "background" a has enough information not available to the adversary (who knows c); the theorem guarantees that there exists a string f that allows the agent to reconstruct b from a, has the minimal possible length among all strings with this property, and does not provide any information about b if the adversary knows only c. (Note that we use the same constant C in all $O(\log N)$ expressions, but this does not matter since increasing C makes the statement only weaker.)

Proof of Theorem 4. Using the conditional description theorem [1], we find a string b' of length $C(b|a)$ such that both complexities $C(b|b',a)$ and $C(b'|b)$ are $O(\log N)$. Then we apply the combinatorial lemma with n equal to the length of a and m equal to the length of b', i.e., to $C(b|a)$. The lemma provides a family $\mathcal{F}$, and we may assume without loss of generality that the complexity of $\mathcal{F}$ is $O(\log N)$ (for given m and n, take the first family with the required properties in some fixed ordering).

Most functions in $\mathcal{F}$ (as well as most objects in any simple set) do not have much information about b when c is known, i.e., the difference $C(b|c) - C(b|c, f)$ is small for most $f \in \mathcal{F}$. Indeed, with logarithmic precision this difference can be rewritten as $C(f|c) - C(f|b, c)$ (recall the formula for pair and conditional complexities), and the average value of both terms in the last expression is $m + O(\log N)$, the difference is of order $O(\log N)$, and we can use the Chebyshev inequality.

Let $\mathcal{F}'$ be functions from this majority. The lemma guarantees that the graphs of these functions cover all pairs $\langle a', b'\rangle$ for all strings a' of length n except for $O(2^m)$ "bad" values of a', and it remains to show that the given string a is not "bad". This is so because

$$C(a'|c) < C(b|c) + C(b|a) + O(\log N)$$

for all "bad" a'. Indeed, knowing b, c, and $C(b|c)$ (the latter contains $O(\log N)$ bits and can be ignored with logarithmic precision), we can enumerate all functions f that do not belong to $\mathcal{F}'$ (i.e., functions that make the complexity of b with condition c smaller), and therefore we can enumerate all $O(2^m)$ "bad" values. (Note that b' can also be obtained from b with a logarithmic advice.) So the complexity of the "bad" values (for known b and c) is at most $m + O(\log N)$:

$$C(a'|b, c) \leq C(b|a) + O(\log N)$$

for all "bad" a'; therefore,

$$C(a'|c) \leq C(a'|b, c) + C(b|c) + O(\log N) \leq C(b|a) + C(b|c) + O(\log N),$$

as we claimed. □

3. NEGATIVE RESULT AND OPEN QUESTIONS

The assumption made in Theorem 4 may look artificial at first glance: for example, if a, b, and c are pairwise independent, we require $C(a)$ to be twice as big as $C(b)$, and it is intuitively unclear why the amount of the background information should be twice as big as the message we want to transmit (the inequality $C(a) > C(b)$ seems more natural). In this section we show that this condition, even if looking artificial, is important: without it, all the strings f that satisfy the claim of Theorem 2 may have exponentially large length. The exact statement (see Theorem 5 below) and its proof are rather technical, so let us start with a simplified example, where, unfortunately, we get a string c of large complexity. Then we explain a more advanced example that does not have this problem.

Let us construct three strings a, b, and c with the following properties: every reasonably long program f (of polynomial or subexponential length) that maps a to b can be used to simplify the transformation of c into b. In our example the string a has complexity $1.3n$, the string b has complexity n, and they are mutually independent (have logarithmic mutual information). (The coefficient 1.3 is chosen arbitrarily; it is important that 1.3 is greater than 1 and less than 2.) The complexity of b when c is known will be about n, so using c as a condition does not make b simpler. But if we add to c any program f that maps a to b, it becomes possible to obtain b using only $0.3n$ bits of advice: the conditional complexity decreases from $C(b|c) \approx n$ to $C(b|f, c) \approx 0.3n$.

The main idea of this example can be explained as follows: the string c itself encodes a function that maps a to b (but still c without a has no information about b). Assume that some program f that maps a to b is given. Why does it help to describe b if c is known in addition to f? We know that both f and c map a to b, so a is one of the solutions of the equation $f(x) = c(x)$. If this equation has not too many solutions, we can describe a (and therefore b) by specifying the ordinal number of a in the enumeration of all solutions. (Note that f may not be defined everywhere, but this does not matter.) In this way we get a conditional description of b (for known c and f) that may have a small length compared to $C(b)$ (and $C(b)$ will be close to $C(b|c)$; we promised that c itself has no information about b).

How do we get a, b, and c with these properties? We get such a triple with high probability if a and b are independently taken at random among strings of length $1.3n$ and n, respectively, and c is a random function whose graph contains the pair $\langle a, b\rangle$. The same distribution on a, b, and c can be described in a different way: we take a random function c and then a random element $\langle a, b\rangle$ of its graph.

With high probability we get strings a and b with the required complexities $1.3n$ and n and small mutual information. We can also show that $C(b|c)$ is close to n with high probability. Indeed, for a typical function c of type $\mathbb{B}^{1.3n} \to \mathbb{B}^n$ most of its values have preimages of size $2^{0.3n}$, and therefore the second component of a random element of its graph has an almost uniform distribution, so most of the values of c have high complexity even with condition c.

Now let f be some program that maps a to b and has not very high complexity (much less than what Theorem 2 gives). How many solutions does the equation $f(x) = c(x)$ have? Typically (for a given f and a random c) we have about $2^{0.3n}$ solutions (for each x the probability of $f(x) = c(x)$ equals 2^{-n}, and there are $2^{1.3n}$ points x); here we assume that f is total, but if it is not, we get even fewer solutions. For a fixed f and a random c, it is very unlikely that the number of solutions is significantly greater than $2^{0.3n}$. In other words, the Hamming ball of the corresponding radius around f has a negligible probability. If the number of these balls (i.e., the number of programs f we consider) is not too large, the union of these events also has small probability, so a randomly chosen c will be outside these balls. This means that for all programs f with bounded complexity the equation $f(x) = c(x)$ has at most $2^{0.3n}$ solutions (or slightly more) and the complexities $C(a|f, c)$ and $C(b|f, c)$ are (almost) bounded by $0.3n$ as we promised.

We do not provide details of this argument since we want to prove a stronger (and more complicated) result. Namely, we want to find a function c that has not very high complexity (while the argument explained gives c that may have exponential complexity): the complexity of c should exceed the complexity of programs f (that it opposes) by $C(b)$. (If we allow more programs, we need more freedom for c.)

The idea of the construction remains the same: we select a random point on the graph of a random function. However, now the function is a random element of some family $\mathcal{C}$ of functions. We formulate some combinatorial properties of $\mathcal{C}$. Then we prove (by a probabilistic argument) that there exists a family with these properties and conclude that there exists a simple family with these properties (the first family found by an exhaustive search). Finally, we prove that for most pairs $\langle a, b\rangle$ there exists a function c in the family that satisfies our requirements. (So we prove even a slightly stronger statement: instead of the existence of a triple a, b, c we prove that for most a and b there exists c.) The size of the family $\mathcal{C}$ provides a bound for the complexity of c (since every element of $\mathcal{C}$ is determined by its index).

Let us formulate the required combinatorial statement starting with some definitions. Fix some sets A and B. We say that some family $\mathcal{F}$ of functions $A \to B$ *rejects* a function $c\colon A \to B$ if there exists $f \in \mathcal{F}$ such that the cardinality of the set $\{a\colon c(a) = f(a)\}$ exceeds $4\#A/\#B$ (note that the "expected" cardinality is $\#A/\#B$). Let $\mathfrak{H}$ be a mapping defined on B; for every $b \in B$ the value $\mathfrak{H}(b)$ is a family of functions of type $A \to B$ (i.e., $\mathfrak{H}(b) \subset B^A$ for every $b \in B$). We say that a function c *covers* the pair $\langle a, b\rangle \in A \times B$ (for given $\mathfrak{H}$ and $\mathcal{F}$) if (1) $c(a) = b$, (2) the function c is not rejected by $\mathcal{F}$, and (3) $c \notin \mathfrak{H}(b)$.

Lemma. *Assume that $\#B \geq 2$ and $\#A \geq 16\#B$. Assume that two numbers $\varepsilon \geq 4\#B/\#A$ and $\Phi \leq 2^{\#A/4\#B}$ are fixed. There exists a family $\mathcal{C}$ of functions $A \to B$ of cardinality*

$$\max\left\{\frac{20\#B}{\varepsilon}, \frac{6\Phi\log_2(\#B)}{\varepsilon}, 6\Phi\cdot\#B\cdot\log_2(\#B)\right\}$$

with the following property: *for every family $\mathcal{F}$ of size at most Φ and for every mapping $\mathfrak{H}$ such that $\#(\mathfrak{H}(b)) \le (1/4)\#\mathcal{C}$ for every $b \in B$, at most an ε-fraction of all pairs $\langle a, b\rangle$ are not covered by any $c \in \mathcal{C}$ (for these $\mathcal{F}$ and $\mathfrak{H}$).*

The statement of this lemma can be written as follows (we omit conditions on the cardinalities of $\mathcal{C}$, $\mathcal{F}$ and $\mathfrak{H}(b)$):

$$\exists \mathcal{C}\ \forall \mathcal{F}, \mathfrak{H}$$
$$\left|\left\{\langle a,b\rangle\colon \forall c\ \left[(c(a)=b) \Rightarrow \left[(c \in \mathfrak{H}(b))\ \vee\ \left(\exists f \in \mathcal{F}\ \#\{x\colon f(x)=c(x)\} \ge \frac{4\#A}{\#B}\right)\right]\right]\right\}\right|$$
$$\le \varepsilon \cdot \#A \cdot \#B.$$

Let us explain informally the meaning of this lemma (how it is used in the sequel). We may assume without loss of generality that the family $\mathcal{C}$ is simple (looking for the first family with the required properties in some ordering). Let $\mathcal{F}$ be the family of all functions that have simple programs (or their extensions, if the functions are partial). Let $\mathfrak{H}(b)$ be the set of all functions that are simple when b is known (having small conditional complexity with condition b). For a pair $\langle a, b\rangle$ that does not belong to the "bad" ε-fraction, there exists a function $c \in \mathcal{C}$ that covers $\langle a, b\rangle$. This function (or, better to say, its index in $\mathcal{C}$) is a counterexample we are looking for. Indeed, if the eavesdropper knows c and gets a simple program f mapping a to b, the complexity of b for him decreases. Indeed, it is enough to specify the ordinal number of a in the enumeration of all solutions of the equation $f(x) = c(x)$, and the eavesdropper can reconstruct a (and therefore b, since $f(a) = b$). In addition, the choice of $\mathfrak{H}$ guarantees that c and b are independent (i.e., c has maximal possible complexity even if b is known). The details of this argument will be explained later, after we prove the lemma.

Proof of the lemma. Using a probabilistic argument, let us consider a random family $\mathcal{C}$ of the size mentioned. We assume that $\mathcal{C}$ is indexed by integers in the range $1, \dots, \#\mathcal{C}$, and for every index i and every point $a \in A$ the value of the ith function on a is an independent random variable uniformly distributed over B. Then we prove that the probability of the event "$\mathcal{C}$ is bad" (i.e., does not have the required property) is strictly less than 1.

To this end we get an upper bound for the probability of the event "$\mathcal{C}$ does not have the required property with respect to a fixed family $\mathcal{F}$" (and then multiply it by the number of different families $\mathcal{F}$). So let us assume $\mathcal{F}$ is fixed. Things are "good" if for every mapping $b \mapsto \mathfrak{H}(b)$ (with our restrictions: all $\mathfrak{H}(b)$ have cardinality at most $(1/4)\#\mathcal{C}$) for ε-almost all pairs $\langle a, b\rangle$ there is a function $c \in \mathcal{C}$ that is not rejected by $\mathcal{F}$, is not in $\mathfrak{H}(b)$, and is such that $c(a) = b$.

Note that the definition of rejection does not refer to $\mathcal{C}$: the set of rejected functions is determined by $\mathcal{F}$ alone. For a given $\mathcal{F}$ there are two possibilities: (1) many functions are rejected (we choose $(1/4)\#\mathcal{C}$ as a threshold) or (2) not many functions are rejected. In the latter case we may add the rejected functions to all $\mathfrak{H}(b)$ (for all b), and the size of all $\mathfrak{H}(b)$ remains bounded by $(1/2)\#\mathcal{C}$.

In other terms, for a fixed $\mathcal{F}$ the "bad" event is covered by the union of the following two events:

(i) $\mathcal{F}$ rejects at least $1/4$ of all functions in $\mathcal{C}$;

(ii) there exists a mapping $b \mapsto \mathfrak{H}(b)$ with all sets $\mathfrak{H}(b)$ of cardinality at most $(1/2)\#\mathcal{C}$ such that the fraction of the pairs $\langle a, b\rangle \in A \times B$ that do not belong to any function $c \in \mathcal{C} \setminus \mathfrak{H}(b)$ exceeds ε.

What we need is the following: the sum of the probabilities of these two events multiplied by the number of possibilities for $\mathcal{F}$ is less than 1. To show this, we prove that each of these two probabilities is less than $1/2$ divided by $(\#B^{\#A})^{\Phi}$ (this expression is an upper bound for the number of different families $\mathcal{F} \subset B^A$ of size Φ).

The first event can be rewritten as follows: *there exists a subfamily $\mathcal{C}' \subset \mathcal{C}$ of size $\#\mathcal{C}/4$ such that for all $c \in \mathcal{C}'$ there exist $A' \subset A$ of size $4\#A/\#B$ and a function $f \in \mathcal{F}$ such that $f(a) = c(a)$ for all $a \in A'$.*

The number of possibilities for $\mathcal{C}'$ does not exceed $2^{\#\mathcal{C}}$, the number of all subsets. For a fixed $\mathcal{C}'$ (or, better to say, for a fixed set of indices) the functions with these indices are chosen independently. So we can estimate the probability of the bad event for one index and then use independence. To get an upper bound for the number of possibilities for A', let us note that the number of r-element subsets of a q-element set, $\binom{q}{r}$, does not exceed $q^r/r! \le q^r/((r/3)^r) = (3q/r)^r$. For $q = \#A$ and $r = 4\#A/\#B$ we get the bound $(3\#B/4)^{4\#A/\#B}$.

Therefore, the probability of the first event does not exceed

$$2^{\#\mathcal{C}}\left(\Phi\left(\frac{3\#B}{4}\right)^{4\#A/\#B}\left(\frac{1}{\#B}\right)^{4\#A/\#B}\right)^{\#\mathcal{C}/4} = \left(2\Phi^{1/4}\left(\frac{3}{4}\right)^{\#A/\#B}\right)^{\#\mathcal{C}}.$$

Multiplied by $(\#B)^{\#A\cdot\Phi}$ (the number of possibilities for $\mathcal{F}$), this probability is less than 1/2, since $\#B \ge 2$, $\#A \ge 16\#B$, $\Phi \le 2^{\#A/4\#B}$, and $\#\mathcal{C} \ge 6\Phi\cdot\#B\cdot\log_2(\#B)$ (according to the assumptions of the lemma). Indeed, the last inequality implies that $\#\mathcal{C} \ge 12$ if $\Phi \ge 1$ (for an empty $\mathcal{F}$ the statement is trivial). Since $\#B \ge 2$, we conclude that $1 + \#\mathcal{C} \le 13\#\mathcal{C}/12$. Then $1 \le \#A/(16\#B)$ implies that $1 + \#\mathcal{C} \le (13/192)(\#A\cdot\#\mathcal{C}/\#B)$. The condition $\log_2\Phi \le \#A/4\#B$ implies that $(\#\mathcal{C}/4)\log_2\Phi \le (1/16)(\#A\cdot\#\mathcal{C}/\#B)$. Finally, the inequality $\#\mathcal{C} \ge 6\Phi\cdot\#B\cdot\log_2\#B$ implies that $\#A\cdot\Phi\log_2\#B \le (1/6)(\#A\cdot\#\mathcal{C}/\#B)$. Adding up these inequalities (note that $19/64 < 1/3 < \log_2(4/3)$) and taking the exponential (with base 2) of both sides, we get the required inequality (after appropriately grouping the factors).

Now let us consider the second event (recall that it depends on $\mathcal{F}$, which is fixed): *there exist a mapping $b \mapsto \mathfrak{H}(b)$ such that every $\mathfrak{H}(b)$ has cardinality at most $\#\mathcal{C}/2$ and a subset $U \subset A\times B$ of size $\varepsilon\cdot\#A\cdot\#B$ such that for every pair $\langle a,b\rangle \in U$ and for every function $c \in \mathcal{C}\setminus\mathfrak{H}(b)$ we have $c(a) \ne b$.*

In the sequel we assume that $\mathfrak{H}(b)$ is not a set of functions, but a set of their indices (numbers in the range $1,\dots,\#\mathcal{C}$); this does not change the event in question.

To estimate the probability of the second event, let us fix not only $\mathcal{F}$ but also $\mathfrak{H}$ and U. The corresponding event can be described as the intersection (taken over all pairs $\langle a,b\rangle$ and over all $i \notin \mathfrak{H}(b)$) of the events $c[i](a) \ne b$ ("the ith function does not map a to b"). The probability bound would be simple if all these events were independent; in this case the probability would be $(1-1/\#B)^d$, where d is the number of all triples $\langle i,a,b\rangle$, i.e., $\varepsilon\cdot\#A\cdot\#B\cdot\#\mathcal{C}/2$ (i.e., d is the product of the number of pairs $\langle a,b\rangle \in U$ and the number of possible values of i for a given b).

Unfortunately, these events are independent only for different a (or different i); the events $c[i](a) \ne b_1$ and $c[i](a) \ne b_2$ are dependent. However, the dependence even helps us: the condition $c[i](a) \ne b_1$ only decreases the probability of the event $c[i](a) \ne b_2$ (the denominator in $1/\#B$ decreases by 1). The same is true for several conditions.

Formally speaking, we may group the events with common a and i and then use the inequality $1 - k/\#B \le (1-1/\#B)^k$, where k is the number of events in a group.

In this way we get an upper bound for the probability of failure: for fixed $\mathcal{F}$, $\mathfrak{H}$ and U, it does not exceed

$$\left(1-\frac{1}{\#B}\right)^{\varepsilon\cdot\#A\cdot\#B\cdot\#\mathcal{C}/2} \le 2^{-\varepsilon\cdot\#A\cdot\#\mathcal{C}/2}.$$

This expression is then multiplied by the number of possibilities for U (which does not exceed $2^{\#A\cdot\#B}$), for $\mathfrak{H}$ (which does not exceed $(2^{\#\mathcal{C}})^{\#B}$) and for $\mathcal{F}$. In total, we get

$$2^{-\varepsilon\cdot\#A\cdot\#\mathcal{C}/2}\cdot 2^{\#A\cdot\#B}\cdot 2^{\#\mathcal{C}\cdot\#B}(\#B)^{\#A\cdot\Phi}.$$

It is easy to check that this expression is less than 1/2 if $\#B \geq 2$, $\varepsilon \geq 4\#B/\#A$, $\#\mathcal{C} \geq 20\#B/\varepsilon$, and $\#\mathcal{C} \geq (6\Phi \log_2 \#B)/\varepsilon$. Indeed, we have $1 + \#A \cdot \#B \leq 3 \cdot \#A \cdot \#B/2$ if A is not empty and $\#B \geq 2$. Therefore, $\#\mathcal{C} \geq 20\#B/\varepsilon$ implies $1 + \#A \cdot \#B \leq (3/40)\varepsilon \cdot \#A \cdot \#\mathcal{C}$. Also $\varepsilon \geq 4\#B/\#A$ implies $\#\mathcal{C} \cdot \#B \leq (1/4)\varepsilon \cdot \#A \cdot \#\mathcal{C}$. Finally, $\#\mathcal{C} \geq (6\Phi \log_2 \#B)/\varepsilon$ implies $\#A \cdot \Phi \log_2(\#B) \leq (1/6)\varepsilon \cdot \#A \cdot \#\mathcal{C}$. Adding up these inequalities, noting that $59/120 < 1/2$, and then taking the exponentials (with base 2), we get the required bound after regrouping the factors.

The lemma is proven. □

Now we use this lemma to prove the promised negative result. Let $\alpha > 0$ be some constant. Let m, n, and l be positive integers such that $n \geq 1$, $m \geq n + 4$, $m - \alpha \log_2 m \geq n + 2$, and $l + 1 + \log_2(l + 1) \leq 2^{m-n-2}$. Let $N = \max\{m, l\}$.

Theorem 5. *Let a be a string of length m and b be a string of length n such that*

$$m + n - C^{\mathbf{0}'}(a, b) < \alpha \log_2 m.$$

Then there exists a string c of complexity $n + l + O(\log N)$ such that

- $C(c|b) = C(c) + O(\log N)$;
- $C(b|a, c) = O(\log N)$;
- *for every f such that $C(f) \leq l - C(b|a, f)$, we have $C(b|c, f) \leq m - n + C(b|a, f) + O(\log N)$.*

(*The constant hidden in $O(\cdot)$ depends on α but does not depend on m, n, or l.*)

Before proving this theorem, let us explain why it shows the importance of the condition in Theorem 4. The equation $C(c|b) = C(c) + O(\log N)$ shows that the strings b and c are independent and $C(b|c) = C(b) = n$ with $O(\log N)$-precision. Since $C(b|a, c) = O(\log N)$, we have $C(a|c) \geq C(b|c) - C(b|a, c) = n$ (with the same $O(\log N)$-precision). Note also that $C(b|a) = n$ (with $O(\log m)$-precision). Therefore, if $C(b|a, f) = O(\log N)$ for some string f of length not exceeding l, then

$$C(b|c, f) < \min\{C(a|c), C(b|c)\} + O(\log N)$$

when $m - n < n + O(\log N)$, i.e., when $C(a) < C(b|c) + C(b|a)$.

Proof of Theorem 5. Let A be the set of all m-bit strings, and let B be the set of all n-bit strings. Let $\varepsilon = 1/m^\alpha$ and $\Phi = 2^{l+1}(l + 1)$. Our assumptions about n, m, and l guarantee that A, B, ε, and Φ satisfy the conditions of the lemma. Therefore, there is a family $\mathcal{C}$ with the properties described in the statement of the lemma. As we have said, we may assume without loss of generality that $\mathcal{C}$ is simple, and in this case the complexity of every element of $\mathcal{C}$ does not exceed $\log_2 \#\mathcal{C}$ plus $O(\log N)$, i.e., does not exceed $n + l + O(\log N)$.

Now let $\mathfrak{H}(b)$ be the set $\{c \in \mathcal{C} \colon C(c|b) < \log_2(\#\mathcal{C}) - 2\}$; then $\#\mathfrak{H}(b) \leq \#\mathcal{C}/4$ for every b.

Now the family $\mathcal{F}$ is constructed as follows. It contains Φ functions numbered by integers in the range $1, \ldots, \Phi$. We enumerate all triples $\langle a, b, f\rangle$, where $a \in A$, $b \in B$, and f is an l-bit string such that $C(f) + C(b|a, f) \leq l$. Some indices (numbers) have labels that are l-bit strings. When a new triple $\langle a, b, f\rangle$ appears, we first try to add $\langle a, b\rangle$ to one of the functions whose index already has label f. If this is not possible (all functions that have label f are already defined at a and have values not equal to b), we take a fresh index (that has no label), assign label f to it and let the corresponding function map a to b. A free index does exist since each f occupies at most $2^{l-C(f)+1}$ indices (if some f needs more, then for some a all $2^{l-C(f)+1}$ functions are defined and have different values, so we have already enumerated more than $2^{l-C(f)+1}$ different elements b such that $C(b|a, f) \leq l - C(f)$; a contradiction), and all f in total require at most $\sum_{C(f)\leq l} 2^{l-C(f)+1} = \sum_{k=0}^{l} \sum_{C(f)=k} 2^{l-k+1} = \Phi$ indices. After all the triples with these properties are enumerated, we extend our functions to total ones (arbitrarily).

Consider the set of pairs $\langle a, b\rangle$ that are not covered by $\mathcal{C}$ (for given $\mathcal{F}$ and $\mathfrak{H}$). The cardinality of this set does not exceed $\varepsilon \cdot 2^{m+n}$. On the other hand, $\mathcal{F}$ and $\mathfrak{H}$ can be computed using the $\mathbf{0}'$-oracle; after that the set of noncovered pairs can be enumerated; therefore, $C^{\mathbf{0}'}(a,b) \le m+n-\alpha\log_2 m$ for every noncovered pair $\langle a, b\rangle$.

Therefore, for any a and b such that $m+n-C^{\mathbf{0}'}(a,b) < \alpha\log_2 m$ there exists $c \in \mathcal{C}$ such that $c(a) = b$, $c \notin \mathfrak{H}(b)$, and for every $f \in \mathcal{F}$ the equation $c(x) = f(x)$ has at most 2^{m-n+2} solutions.

Since $c(a) = b$, we have $C(b|a,c) = O(\log N)$.

Since $c \notin \mathfrak{H}(b)$, we have $C(c|b) \ge \log_2(\#\mathcal{C}) - 2$, i.e., $C(c) = C(c|b) + O(\log N)$.

Finally, we have to estimate $C(b|c,f)$ for strings f such that $C(f) \le l - C(b|a,f)$. Knowing f, we enumerate the functions in $\mathcal{F}$ that have label f. One of them, say, $\widetilde{f}$, passes through $\langle a, b\rangle$ (i.e., $\widetilde{f}(a) = b$). To specify this function, we need at most $C(b|a,f) + O(\log N)$ additional bits. Knowing $\widetilde{f}$ and c, we may enumerate all x such that $c(x) = \widetilde{f}(x)$. (More precisely, we specify the index of $\widetilde{f}$ in $\mathcal{F}$ rather than $\widetilde{f}$ itself. However, to enumerate the solutions of the equation $c(x) = \widetilde{f}(x)$, it is enough to enumerate the pairs $\langle x, y\rangle$ such that $y = \widetilde{f}(x)$ by repeating the construction of $\mathcal{F}$.) This set contains a and has cardinality at most 2^{m-n+2}, so we can specify a using $m-n+2$ additional bits. Altogether, $C(b|c,f) \le C(a|c,f) + O(\log N) \le C(b|a,f) + m - n + O(\log N)$, as we claimed.

Theorem 5 is proven. □

Open questions. 1. Is it possible to strengthen Theorem 5 and have c of complexity at most $n + O(\log N)$ instead of $n + l + O(\log N)$? (An.A. Muchnik in his talk claimed that this can be done by a more complicated combinatorial argument, which was not explained in the talk.)

2. Theorem 5 shows that if a is only slightly more complex than b, then for some c short messages do not work. On the other hand, the alternative proof of Theorem 1 works for empty c. What can be said about other c? What are the conditions that make short messages possible?

3. What can be said about the possible complexities $C(f|b)$, $C(f|a,b)$, and $C(f|a,b,c)$ if f is a message with the required properties?

REFERENCES

1. An. A. Muchnik, "Conditional Complexity and Codes," Theor. Comput. Sci. **271** (1–2), 97–109 (2002). [Preliminary version: An. Muchnik and A. Semenov, "Multi-conditional Descriptions and Codes in Kolmogorov Complexity," ECCC Tech. Rep. No. 15 (Jan. 27, 2000).]
2. A. Shen and N. Vereshchagin, "Logical Operations and Kolmogorov Complexity," Theor. Comput. Sci. **271** (1–2), 125–129 (2002).

Translated by A. Chernov and A. Shen