Volume 73 Number 6 September 2007 ISSN 0022-0000

ELSEVIER

JOURNAL OF COMPUTER AND SYSTEM SCIENCES

Available online at www.sciencedirect.com
ScienceDirect

ELSEVIER

Journal of Computer and System Sciences 73 (2007) 973–985

JOURNAL OF COMPUTER AND SYSTEM SCIENCES

www.elsevier.com/locate/jcss

Individual communication complexity ☆

Harry Buhrman [a], Hartmut Klauck [b], Nikolai Vereshchagin [c], Paul Vitányi [a,*]

[a] *CWI, Kruislaan 413, 1098 SJ Amsterdam, The Netherlands*
[b] *Computer Science Department, University of Frankfurt, Germany*
[c] *Department of Mathematical Logic and Theory of Algorithms, Faculty of Mechanics and Mathematics, Moscow State University, Leninskie gory, Moscow 119992, Russia*

Received 13 September 2006; received in revised form 7 March 2007

Available online 15 March 2007

Abstract

We initiate the theory of communication complexity of individual inputs held by the agents. This contrasts with the usual communication complexity model, where one counts the amount of communication for the worst-case or the average-case inputs. The individual communication complexity gives more information (the worst-case and the average-case can be derived from it but not vice versa) and may in some cases be of more interest. It is given in terms of the Kolmogorov complexities of the individual inputs. There are different measures of communication complexity depending on whether the protocol is guaranteed to be correct for all inputs or not, and whether there's one-way or two-way communication. Bounds are provided for the communication of specific functions and connections between the different communication measures are shown. Some counter-intuitive results: for deterministic protocols that need to communicate Bob's input to Alice they need to communicate all of Bob's input (rather than the information difference with Alice's input), and there are so-called "non-communicable" inputs.

Keywords: Communication complexity; Kolmogorov complexity; Individual complexity

1. Introduction

In this paper we restrict ourselves to the basic two-party model of communication introduced in [12]. Applications to distributed computing, networks, VLSI design are discussed in [5]. Suppose Alice has input x, Bob has input y, and they want to compute a function $f(x, y)$ by communicating information and local computation according to a fixed protocol. To be more precise, let us assume that Alice outputs $f(x, y)$. Local computation costs are ignored; we are only interested in minimizing the number of bits communicated between Alice and Bob. Usually, one considers the worst-case or the average-case over all inputs x, y of given length n. But in many situations, for example replicated

☆ A preliminary version of this paper was presented in part in: Proc. 21st Symp. Theoret. Aspects Comput. Sci. (STACS), in: Lecture Notes in Comput. Sci., vol. 2996, Springer-Verlag, Berlin, 2004, pp. 19–30. The authors are partially supported by EU project QAIP, IST-1999-11234, EU project RESQ, IST-2001-37559, the NoE QUIPROCONE IST-1999-29064, the ESF QiT Programme, the EU NeuroCOLT II Working Group EP 27150, the EU NoE PASCAL, and by the Netherlands Organization for Scientific Research (NWO).

[*] Corresponding author.

E-mail addresses: buhrman@cwi.nl (H. Buhrman), klauck@thi.informatik.uni-frankfurt.de (H. Klauck), ver@mccme.ru (N. Vereshchagin), paulv@cwi.nl (P. Vitányi).

0022-0000/$ – see front matter © 2007 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcss.2007.03.015

file systems and cache coherence algorithms in multiprocessor systems and computer networks, the worst-case or the average-case are not necessarily significant. The files x and y can be very large, but in real life both x and y may be non-random, for example pictures that have considerable regularities. Furthermore, x and y can be correlated in ways that allow the communicated information to be greatly compressible, for example consecutive picture frames in movies. As another example, in version management of copies of the (originally) same file in different locations in a distributed system one wants to transmit only the updates that happened since the last time the versions were synchronized [7]. Let us go even further, as in [3], and suppose that neither the updates or a log of them, are available, nor that the original document is available. Each agent has only its current version. In all such situations, the worst-case is the wrong quantity to analyze. It also may be hard to find and express the probability distributions capturing regularities present in x and y, and a probability distribution on pairs of files capturing the correlation between x and y. This being the case, we are motivated to analyze the individual case. This gives also more information: from the individual-case analysis one can derive the worst-case and the average-case for given distributions, but not the other way around. Of course, formally speaking the individual case corresponds to the average-case where we have concentrated all probability on this individual case.

Our results are expressed in terms of Kolmogorov complexity [6], the minimal number of bits from which the data can be decompressed by effective computation. We use the "plain" Kolmogorov complexity denoted as $C(x)$, $C(x|y)$ for the absolute complexity of x and the conditional complexity of x given y. Increased compression of the data approximates the Kolmogorov complexity more and more, but the actual value is uncomputable in general. Given x, y, and assuming that Alice and Bob have a protocol P that works correctly on x, y, we study the *individual communication complexity* $CC^P(x, y)$ defined as the number of bits Alice with input x and Bob with input y exchange using protocol P. The use of Kolmogorov complexity results in establishing the ultimate limits on individual communication complexity, taking all effective regularities in the inputs into account. No computable compression method can improve on these limits in any individual case. The most significant problem in this setting appears to us to be $f(x, y) = y$ where Bob's input y is communicated to Alice. This is the essence of all applications motivating this work, and the main problem that has to be analyzed initially.

1.1. Results and related work

We use the framework of communication complexity as in [5,12]. As far as we are aware there is no previous work on individual communication complexity apart from [3], based on Lempel–Ziv compression. We formulate a theory of individual communication complexity, and first analyze the "mother" problem, the identity function, where Alice outputs the input of Bob (Theorem 1). Here we consider only deterministic protocols; randomized protocols are the subject of the follow-up paper [2]. We look at special functions such as the inner product (Theorem 2), random functions (Theorem 3), and equality (Theorem 4). We then turn to the question of analyzing the communication complexity, with respect to the best protocol of given complexity, for the mother problem, the identity function. We resolve the question whether there are pairs (x, y) such that 2-way protocols of complexity not exceeding certain level are more powerful than 1-way ones. We consider total protocols (defined for all input pairs), partial protocols (not necessarily defined for all input pairs), both of the variety that is always correct and of the variety that can possibly err (Theorem 8 and Corollary 1). We show that for total protocols that are always correct, the power of one-way protocols equals that of two-way protocols (Theorem 5), but for erring total protocols or partial protocols, two-way protocols are remarkably more powerful (Corollary 2). We establish a relation with Kolmogorov's Structure function (Theorem 6), and the existence of strange "non-communicable" inputs of possibly low Kolmogorov complexity for total protocols—for which the communication complexity of every total protocol is necessarily very large (almost the literal uncompressed input needs to be communicated) unless all of the input is hard-wired in the protocol (Theorem 7).

2. Preliminaries

2.1. Kolmogorov complexity

Roughly speaking, the Kolmogorov complexity of a binary string x is defined as the minimal length of a program that generates x; the conditional complexity $C(x|y)$ of x conditional to y is the minimal length of a program that produces x having y as input. There are different refinements of this idea (called *plain* Kolmogorov complexity, *monotone* complexity, *prefix* complexity, *decision* complexity, see [6,10]). In this paper we use the plain Kolmogorov complexity, $C(x)$, and the plain conditional complexity, $C(x|y)$. Their definitions follows.

A *conditional description method* is a partial computable function F (that is, a Turing machine) mapping pairs of binary strings to binary strings. A string p is called a *description of x conditional to y* with respect to F if $F(p,y)=x$. The complexity of x conditional to y with respect to F is defined as the minimal length of a description of x conditional to y with respect to F:

$$C_F(x|y) = \min\bigl\{|p|:\ F(p,y)=x\bigr\}.$$

A conditional description method U is called *universal* if for all other conditional description methods F there is a constant C such that

$$C_U(x|y) \leqslant C_F(x|y) + C$$

for all x, y. The Kolmogorov–Solomonoff theorem [4,9] (see also the textbook [6]) states that universal methods exist. We fix a universal U and define *conditional Kolmogorov complexity* $C(x|y)$ as $C_U(x|y)$. We call this U the reference universal Turing machine. The (unconditional) Kolmogorov complexity $C(x)$ is defined as Kolmogorov complexity of x conditional to the empty string. Comparing the universal function U with the function $F(p,y) = U(p, \text{empty string})$ we see that the conditional Kolmogorov complexity does not exceed the unconditional one:

$$C(x|y) \leqslant C(x) + O(1).$$

Comparing the universal function U with the function $F'(p,y) = p$ we see that Kolmogorov complexity does not exceed the length:

$$C(x) \leqslant |x| + C \tag{1}$$

for some C and all x. For most strings this inequality is close to an equality: the number of strings x of length n with

$$C(x) < n - m$$

is less than 2^{n-m}. Indeed, the total number of descriptions of length less than $n-m$ is equal to

$$1 + 2 + \cdots + 2^{n-m-1} = 2^{n-m} - 1.$$

In particular, for every n there is a string x of length n and complexity at least n. Such strings are called *incompressible, or random*. By the same reason for every y and n there is a string x of length n with $C(x|y) \geqslant n$.

The Kolmogorov complexity of other finite objects can be defined as follows. For example, to define the Kolmogorov complexity $C(x,y)$ of the ordered pair $\langle x,y\rangle$ of binary strings fix a computable injective function $x, y \mapsto [x,y]$ encoding pairs of binary strings by binary strings and let $C(x,y) = C([x,y])$. Different computable encodings lead to complexities of $C(x,y)$ that differ only by $O(1)$.

To describe the pair of strings $\langle x,y\rangle$ it is enough to concatenate the shortest descriptions of x and y. Thus we obtain:

$$C(x,y) \leqslant C(x) + C(y) + O\bigl(\log C(x)\bigr). \tag{2}$$

The term $O(\log C(x))$ is needed, as we have to separate the description of x from that of y. To this end we prefix the concatenation of the shortest descriptions of x and y by the binary notation of $C(x)$, written in a self-delimiting way. As a self-delimiting description of a string u we can take the string $\bar{u}$ obtained from u by doubling all its bits and appending the pattern 01. For instance, $\overline{001} = 00001101$. The inequality (2) can be easily strengthened:

$$C(x,y) \leqslant C(x) + C(y|x) + O\bigl(\log C(x)\bigr). \tag{3}$$

Indeed, concatenate the shortest descriptions of x with the shortest description of y conditional to x. Its conditional version is used several times throughout the paper:

$$C\bigl([x,y]|z\bigr) \leqslant C(x|z) + C\bigl(y|[x,z]\bigr) + O\bigl(\log C(x|z)\bigr) \leqslant C(x|z) + C(y|x) + O\bigl(\log C(x|z)\bigr). \tag{4}$$

2.2. Communication protocols

Let f be a function defined on pairs of strings of the same length. Assume that Alice has x, Bob has y and Alice wants to compute $f(x, y)$. A (total) *communication protocol* P over domain X with range Z is a finite rooted binary tree, whose internal nodes are divided into two parts, A and B, called Alice's nodes and Bob's nodes. (They indicate the turn of move.) Each internal node v is labeled by a function $r_v : X \to \{0, 1\}$ and each leaf v is labeled by a function $r_v : X \to Z$. A node *reached* by a protocol P on inputs x, y is the leaf reached by starting at the root of P and walking towards leaves where in each encountered internal node we go left if $r_v(x) = 0$, and we go right otherwise. This leaf is called the *conversation* on x, y. We say that using P on input x and y, *Alice computes* $z \in Z$, if the leaf v reached on x and y satisfies $z = r_v(x)$. We say that a protocol computes a function $f : X \to Z$ if Alice computes $f(x, y)$ for all $x, y \in X$. The domain X of protocols considered in the paper is always equal to the set $\{0, 1\}^n$ of binary strings of certain length n. As Z we will take either $\{0, 1\}$ or $\{0, 1\}^n$.

Definition 1. The *length of communication* $CC^P(x, y)$ of the protocol P on inputs x, y is the length of the path from the root of P to the leaf reached on x, y. By the complexity of a protocol P we mean its Kolmogorov complexity conditional to n, denoted by $C(P|n)$.

Informally, a partial protocol is a protocol which on some x, y is allowed to get stuck, that is, give no instructions at all about how to proceed. Formally, a *partial* protocol is a protocol, as defined above, but the functions r_v may be partial. The complexity $C(P|n)$ of a partial protocol P is defined as the minimal Kolmogorov complexity of a program that given n, v determines whether v is a leaf or Alice's internal node or Bob's internal node, and given n, v, x computes $r_v(x)$. If a partial protocol happens to be total (all r_v are total functions) then the new definition of $C(P|n)$ coincides with the old one.

3. The mother function: Identity

Let $I(x, y) = (x, y)$ be the identity function: Alice has to learn Bob's string. This is the "mother" function: for if Alice can compute I then she can compute every computable function f. In the following theorem we consider protocols that compute I on all strings x, y of length n. Roughly speaking, this theorem states that Alice's string x cannot be used to simplify her job: for all x, y we need to transmit between $C(y|P)$ and $C(y|n)$ bits.

Theorem 1.

(i) *For all n there is a protocol P of complexity $n + O(1)$ such that*

$$CC_I^P(x, y) \leqslant C(y|n) \tag{5}$$

for all x, y. The complexity of every protocol P satisfying inequality (5) *is at least $n - O(\log n)$.*

(ii) *For every protocol P and every x, y we have*

$$CC_I^P(x, y) \geqslant C(y|P) - O(1) \tag{6}$$

$$\geqslant C(y|n) - C(P|n) - O\big(\log C(P|n)\big). \tag{7}$$

(iii) *For all P there are x, y with*

$$CC_I^P(x, y) \geqslant C(y|x) + n - O(1)$$

(*no protocol can compute the identity in about $C(y|x)$ communicated bits for all x, y*).

Proof. (i) Assume that Bob knows $L_n = |\{p\colon\ |p| \leqslant n + C,\ U(p) \text{ halts}\}|$. Here U is the reference universal Turing machine and C is the constant from Eq. (1). Then Bob can find all halting programs of length at most $n + C$ by enumerating them until he obtains L_n halting programs. This allows him to find a shortest description for y of $C(y)$ bits. He transmits that program to Alice and Alice computes y. The complexity of this protocol is $C(L_n) + O(1) = n + O(1)$.

Let P satisfy (5) for all x, y and y be the first string such that $CC_I^P(0^n, y) \geqslant n$ (by counting arguments there is such y). As y can be found by an exhaustive search from P, we have $C(y|P) = O(1)$ and hence

$$n \leqslant CC_I^P\big(0^n, y\big) \leqslant C(y|n) \leqslant C(y|P) + C(P|n) + O(\log n) = C(P|n) + O(\log n).$$

(ii) Let c be the conversation between Alice and Bob on inputs x, y. To prove (6) it suffices to show that given P, c we can find y. The definition of a communication protocol implies that the set of all pairs (x', y') such that the conversation between Alice and Bob on input (x', y') is equal to c is a "rectangle," that is, has the form $X \times Y$, for some $X, Y \subset \{0,1\}^n$. The set Y is a one-element set, as for every $y' \in Y$ Alice outputs y also on the input (x, y') (the output of Alice depends on c, P, x only). We can find Y given P, c and since $Y = \{y\}$ we are done. The inequality (7) follows from (6) and (4).

(iii) Let y be a string of length n with $C(y|P) \geqslant n$ and let $x = y$. Then $C(y|x) = O(1)$ and by (6), we have

$$CC^P(x, y) \geqslant C(y|P) - O(1) \geqslant C(y|x) + n - O(1). \qquad \square$$

4. Other functions

In this section we establish some non-trivial lower bounds on $CC^P(x, y)$ for P computing f on all arguments for the inner product function and for random Boolean functions.

4.1. Inner product

Initially, Alice has a string $x = x_1, \ldots, x_n$ and Bob has a string $y = y_1, \ldots, y_n$ with $x, y \in \{0,1\}^n$. Alice and Bob compute the inner product of x and y modulo 2

$$f(x, y) = \sum_{i=1}^{n} x_i \cdot y_i \mod 2$$

with Alice ending up with the result. The following result is proved by extending an argument introduced in [1].

Theorem 2. *Every protocol P computing the inner product function f requires at least $CC^P(x, y) \geqslant C(x, y|P) - n - O(1)$ bits of communication on all x, y.*

Proof. Fix a communication protocol P that computes the inner product. Let Alice's input be $x = x_1 \ldots x_n$ and Bob's input be $y_1 \ldots y_n$. Run the communication protocol P on x, y and let $c(x, y)$ be the communication between Alice and Bob. Recall that P is a tree with $c(x, y)$ a path in that tree. Hence $c(x, y)$ form a prefix free set. Consider the set $S = S(x, y)$ defined by

$$S := \big\{(a, b)\colon\ c(a, b) = c(x, y),\ \text{and Alice outputs } f(x, y) \text{ having the conversation } c(x, y) \text{ and input } a\big\}.$$

We claim that $|S| \leqslant 2^n$. To prove the claim assume first that $f(x, y) = 0$. Let X be the first projection of S and Y be the second projection of S. Being an intersection of two rectangles, S is a rectangle too. As P computes f we know that $f(a, b) = 0$ for all $(a, b) \in S$. In other words, every element of X is orthogonal to every element in Y hence $\operatorname{rank}(X) + \operatorname{rank}(Y) \leqslant n$. Thus $|S| = |X| \cdot |Y| \leqslant 2^{\operatorname{rank}(X)+\operatorname{rank}(Y)} \leqslant 2^n$. Assume now that $f(x, y) = 1$. Again $S = X \times Y$ for some X, Y and $f(a, b) = 1$ for all $(a, b) \in S$. Subtracting x from the first component of all pairs in S we obtain a rectangle S' such that $f(a, b) = 0$ for all $(a, b) \in S'$. By above argument, we have $|S'| \leqslant 2^n$. As $|S'| = |S|$ we are done.

Given P, $c(x, y)$, $f(x, y)$ and the index of (x, y) in S we can compute (x, y). By the prefix free property, $c(x, y)$ and the index of (x, y) can be concatenated without delimiters. Consequently, $C(x, y|P) \leqslant |c(x, y)| + n + O(1)$. $\square$

Remark 1. The result of the theorem is only significant for $C(x, y|P) > n$, but for some x, y it cannot be improved. Namely, if $x = 00\ldots0$ then $f(x, y) = 0$ for all y's and there is a protocol P computing the inner product function such that $CC^P(x, y) = 0$ for all such x, y. If y is any random string (relative to P) then the right-hand side of the inequality $CC^P(x, y) \geqslant C(x, y|P) - n - O(1)$ becomes $O(1)$ while the left-hand side is equal to 0, thus both sides are almost the same.

4.2. Random functions

Assume that a function $f:\{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ satisfies

$$C(f|n) \geqslant 2^{2n} - n. \tag{8}$$

The latter condition means that the truth table describing the outcomes of f for the 2^n possible inputs x (the rows) and the 2^n possible inputs for y (the columns) has high Kolmogorov complexity. If we flip the truth table for a prospective f using a fair coin, then with probability at least $1 - 2^{-n}$ it will satisfy (8).

Theorem 3. *Every deterministic protocol P computing a function f satisfying* (8) *requires at least $CC^P(x,y) \geqslant \min\{C(x|P), C(y|P)\} - \log n - O(1)$.*

Proof. Run the communication protocol P on x, y and let $c(x,y)$ be the communication between Alice and Bob. Consider the set $S = S(x,y)$ defined by

$$S = \big\{(x', y'):\ c(x', y') = c(x,y),\ \text{and Alice outputs } f(x,y) \text{ having the conversation } c(x,y) \text{ and input } x'\big\}.$$

Then S is a *monochromatic rectangle* in the function table of f (that is, $f(x', y') = f(x,y)$ for all $(x', y') \in S$). Suppose the rectangle S has dimensions $a \times b$. Then we can describe f by giving $f(x,y)$, the values of a, b, the positions R of the rows of the rectangle, the positions C of the columns of the rectangle, and T, all of the table except the rectangle, in row-major order. This description must have length at least the Kolmogorov complexity, so by (8) we find

$$\begin{aligned} 2^{2n} - n &\leqslant C(f|n) \leqslant C\big([R,C,T]|n\big) + O(1) \leqslant C\big([R,C,T]|[a,b,n]\big) + O\big(C\big([a,b]|n\big)\big) \\ &\leqslant an + bn + \big(2^{2n} - ab\big) + O(\log ab). \end{aligned}$$

Assume w.l.o.g. that $b \geqslant a$. Then $a < 3n$ if n is large enough, as otherwise we would have $3bn \leqslant ab \leqslant (2b+1)n + O(\log b)$. Given the communication sequence $c(x,y)$, n and $f(x,y)$ we can find the rectangle S that it defines. Then, we can reconstruct x by indicating its row in the rectangle. Then $C(x|P) \leqslant |c(x,y)| + \log n + O(1)$. □

4.3. Equality

Let f be the equality function, with $f(x,y) = 1$ if $x = y$ and 0 otherwise.

Theorem 4. *For every deterministic protocol P computing f we have $CC^P(x,x) \geqslant C(x|P) - O(1)$ for all x, y. On the other hand, there is P of complexity $O(1)$ such that there are x, y ($x \neq y$) with $C(x|P), C(y|P) \geqslant n - 1$ for which $CC_f^P(x,y) = 2$.*

Proof. Lower bound: Since trivially the communication sequence must be different and uniquely identify x if both Alice and Bob have input x, we have $CC_f^P(x,x) + O(1) \geqslant C(x|P)$.

Upper bound: In the function table the lower left rectangle consisting of all x's beginning with 0 and all y's beginning with 1 is monochromatic (entries are all 0). Thus, a protocol where Bob communicates one bit to Alice indicating whether x starts with 0 allows Alice, in case y starts with 1, to output 0. Otherwise Alice and Bob start the default protocol. Thus, for such x, y and P we have $CC_f^P(x,y) = 2$. By simple counting for some such inputs we have $C(x|P), C(y|P) \geqslant n - 1$. □

Generalizing this idea, every function that contains large monochromatic rectangles, of size say $2^{2n}/n^{O(1)}$, has many pairs x, y of complexity close to n for which the individual communication complexity drops to $O(\log n)$: In round 1 Bob tells Alice in which large rectangle (if any) his input is situated, by sending the index of the rectangle to Alice, and 0 otherwise. If Bob did send an index, and Alice's input is in that rectangle as well, then Alice outputs the color ("0" or "1") of the rectangle. Otherwise, Alice starts a default protocol.

5. Protocol-independent individual communication complexity of Identity

As before, let f be a function defined on pairs of strings of the same length. Assume that Alice has x, Bob has y and Alice wants to compute $f(x, y)$. A naive definition of the individual communication complexity of the value of the function f on the argument (x, y) is the number of communicated bits in the "best" communication protocol. Then, for every x, y there is a protocol with no communication at all on (x, y): the string y is hard wired into the protocol. To meaningfully capture the individual communication complexity of computing a function $f(x, y)$ we define now the following notion.

Definition 2. Let α be a natural number parameter. Let $TCC_f^\alpha(x, y)$ stand for the minimum $CC^P(x, y)$ over all total protocols P of complexity at most α that compute f (an all inputs, not only on x, y).

For $\alpha = n + O(1)$ we have $TCC_f^\alpha(x, y) = 0$ for all computable f and all x, y, since we can hard wire y into the protocol. Therefore it is natural to consider only α that are much smaller than n, say $\alpha = O(\log n)$. Since computation of the Identity function suffices to compute all other functions we have $TCC_f^{\alpha+O(1)}(x, y) \leqslant TCC_I^\alpha(x, y)$. The trivial lower bound is $TCC_f^\alpha(x, y) \geqslant C(f(x, y)|x) - \alpha - O(\log\alpha)$. For $f = I$ this gives $TCC_I^\alpha(x, y) \geqslant C(y|x) - \alpha - O(\log\alpha)$.

5.1. *One-way equals two-way for identity*

Let $TCC_{f,1\text{-way}}^\alpha(x, y)$ stand for the minimum $TCC^P(x, y)$ over all one-way (Bob sends a message to Alice) total protocols P of complexity at most α computing f (over all inputs, and not only on (x, y)). It is clear that $TCC_{f,1\text{-way}}^\alpha(x, y)$ does not depend on x: indeed, consider for given (x, y) the best protocol P; that protocol sends the same message on every other pair (x', y) hence

$$TCC_{f,1\text{-way}}^\alpha(x', y) \leqslant TCC_{f,1\text{-way}}^\alpha(x, y).$$

Therefore, we will henceforth drop x in the notation "$TCC_{f,1\text{-way}}^\alpha(x, y)$" replacing it by "$TCC_{f,1\text{-way}}^\alpha(y)$." Obviously,

$$TCC_f^\alpha(x, y) \leqslant TCC_{f,1\text{-way}}^\alpha(y)$$

for all α, x, y, f.

Surprisingly, for $f = I$, the Identity function, this inequality is an equality. That is, for total protocols "1-way" is as powerful as "2-way." More specifically, the following holds.

Theorem 5. *There is a constant C such that for all α, x, y we have*

$$TCC_{I,1\text{-way}}^{\alpha+C}(y) \leqslant TCC_I^\alpha(x, y).$$

Proof. Pick a two-way protocol P witnessing $TCC_I^\alpha(x, y) = l$. Let $c = c(x, y)$ be the conversation according to P between Alice and Bob on inputs x, y. The set of all pairs (x', y') such that the conversation between Alice and Bob on input (x', y') is equal to c is a rectangle, that is, has the form $X \times Y$, for some $X, Y \subset \{0, 1\}^n$. The set Y is a one-element set, as for every $y' \in Y$ Alice outputs y also on the input (x, y') (the output of Alice depends on c, P, x only).

Consider the following 1-way protocol P': find an x' with minimum $|c(x', y)|$ and send $c(x', y)$ to Alice. Alice then finds the set of all pairs $(x,'' y')$ such that the conversation between Alice and Bob on input $(x,'' y')$ is equal to $c(x', y)$. As we have seen that set has the form $X \times \{y\}$ for some X. Thus Alice knows y. As $|c(x', y)| \leqslant |c(x, y)| = TCC_I^\alpha(x, y)$ and $C(P'|P) = O(1)$ we are done. □

5.2. Possible behaviors of $TCC^{\alpha}_{I,1\text{-way}}(y)$

The function $TCC^{\alpha}_{I,1\text{-way}}(y)$, as a function of y, α, essentially coincides with *Kolmogorov Structure function* $f_y(i)$, studied in slightly different formulation in [11]. [1] It is defined by

$$f_y(i) = \min\{C(S)\colon\ S \ni y,\ |S| \leqslant 2^i\},$$

where $S \subseteq \{0,1\}^*$ is a finite set and $C(S)$ is the length (number of bits) in the shortest binary program from which the reference universal machine U computes a listing of the elements of S and then halts.

Definition 3. The *protocol size* function $g_y(i)$ is defined by

$$g_y(i) = \min\{\alpha\colon\ TCC^{\alpha}_{I,1\text{-way}}(y) \leqslant i\}$$

gives the minimal number of bits of a total protocol that transmits $y \in \{0,1\}^*$ in at most i bits of communication.

Theorem 6. *The protocol size function g_y coincides with the Kolmogorov Structure function f_y with logarithmic accuracy*:

$$g_y(i) = f_y(i) + O(\log n). \tag{9}$$

Proof. To prove the left inequality we have to transform a finite set $S \ni y$ into a one-way protocol P of complexity at most $\alpha = C(S) + O(1)$ communicating y in at most $\log|S|$ bits. The protocol just sends the index of y in S, or y literally if $y \notin S$.

To prove the right inequality we have to transform the one-way protocol P witnessing $TCC^{\alpha}_{I,1\text{-way}}(y)$ into a finite set $S \ni y$ of complexity at most $\alpha + O(\log n)$ with $\log|S| \leqslant TCC^{\alpha}_{I,1\text{-way}}(y)$. The set consists of all y' on which P sends the message of the same length l as the length of the message on y. Obviously, $|S| \leqslant 2^l = 2^{TCC^{\alpha}_{I,1\text{-way}}(y)}$ and to specify S we need a program describing P and l. Thus $C(S) \leqslant C(P) + O(\log l) \leqslant C(P) + O(\log n)$. □

For the properties of $f_y(i)$, which are also properties of $g_y(i)$ we refer to [11]. The following theorem is a reformulation of a result in [11] on possible behaviors of f_y.

Theorem 7.

(i) *For every string y of length n we have*

$$g_y(n) = O(1), \tag{10}$$
$$0 \leqslant g_y(i) - g_y(j) \leqslant j - i + O(\log n) \tag{11}$$

for all $i < j \leqslant n$.

(ii) *Conversely, let g be a function from* $\{0,1,\dots,n\}$ *to the naturals satisfying* (10) *and* (11) *with* $O(1)$, $O(\log n)$ *terms replaced by* 0. *Then there is a string y of length n with*

$$g_y(i) = g(i) + O(\log n + C(g)),$$

where $C(g)$ stands for the complexity of the graph of the function g.

Proof. (i) The equality $g_y(n) = O(1)$ is witnessed by the protocol that communicates y literally. The inequality $g_y(i) \geqslant g_y(j)$ for $i < j$ is straightforward. Let us prove the inequality $g_y(i) \leqslant g_y(j) + j - i + O(\log n)$. Let P be a protocol that communicates j bits on y and has complexity $g_y(j)$. Consider the new protocol P' that communicates

[1] We use the slightly non-standard notation "$S \ni y$" instead of the customary "$y \in S$" to stress that y is fixed and we range over all sets S that contain y as an element.

only the first i bits of the message communicated by P and the remaining $j - i$ of its bits are hard wired into P'. Obviously

$$C(P') \leqslant C(P) + (j - i) + O(\log n) = g_y(j) + (j - i) + O(\log n).$$

(ii) Let g satisfy the conditions of the theorem.

Claim 1. There is a string y of length n such that $g_y(i) \geqslant g(i) - \log(n + 1)$ for every $i = 0, 1, \ldots, n$.

Proof. We show that the number of strings that do not satisfy the claim is less than 2^n. Fix i. If $g_y(i) < g(i) - \log(n + 1)$ then there is a 1-way protocol P computing I of complexity less than $g(i) - \log(n + 1)$ communicating at most i bits on input y. Let $B_i(P)$ denote the set of all y' printed out by Alice if she runs P after receiving at most i bits from Bob. As $B_i(P) \leqslant 2^i$, summing over P we obtain at most $2^{i+g(i)-\log(n+1)}$ strings in the union of all $B_i(P)$. By condition of the theorem the function $i + g(i)$ is non-decreasing, therefore,

$$2^{i+g(i)-\log(n+1)} \leqslant 2^{n+g(n)-\log(n+1)} = 2^n/(n + 1).$$

Summing over i we obtain less than 2^n different y's with $g_y(i) < g(i) - \log(n + 1)$. □

We demonstrate that the lexicographically first y, as defined in Claim 1, also satisfies $g_y(i) \leqslant g(i) + O(\log n)$. Fix i. It suffices to construct a set $S \ni y$ of cardinality 2^i and of complexity at most $g(i) + O(\log n)$. To this end run the following:

Algorithm. Let A be a set variable initially containing all strings of length n, and let S be a set variable initially containing the 2^i first strings of A in lexicographical order. Run all programs of length at most n dovetail style. Every time, for some j, a program p of length less than $g(j) - \log(n + 1)$ halts, and p prints a protocol P, we remove all the elements of $B_j(P)$ from A (but not from S); we call a step at which this happens a *j-step*. Every time $S \cap A$ becomes empty at a j-step, we replace the contents of S by the set of the 2^i first strings in lexicographical order of (the current contents of) A. Possibly, the last replacement of S is incomplete because there are less than 2^i elements left in A. It is easy to see that $y \in S \setminus A$ just after the final replacement, and stays there forever after, even though some programs in the dovetailing process may still be running and elements from A may still be eliminated.

Claim 2. The contents of the set S is replaced at most $2^{g(i)+1}$ times.

Proof. There are two types of replacements that will be treated separately.

Case 1. Replacement of the current contents of S where at some j-step with $j \geqslant i$ *at least one* element was removed from the current contents $S \cap A$. Trivially, the number of this type of replacements is bounded by the number of j-steps with $j \geqslant i$, and hence by the number of programs of length less than $g(j) \leqslant g(i)$, that is, by $2^{g(i)}$. Here the inequality $g(j) \leqslant g(i)$ holds by the conditions of the theorem.

Case 2. Replacement of the current contents of S where *every one* of the 2^i elements of the current contents of S is removed from A by j-steps with $j < i$. Let us estimate the number of this type of replacements: Every element y removed at a j-step with $j < i$ belongs to a set $B_j(P)$ with $C(P) < g(j) - \log(n + 1)$. The overall cumulative number of elements removed from A on j-steps with $j < i$ is bounded by

$$\sum_{j<i} 2^{g(j)+j-\log(n+1)} \leqslant \sum_{j<i} 2^{g(i)+i-\log(n+1)} \leqslant 2^{g(i)+i},$$

where the first inequality holds by the conditions of the theorem. Hence replacements of the second type can happen at most $2^{g(i)+i-i} = 2^{g(i)}$ times. □

By Claim 2, S stabilizes after a certain number of j-steps. That number may be large. However, the number of replacements of S is small. The final set $S \ni y$ has cardinality 2^i, and can be specified by the number of replacements resulting in its current contents (as in Claim 2), and by i, n, g. This shows that $C(S) \leqslant g(i) + O(\log n + C(g))$. □

Example 1. One consequence of the theorem is as follows. Let k be an integer less than n. Apply the theorem to the function g where $g(i) = k$ for $i \leqslant n - k$ and $g(i) = n - i$ for $i \geqslant n - k$. For the string y of length n existing by the theorem we have $g_y(0) = k + O(\log n)$ (thus $C(y) = k + O(\log n)$) and

$$TCC^{\alpha}_{I,\text{1-way}}(y) > n - \alpha - O(\log n)$$

for all $\alpha < k - O(\log n)$. We call such strings y *non-communicable*. For example, with $k = (\log n)^2$ this shows that there are y of complexity $C(y) \approx (\log n)^2$ with $TCC^{\alpha}_{I,\text{1-way}}(y) \geqslant n - (\log n)^2$ for all $\alpha < C(y) - O(\log n)$ and $TCC^{\alpha}_{I,\text{1-way}}(y) = 0$ for all $\alpha > C(y) + O(1)$. That is, Bob can hold a highly compressible string y, but cannot use that fact to reduce the communication complexity significantly below $|y|$! Unless *all* information about y is hard wired in the (total) protocol the communication between Bob and Alice requires sending y almost completely literally. For such y, irrespective of x, the communication complexity is *exponential* in the complexity of y for all protocols of complexity less than that of y; when the complexity of the protocol is allowed to pass the complexity of y then the communication complexity suddenly drops to 0.

5.3. Protocols that can err or not halt on some input pairs

In the last section we allowed the protocol to depend on the strings x, y (as long as the complexity of the protocol does not exceed a certain level). Once we do it, it is natural to allow the protocol to err on inputs different from x, y and even not halt on them.

Definition 4. Let $CC^{\alpha}_{f}(x, y)$ stand for the minimum $CC^{P}(x, y)$ over all total protocols P of complexity at most α computing f correctly on input (x, y) (on other inputs P may output incorrect result). The minimum of the empty set is defined as ∞. The notation $CC^{\alpha}_{f,\text{1-way}}(x, y)$ is understood in the similar way. Let $PCC^{\alpha}_{f}(x, y)$ $[PCC^{\alpha}_{f,\text{1-way}}(x, y)]$ stand for the minimum $CC^{P}(x, y)$ over all partial [1-way] protocols P of complexity at most α computing f correctly on input (x, y) (on other inputs P may output incorrect result or not halt).

For instance, if f is a Boolean function then $CC^{O(1)}_{f}(x, y) = 0$ for all x, y (either the protocol outputting always 0 or the protocol outputting always 1 computes $f(x, y)$ for specific pair (x, y)). The definitions imply that

$$PCC^{\alpha}_{f}(x, y) \leqslant CC^{\alpha}_{f}(x, y) \leqslant TCC^{\alpha}_{f}(x, y).$$

Consider again the Identity function. We have the following obvious lower bound and upper bounds for $PCC^{\alpha}_{I}(x, y)$:

$$C(y|x) - \alpha - O(\log \alpha) \leqslant PCC^{\alpha}_{I}(x, y) \leqslant PCC^{\alpha}_{I,\text{1-way}}(x, y) \leqslant C(y) \tag{12}$$

for all α, x, y such that α is at least $\log C(y) + O(1)$. (To prove the last inequality, we hardwire the value $C(y)$ in the protocol using $\log C(y)$ bits. This enables Bob to find a shortest description of y of $C(y)$ bits and to send it to Alice; subsequently Alice decompresses the message received from Bob. Note Bob gets no instruction what to send if the complexity of his input is greater than $C(y)$. Therefore, this protocol is not total.)

In the next theorem we observe three facts: (i) In computing the Identity function, for some (x, y) total protocols that may err are more powerful than totally correct ones (the first statement in the theorem) and less powerful than partial protocols (the last statement in the theorem). (ii) In contrast to totally correct protocols, in partial and erring total protocols one can use Alice's string x: there are pairs (x, y) with

$$CC^{\alpha}_{I}(x, y) \ll PCC^{\alpha}_{I}\bigl(0^{n}, y\bigr)$$

(the first statement in the theorem). (iii) $CC^{\alpha}_{I}(x, y)$ can be much greater than the conditional complexity $C(y|x)$ (the second statement in the theorem for, say, $\alpha = n/3$). We use "$\gg$" to express that the left-hand side in the equation is much greater than the right-hand side.

Theorem 8. *For all n there is x such that*

$$TCC^{n/2-O(1)}_{I}(x, x) \geqslant TCC^{n/2}_{I,\text{1-way}}(x) \geqslant PCC^{n/2}\bigl(0^{n}, x\bigr) \geqslant n/2 - O(\log n) \gg CC^{O(1)}(x, x) = 0.$$

For all α, n, x there is y of length n such that $CC_I^{\alpha}(x, y) \geqslant n - \alpha$ and $C(y|x) \leqslant \alpha + O(1)$. For all n there are x, y with

$$CC_I^{n/3}(x, y) \geqslant 2n/3 \gg n/3 + O(1) \geqslant PCC_{I,\text{1-way}}^{\log n + O(1)}(x, y).$$

Proof. To prove the first statement, observe that for every n there is a 1-way protocol $P = P_n$ computable from n such that $CC^P(x, x) = 0$ (Alice outputs her string), thus $CC_I^{\alpha}(x, x) = 0$ if α exceeds a certain constant C. Let x be a string of length n with $C(x|0^n) \geqslant n$. Then

$$PCC_I^{n/2}(0^n, x) \geqslant C(x|0^n) - n/2 - O(\log n) \geqslant n/2 - O(\log n),$$

where the first inequality holds by (12).

To prove the second statement, fix a string x. By counting arguments, there is a string y with $CC_I^{\alpha}(x, y) \geqslant n - \alpha$. Indeed, there are less than $2^{\alpha+1}$ total protocols of complexity at most α. For each total protocol P there are at most $2^{n-\alpha-1}$ different y's with $CC^P(x, y) < n - \alpha$. Therefore the total number of y's with $CC_I^{\alpha}(x, y) < n - \alpha$ is less than $2^{\alpha+1}2^{n-\alpha-1} = 2^n$.

Let y be the first string with $CC_I^{\alpha}(x, y) \geqslant n - \alpha$. To identify y conditional on x we only need to know the number of total protocols of complexity at most α: given that number we enumerate all such protocols until we find all them. Given all those protocols and x we run all of them on all pairs (x, y) to find $CC_I^{\alpha}(x, y)$ (here we use that the protocols are total), and determine the first y for which $CC_I^{\alpha}(x, y) \geqslant n - \alpha$. Hence $C(y|x) \leqslant \alpha + O(1)$.

Applying the proved statement to the empty string x and to $\alpha = n/3$ we obtain a y of length n with $C(y) \leqslant n/3 + O(1)$ and $CC_I^{n/3}(x, y) \geqslant 2n/3$. The last inequality in (12) implies that

$$PCC_{I,\text{1-way}}^{\log n + O(1)}(x, y) \leqslant C(y) \leqslant n/3 + O(1). \qquad \square$$

The above results leave open the question of whether it is possible to separate PCC_I^{α} from $C(y|x)$. A deep result of An. Muchnik [8] implies that actually $PCC_I^{\alpha}(x, y)$ is close to $C(y|x)$ provided $\alpha \geqslant O(\log n)$.

Theorem 9 *(An. Muchnik). For all x, y of length n there is p such that $|p| \leqslant C(y|x) + O(\log n)$, $C(p|y) = O(\log n)$ and $C(y|p, x) = O(\log n)$, where the constants in $O(\log n)$ do not depend on n, x, y.*

Proof *(Sketch).* First we prove non-constructively that for all $k \leqslant n$ there exists a family $\{h_i\}$ of poly(n) hash functions mapping strings of length n to strings of length $k + O(\log n)$ having the following property. For every 2^k-element set of n-bit strings A and almost all $y \in A$ (the number of exceptions is less than $2^k/n^c$) there is a hash function h_i distinguishing y from other strings in A in the following sense. There are at most poly(n) strings y' in A with $h_i(y') = h_i(y)$. Such family can be found by exhaustive search, hence for every hash function h_i in the family we have $C(h_i) = O(\log n)$. Let then $k = C(y|x)$ and $A = \{y'\colon C(y'|x) \leqslant k\}$. We know that y is not an exception, as otherwise $C(y|x) \leqslant k - c\log n + O(\log n) < k$ provided c is large enough. Thus, there is a hash function h_i distinguishing y and we can let $p = h_i(y)$. $\square$

Corollary 1. *For all x, y of length n we have*

$$PCC_{I,\text{1-way}}^{O(\log n)}(x, y) \leqslant C(y|x) + O(\log n).$$

Proof. Let p be the program of Muchnik's theorem, let q be the program of length $O(\log n)$ for the reference computer to reconstruct p from y and let r the program of length $O(\log n)$ for the reference computer to reconstruct y from the pair (x, p). The protocol is as follows: Bob finds p from y, q and sends p to Alice; Alice reconstructs y from x, r. Both q and r are hardwired into the protocol, so its complexity is $O(\log n)$. This protocol is partial, as both Bob and Alice may be stuck when reconstructing p from y', q and y from x', r. $\square$

5.4. Two-way is better than one-way for partial and erring total protocols for Identity

Note that for the Identity function all our upper bounds hold for one-way protocols and all our lower bounds hold for two-way protocols. The following question arises: are two-way protocols more powerful than one-way ones (to compute the Identity function)? Theorem 5 implies that for total protocols it does not matter whether the communication is one-way or two-way. Muchnik's theorem implies that for some constant c for $\alpha = c\log n$ and for partial protocols again one-way is as powerful as two-way, as both one-way and two-way communication complexities are close to $C(y|x)$:

$$C(y|x) - O(\log n) \leqslant PCC(x,y)_I^\alpha(x,y) \leqslant PCC(x,y)_{I,\text{1-way}}^\alpha(x,y) \leqslant C(y|x) + O(\log n).$$

For erring total protocols and partial protocols and $\alpha < \log n$ the situation is different. It turns out that erring total two-way protocols are stronger than even partial one-way protocols (Corollary 2 below).

Theorem 10. *For every k, l, s such that $k \geqslant s + l2^s$ there are strings x, y of length $(2^s+1)k$ such that $CC_I^{O(1)}(x,y) \leqslant 2^s\log(2k)$ but $PCC_{I,\text{1-way}}^s(x,y) \geqslant l$.*

Proof. The string x will consist of 2^s+1 blocks, each of length k: $x = z_0z_1\dots z_{2^s}$. The string y will have the form $z_j00\dots0$ where z_j is a block from x.

To prove the upper bound consider the following two-way protocol: Alice finds a set of indexes $I = \{i_1,\dots,i_{2^s}\}$ such that for every distinct j, m there is $i \in I$ such that ith bit of z_j is different from ith bit of z_m (such set does exist, which may be shown by induction). Then she sends to Bob the string $i_1\dots i_{2^s}$ and Bob sends to Alice ith bit of y for all $i \in I$. Alice knows now y.

We need to find now particular $z_0, z_1, \dots, z_{2^s}$ such that no one-way protocol is effective on the pair (x,y) obtained from them in the specified way. To this end let $P_1,\dots,P_N$ be all the one-way partial protocols of complexity less than s computing the identity function. For every z and $i \leqslant N$ let $c(z,i)$ denote the message sent by Bob in protocol P_i when his input is $z00\dots0$ provided the length of the message is less than l. Otherwise let $c(z,i) = \infty$. Let $c(z)$ stand for the concatenation of $c(z,i)$ over all i. The range of $c(z)$ has $(2^l)^N < 2^{l2^s}$ elements. Hence there is c such that for more than $2^{k-2^sl} \geqslant 2^s$ different z's we have $c(z) = c$. Pick such c and pick different $z_0, z_1, \dots, z_{2^s}$ among those z's. Let y_j stand for the string obtained from z_j by appending 0s. We claim that $CC_I^{P_i}(x, y_j) \geqslant l$ for some j for all $i \leqslant N$. Assume that this is not the case. That is, for every j there are i such that $CC_I^{P_i}(x, y_j) < l$. There are $j_1 \neq j_2$ for which i is the same. As $c(z_{j_1}, i) = c(z_{j_2}, i) \neq \infty$ Alice receives the same message in P_i on inputs (x, y_{j_1}), (x, y_{j_2}) and should output both answers y_{j_1}, y_{j_2}, which is a contradiction. □

Corollary 2. *Let in the above theorem $s = (\log k)/3$ and $l = k^{2/3}/\log k$. These values satisfy the condition $k \geqslant s + l2^s$ and hence there are x, y of length about $k^{4/3}$ with almost quadratic gap between $CC_I^\alpha(x,y)$ and $PCC_{I,\text{1-way}}^\alpha(x,y)$:*

$$CC_I^{O(1)}(x,y) \leqslant k^{1/3}\log 2k \ll k^{2/3}/\log k \leqslant PCC_{I,\text{1-way}}^{(\log k)/3}(x,y).$$

Letting $s = \log\log k$ and $l = k/(2\log k)$ we obtain x, y of length about $k\log k$ with an exponential gap between $CC_I^\alpha(x,y)$ and $PCC_{I,\text{1-way}}^\alpha(x,y)$:

$$CC_I^{O(1)}(x,y) \leqslant \log k\log(2k) \ll k/(2\log k) \leqslant PCC_{I,\text{1-way}}^{\log\log k}(x,y).$$

We leave open the question whether $CC_I^\alpha(x,y)$ can be much less than $CC_{I,\text{1-way}}^\alpha(x,y)$ for α greater than $\log n$:

Question. Is it true that there for all c_1 there is c_2 such that for all x, y of length n we have

$$CC_{I,\text{1-way}}^{c_2\log n}(x,y) < CC_I^{c_1\log n}(x,y) + c_2\log n?$$

References

[1] H. Buhrman, T. Jiang, M. Li, P.M.B. Vitányi, New applications of the incompressibility method: Part II, Theoret. Comput. Sci. 235 (1) (2000) 59–70.

[2] H. Buhrman, M. Koucký, N. Vereshchagin, Randomized individual communication complexity, manuscript, CWI, 2006.

[3] G. Cormode, M. Paterson, S. Sahinalp, U. Vishkin, Communication complexity of document exchange, in: Proc. of the ACM–SIAM Symp. on Discrete Algorithms, 2000, pp. 197–206.

[4] A.N. Kolmogorov, Three approaches to the quantitative definition of information, Problems Inform. Transmission 1 (1) (1965) 1–7.

[5] E. Kushilevitz, N. Nisan, Communication Complexity, Cambridge University Press, 1997.

[6] M. Li, P.M.B. Vitányi, An Introduction to Kolmogorov Complexity and Its Applications, second ed., Springer-Verlag, New York, 1997.

[7] N. Lynch, Distributed Algorithms, Morgan Kaufmann, 1997.

[8] An.A. Muchnik, Conditional complexity and codes, Theoret. Comput. Sci. 271 (1/2) (2002) 97–111.

[9] R.J. Solomonoff, A formal theory of inductive inference, Part 1, Inform. Control 7 (1964) 1–22, Part 2, Inform. Control 7 (1964) 224–254.

[10] V.A. Uspensky, A. Shen, Relations between varieties of Kolmogorov complexities, Mathematical Systems Theory 29 (3) (1996) 271–292.

[11] N.K. Vereshchagin, P.M.B. Vitányi, Kolmogorov's structure functions and model selection, IEEE Trans. Inform. Theory 50 (12) (2004) 3265–3290.

[12] A.C. Yao, Some complexity questions related to distributive computing, in: Proc. 11th ACM Symposium on Theory of Computing, 1979, pp. 209–213.