Communication Complexity

Alexander A. Razborov

Abstract. When I was asked to write a contribution for this book about something related to my research, I immediately thought of Communication Complexity. This relatively simple but extremely beautiful and important sub-area of Complexity Theory studies the amount of *communication* needed for several distributed parties to learn something new. We will review the basic communication model and some of the classical results known for it, sometimes even with proofs. Then we will consider a variant in which the players are allowed to flip fair unbiased coins. We will finish with a brief review of more sophisticated models in which our current state of knowledge is less than satisfactory. All our definitions, statements and proofs are completely elementary, and yet we will state several open problems that have evaded strong researchers for decades.

1 Introduction

As the reader can guess from the name, Communication Complexity studies ways to arrange *communication* between several parties so that at the end of the day they learn what they are supposed to learn, and to do this in the most efficient, or least *complex* way. This theory constitutes a small but, as we will see below, very beautiful and important part of *Complexity Theory* which, in turn, is situated right at the intersection of Mathematics and Theoretical Computer Science. For this reason I would like to begin with a few words about Complexity Theory in general and what kind of problems researchers

Alexander A. Razborov
Department of Computer Science, The University of Chicago, 1100 East 58th Street, Chicago, IL 60637, USA; and Steklov Mathematical Institute, Moscow, Russia, 117 966. e-mail: `razborov@cs.uchicago.edu`

are studying there. The reader favoring concrete mathematical content over philosophy should feel free to skip the introduction and go directly to Section 2.

Complexity Theory is interested in problems that in most cases can roughly be described as follows. Assume that we have a task T that we want to accomplish. In most cases this involves one or more computers doing something, but it is not absolutely necessary. There can be many different ways to achieve our goal, and we denote the whole set of possibilities by $\mathcal{P}_T$. Depending on the context, elements of $\mathcal{P}_T$ can be called *algorithms* or, as in our article, *protocols*. In most cases of interest it is trivial that there is at least one algorithm/protocol to solve T, that is the set $\mathcal{P}_T$ is non-empty.

While all $P \in \mathcal{P}_T$ solve our original task T, not all solutions are born equal. Some of them may be better than others because they are shorter, consume less resources, are simpler, or for any other reason. The main idea of mathematical complexity theory is to try to capture our intuitive preferences by a positive real-valued function $\mu(P)$ $(P \in \mathcal{P}_T)$ called *complexity measure* with the idea that the smaller $\mu(P)$ is, the better is our solution P. Ideally, we would like to find the *best* solution $P \in \mathcal{P}_T$, that is the one minimizing the function $\mu(P)$. This is usually very hard to do, so in most cases researchers try to approach this ideal from two opposite sides as follows:

- try to find “reasonably good” solutions $P \in \mathcal{P}_T$ for which $\mu(P)$ may perhaps not be minimal, but still is “small enough”. Results of this sort are called “upper bounds” as what we are trying to do mathematically is to prove *upper bounds* on the quantity

$$\min_{P \in \mathcal{P}_T} \mu(P)$$

  that (not surprisingly!) is called the *complexity* of the task T.
- *Lower bound* problems: for some $a \in \mathbb{R}$ we try to show that $\mu(P) \geq a$ for *any* P, that is that there is no solution in $\mathcal{P}_T$ better than a. The class $\mathcal{P}_T$ is usually very rich and solutions $P \in \mathcal{P}_T$ can be based upon very different and often unexpected ideas. We have to take care of all of them with a uniform argument. This is why lower bound problems are amongst the most difficult in modern mathematics, and the overwhelming majority of them is still wide open.

All right, it is a good time for some examples. A great deal of mathematical olympiad problems are actually of complexity flavor even if it is not immediately clear from their statements.

> *You have 7 (or 2010, n, ...) coins, of which* 3 *(at most* 100*, an unknown number, ...) are counterfeit and are heavier (are lighter, weigh one ounce more, ...) than the others. You also have a scale that can weigh (compare, ...) as many coins as you like (compare at most 10 coins, ...). How many weighings do you need to identify all (one counterfeit, ...) coin(s)?*

These are typical complexity problems, and they are very much related to what is called *sorting networks and algorithms* in the literature. The task T is to identify counterfeit coins, and $\mathcal{P}_T$ consists of all sequences of weighings that allow us to accomplish it. The complexity measure $\mu(P)$ is just the length of P (that is, the number of weighings used).

> *You have a number (polynomial, expression, ...), how many additions/ multiplications do you need to build it from certain primitive expressions?*

Not only is this a complexity problem, but also a paradigmatic one. Can you describe $T, \mathcal{P}_T$ and μ in this case? And, by the way, if you think that the "school" method of multiplying integers is optimal in terms of the number of digit operations used, then this is incorrect. It was repeatedly improved in the work of Karatsuba [13], Toom and Cook (1966), Schönhage and Strassen [26] and Fürer [11], and it is still open whether Fürer's algorithm is the optimal one. It should be noted, however, that these advanced algorithms become more efficient than the "school" algorithm only for rather large numbers (typically at least several thousand digits long).

If you have heard of the famous **P** vs. **NP** question (otherwise I recommend to check out e.g. `http://www.claymath.org/millennium/P_vs_NP`), it is another complexity problem. Here T is the task of solving a fixed **NP**-complete problem, e.g. SATISFIABILITY, and $\mathcal{P}_T$ is the class of all deterministic algorithms fulfilling this task.

In this article we will discuss complexity problems involving *communication.* The model is very clean and easy to explain, but quite soon we will plunge into extremely interesting questions that have been open for decades... And, even if we will not have time to discuss it here at length, the ideas and methods of communication complexity penetrate today virtually all other branches of complexity theory.

Almost all material contained in our article (and a lot more) can be found in the classical book [17]. The recent textbook [3] on Computational Complexity has Chapter 13 devoted entirely to Communication Complexity, and you can find its applications in many other places all over the book.

Since the text involves quite a bit of notation, some of it is collected together at the end of the article, along with a brief description.

2 The Basic Model

The basic (deterministic) model was introduced in the seminal paper by Yao [27]. We have two players traditionally (cf. the remark below) called Alice and Bob, we have finite sets X, Y and we have a function $f : X \times Y \longrightarrow \{0, 1\}$. The task T_f facing Alice and Bob is to evaluate $f(x, y)$ for a given input (x, y). The complication that makes things interesting is that Alice holds the first part $x \in X$ of their shared input, while Bob holds another part $y \in Y$. They do have a two-sided communication channel, but it is something like a transatlantic phone line or a beam communicator with a spacecraft orbiting Mars. Communication is expensive, and Alice and Bob are trying to minimize the number of bits exchanged while computing $f(x, y)$.

Thus, a protocol $P \in \mathcal{P}_T$ looks like this (see Figure 1). Alice sends a message encoded for simplicity as a *binary string* a_1 (i.e., a finite sequence of zeroes and ones). Bob responds with some b_1 that depends only on his y *and* Alice's message a_1. They continue in this way until one of them (say, Bob) is able to compute the value of $f(x, y)$ and communicate it to Alice in the t-th round.

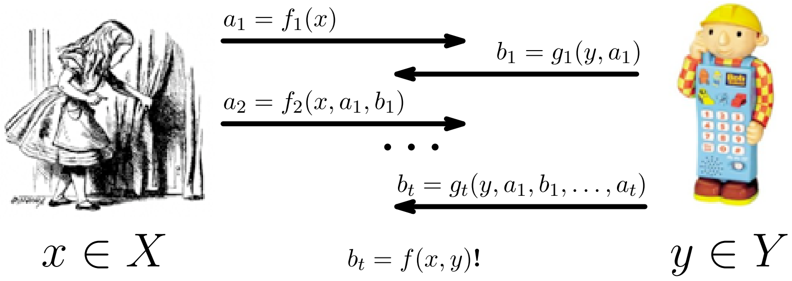

Fig. 1. Protocol P for computing $f(x, y)$. (Picture of Alice by John Tenniel.)

Remark 1. It should be noted that Alice and Bob are by far the most lovable and popular heroes in the whole literature on Complexity and the closely related field of Cryptography. As such, they are summoned up in many other episodes of this ongoing story, and, just like their real-life prototypes, they live a busy life. Sometimes their goals coincide only partially, and they are very cautious about leaking out unwanted information, then it is called Cryptography. Often there is an evil eavesdropper (usually called, for obvious reasons, Eve). Sometimes Alice and Bob do not even trust that the other party will truthfully follow the protocol, although in such cases they usually change their names to Arthur and Merlin. But in our article we will consider only the simplest scenario: complete mutual trust, nothing to hide, perfect and secure communication channel.

In this definition we deliberately left a few things imprecise. For example, is the *length* of Alice's message a_1 fixed or is it allowed to depend on x? Likewise, can the number of rounds t depend on x and y and, if so, how can Alice know that Bob's message b_t is actually the last one and already gives the final answer? It turns out, however, that all these details are very inessential, and the reader can fill them any way he or she likes — this will change the complexity only by a small additive factor.

How to measure the complexity $\mu(P)$ of this protocol P? There are several ways of doing this, all of them reasonable. In this article we will focus only on the most important and popular model called *worst-case complexity.* For any *given* input $(x, y) \in X \times Y$ we define the *cost* of the protocol P on this input as the total number of bits[1] $|a_1| + |b_1| + \ldots + |b_t|$ exchanged on this input (cf. Figure 1). And then we define the complexity (that, for historical reasons, is also called *cost* in this case) $\mathrm{cost}(P)$ of the protocol P as the *maximal* cost of P over all inputs $(x, y) \in X \times Y$. Finally, the *communication complexity* $C(f)$ of (computing) the function $f : X \times Y \longrightarrow \{0, 1\}$ is defined as the *minimum* $\min_{P \in \mathcal{P}_f} \mathrm{cost}(P)$ taken over all legitimate protocols P, i.e., those protocols that correctly output the value $f(x, y)$ for all possible inputs. We would like to be able to compute $C(f)$ for "interesting" functions f, or at least get good estimates for it.

The first obvious remark is that

$$C(f) \leq \lceil \log_2 |X| \rceil + 1 \tag{1}$$

for any problem[2] f. The protocol of this cost is very simple: Alice encodes her input x as a binary string of length $\lceil \log_2 |X| \rceil$ using any injective encoding $f_1 : X \longrightarrow \{0, 1\}^{\lceil \log_2 |X| \rceil}$ and sends $a_1 = f_1(x)$ to Bob. Then Bob decodes the message (we assume that the encoding scheme f_1 is known to both parties in advance!) and sends the answer $f(f_1^{-1}(a_1), y)$ back to Alice.

Surprisingly, there are only very few interesting functions f for which we can do significantly better than (1) in the basic model. One example that is sort of trivial is this. Assume that X and Y consist of integers not exceeding some fixed N: $X = Y = \{1, 2, \ldots, N\}$. Alice and Bob want to compute the $\{0, 1\}$-valued function $f_N(x, y)$ that outputs 1 if and only if $x + y$ is divisible by 2010. A much more economical way to solve this problem would be for Alice to send to Bob not her whole input x, but only its remainder $x \mod 2010$. Clearly, this still will be sufficient for Bob to compute $x + y \mod 2010$ (and hence also $f_N(x, y)$), and the cost of this protocol is only $\lceil \log_2 2010 \rceil + 1 \, (= 12)$. Thus,

$$C(f_N) \leq \lceil \log_2 2010 \rceil + 1\,. \tag{2}$$

[1] $|a|$ is the length of the binary word a.

[2] Note that complexity theorists often identify functions f with computational problems they naturally represent. For example, the equality *function* EQ_N defined below is also viewed as the *problem* of checking if two given strings are equal.

Now, complexity theorists are lazy people, and not very good at elementary arithmetic. What is really remarkable about the right-hand side of (2) is that it represents *some* absolute constant that magically does not depend on the input size at all! Thus, instead of calculating this expression, we prefer to stress this fact using the mathematical *big-O* notation and write (2) in the simpler, even if weaker, form

$$C(f_N) \leq O(1)\,.$$

This means that there exists a positive universal constant $K > 0$ that anyone interested can (usually) extract from the proof such that for all N we have $C(f_N) \leq K \cdot 1 = K$. Likewise, $C(f_N) \leq O(\log_2 N)$ would mean that $C(f_N) \leq K \log_2 N$ etc. We will extensively use this standard[3] notation in our article.

Let us now consider a simpler problem that looks as fundamental as it can only be. We assume that $X = Y$ are equal sets of cardinality N. The reader may assume that this set is again $\{1, 2, \ldots, N\}$, but now this is not important. The *equality function* EQ_N is defined by letting $\mathrm{EQ}_N(x, y) = 1$ if and only if $x = y$. In other words, Alice and Bob want to check if their files, databases etc. are equal, which is clearly an extremely important task in many applications.

We can of course apply the trivial bound (1), that is, Alice can simply transmit her whole input x to Bob. But can we save even a little bit over this trivial protocol? At this point I would like to strongly recommend you to put this book aside for a while and try out a few ideas toward this goal. That would really help to better appreciate what will follow.

[3] We should warn the reader that in most texts this notation is used with the equality, rather than inequality, sign, i.e., $C(f_N) = O(\log_2 N)$ in the previous example. However, we see numerous issues with this usage and in particular it becomes rather awkward and uninformative in complicated cases.

3 Lower Bounds

Did you have any luck? Well, you do not have to be distressed by the result since it turns out that the bound (1) actually can *not* be improved, that is *any* protocol for EQ_N must have cost at least $\log_2 N$. This was proven in the same seminal paper by Yao [27], and many ideas from that paper determined the development of Complexity Theory for several decades to follow. Let us see how the proof goes, the argument is not very difficult but it is very instructive.

We are given a protocol P of the form shown on Figure 1, and we know that upon executing this protocol Bob knows $\mathrm{EQ}_N(x, y)$. We should somehow conclude that $\mathrm{cost}(P) \geq \log_2 N$.

One very common mistake often made by new players in the lower bounds game is that they begin telling P what it "ought to do", that is, consciously or unconsciously, begin making assumptions about the best protocol P based on the good common sense. In our situation a typical argument would start off by something like "let i be the first bit in the binary representation of x and y that the protocol P compares". "Arguments" like this are dead wrong since it is not clear at all that the best protocol should proceed in this way, or, to that end, in any other way we would consider "intelligent". Complexity Theory is full of ingenious algorithms and protocols that do something strange and apparently irrelevant almost all the way down, and only at the end of the day they conjure the required answer like a rabbit from the hat — we will see one good example below. The beauty and the curse of Complexity Theory is that we should take care of all protocols with seemingly irrational (in our opinion) behavior all the same, and in our particular case we may not assume *anything* about the protocol P besides what is explicitly shown on Figure 1.

Equipped with this word of warning, let us follow Yao and see what useful information we still can retrieve from Figure 1 alone. Note that although we are currently interested in the case $f = \mathrm{EQ}_N$, Yao's argument is more general and can be applied to any function f. Thus, for the time being we assume that f is an arbitrary function whose communication complexity we want to estimate; we will return to EQ_N in Corollary 2.

The first thing to do is to introduce an extremely useful concept of a *history* or a *transcript*: this is the whole sequence $(a_1, b_1, \ldots, a_t, b_t)$ of messages exchanged by Alice and Bob during the execution of the protocol on some particular input. This notion is very broad and general and is successfully applied in many different situations, not only in communication complexity.

Next, we can observe that there are at most $2^{\mathrm{cost}(P)}$ different histories as there are only that many different strings[4] of length $\mathrm{cost}(P)$. Given any fixed history h, we can form the set R_h of all those inputs (x, y) that lead to this history. Let us see what we can say about these sets.

[4] Depending on finer details of the model, histories may have different length, the placement of commas can be also important etc. that might result in a slight increase of this number. But remember that we are lazy and prefer to ignore small additive, or even multiplicative factors.

First of all, every input (x, y) leads to one and only one history. This means that the collection $\{R_h\}$ forms a *partition* or *disjoint covering* of the set of all inputs $X \times Y$:

$$X \times Y = \dot{\bigcup}_{h\in\mathcal{H}} R_h \,, \tag{3}$$

where $\mathcal{H}$ is the set of all possible histories. The notation $\dot{\bigcup}$ stands for *disjoint union* and simultaneously means two different things: that $X \times Y = \bigcup_{h\in\mathcal{H}} R_h$, and that $R_h \cap R_{h'} = \emptyset$ for any two different histories $h \neq h' \in \mathcal{H}$.

Now, every history h includes the value of the function $f(x, y)$ as Bob's last message b_t. That is, any R_h is an *f-monochromatic* set, which means that either $f(x, y) = 0$ for all $(x, y) \in R_h$ or $f(x, y) = 1$ for all such (x, y).

Finally, and this is very crucial, every R_h is a *combinatorial rectangle* (or simply a rectangle), that is it has the form $R_h = X_h \times Y_h$ for some $X_h \subseteq X$, $Y_h \subseteq Y$. In order to understand why, we should simply expand the sentence "(x, y) leads to the history $(a_1, b_1, \ldots, a_t, b_t)$". Looking again at Figure 1, we see that this is equivalent to the set of "constraints" on (x, y) shown there: $f_1(x) = a_1,\ g_1(y, a_1) = b_1,\ f_2(x, a_1, b_1) = a_2, \ldots, g_t(y, a_1, \ldots, a_t) = b_t$. Let us observe that odd-numbered constraints in this chain depend only on x (remember that h is fixed!); let us denote by X_h the set of those $x \in X$ that satisfy all these constraints. Likewise, let Y_h be the set of all $y \in Y$ satisfying even-numbered constraints. Then it is easy to see that we precisely have $R_h = X_h \times Y_h$!

Let us summarize a little bit. For any protocol P solving our problem $f : X \times Y \longrightarrow \{0, 1\}$, we have been able to chop $X \times Y$ into at most $2^{\mathrm{cost}(P)}$ pieces so that each such piece is an f-monochromatic combinatorial rectangle. Re-phrasing it a little bit differently, let us denote by $\chi(f)$ (yes, complexity theorists love to introduce complexity measures!) the minimal number of f-monochromatic rectangles into which we can partition $X \times Y$. We thus have proved, up to a small multiplicative constant that may depend on finer details of the model:

Theorem 1 (Yao). *$C(f) \geq \log_2 \chi(f)$.* □

Let us return to our particular case $f = \mathrm{EQ}_N$. All f-monochromatic combinatorial rectangles can be classified into 0-*rectangles* (i.e., those on which f is identically 0) and 1-*rectangles*. The function EQ_N has many large 0-rectangles. (Can you find one?) But all its 1-rectangles are very primitive, namely every such rectangle consists of just one point (x, x). Therefore, in order to cover even the "diagonal" points $\{\,(x, x) \mid x \in X\,\}$, one needs N different 1-rectangles, which proves $\chi(\mathrm{EQ}_N) \geq N$. Combining this with Theorem 1, we get the result we were looking for:

Corollary 2. *$C(\mathrm{EQ}_N) \geq \log_2 N$.* □

Exercise 1. *The function* LE_N *(less-or-equal) is defined on* $\{1, 2, \ldots, N\} \times \{1, 2, \ldots, N\}$ *as*

$$\mathrm{LE}_N(x, y) = 1 \text{ iff } x \leq y \,.$$

Prove that $C(\mathrm{LE}_N) \geq \log_2 N$.

Exercise 2 (difficult). *The function* DISJ_n *is defined on* $\{0,1\}^n \times \{0,1\}^n$ *as*

$$\mathrm{DISJ}_n(x,y) = 1 \textit{ iff } \forall i \leq n : \ x_i = 0 \vee y_i = 0 \ ,$$

that is, the sets of positions where the strings x *and* y *have a 1 are disjoint. Prove that* $C(\mathrm{DISJ}_n) \geq \Omega(n)$.

(Here Ω is yet another notation that complexity theorists love. It is dual to "big-O" and means that there exists a constant $\varepsilon > 0$ that we do not want to compute such that $C(\mathrm{DISJ}_n) \geq \varepsilon n$ for all n.)

Hint. How many points (x,y) with $\mathrm{DISJ}_n(x,y) = 1$ do we have? And what is the maximal size of a 1-rectangle?

4 Are These Bounds Tight?

The next interesting question is, how good is Theorem 1 in general? Can it be the case that $\chi(f)$ is small, that is we do have a good disjoint covering by f-monochromatic rectangles, and nonetheless $C(f)$ is large, so that in particular we can not convert our covering into a decent communication protocol? Figure 2 suggests at least that this question may be non-trivial: it gives an example of a disjoint covering by only five rectangles that does not correspond to any communication protocol.

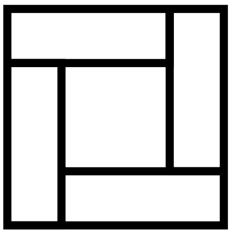

Fig. 2. What should Alice do?

As in many similar situations, the answer depends on how precise you want it to be. In the next influential paper on communication complexity [1], the following was proved among other things:

Theorem 3 (Aho, Ullman, Yannakakis). $C(f) \leq O(\log_2 \chi(f))^2$.

The proof is not very difficult, but still highly non-trivial. The reader can try to find it by himself or consult e.g. [17].

Can we remove the square in Theorem 3? For almost thirty years that have elapsed since the paper [1], many people have tried to resolve the question one or the other way. But it has resisted all efforts so far...

Open Problem 1. *Is it true that* $C(f) \leq O(\log_2 \chi(f))$*?*

Besides Theorem 3, the paper [1] contains many other great things pertaining to the so-called *non-deterministic communication complexity.* In this model, Alice and Bob are also given access to a shared string z not determined by the protocol (whence comes the name) but rather given to them by a third all-powerful party trying to convince them that $f(x, y) = 1$. We require that a convincing string z exists if and only if $f(x, y)$ is *indeed* equal to 1, and we note that in this definition we give up on the symmetry of answers 0 and 1. Due to lack of space we discuss this important concept only very briefly, and complexity measures we mention during the discussion will hardly be used in the rest of the article.

Define $t(f)$ in the same way as $\chi(f)$, only now we allow the monochromatic rectangles in our cover to overlap with each other. Clearly, $t(f) \leq \chi(f)$, but it turns out that the bound of Theorem 3 still holds: $C(f) \leq O(\log_2 t(f))^2$. On the other hand, there are examples for which $C(f)$ is of order $(\log_2 t(f))^2$. This means that the (negative) solution to the analogue of Problem 1 for not necessarily disjoint coverings is known.

Let $\chi_0(f)$ and $\chi_1(f)$ be defined similarly to $\chi(f)$, except now we are interested in a disjoint rectangular covering of only those inputs that yield value 0 (respectively, value 1); note that $\chi(f) = \chi_0(f) + \chi_1(f)$. Then still $C(f) \leq O(\log_2 \chi_1(f))^2$ and (by symmetry) $C(f) \leq O(\log_2 \chi_0(f))^2$. By analogy, we can also define the quantities $t_0(f)$ and $t_1(f)$ (the non-deterministic communication complexity we mentioned above turns out to be equal to $\log_2 t_1(f)$). We cannot get any reasonable (say, better than exponential) bound on $C(f)$ in terms of $\log_2 t_1(f)$ or $\log_2 t_0(f)$ only: for example, $t_0(\mathrm{EQ}_N) \leq O(\log_2 N)$ (why?) while, as we already know, $C(\mathrm{EQ}_N) \geq \log_2 N$. In conclusion, there is no good bound on the deterministic communication complexity in terms of the non-deterministic one, but such a bound becomes possible if we know that the non-deterministic communication complexity of the negated function is also small.

The next landmark paper we want to discuss is the paper [19] that introduced to the area *algebraic methods.* So far all our methods for estimating $\chi(f)$ from below (Corollary 2 and Exercises 1 and 2) were based on the same unsophisticated idea: select "many" inputs $D \subseteq X \times Y$ such that every f-monochromatic rectangle R may cover only "a few" of them, and then apply the pigeonhole principle. This method does not use anyhow that the covering (3) is disjoint or, in other words, it can be equally well applied to bounding from below $t(f)$ as well as $\chi(f)$. Is it good or bad? The answer depends. It is always nice, of course, to be able to prove more results, like lower bounds on the *non-deterministic* communication complexity $\log_2 t_1(f)$, with the same shot. But sometimes it turns out that the quantity analogous to $t(f)$ is *always* small and, thus, if we still want to bound $\chi(f)$ from below, we must use methods that "feel" the difference between these two concepts. The *rank lower bound* of Mehlhorn and Schmidt [19] was the first of such methods.

We will need the most basic concepts from linear algebra like a *matrix* M or its *rank* $\mathrm{rk}(M)$, as well as their simplest properties. If the reader is not yet familiar with them, then this is a perfect opportunity to grab any textbook in basic linear algebra and read a couple of chapters from it. You will have to learn this eventually anyway, but now you will also immediately see quite an unexpected and interesting application of these abstract things.

Given any function $f : X \times Y \longrightarrow \{0,1\}$, we can arrange its values in the form of the *communication matrix* M_f. Rows of this matrix are enumerated by elements of X, its columns are enumerated by elements of Y (the order is unimportant in both cases), and in the intersection of the x-th row and the y-th column we write $f(x,y)$. The following result relates two quite different worlds, those of combinatorics and linear algebra.

Theorem 4. $\chi(f) \geq \mathrm{rk}(M_f)$.

Proof. The proof is remarkably simple. Let $R_1, \ldots, R_\chi$ be disjoint 1-rectangles covering all (x,y) with $f(x,y) = 1$ so that $\chi \leq \chi(f)$. Let $f_i : X \times Y \longrightarrow \{0,1\}$ be the *characteristic* function of the rectangle R_i, i.e., $f_i(x,y) = 1$ if and only if $(x,y) \in R_i$, and let $M_i = M_{f_i}$ be its communication matrix. Then $\mathrm{rk}(M_i) = 1$ (why?) and $M_f = \sum_{i=1}^{\chi} M_i$. Therefore, $\mathrm{rk}(M_f) \leq \sum_{i=1}^{\chi} \mathrm{rk}(M_i) \leq \chi \leq \chi(f)$. □

In order to fully appreciate how useful Theorem 4 is, let us note that M_{EQ_N} is the identity matrix (we tacitly assume that if $X = Y$ then the orders on rows and columns are consistent) and, therefore, $\mathrm{rk}(M_{\mathrm{EQ}_N}) = N$. This immediately gives Corollary 2. M_{LE_N} is the upper triangular matrix, and therefore we also have $\mathrm{rk}(M_{\mathrm{LE}_N}) = N$. Exercise 1 follows. It does require a little bit of thinking to see that the communication matrix M_{DISJ_n} is nonsingular, that is $\mathrm{rk}(M_{\mathrm{DISJ}_n}) = 2^n$. But once it is done, we immediately obtain $C(\mathrm{DISJ}_n) \geq n$ which is essentially tight by (1) and also *stronger* than what we could do with combinatorial methods in Exercise 2 (the Ω is gone).

How tight is the bound of Theorem 4? It had been conjectured for a while that perhaps $\chi(f) \leq (\mathrm{rk}(M_f))^{O(1)}$ or maybe even $\chi(f) \leq O(\mathrm{rk}(M_f))$. In this form the conjecture was disproved in the series of papers [2, 23, 21]. But it is still possible and plausible that, say,

$$\chi(f) \leq 2^{O(\log_2 \mathrm{rk}(M_f))^2} \ ;$$

note that in combination with Theorem 3 that would still give a highly nontrivial inequality $C(f) \leq O(\log_2 \mathrm{rk}(M_f))^4$.

Despite decades of research, we still do not know the answer, and we actually do not have a very good clue how to even approach this problem that has become notoriously known as the *Log-Rank Conjecture*:

Open Problem 2 (Log-Rank Conjecture). *Is it true that*

$$\chi(f) \leq 2^{(\log_2 \mathrm{rk}(M_f))^{O(1)}} \quad ?$$

Equivalently (by Theorems 1, 3), is it true that $C(f) \leq (\log_2 \mathrm{rk}(M_f))^{O(1)}$*?*

5 Probabilistic Models

This is all we wanted to say about the basic model of communication complexity. Even more fascinating and difficult problems arise when we introduce some variations. The most important of them, and the only one that we treat in sufficient detail in the rest of this article, is the model of *probabilistic communication complexity.*

Assume that Alice and Bob are now slightly less ambitious and agree to tolerate some small probability of error when computing the value of $f(x, y) \in \{0, 1\}$. Both of them are equipped with a fair unbiased coin (scientifically known as *generator of random bits*) that they can toss during the execution of the protocol, and adjust the messages they send to each other according to the result. Everything else is the same (that is, as on Figure 1) but we have to specify what it means that the protocol P correctly computes the function f.

Fix an input (x, y) and assume that Alice and Bob together flip their coins r times during the execution, which gives 2^r possible outcomes of these coin tosses. Some of them are *good* in the sense that Bob outputs the correct value $f(x, y)$, but some are *bad* and he errs. Let $\mathrm{Good}(x, y)$ be the set of all good outcomes, then the quantity

$$p_{xy} = \frac{|\,\mathrm{Good}(x, y)|}{2^r} \tag{4}$$

is for obvious reasons called the *probability of success* on the input (x, y).

What do we want from it? There is a very simple protocol of cost 1 that achieves $p_{xy} = 1/2$: Bob simply tosses his coin and claims that its outcome is $f(x, y)$. Thus, we definitely want to demand that

$$p_{xy} > 1/2\,. \tag{5}$$

But how well should the probability of success be separated from 1/2?

It turns out that there are essentially only three different possibilities (remember that we are lazy and do not care much about exact values of our constants). In the most popular and important version we require that $p_{xy} \geq 2/3$ for *any* input (x, y). The minimal cost of a probabilistic protocol that meets this requirement is called *bounded-error probabilistic communication complexity of the function* f and denoted by $R(f)$. If for any input pair (x, y) we only require (5) then the model is called *unbounded-error*, and the corresponding complexity measure is denoted by $U(f)$. In the third model (that is less known and will not be considered in our article), we still require (5), but now Alice and Bob are also charged for coin tosses. This e.g. implies that in any protocol of cost $O(\log_2 n)$, (5) automatically implies the better bound $p_{x,y} \geq \frac{1}{2} + \frac{1}{p(n)}$ for some polynomial $p(n)$.

Why, in the definition of $R(f)$, did we request that $p_{xy} \geq 2/3$, not $p_{xy} \geq 0.9999$? By using quite a general technique called *amplification*, it can be shown not to be very important. Namely, assume that Alice and Bob have at their disposal a protocol of cost $R(f)$ that achieves $p_{xy} \geq 2/3$, they repeat it independently 1000 times and output at the end the most frequent answer. Then the error probability of this repeated protocol of cost only $1000\,R(f)$ will not exceed 10^{-10}... (In order to prove this statement, some knowledge of elementary probability theory, like Chernoff bounds, is needed.)

Are coins really helpful for anything, that is are there any interesting problems that can be more efficiently solved using randomization than without it? An ultimate answer to this question is provided by the following beautiful construction, usually attributed to Rabin and Yao, that has to be compared with Corollary 2.

Theorem 5. *$R(\mathrm{EQ}_N) \leq O(\log_2 \log_2 N)$.*

Proof. In order to prove this, it is convenient to represent elements of X and Y as binary strings of length n, where $n = \lceil \log_2 N \rceil$. Furthermore, we want to view the binary string $x_1 x_2 \dots x_n$ as the polynomial $x_1 + x_2\xi + \dots + x_n\xi^{n-1}$ in a variable ξ. Thus, Alice and Bob hold two polynomials $g(\xi)$ and $h(\xi)$ of the form above, and they want to determine if these polynomials are equal. For doing that they agree beforehand on a fixed prime number $p \in [3n, 6n]$ (such a prime always exists by Chebyshev's famous theorem). Alice tosses her coin to pick a random element $\xi \in \{0, 1, \dots, p-1\}$. Then she computes the remainder (!) $g(\xi) \mod p$ and sends the pair $(\xi,\ g(\xi) \mod p)$ to Bob. Bob evaluates $h(\xi) \mod p$ and outputs 1 if and only if $h(\xi) \mod p$ is equal to the value $g(\xi) \mod p$ he received from Alice.

The cost of this protocol is only $O(\log_2 n)$ as required: this is how many *bits* you need to transmit a pair of integers $(\xi,\ g(\xi) \mod p)$ not exceeding $p \leq O(n)$ each. What about the probability of success? If $\mathrm{EQ}(g, h) = 1$ then $g = h$ and Bob clearly always outputs 1, there is no error in this case at all. But what will happen if $g \neq h$? Then $(h - g)$ is a *non-zero* polynomial of degree at most n. And any such polynomial can have at most n different roots in the finite field $\mathbb{F}_p$. If you do not understand the last sentence, then you can simply trust me that the number of bad $\xi \in \{0, 1, \dots, p-1\}$, i.e., those for which Bob is fooled by the fact $g(\xi) = h(\xi) \mod p$, does not exceed $n \leq \frac{p}{3}$. And since ξ was chosen completely at random from $\{0, 1, \dots, p-1\}$, this precisely means that the probability of success is at least $2/3$. □

Let us now review other problems that we already saw before in the light of probabilistic protocols. The function less-or-equal from Exercise 1 also gives in to such protocols: $R(\mathrm{LE}_N) \leq O(\log_2 \log_2 N)$, although the proof is *way* more complicated than for equality [17, Exercise 3.18]. On the other hand, randomization does not help much for computing the disjointness function [4, 12, 24]:

Theorem 6. *$R(\mathrm{DISJ}_n) \geq \Omega(n)$.*

The proof is too complicated to discuss here. It becomes slightly easier for another important function, *inner product mod 2,* that we now describe.

Given $x, y \in \{0,1\}^n$, we consider, like in the case of disjointness, the set of all indices i for which $x_i = 1$ and $y_i = 1$. Then $\mathrm{IP}_n(x,y) = 1$ if the cardinality of this set is odd, and $\mathrm{IP}_n(x,y) = 0$ if it is even. Chor and Goldreich [9] proved the following:

Theorem 7. *$R(\mathrm{IP}_n) \geq \Omega(n)$.*

The full proof is still too difficult to be included here, but we would like to highlight its main idea.

So far we have been interested only in f-monochromatic rectangles, i.e., those that are composed either of zeros only or of ones only. We typically wanted to prove that every such rectangle is small in a sense. In order to tackle probabilistic protocols, we need to consider *arbitrary* rectangles R. Every such rectangle has a certain number $N_0(f,R)$ of points with $f(x,y) = 0$, and $N_1(f,R)$ points with $f(x,y) = 1$. We need to prove that even if R is "large" then it is still "well balanced" in the sense that $N_0(f,R)$ and $N_1(f,R)$ are "close" to each other. Mathematically, the *discrepancy under uniform distribution*[5] of the function $f : X \times Y \longrightarrow \{0,1\}$ is defined as

$$\mathrm{Disc}_u(f) = \max_R \frac{|N_0(f,R) - N_1(f,R)|}{|X| \times |Y|}\,,$$

where the maximum is taken over all possible combinatorial rectangles $R \subseteq X \times Y$.

It turns out that

$$R(f) \geq \Omega(\log_2(1/\,\mathrm{Disc}_u(f)))\,, \tag{6}$$

that is, low discrepancy implies good lower bounds for probability protocols. Then the proof of Theorem 7 is finished by proving $\mathrm{Disc}_u(\mathrm{IP}_n) \leq 2^{-n/2}$ (which is rather non-trivial).

What happens if we go further and allow probabilistic protocols with unbounded error, that is we only require the success probability (4) to be strictly greater than 1/2? The complexity of the equality function deteriorates completely [20]:

Theorem 8. *$U(\mathrm{EQ}_N) \leq 2$.*

[5] This concept can be generalized to other distributions as well.

The disjointness function also becomes easy, and this is a good exercise:

Exercise 3. *Prove that $U(\mathrm{DISJ}_n) \leq O(\log_2 n)$.*

The inner product, however, still holds the fort:

Theorem 9. *$U(\mathrm{IP}_n) \geq \Omega(n)$.*

This result by Forster [10] is extremely beautiful and ingenious, and it is one of my favorites in the whole Complexity Theory.

6 Other Variations

We conclude with briefly mentioning a few modern directions in communication complexity where current research is particularly active.

6.1 Quantum Communication Complexity

Well, I will not even attempt to define what *quantum computers* are or if they have anything to do with the Quantum of Solace — most readers have probably heard of these still imaginary devices. Let me just say that they can be utilized for solving communication problems as well [28] and denote by $Q(f)$ the corresponding complexity measure. Quantum computers have an implicit access to random bits that implies $Q(f) \leq R(f)$. On the other hand, the discrepancy lower bound (6) still holds for quantum protocols [16] that gives for them the same bound as in Theorem 7. Something more interesting happens to the disjointness function: its complexity drops from n to $\sqrt{n}$ [7, 25]. Can a quantum communication protocol save more than a quadratic term over the best probabilistic protocol? This is one of the most important and presumably very difficult problems in the area:

Open Problem 3. *Is it true that $R(f)$ is bounded by a polynomial in $Q(f)$ for functions $f : X \times Y \longrightarrow \{0,1\}$?*

6.2 Multiparty Communication Complexity

Now we have more than 2 players, Alice, Bob, Claire, Dylan, Eve..., who collectively want to evaluate some function f. Depending on how the input to f is distributed among the players, there are several different models, the simplest being the scenario in which every player is holding her own set of data not known by any of the others. It turns out, however, that the most important one of them (by the token of having a *really* great deal of various applications) is the following *number-on-the-forehead* model. In this model, k players still want to evaluate a function $f(x^1, \ldots, x^k)$, $x^i \in \{0,1\}^n$. An interesting twist is that the i-th player has x^i written on his forehead, so he can actually see *all pieces of the input except for his own.* Let $C^k(f)$ as always be the minimal number of bits the players have to exchange to correctly compute $f(x^1, \ldots, x^k)$; for simplicity we assume that every message is broadcasted to all other players at once.

Our basic functions DISJ_n and IP_n have "unique" natural generalizations DISJ_n^k and IP_n^k in this model. (Can you fill in the details?) The classical paper [5] proved the following bound:

Theorem 10. *$C^k(\mathrm{IP}_n^k) \geq \Omega(n)$ as long as $k \leq \varepsilon \log_2 n$ for a sufficiently small constant $\varepsilon > 0$.*

If we only could improve this result to a larger number of players (even for any other "good" function f), that would give absolutely fantastic consequences in complexity theory, some of which are outlined already in [5]. But this seems to be well out of reach of all methods that we currently have at our disposal.

Open Problem 4. *Prove that $C^k(\mathrm{IP}_n^k) \geq n^\varepsilon$ for, say, $k = \lceil (\log_2 n)^2 \rceil$ and some fixed constant $\varepsilon > 0$.*

The multiparty communication complexity of DISJ_n^k was completely unknown for quite a while even for $k = 3$. Very recent breakthrough [8, 18, 6] gives lower bounds on $C^k(\mathrm{DISJ}_n^k)$ that are non-trivial up to $k = \varepsilon(\log_2 n)^{1/3}$ players.

6.3 Communication Complexity of Search Problems

So far we have been considering functions that assume only two values, 0 and 1. In complexity theory such functions are often identified with *decision problems* or *languages*. But we can also consider functions of more general form $f : X \times Y \longrightarrow Z$, where Z is some more complicated finite set. Or we can go even one step further and assume that the function f is *multi-valued*, or in other words, we have a ternary relation $R \subseteq X \times Y \times Z$ such that for any pair (x, y) there exists at least one $z \in Z$ (a "value" of the multi-valued function f) such that $(x, y, z) \in R$. Given (x, y), the protocol P is supposed to output *some* $z \in Z$ with the property $(x, y, z) \in R$. Otherwise this z can be arbitrary. This kind of problems is called *search problems*.

The complexity of search problems is typically even more difficult to analyze than the complexity of decision problems. Let us consider just one important example, somewhat inspired by the equality function.

Assume that $X, Y \subseteq \{0,1\}^n$, but that these sets of strings are disjoint: $X \cap Y = \emptyset$. Then $\mathrm{EQ}(x, y) = 0$ for any $x \in X,\ y \in Y$ and there always exists a position i where they differ: $x_i \neq y_i$. Assume that the task of Alice and Bob is to actually *find* any such position.

This innocently-looking communication problem turns out to be equivalent to the second major open problem in computational complexity concerning computational depth [15, 22] (the first place being taken by the famous **P** vs. **NP** question). We do not have any clue as to how to prove lower bounds here. A simpler problem is obtained in a similar fashion from the disjointness function. That is, instead of $X \cap Y = \emptyset$ we assume that for any input $(x, y) \in X \times Y$ there is a position i such that $x_i = y_i = 1$. The task of Alice and

Bob is once again to exhibit any such i. Lower bounds for this problem were indeed proved in [15, 22, 14], and they lead to very interesting consequences about the *monotone* circuit depth of Boolean functions.

7 Conclusion

In this article we tried to give some impression of how soon simple, elementary and innocent questions turn into open problems that have been challenging us for decades. There are even more such challenges in the field of computational complexity, and we are in the need of young and creative minds to answer these challenges. If this article has encouraged at least some of the readers to look more closely into this fascinating subject, the author considers its purpose fulfilled in its entirety.

List of Notation

Since this text uses quite a bit of notation, some of the most important notations are collected here together with a brief description, as well as the page of first appearance.

Complexity Measures

$\mathrm{cost}(P)$ *cost of protocol P* — maximal number of bits to transmit in order to calculate the value of a function on any input (x, y) using protocol P **5**

$C(f)$ *(worst-case) communication complexity of function f* — minimal cost of any protocol computing f **5**

$\chi(f)$ *partition number of function f* — minimal number of pairwise disjoint f-monochromatic rectangles covering domain of f **9**

$t(f)$ *cover number of function f* — minimal number of f-monochromatic rectangles covering domain of f **11**

$\chi_0(f)$ minimal number of pairwise disjoint f-monochromatic rectangles covering $f^{-1}(\{0\})$ **11**

$\chi_1(f)$ minimal number of pairwise disjoint f-monochromatic rectangles covering $f^{-1}(\{1\})$ **11**

$t_0(f)$ minimal number of f-monochromatic rectangles covering $f^{-1}(\{0\})$ **11**

$t_1(f)$ minimal number of f-monochromatic rectangles covering $f^{-1}(\{1\})$ ($\log_2 t_1(f)$ *is called non-deterministic communication complexity of f*) **11**

$R(f)$ *bounded-error probabilistic communication complexity of function f* — minimal cost of randomized protocol that assures that for any input the output will be correct with probability at least $\frac{2}{3}$ **13**

$U(f)$ *unbounded-error probabilistic communication complexity of function f* — minimal cost of randomized protocol that assures that for any input the output will be correct with probability greater than $\frac{1}{2}$ **14**

$\mathrm{Disc}_u(f)$ *discrepancy (under uniform distribution) of function f* — maximal difference of how often values 0 and 1 occur on any rectangle (divided by $|X \times Y|$, where $X \times Y$ is the domain of f) **15**

$Q(f)$ *quantum communication complexity of function f* — minimal cost of quantum computer protocol evaluating f **16**

$C^k(f)$ *multi-party communication complexity of function f* — minimal number of bits that k players have to transmit in order to correctly compute the value of f (in number-on-the-forehead model) **17**

Binary Functions

EQ_N *equality function* — maps $\{1, 2, \ldots, N\} \times \{1, 2, \ldots, N\}$ to $\{0, 1\}$ with $\mathrm{EQ}_N(x, y) = 1$ iff $x = y$ **6**

LE_N *less-or-equal function* — maps $\{1, 2, \ldots, N\} \times \{1, 2, \ldots, N\}$ to $\{0, 1\}$ with $\mathrm{LE}_N(x, y) = 1$ iff $x \leq y$ **9**

DISJ_n *disjointness function ("NAND")* — maps $\{0, 1\}^n \times \{0, 1\}^n$ to $\{0, 1\}$ with $\mathrm{DISJ}_n(x, y) = 1$ iff for all $i \leq n$ we have $x_i = 0$ or $y_i = 0$ **10**

IP_n *inner product mod 2* — maps $\{0, 1\}^n \times \{0, 1\}^n$ to $\{0, 1\}$ with $\mathrm{IP}_n(x, y) = 1$ iff $x_i = y_i = 1$ for an odd number of indices i **15**

DISJ_n^k *generalized disjointness function* — maps $(\{0, 1\}^n)^k$ to $\{0, 1\}$ with $\mathrm{DISJ}_n^k(x^1, \ldots, x^k) = 1$ iff for all $i \leq n$ there exists $\nu \in \{1, \ldots, k\}$ with $x_i^\nu = 0$ **17**

IP_n^k *generalized inner product mod 2* — maps $(\{0, 1\}^n)^k$ to $\{0, 1\}$ with $\mathrm{IP}_n^k(x^1, \ldots, x^k) = 1$ iff the number of indices $i \leq n$ for which $x_i^1 = x_i^2 = \ldots = x_i^k = 1$ is odd **17**

Growth of Functions[6] and Other

$O(f(n))$ $g(n) \leq O(f(n))$ iff there is $C > 0$ with $g(n) \leq Cf(n)$ for all n **6**

$\Omega(f(n))$ $g(n) \geq \Omega(f(n))$ iff there is $\varepsilon > 0$ with $g(n) \geq \varepsilon f(n)$ for all n **10**

$\lceil x \rceil$ the smallest integer $n \geq x$, for $x \in \mathbb{R}$ **5**

[6] The more traditional notation is $g(n) = O(f(n))$ and $g(n) = \Omega(f(n))$; see also footnote 3.

References

[1] Alfred V. Aho, Jeffrey D. Ullman, and Mihalis Yannakakis, *On notions of information transfer in VLSI circuits.* In: *Proceedings of the* 15*th ACM symposium on the theory of computing*, ACM Press, New York, 1983, 133–139.

[2] Noga Alon and Paul Seymour, *A counterexample to the rank-coloring conjecture.* Journal of Graph Theory **13** (1989), 523–525.

[3] Sanjeev Arora and Boaz Barak, *Computational complexity: a modern approach.* Cambridge University Press, Cambridge, 2009.

[4] László Babai, Peter Frankl, and Janos Simon, *Complexity classes in communication complexity theory.* In: *Proceedings of the* 27*th IEEE symposium on foundations of computer science*, IEEE Computer Society, Los Alamitos, 1986, 337–347.

[5] László Babai, Noam Nisan, and Márió Szegedy, *Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs.* Journal of Computer and System Sciences **45** (1992), 204–232.

[6] Paul Beame and Dang-Trinh Huynh-Ngoc, *Multiparty communication complexity and threshold circuit size of AC^0.* Technical Report TR08-082, Electronic Colloquium on Computational Complexity, 2008.

[7] Harry Buhrman, Richard Cleve, and Avi Wigderson, *Quantum vs. classical communication and computation.* In: *Proceedings of the* 30*th ACM symposium on the theory of computing*, ACM Press, New York, 1998, 63–86; preliminary version available at `http://arxiv.org/abs/quant-ph/9802040`.

[8] Arkadev Chattopadhyay and Anil Ada, *Multiparty communication complexity of disjointness.* Technical Report TR08-002, Electronic Colloquium on Computational Complexity, 2008.

[9] Benny Chor and Oded Goldreich, *Unbiased bits from sources of weak randomness and probabilistic communication complexity.* SIAM Journal on Computing **17** 2 (1988), 230–261.

[10] Jürgen Forster, *A linear lower bound on the unbounded error probabilistic communication complexity.* Journal of Computer and System Sciences **65** 4 (2002), 612–625.

[11] Martin Fürer, *Faster integer multiplication.* SIAM Journal on Computing **39** 3 (2009), 979–1005.

[12] Bala Kalyanasundaram and Georg Schnitger, *The probabilistic communication complexity of set intersection.* SIAM Journal on Discrete Mathematics **5** 4 (1992), 545–557.

[13] Anatolii A. Karatsuba and Yuri P. Ofman, *Multiplication of many-digital numbers by automatic computers.* Proceedings of the USSR Academy of Sciences **145** (1962), 293–294.

[14] Mauricio Karchmer, Ran Raz, and Avi Wigderson, *Super-logarithmic depth lower bounds via direct sum in communication complexity.* Computational Complexity **5** (1995), 191–204.

[15] Mauricio Karchmer and Avi Wigderson, *Monotone circuits for connectivity require super-logarithmic depth.* SIAM Journal on Discrete Mathematics **3** 2 (1990), 255–265.

[16] Ilan Kremer, *Quantum communication.* Master's thesis, Hebrew University, Jerusalem, 1995.

[17] Eyal Kushilevitz and Noam Nisan, *Communication complexity.* Cambridge University Press, Cambridge, 1997.

[18] Troy Lee and Adi Shraibman, *Disjointness is hard in the multiparty number-on-the-forehead model.* Computational Complexity **18** 2 (2009), 309–336.

[19] Kurt Mehlhorn and Erik M. Schmidt, *Las Vegas is better than determinism in VLSI and distributive computing.* In: *Proceedings of the* 14*th ACM symposium on the theory of computing*, ACM Press, New York, 1982, 330–337.

[20] Ramamohan Paturi and Janos Simon, *Probabilistic communication complexity.* Journal of Computer and System Sciences **33** 1 (1986), 106–123.

[21] Ran Raz and Boris Spieker, *On the "log-rank"-conjecture in communication complexity.* Combinatorica **15** 4 (1995), 567–588.

[22] Alexander Razborov, *Applications of matrix methods to the theory of lower bounds in computational complexity.* Combinatorica **10** 1 (1990), 81–93.

[23] Alexander Razborov, *The gap between the chromatic number of a graph and the rank of its adjacency matrix is superlinear.* Discrete Mathematics **108** (1992), 393–396.

[24] Alexander Razborov, *On the distributional complexity of disjointness.* Theoretical Computer Science **106** (1992), 385–390.

[25] Alexander Razborov, *Quantum communication complexity of symmetric predicates.* Izvestiya: Mathematics **67** 1 (2003), 145–159.

[26] Arnold Schönhage and Volker Strassen, *Schnelle Multiplikation großer Zahlen.* Computing **7** (1971), 281–292.

[27] Andrew Yao, *Some complexity questions related to distributive computing.* In: *Proceedings of the* 11*th ACM symposium on the theory of computing*, ACM Press, New York, 1979, 209–213.

[28] Andrew Yao, *Quantum circuit complexity.* In: *Proceedings of the* 34*th IEEE symposium on foundations of computer science*, IEEE Computer Society, Los Alamitos, 1993, 352–361.