Linear Algebra and its Applications 436 (2012) 4459–4472

Contents lists available at SciVerse ScienceDirect

Linear Algebra and its Applications

journal homepage: www.elsevier.com/locate/laa

Linear algebraic methods in communication complexity ☆

Louis Deaett [a,*], Venkatesh Srinivasan [b]

[a] Department of Mathematics and Statistics, University of Victoria, P.O. Box 3060 STN CSC, Victoria, BC, Canada V8W 3R4
[b] Department of Computer Science, University of Victoria, P.O. Box 3055 STN CSC, Victoria, BC, Canada V8W 3P6

ARTICLE INFO

Article history:
Received 11 May 2011
Accepted 7 July 2011
Available online 6 August 2011

Submitted by P. van den Driessche

AMS classification:
15-02
68-02
68Q05
68Q17
68Q25

Keywords:
Communication complexity
Matrix rank
Minimum rank
Sign rank
Randomization

ABSTRACT

The notion of communication complexity seeks to capture the amount of communication between different parties that is required to find the output of a Boolean function when each party is provided with only part of the input. Different variants of the model governing the rules of this communication lead to different connections with problems in combinatorial linear algebra. In particular, problems arise in this context that concern the rank of a (0, 1)-matrix and the minimum rank of a matrix meeting a given combinatorial description. This paper surveys these connections.

1. Introduction

Combinatorial matrix theory is a branch of linear algebra devoted to problems that call for the properties of a matrix as a linear operator to be related to some combinatorial description of the matrix. As it turns out, this sort of problem arises naturally in the context of computational complexity theory [1], where it is of interest to study the "complexity" of a Boolean function f in terms of the resources required to compute it.

☆ Supported by NSERC Discovery Grant.
* Corresponding author.
E-mail addresses: deaett@math.uvic.ca, deaett@uvic.ca (L. Deaett), venkat@cs.uvic.ca (V. Srinivasan).

0024-3795/$ - see front matter © 2011 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.laa.2011.07.008

In this paper, we discuss some particular connections of this type which came to our attention during the 2010 BIRS workshop *Theory and Applications of Matrices Described by Patterns*. In particular, the notion of the communication complexity of a Boolean function (introduced in [26]) has in several instances been connected to problems of the sort usually referred to as "minimum rank problems" in the linear algebra literature.

The essence of a "minimum rank problem" is to determine, given some combinatorial description of a matrix, the smallest possible rank of a matrix meeting that description. (For a survey of such problems, see [7], updated with [8].) Some variation exists in terms of what sort of combinatorial description is considered, but often the essential quality is that the magnitudes of the entries of the matrix are disregarded, and only their signs or zero/nonzero character is retained. In the context of the present survey, the most important combinatorial description of a matrix is provided by the following definition.

Definition 1.1. Given a real $m \times n$ matrix A, the *sign pattern* of A is the unique $m \times n$ matrix $\mathcal{A}$ with entries from the set $\{+, -, 0\}$ that results by replacing each positive entry of A with the symbol $+$ and each negative entry with the symbol $-$. That is,

$$\mathcal{A}(i,j) = \begin{cases} + & \text{if } A(i,j) > 0, \\ 0 & \text{if } A(i,j) = 0, \\ - & \text{if } A(i,j) < 0. \end{cases}$$

What does the sign pattern of a real matrix imply about its rank? In particular, how small might the rank of a matrix be, given that it has a particular sign pattern? This question arises in connection with problems in computational complexity theory.

Computational complexity theory is an area of computer science that aims to study and classify various fundamental computational problems based on the resources needed to solve them. In a situation where the various inputs required to solve a computational problem are split among many parties or processors, the parties need to communicate with each other using messages and the total amount of communication becomes an important resource worth investigating.

In computational complexity theory, mathematical models are needed to capture the intuitive notion of efficient computation. While the Turing machine model serves this purpose in order to study the usage of resources such as time and space, no such mathematical model was available in order to study communication as a resource until Yao in [26] proposed the following two-party communication model.

1.1. The model

Let X and Y be finite sets. Alice and Bob are two players or processors interested in computing a Boolean function f of two inputs, $f : X \times Y \to \{0, 1\}$. Alice is given an input $x \in X$ and Bob is given an input $y \in Y$ and they need to compute $f(x, y)$. We assume that both of them have unlimited computational power. Since Alice does not know y and Bob does not know x, they have to communicate by sending messages to one another, so they must agree in advance upon a *communication protocol* they will use to exchange information about their inputs in order to compute f. Of course, they wish to minimize the total communication needed.

A communication protocol to compute f describes a complete set of rules that Alice and Bob agree to follow while they send messages to each other. In particular, it must do two things:

(1) For every sequence of bits exchanged so far, the protocol specifies whether or not the communication is complete. If it is, both Alice and Bob must know the output bit, i.e. the value of the function f on their pair of inputs.
(2) If the communication is not complete, then the protocol specifies which player is responsible for transmitting the next bit, and how that player is to determine this bit based upon their input and the total communication (sequence of bits transmitted) so far.

It is natural to view a communication protocol as a binary tree, called the *protocol tree*. Each internal node of this tree indicates which player is responsible for transmitting the next bit of information, and how that bit is determined based on the "knowledge" of that player. Each leaf node is labeled by a Boolean value indicating the protocol's output. To follow the protocol, Alice and Bob start at the root node, and then at each node traverse the tree to the left if the responsible player transmits a 0 and to the right if the player transmits a 1. We will return to this representation of a protocol often.

We wish to study how much communication Alice and Bob use to compute the function. However, the total number of bits communicated in following a protocol to completion may not be the same for all input pairs. Hence, we focus on the *worst case* performance of the protocol; that is, we consider the maximum, over all input pairs $(x, y) \in X \times Y$, of the total number of bits that Alice and Bob transfer in computing $f(x, y)$. This maximum is the *cost* of the protocol. This cost has a very simple interpretation in terms of the protocol tree; it is simply the depth of this tree.

1.2. Two examples

In this survey, we are interested in techniques to prove upper and lower bounds on the amount of communication required to compute explicit functions. We will use the following two examples.

Definition 1.2. The EQUALITY function, $EQ_n : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}$, is defined by

$$EQ_n(x, y) = \begin{cases} 1 & \text{if } x = y, \\ 0 & \text{otherwise.} \end{cases}$$

Definition 1.3. The INNER PRODUCT function, $IP_n : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}$, is defined by

$$IP_n(x, y) = \sum_{i=1}^{n} x_i y_i \pmod 2.$$

Definitions 1.2 and 1.3 each define a *family* of Boolean functions, i.e. one function with domain $\{0, 1\}^n$ for each natural number n. (For convenience, we will refer to the two families as EQ and IP, respectively.) The notion of such a family of functions is a fundamental one in computational complexity theory, where the central question of interest is usually how the complexity of the functions in the family behaves as n grows.

Having described communication protocols, we now ask ourselves the following question: What does it mean for a communication protocol to compute a function f? As we shall see in the rest of this survey, there are many possible answers to this question, resulting in many interesting communication models.

2. Rank and deterministic communication complexity

The most straightforward way in which a protocol may compute a function is captured by the deterministic communication model.

A deterministic communication protocol over the domain $X \times Y$ is a binary tree in which every internal node is labeled by a Boolean function on X or a Boolean function on Y. Every leaf is labeled by 0 or 1. On any input (x, y), the *output* of the protocol is the label of the leaf reached by the following process: Start at the root and walk down the tree. For each internal node labeled by a Boolean function f on X, walk left if $f(x) = 0$ and right if $f(x) = 1$. Similarly, for each internal node labeled by a Boolean function g on Y, walk left if $g(y) = 0$ and right if $g(y) = 1$.

At any internal node labeled by a Boolean function on X, walking to the left child corresponds to the communication of a 0 bit from Alice to Bob while walking to the right child corresponds to communication of a 1 bit from Alice to Bob. Similarly, at any internal node labeled by a Boolean function on Y, walking to the left child corresponds to communication of a 0 bit from Bob to Alice while walking to the right child corresponds to the communication of a 1 bit from Bob to Alice.

Definition 2.1. A protocol P is said to compute the function f *deterministically* if, for every input $(x, y) \in X \times Y$, the output of P on input (x, y) is equal to $f(x, y)$.

Recall that the cost of a protocol, viewed as a binary tree, is the depth of the tree, as this represents the maximum amount of communication between Alice and Bob on any input pair.

Definition 2.2. The *deterministic communication complexity* of f, which we denote $D(f)$, is the minimum cost of a protocol that computes f deterministically.

For any function f, there is a naive protocol, called the *trivial protocol*, that computes f deterministically. In this protocol, Alice sends her entire input x (comprising n bits) to Bob. Bob then knows both x and y and computes $f(x, y)$ and sends this single bit back to Alice. The existence of this protocol gives the following result.

Theorem 2.3. *For any function* $f : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}$,

$$D(f) \leq n + 1.$$

For either of our examples *EQ* and *IP*, is there a deterministic protocol that does better? This is a question that can be answered with linear algebra.

To any function $f : X \times Y \to \{0, 1\}$, we associate a $(0, 1)$-matrix M_f. This M_f is a $|X| \times |Y|$ matrix with the rows indexed by X and the columns indexed by Y, and the (x, y) entry defined to be $f(x, y)$. Let $\operatorname{rank}(M_f)$ denote the real rank of M_f. Melhhorn and Schmidt [18] proved the following result.[1]

Theorem 2.4. *For any function* f, $D(f) \geqslant \log(\operatorname{rank}(M_f))$.

Proof. Recall that a deterministic protocol to compute f can be viewed as a binary tree. On every input, the protocol starts at the root and walks down the tree moving either left or right at each step until it reaches a leaf. As each input from $X \times Y$ leads to a unique leaf, the protocol induces a partition of $X \times Y$ into many sets, one for each leaf, in the following way. For a fixed leaf l, let S_l denote the set of input pairs (x, y) that reach l. It is easy to check, by induction on the depth of a node, that S_l is of the form $A \times B$ for some $A \subseteq X$ and $B \subseteq Y$.

Now let us denote by L_1 the set of all leaves labeled by 1. For each $l \in L_1$, define a matrix M_l by

$$M_l(x, y) = \begin{cases} 1 & \text{if } (x, y) \in S_l, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

It follows from the form of S_l that $\operatorname{rank}(M_l) = 1$ for each $l \in L_1$. Moreover, since each input reaches a unique leaf, $M_f = \sum_{l \in L_1} M_l$. Therefore,

$$\operatorname{rank}(M_f) \leqslant \sum_{l \in L_1} \operatorname{rank}(M_l) = |L_1|.$$

Now $D(f)$ is at least the depth of the binary tree for this protocol, and this in turn is at least the logarithm of the number of leaves in the tree. Thus, we conclude that $D(f) \geqslant \log(\operatorname{rank}(M_f))$. □

Noting that M_{EQ} is the $2^n \times 2^n$ identity matrix, we see that it has full rank, and thus, by Theorem 2.4, $D(EQ) \geq n$. Similarly, the matrix M_{IP} can be obtained by adding the matrix of all 1s to a $2^n \times 2^n$ Hadamard matrix, and then scaling the result. (See [12, Chapter 14] for the basics facts about Hadamard matrices used here.) It follows that M_{IP} has rank $2^n - 1$ and hence $D(IP) \geq n$ for $n \geq 2$. In light of Theorem 2.3, this means that it is impossible for any deterministic protocol for *EQ* or *IP* to do significantly better than the trivial protocol.

[1] All logarithms in this paper are taken with a base of 2.

Theorem 2.4 gives an important relationship between the deterministic communication complexity of f and the real rank of M_f. The following conjecture from [17], known as the *log-rank conjecture*, proposes a strengthening of this relationship.

Conjecture 2.5. *There is some positive constant c such that, for any function f, $D(f) \leqslant (\log(\operatorname{rank}(M_f)))^c$.*

The resolution of this conjecture remains the biggest open problem in communication complexity. One remarkable result of [19] is that the truth of the conjecture is unchanged if $D(f)$ is replaced by $-\log(\operatorname{mono}(M))$, where $\operatorname{mono}(M)$ is the largest fraction of the entries of M to be found within a single *monochromatic* submatrix, i.e. a submatrix in which either every entry is 1 or every entry is 0. Also in [19], a construction is given of an explicit function f with communication complexity at least $(\log(\operatorname{rank}(M_f)))^\alpha$ for $\alpha = \log_3 6 \approx 1.63$, giving the best known lower bound on the c of Conjecture 2.5. For details about this conjecture and related research subjects in linear algebra, the survey [4] is highly recommended.

3. Nonnegative sign rank and nondeterministic communication complexity

When studying the power of a certain model of computation, it is often illuminating to investigate its nondeterministic analog. (Most famously, in the case of Turing machine time complexity, this gives rise to the P vs. NP problem.) In the communication protocol model, nondeterminism is introduced by allowing certain points in the protocol at which a bit is to be sent nondeterministically. That is, at such a point, whether the next bit to be sent is 0 or 1 is not determined by the input, nor by anything else – hence the term "nondeterminism". (One way to think of this is that the protocol may occasionally require Alice or Bob to send a bit based simply on a "guess".)

Formally, a nondeterministic protocol P over the domain $X \times Y$ is a binary tree in which each internal node is labeled by a function $f : X \to \{0, 1, *\}$ or a function $g : Y \to \{0, 1, *\}$ while each leaf is labeled by 0 or 1. On input (x, y), the *output* of such a protocol is the label of the leaf reached by the following process: Start at the root and walk down the tree. For each internal node labeled by a function f on X, walk left if $f(x) = 0$, right if $f(x) = 1$ and choose nondeterministically to move either left or right if $f(x) = *$. Similarly, for each internal node labeled by a function g on Y, walk left if $g(y) = 0$, right if $g(y) = 1$ and move either left or right if $g(y) = *$.

In contrast with the deterministic case, the output of a nondeterministic protocol is not determined solely by the input, but also by the (possibly empty) sequence of nondeterministic 'guesses' the players make for the appropriate bits in the communication. Thus, it is necessary to explain what it means for a protocol to compute a function f with respect to this nondeterministic model. That is the purpose of the following definition.

Definition 3.1. A protocol is said to compute the function f *nondeterministically* if both of the following hold.

(1) If $f(x, y) = 1$, then the protocol on input (x, y) gives an output of 1 for at least one sequence of choices for the nondeterministic bits.
(2) If $f(x, y) = 0$, then on input (x, y) every possible sequence of choices for the nondeterministic bits results in an output of 0.

Note the asymmetry in this definition: The protocol is allowed to give an incorrect output some of the time if $f(x, y) = 1$, but must give the correct output every time if $f(x, y) = 0$.

We have discussed above how a nondeterministic protocol may be represented as a binary tree. As always, the cost of the protocol is the depth of this tree.

Definition 3.2. The smallest cost of a protocol computing f nondeterministically is the *nondeterministic communication complexity* of f, which we denote $N(f)$.

In the tree for a deterministic protocol, each possible input leads to a unique leaf. In the tree for a nondeterministic protocol, a given input may lead to any one of a set of leaves. Hence, in the tree for a nondeterministic protocol computing f, the leaves no longer induce a decomposition of M_f into $(0, 1)$-matrices of rank one as they did (c.f. the proof of Theorem 2.4) in the deterministic case; instead, they induce a covering of the 1s in M_f by such matrices. We make this notion precise with the following definition.

Definition 3.3. Let M and $C_1, \ldots, C_k$ be $(0, 1)$-matrices of dimension $m \times n$. The matrices C_i *cover the* 1*s in* M if it is the case that $M(x, y) = 1$ if and only if $C_i(x, y) = 1$ for some i.

Definition 3.4. Let M be a $(0, 1)$-matrix. The 1-*cover number* of M, denoted $C^1(M)$, is the smallest k such that there exist $(0, 1)$-matrices $L_1, \ldots, L_k$ of rank one that cover the 1s in M.

The 1-cover number provides a combinatorial description of M_f that completely captures the nondeterministic communication complexity of f.

Theorem 3.5. $\lceil \log C^1(M_f) \rceil \leq N(f) \leq \lceil \log C^1(M_f) \rceil + 2$.

Proof. Consider the protocol tree for a nondeterministic protocol computing f with cost d. For each leaf l, the input pairs (x, y) that reach l (for some sequence of choices for the nondeterministic bits) form a set $S_l = A \times B \subseteq X \times Y$. For each leaf l labeled with a 1, let M_l be defined as in (1).

Now if $f(x, y) = 1$, then there is at least one leaf l reachable on input (x, y) that is labeled with a 1 and hence $M_l(x, y) = 1$. Thus, the matrices M_l together cover the 1s in M_f. Hence, $C^1(M_f) \leq 2^d$, implying that $\lceil \log C^1(M_f) \rceil \leq d \leq N(f)$.

On the other hand, given a collection of $(0, 1)$-matrices $L_1, \ldots, L_k$ of rank one that cover the 1s in M, a suitable protocol to compute f proceeds as follows. On input (x, y), Alice sends $\lceil \log k \rceil$ bits nondeterministically that are then interpreted by both parties as a "guess" of one of the L_i. As L_i has rank one, its support is some $A \times B \subseteq X \times Y$. Alice now sends a 1 to Bob if and only if $x \in A$. Then Bob is able to check if $(x, y) \in A \times B$, sending a 1 if this is so and a 0 otherwise; this final bit is the output of the protocol. It is easy to see that this protocol computes f nondeterministically, and that therefore $N(f) \leq \lceil \log k \rceil + 2$. □

We now show that the combinatorial description of M provided by $C^1(M)$ is in fact equivalent to an algebraic description with a "minimum rank" flavor. To this end, we introduce the following definitions.

Definition 3.6. Given a nonnegative $m \times n$ matrix M, the *nonnegative rank* of M is the smallest p such that there exist nonnegative matrices A of dimension $m \times p$ and B of dimension $p \times n$ such that $M = AB$.

Of course, if we remove from Definition 3.6 the requirement that A and B be nonnegative, what results is the ordinary rank of M. As M is itself nonnegative, the fact that its nonnegative rank may differ from its rank is perhaps surprising; see [5] for a detailed discussion of nonnegative rank.

A connection between the nondeterministic communication complexity of f and the nonnegative rank of M_f has been noted in [25]. In order to strengthen this connection, we define the corresponding minimum rank notion. We believe that this notion has not been explicitly introduced previously, but it arises quite naturally in the present context.

Definition 3.7. Given a nonnegative $m \times n$ matrix M, the *nonnegative sign rank* of M is the smallest p such that there exist nonnegative matrices A of dimension $m \times p$ and B of dimension $p \times n$ such that the sign pattern of AB is the same as that of M. We denote this value by $\text{sign-rank}_+(M)$.

Note that the nonnegative rank of M depends only on its sign pattern. The following well-known argument shows that the nonnegative sign rank of a $(0, 1)$-matrix and its 1-cover number are identical.

Theorem 3.8. *For any* $(0, 1)$*-matrix* M, $\text{sign-rank}_+(M) = C^1(M)$.

Proof. Suppose M is a $(0, 1)$-matrix. Let $k = C^1(M)$ and $p = \text{sign-rank}_+(M)$. By definition, there exist $(0, 1)$-matrices $L_1, \ldots, L_k$ of rank one that cover the 1s in M. It follows that $\sum_{i=1}^{k} L_i$ has the same sign pattern as M. Each L_i can be written as $u_i v_i^T$, where u_i is the unique nonzero column of L_i and v_i^T is the unique nonzero row of L_i. Thus, taking A to be the $n \times k$ matrix whose columns are the u_i and B to be the $k \times n$ matrix whose rows are the v_i yields $M = AB$. This shows that $k \geq p$.

For the other direction, take nonnegative matrices A of dimension $n \times p$ and B of dimension $p \times n$ such that AB has the same sign pattern as M. Then, writing a_i for the ith column of A and b_i^T for the ith row of B, we have

$$AB = \sum_{i=1}^{p} a_i b_i^T.$$

Each term $a_i b_i^T$ in the above is an $n \times n$ matrix of rank one, and so each of its nonzero columns has the same support. As a result, if we let L_i be the unique $(0, 1)$-matrix obtained by replacing each nonzero entry in $a_i b_i^T$ with a 1, then each L_i has rank one, and it is clear that $L_1, \ldots, L_p$ cover the 1s in M. Hence, $p \geq k$. □

Theorems 3.5 and 3.8 together give the following connection between nondeterministic communication complexity and the minimum rank notion introduced in Definition 3.7.

Corollary 3.9. $\lceil \log(\text{sign-rank}_+(M_f)) \rceil \leq N(f) \leq \lceil \log(\text{sign-rank}_+(M_f)) \rceil + 2$.

Generally, the important questions about communication complexity concern how the complexity of a family of functions may grow asymptotically as n becomes large. Hence, the gap between the lower and upper bounds in Corollary 3.9 (and the analogous gap in Theorem 5.4) are of little significance.

Given that M_{EQ} is the $2^n \times 2^n$ identity matrix, clearly $\text{sign-rank}_+(M_{EQ}) = C^1(M_{EQ}) = 2^n$. By Corollary 3.9, then, $n \leq N(EQ) \leq n + 2$. By Theorem 2.3, together with the obvious fact that $N(f) \leq D(f)$ for every f, this means that $N(EQ)$ is essentially as large as possible.

On the other hand, if we define the function $NEQ_n : \{0, 1\}^n \times \{0, 1\}^n \to \{0, 1\}$ by $NEQ_n(x, y) = 1$ if and only if $x \neq y$, then a simple nondeterministic protocol for NEQ has Alice begin by sending $\lceil \log n \rceil$ bits nondeterministically to "guess" a certain integer $k \in \{1, \ldots, n\}$. Then Alice sends the value of the kth bit of her input. Bob then sends the final bit, the output of the protocol, as 1 if his input differs in the kth bit, and 0 otherwise. This shows that $N(NEQ) \leq \lceil \log n \rceil + 2$. Thus, NEQ requires exponentially less communication than EQ in the nondeterministic model; this reflects the fundamental asymmetry of nondeterminism, manifest in Definition 3.1.

To settle $N(IP)$, we examine the size of a largest monochromatic submatrix in M_{IP}. Suppose some submatrix $M_{IP}[R, S]$ has every entry equal to 0. Note that the subspace of $\mathbb{F}_2^n$ spanned by the vectors in R has dimension at least $\log |R|$, while that spanned by the vectors in S has dimension at least $\log |S|$. Since $M_{IP}[R, S]$ has only zero entries, the two subspaces are orthogonal, so $\log |R| + \log |S| \leq n$, giving $|R||S| \leq 2^n$. Hence, any zero submatrix of M_{IP} may have no more than 2^n entries.

On the other hand, if every entry of $M_{IP}[R, S]$ is 1, choose any $v_r \in R$ and $v_s \in S$. Neither of these vectors can be zero, so

$$\tilde{R} = \{v_r + w : w \in R\} \text{ and } \tilde{S} = \{v_s + w : w \in S\}$$

are disjoint from R and S, respectively, with $|R| = |\tilde{R}|$ and $|S| = |\tilde{S}|$. As $M_{IP}[\tilde{R}, \tilde{S}]$ has every entry equal to zero, $|\tilde{R}||\tilde{S}| = |R||S| \leq 2^n$.

Having shown that a submatrix of M_{IP} containing only 1s has at most 2^n entries, it follows that $C^1(M_{IP}) \geq 2^{2n}/2^n = 2^n$. By Theorem 3.5, then, $n \leq N(IP) \leq n + 2$. Again, this is essentially as large as possible.

4. Bounded-error randomized communication complexity

In a *randomized protocol* for computing a Boolean function f, the player whose turn it is to speak computes the next bit of communication with a random coin flip whose distribution is a function of the player's input and the sequence of communication up to that point. One can imagine that each non-leaf node in the protocol tree comes with a bag of weighted coins, one for each of the inputs that may have led the player to that point in the protocol. At each step, Alice and Bob simply select from that bag the coin that corresponds to their input, flip it, and transmit a 0 or 1 accordingly.

Of course, for a fixed input (x, y), such a protocol can result in different communication transcripts – and hence possibly in different outputs – depending on the outcomes of the coin flips. This is similar to the way in which a nondeterministic protocol may result in different outputs depending on the results of the nondeterministic "guesses" involved.

Formally, a randomized protocol P over the domain $X \times Y$ is a binary tree in which each internal node is labeled by a function $f : X \to [0, 1]$ or a function $g : Y \to [0, 1]$ while each leaf is labeled by 0 or 1. On any input (x, y), the *output* of the protocol is the label of a leaf reached by the following process: Start at the root and walk down the tree. For each internal node labeled by a function f on X, walk left with probability $p = f(x)$ and right with probability $1 - p$. Similarly, for each internal node labeled by a function g on Y, walk left with probability $p = g(y)$ and right with probability $1 - p$. Thus, for a randomized protocol P on input (x, y), the output of the protocol $P(x, y)$ is a Boolean random variable.

When should we say that a randomized protocol P computes a function f? A natural choice is to say that it should do better than a random guess. That is, the output of P should coincide with the value of f with probability $1/2 + \epsilon$ for some $\epsilon > 0$. This sounds simple enough, but an important choice needs to be made regarding how ϵ is allowed to change as the input size grows. We can merely require that ϵ be positive for each n (possibly with $\epsilon \to 0$ as $n \to \infty$) or we can fix some bound $\delta > 0$ and require that $\epsilon \geq \delta$ for all n. These two choices lead to two different probabilistic communication models that we study in this and the next section.

We say that P computes f with *bounded error* if, for some fixed $\delta > 0$, the output of P on any input (x, y) is equal to $f(x, y)$ with probability $\frac{1}{2} + \epsilon$ for $\epsilon > \delta$. (That is, the error probability is bounded above by $\frac{1}{2} - \delta$.) Note that, given a protocol whose error is bounded in this way for some value of δ, the error probability can be brought below any fixed bound by repeating the protocol some fixed number of times and choosing the majority answer. Moreover, this will affect the cost by only a constant factor. Given that the value of δ is arbitrary according to this consideration, we will take $\delta = 1/6$ for convenience. This results in the following formal definition.

Definition 4.1. A randomized protocol P computes $f : X \times Y \to \{0, 1\}$ with *bounded error* if, for all $(x, y) \in X \times Y$,

$$\Pr[P(x, y) = f(x, y)] \geqslant \frac{2}{3}.$$

As with deterministic protocols, we will be interested in the most efficient randomized protocol for computing a function f. This motivates the following definition.

Definition 4.2. The *bounded-error randomized communication complexity* of a function f, which we denote by $R(f)$, is the minimum cost of a randomized protocol that computes f with bounded error.

The following theorem shows that randomized protocols for the equality function require significantly less communication than deterministic protocols.[2] This result is due originally to Rabin and Yao (see [27]). The proof presented here is that in [15].

[2] We recommend [11, Chapter 9] to any reader unfamiliar with the O and Ω notations used here.

Theorem 4.3. $R(EQ) = O(\log n)$.

Proof. Our goal is to exhibit a randomized protocol for EQ with cost $O(\log n)$ and error probability at most 1/3.

For convenience, we will refer to the input string for Alice as $a = a_0 \dots a_{n-1}$ and the input string for Bob as $b = b_0 \dots b_{n-1}$. In the protocol, both Alice and Bob view their input string as a polynomial of degree $n-1$. That is, Alice has the polynomial $a(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1}x^{n-1}$ and Bob has the polynomial $b(x) = b_0 + b_1 x + b_2 x^2 + \cdots + b_{n-1}x^{n-1}$.

The protocol proceeds as follows:

(1) Alice picks a prime p, $3n \leqslant p \leqslant 6n$. Then Alice picks a random element $r \in \{0, 1, \dots, p-1\}$. Alice sends $(p, r, a(r) \bmod p)$ to Bob.
(2) Bob computes $b(r) \bmod p$ and checks if $a(r) \bmod p = b(r) \bmod p$. He sends a 1 to Alice if the equality holds, and sends a 0 otherwise.

Alice sends three numbers between $3n$ and $6n$, and Bob replies with a single bit, so the total number of bits of communication is $O(\log n)$. We now check that the error probability is at most 1/3. If $a = b$, then $a(r) \bmod p = b(r) \bmod p$ and the protocol always outputs 1 irrespective of the choice of the random element r. If $a \neq b$, then the probability (over all possible choices for r) that the protocol outputs 1 is at most the probability that r is a root of the nonzero polynomial $a(x) - b(x)$ of degree at most $n-1$. Therefore, the probability is at most $(n-1)/3n < 1/3$. □

In contrast, randomized protocols for the inner product function require asymptotically the same amount of communication as deterministic protocols. A general technique for proving lower bounds on the complexity of randomized protocols by relating it to the distributional complexity of deterministic protocols was introduced by Yao [27]. Using this technique, Vazirani [24] showed the following result. The proof technique described here is from [3].

Theorem 4.4. $R(IP) = \Omega(n)$.

The proof of this result is somewhat technical, so we simply discuss the ideas behind it. Recall that a deterministic protocol P of cost k for computing a function f induces a partition of M_f into at most 2^k monochromatic submatrices. Each such submatrix corresponds to a leaf in the protocol tree of P. So a strategy that works quite well to prove lower bounds for deterministic communication complexity is to show that any monochromatic submatrix in M_f is small in size and hence any partition of M_f into monochromatic submatrices must contain many such matrices. More concretely, if we can show that any partition of M_f into monochromatic matrices is of size at least m, we can conclude that the deterministic communication complexity of f is at least $\log m$.

Since randomized protocols are allowed to make a small error, lower bounds for randomized protocols can be obtained by considering partitions of M_f into "nearly monochromatic" submatrices, i.e. submatrices with mostly 0s and a few 1s or vice versa. The number of such submatrices required to partition the matrix is related to the notion of *discrepancy*, defined as follows.

Definition 4.5. Given $f : X \times Y \to \{0, 1\}$ and $R = A \times B \subseteq X \times Y$, let $n_0(f, R)$ (resp. $n_1(f, R)$) denote the number of entries (a, b) in R such that $f(a, b) = 0$ (resp. $f(a, b) = 1$). Then

$$\mathrm{Disc}(f) = \max_R \frac{|n_1(f, R) - n_0(f, R)|}{|X|\,|Y|}. \tag{2}$$

Intuitively, a small value for $\mathrm{Disc}(f)$ implies that large submatrices of M_f have almost equal numbers of 0s and 1s. Therefore, any nearly monochromatic submatrix must be small in size and hence any partition of M_f will require many such submatrices. Formally, it has been proved [27, Lemma 3] that $R(f) = \Omega(\log(1/\mathrm{Disc}(f)))$. In other words, small discrepancy implies good lower bounds for $R(f)$.

Can we prove that Disc(IP) is small? Indeed, it is shown in [3] (by adapting an argument of J. H. Lindsey, presented in [6], bounding the sum of the entries of a Hadamard matrix) that $\mathrm{Disc}(IP) \leqslant 2^{-n/2}$. This is sufficient to prove Theorem 4.4, that $R(IP) = \Omega(n)$.

Finally, we note that the notion of the *cut norm* of a matrix, introduced in [10] in the context of approximation algorithms for graph-theoretic optimization problems, is equivalent to discrepancy in the sense that the discrepancy of f is the cut norm of M_f. (Some authors define the cut norm exactly as in (2), while others omit the denominator.)

5. Sign-rank and unbounded-error randomized communication complexity

In this section, we study randomized protocols with *unbounded error*. A randomized protocol is said to compute a function f with unbounded error if it computes f correctly with probability better than a random guess. In other words, the success probability of such a protocol on input (x, y) is $1/2 + \epsilon$ for some $\epsilon > 0$.

Definition 5.1. A randomized protocol P computes $f : X \times Y \to \{0, 1\}$ with *unbounded error* if, for all $(x, y) \in X \times Y$,

$$\Pr[P(x, y) = f(x, y)] > \frac{1}{2}.$$

Definition 5.2. The *unbounded-error randomized communication complexity* of a function f, which we denote $U(f)$, is the minimum cost of a randomized protocol that computes f with unbounded error.

Unbounded-error randomized communication complexity turns out to have a very close relationship with the following "minimum rank" notion, which should be compared to that introduced in Definition 3.7.

Definition 5.3. The *sign rank* of a real matrix A is the smallest rank of a real matrix having the same sign pattern as A. We denote the sign rank of A by $\text{sign-rank}(A)$.

Hence, the sign rank is simply the smallest rank of a real matrix with the given sign pattern. Little seems to be understood about the combinatorial behavior of this parameter; the current state of knowledge is laid out in [13]. The most powerful general result about sign rank is due to Forster [9] and has an analytic flavor; we present it as Lemma 5.7 below.

With any function $f : X \times Y \to \{0, 1\}$, we associate an $|X| \times |Y|$ matrix N_f in which the rows are indexed by X and the columns are indexed by Y. The matrix is defined by

$$N_f(x, y) = \begin{cases} 1 & \text{if } f(x, y) = 1, \text{ and} \\ -1 & \text{if } f(x, y) = 0. \end{cases}$$

The following theorem of Paturi and Simon [20] says that the unbounded-error randomized communication complexity of f is essentially captured by $\text{sign-rank}(N_f)$.

Theorem 5.4. $\lceil \log(\text{sign-rank}(N_f)) \rceil \leqslant U(f) \leqslant \lceil \log(\text{sign-rank}(N_f)) \rceil + 1$.

Recall that the rank of a real $m \times n$ matrix A is equal to the smallest dimension d such that there exist vectors $x_1, \ldots, x_m$ and $y_1, \ldots, y_n$ in $\mathbb{R}^d$ with $A(i, j) = \langle x_i, y_j \rangle$ for every i and j. Thus, $\text{sign-rank}(A)$ is the smallest d such that there exist vectors $x_1, \ldots, x_m$ and $y_1, \ldots, y_n$ in $\mathbb{R}^d$ with

$$\text{sign}(A(i, j)) = \text{sign}(\langle x_i, y_j \rangle).$$

The following application of Theorem 5.4 shows that, for the equality function, when we do not require the success probability of a randomized protocol to be bounded away from 1/2 by a constant,

we get amazing savings in communication. This result was originally shown in [20]; the argument here is from [23].

Theorem 5.5. $U(EQ) = O(1)$.

Proof. First note that

$$N_{EQ}(i,j) = \begin{cases} 1 & \text{if } i = j, \\ -1 & \text{otherwise.} \end{cases}$$

By Theorem 5.4, it suffices to show that sign-rank(N_{EQ}) is a constant independent of n. To see this, pick 2^n distinct unit vectors $u_1, u_2, \ldots, u_{2^n}$ in the first quadrant of $\mathbb{R}^2$. Let $\epsilon > 0$ be smaller than $1 - \max_{i \neq j}(\langle u_i, u_j \rangle)$. Then it is easy to check that the matrix A defined by

$$A(i,j) = \langle u_i, u_j \rangle - (1 - \epsilon) \tag{3}$$

has the same sign pattern as does N_{EQ}. It follows from the choice of the vectors $u_1, u_2, \ldots, u_{2^n}$ that their Gram matrix has rank 2, so from (3) it is clear that the rank of A is at most 3. □

In contrast, moving from the bounded-error to the unbounded-error model has no impact asymptotically on the amount of communication required to compute the inner product function.

Theorem 5.6. $U(IP) = \Omega(n)$.

To establish this theorem, Forster [9] proved the following general result relating the sign rank of a $(-1, 1)$-matrix to its spectral norm. It says that if the spectral norm of the matrix is small, then its sign rank must be high.

Lemma 5.7. *For any* $(-1, 1)$*-matrix* A *of dimension* $m \times n$, sign-rank$(A) \geqslant \sqrt{mn}/\|A\|$.

Proof. Let $d =$ sign-rank(A) and let $x_1, \ldots, x_m$ and $y_1, \ldots, y_n$ be unit vectors in $\mathbb{R}^d$ such that for each i and j, $\text{sign}(\langle x_i, y_j \rangle) = \text{sign}(A(i,j))$. To prove the lemma, we will focus on the parameter

$$S = \sum_{j=1}^{n} \left(\sum_{i=1}^{m} |\langle x_i, y_j \rangle| \right)^2.$$

The method is to establish upper and lower bounds on S.

Claim 1. $S \leqslant m\|A\|^2$.
Proof of claim 1. For any j with $1 \leqslant j \leqslant n$,

$$\sum_{i=1}^{m} |\langle x_i, y_j \rangle| = \sum_{i=1}^{m} a_{ij} \langle x_i, y_j \rangle = \left\langle \sum_{i=1}^{m} a_{ij} x_i, y_j \right\rangle \leqslant \left\| \sum_{i=1}^{m} a_{ij} x_i \right\|,$$

where the inequality is by Cauchy–Schwartz. Therefore,

$$\begin{aligned} S = \sum_{j=1}^{n} \left(\sum_{i=1}^{n} |\langle x_i, y_j \rangle| \right)^2 &\leq \sum_{j=1}^{n} \left\| \sum_{i=1}^{m} a_{ij} x_i \right\|^2 \\ &= \sum_{j=1}^{n} \left\langle \sum_{i=1}^{m} a_{ij} x_i, \sum_{l=1}^{m} a_{lj} x_l \right\rangle \\ &= \sum_{k,l} (AA^T)_{kl} \langle x_k, x_l \rangle. \end{aligned} \tag{4}$$

It is easily verified that the matrix $\|A\|^2 I_m - AA^T$ is positive semidefinite. As the cone of positive semidefinite matrices is self-dual (see e.g. [2, Example 2.24]) the entry-wise inner product of any two positive semidefinite matrices must be nonnegative. In particular,

$$\sum_{k,l} (\|A\|^2 I_m - AA^T)_{kl} \langle x_k, x_l \rangle \geq 0,$$

so that

$$\sum_{k,l} (AA^T)_{kl} \langle x_k, x_l \rangle \leq \sum_{k,l} (\|A\|^2 I_m)_{k,l} \langle x_k, x_l \rangle.$$

Combining this with (4) gives

$$S \leq \sum_{k,l} (\|A\|^2 I_m)_{k,l} \langle x_k, x_l \rangle = \|A\|^2 \sum_k \langle x_k, x_k \rangle = \|A\|^2 m,$$

as claimed.

The bulk of the effort in Forster's proof (which we are following here) comes in establishing that the vectors $x_1, \ldots, x_m$ can be taken to be "balanced" in the sense that

$$\sum_{i=1}^{m} x_i x_i^T = \frac{m}{d} I_d. \tag{5}$$

This fact is used to establish the following bound.

Claim 2. $S \geqslant n(m/\text{sign-rank}(A))^2$.
Proof of claim 2. Since

$$\sum_{i=1}^{m} |\langle x_i, y_j \rangle| \geqslant \sum_{i=1}^{m} \langle x_i, y_j \rangle^2 = \sum_{i=1}^{m} y_j^T x_i x_i^T y_j = y_j^T \left(\sum_{i=1}^{m} x_i x_i^T \right) y_j,$$

we may apply (5) above to obtain

$$\sum_{i=1}^{m} |\langle x_i, y_j \rangle| \geqslant y_j^T \left(\sum_{i=1}^{m} x_i x_i^T \right) y_j = y_j^T \left(\frac{m}{d} I_d \right) y_j = \frac{m}{d} (y_j^T y_j) = \frac{m}{d}.$$

Therefore,

$$S = \sum_{j=1}^{n} \left(\sum_{i=1}^{m} |\langle x_i, y_j \rangle| \right)^2 \geqslant \sum_{j=1}^{n} \left(\frac{m}{d} \right)^2 \geqslant n \left(\frac{m}{d} \right)^2,$$

as claimed.

Combining the two claims yields $\|A\|^2 m \geqslant n(m/d)^2$, and hence $d \geqslant \sqrt{mn}/\|A\|$. □

Proof of Theorem 5.6. By observing that N_{IP} is a Hadamard matrix, it follows that $\|N_{IP}\| = 2^{n/2}$. By Lemma 5.7, this gives

$$\text{sign-rank}(N_{IP}) \geqslant \sqrt{2^{2n}}/2^{n/2} = 2^{n/2}.$$

By Theorem 5.4, then, $U(IP) \geqslant \log(\text{sign-rank}(N_{IP})) \geqslant n/2$. □

6. Discussion and further reading

This survey article gives only a glimpse into the wonderful world of communication complexity. Motivated by applications, many other models of communication have also been studied. These include the multiparty communication model and the quantum communication model. Lower bound results in such models have been shown to have interesting connections to other models of computation in which communication plays only an implicit role. Examples of such models include Boolean circuits [14] and the cell probe model for data structures [22]. We refer the reader to the book by Kushilevitz and Nisan [15] and the references therein for an encyclopedic treatment of this area.

A recent monograph by Lokam [16] surveys the use of linear algebraic techniques in complexity theory. In particular, it explains how lower bounds on various robustness measures of matrix rank lead to interesting consequences for circuit and communication models. Recently, Sherstov [23] has introduced a new technique, the pattern matrix method, for proving communication lower bounds. By combining this method with Forster's technique, Razborov and Sherstov [21] were able to prove a lower bound on the sign-rank of a Boolean function in AC^0 (that is, computable by a circuit with polynomial size and constant depth). The reader can see the Ph.D. thesis of Sherstov [23] for many interesting recent results.

Acknowledgments

We would like to thank Richard Brualdi, Shaun Fallat, Leslie Hogben, Bryan Shader and Pauline van den Driessche, the organizers of the BIRS Workshop on Theory and Applications of Matrices Described by Patterns, for inviting us to the workshop and for encouraging us to write this survey article.

References

[1] Sanjeev Arora, Boaz Barak, Computational Complexity, Cambridge University Press, Cambridge, 2009.
[2] Stephen Boyd, Lieven Vandenberghe, Convex Optimization, Cambridge University Press, Cambridge, 2004.
[3] Benny Chor, Oded Goldreich, Unbiased bits from sources of weak randomness and probabilistic communication complexity, SIAM J. Comput. 17 (2) (1988) 230–261.
[4] Bruno Codenotti, Gianna Del Corso, Giovanni Manzini, Matrix rank and communication complexity, Linear Algebra Appl. 304 (1–3) (2000) 193–200.
[5] Joel E. Cohen, Uriel G. Rothblum, Nonnegative ranks, decompositions, and factorizations of nonnegative matrices, Linear Algebra Appl. 190 (1993) 149–168.
[6] Paul Erdös, Joel Spencer, Probabilistic Methods in Combinatorics, Probability and Mathematical Statistics, vol. 17, Academic Press, New York, London, 1974.
[7] Shaun M. Fallat, Leslie Hogben, The minimum rank of symmetric matrices described by a graph: a survey, Linear Algebra Appl. 426 (2–3) (2007) 558–582.
[8] Shaun M. Fallat, Leslie Hogben, Variants on the minimum rank problem: a survey II, preprint. arXiv:1102.5142v1.
[9] Jürgen Forster, A linear lower bound on the unbounded error probabilistic communication complexity, J. Comput. System Sci. 65 (4) (2002) 612–625.
[10] Alan Frieze, Ravi Kannan, Quick approximation to matrices and applications, Combinatorica 19 (2) (1999) 175–220.
[11] Ronald L. Graham, Donald E. Knuth, Oren Patashnik, A Foundation for Computer Science, Concrete Mathematics, second ed., Addison-Wesley Publishing Company, Reading, MA, 1994.
[12] Marshall Hall Jr., Combinatorial Theory, Wiley Classics Library, second ed., John Wiley & Sons Inc., New York, 1998.
[13] Leslie Hogben, A note on minimum rank and maximum nullity of sign patterns, Electron. J. Linear Algebra 22 (2011) 203–213.
[14] Mauricio Karchmer, Avi Wigderson, Monotone circuits for connectivity require super-logarithmic depth, SIAM J. Discrete Math. 3 (2) (1990) 255–265.
[15] Eyal Kushilevitz, Noam Nisan, Communication Complexity, Cambridge University Press, Cambridge, 1997.
[16] Satyanarayana V. Lokam, Complexity lower bounds using linear algebra, Found. Trends Theor. Comput. Sci. 4 (1–2) (2008)
[17] László Lovász, Michael Saks, Communication complexity and combinatorial lattice theory, J. Comput. System Sci. 47 (2) (1993) 322–349. (29th Annual IEEE Symposium on Foundations of Computer Science, White Plains, NY, 1988).
[18] Kurt Mehlhorn, Erik M. Schmidt, Las Vegas is better than determinism in VLSI and distributed computing (extended abstract), in: Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing, STOC '82, New York, NY, USA, ACM, 1982, pp. 330–337.
[19] Noam Nisan, Avi Wigderson, On rank vs. communication complexity, Combinatorica 15 (4) (1995) 557–565.
[20] Ramamohan Paturi, Janos Simon, Probabilistic communication complexity, J. Comput. System Sci. 33 (1) (1986) 106–123. (Twenty-fifth Annual Symposium on Foundations of Computer Science, Singer Island, Fla., 1984).
[21] Alexander A. Razborov, Alexander A. Sherstov, The sign-rank of AC^0, SIAM J. Comput. 39 (5) (2010) 1833–1855.
[22] Pranab Sen, S. Venkatesh, Lower bounds for predecessor searching in the cell probe model, J. Comput. System Sci. 74 (3) (2008) 364–385.

[23] Alexander A. Sherstov, Lower bounds in communication complexity and learning theory via analytic methods, Ph.D. thesis, The University of Texas at Austin, 2009.
[24] U.V. Vazirani, Strong communication complexity or generating quasirandom sequences from two communicating semirandom sources, Combinatorica 7 (4) (1987) 375–392.
[25] Mihalis Yannakakis, Expressing combinatorial optimization problems by linear programs, J. Comput. System Sci. 43 (3) (1991) 441–466.
[26] Andrew C. Yao, Some complexity questions related to distributive computing (preliminary report), in: Proceedings of the Eleventh Annual ACM Symposium on Theory of Computing, STOC '79, New York, NY, USA, ACM, 1979, pp. 209–213.
[27] Andrew C. Yao, Lower bounds by probabilistic arguments, in: Proceedings of the 24th Annual Symposium on Foundations of Computer Science, Washington, DC, USA, IEEE Computer Society, 1983, pp. 420–428.