BOOLEAN CIRCUITS, TENSOR RANKS, AND COMMUNICATION COMPLEXITY

PAVEL PUDLÁK[†¶], VOJTĚCH RÖDL[‡], AND JIŘÍ SGALL[§¶]

Abstract.
We investigate two methods for proving lower bounds on the size of small depth circuits, namely the approaches based on multiparty communication games and algebraic characterizations extending the concepts of the tensor rank and rigidity of matrices. Our methods are combinatorial, but we think that the main contribution concerns the algebraic concepts used in this area (tensor ranks and rigidity). Our main results are following.

(i) An $o(n)$ bit protocol for a communication game for computing shifts, which also gives an upper bound of $o(n^2)$ on the contact rank of the tensor of multiplication of polynomials; this disproves some earlier conjectures. A related probabilistic construction gives $o(n)$ upper bound for computing all permutations and $O(n \log\log n)$ upper bound on the communication complexity of pointer jumping with permutations.

(ii) A lower bound on certain restricted circuits of depth 2 which are related to the problem of proving a superlinear lower bound on the size of logarithmic-depth circuits; this bound has interpretations both as a lower bound on the rigidity of the tensor of multiplication of polynomials, and as a lower bound on the communication needed to compute the shift function in a restricted model.

(iii) An upper bound on boolean circuits of depth 2 for computing shifts, and more generally all permutations; this shows that such circuits are more efficient than the model based on sending bits along vertex disjoint paths.

Key words. Circuit, tensor rank, communication complexity, random graph.

AMS subject classifications. 68Q15, 68Q25, 68R10.

1. Introduction. The problem of proving superlinear lower bounds on the size of circuits for an explicitly defined sequence of boolean functions is, perhaps, the most persistent problem in complexity theory. Attempts to solve it have led to several weaker problems which are often of independent interest. The problem is still open even if we look for functions of n input bits with n outputs and we impose an additional restriction that the depth of the circuit is $O(\log n)$.

The main motivation for this paper is a related problem in which the computation is restricted to *linear circuits* over a field $\mathbf{F}$, i.e., circuits which have linear functions as gates. The question is to find an explicit linear function which cannot be computed by a linear circuit of size $O(n)$ and depth $O(\log n)$. Clearly such a function must have non-constant number of output bits, as every one-output linear function can be computed by a balanced tree with linear gates. If the field is GF_2, this is a variant of the problem for general boolean circuits. However, strictly speaking, the problem about linear circuits over GF_2 is not a weaker problem, since the class of computable functions is smaller.

[†] E-mail pudlak@mbox.cesnet.cz. Mathematical Institute AV ČR, Žitná 25, 11567 Praha 1, Czech Republic. This research was done partly while visiting Dept. of Mathematics and Computer Science, Emory University, Atlanta GA, USA.

[‡] E-mail rodl@mathcs.emory.edu. Dept. of Mathematics and Computer Science, Emory University, Atlanta GA 30322, USA. This research was done partly while visiting Mathematical Institute AV ČR, Praha, Czech Republic.

[§] E-mail sgallj@mbox.cesnet.cz. Mathematical Institute AV ČR, Žitná 25, 11567 Praha 1, Czech Republic. This research was done at School of Computer Science, Carnegie-Mellon University, Pittsburgh PA, USA.

[¶] Partially supported by grant A119107 of AV ČR and by grant # 93 025 of the US-Czechoslovak Science and Technology Program.

Many natural functions are not linear in all of the inputs, but only in some subset of them. For example, matrix multiplication is a bilinear function, which means that whenever one of the matrices is fixed, it is a linear function of the entries of the other matrix. A natural extension of linear circuits which includes circuits for such functions are *semilinear circuits*. Suppose that a function $F(p, \vec{x})$ is linear for every fixed parameter p. We say that a circuit C (with $\vec{x}$ as an input) is a *semilinear circuit* for $F(p, \vec{x})$ if for any fixed parameter p we can assign linear functions to the gates of C so that we get a linear circuit for $F(p, \vec{x})$. We want to prove a lower bound on the size of a semilinear circuit computing an explicit $F(p, \vec{x})$.

So far, superlinear lower bounds have been proved only for infinite fields. Shoup and Smolensky proved a lower bound of $\Omega(n \log n / \log \log n)$ for linear circuits of polylogarithmic depth [30]; however, their proof works only for functions with very large values. For finite fields no explicit functions not computable by linear or semilinear circuits of size $O(n)$ and depth $O(\log n)$ are known.

Some superlinear lower bounds have been proved for a much more restricted model of constant depth circuits with linear gates of unbounded fan-in (this is again nontrivial only for multi-output functions). Some of these bounds use results on the complexity of communication networks [13, 24, 26]. The bounds can be extended to bounded depth boolean circuits with arbitrary boolean functions as gates, which proves, for instance, that addition and multiplication cannot be computed by a constant depth circuit of size $O(n)$.

We are interested in various algebraic and combinatorial concepts related to circuit complexity. First we review several such concepts that have been devised for proving circuit lower bounds.

The algebraic approach dates back to Strassen who introduced the concept of *the rank of a tensor* [31]. This rank characterizes up to a constant factor the number of multiplications needed to compute an explicit bilinear function, which is called multiplicative complexity. It is a major open problem in algebraic complexity to prove a superlinear lower bound on the rank of an explicit tensor. Valiant defined *the rigidity of a matrix* [32]. Sufficiently large lower bounds on the rigidity of the matrix defining a linear function would imply superlinear lower bounds on the size of linear circuits of depth $O(\log n)$. However, so far the best bounds for explicit matrices are too small [14]. Later, Razborov considered a modification of the tensor rank, called *contact rank* [27]. This rank characterizes the complexity of certain restricted algebraic circuits (where only multiplications by a variable or a scalar are allowed). He proved a lower bound $\Omega(n^{3/2})$ on the contact rank of the tensor of multiplication of polynomials and on the contact rank of the tensor of matrix multiplication. He also used the contact rank to prove a lower bound on the rigidity of certain matrices. The combinatorial approach (which can be also characterized as information-theoretic) leads to the *multiparty communication complexity* introduced by Chandra, Furst and Lipton [16], which was used to prove lower bounds on circuit complexity in various situations [7, 15, 20] and to the concept of computation with *common bits* introduced by Valiant [32, 33].

In this paper we generalize the concept of rigidity of a matrix to the *rigidity of a tensor*, where a tensor is essentially a set of matrices. This gives a tool for proving lower bounds on semilinear circuits in the same way as the original concept does for linear circuits. We also define another variant of tensor rank, which we call *the rigidity rank* because of its relation to the rigidity of tensors. Furthermore, we define certain refinements of the multiparty communication complexity and computation with common bits and show that they are closely related to the rigidity rank. Thus some bounds can be transferred from communication complexity to the algebraic framework and vice versa.

We consider the complexity of computing all n cyclic shifts of input bits. We prove an $o(n)$ *upper bound* on the corresponding three-party communication game, thus giving a negative answer to a communication complexity question of N. Nisan and A. Wigderson motivated by the problem of circuits of depth $O(\log n)$ and size $O(n)$ [20]. A similar problem about computation with common bits was also posed by Valiant [33]. Since each shift is a linear function, computing all shifts can be presented as a problem about semilinear circuits, or equivalently as a problem about the rigidity of the diagonal tensor, which is the corresponding tensor. We get an upper bound of $o(n^2)$ on the rigidity rank of this tensor. This disproves a conjecture of Razborov [27] that the contact rank of this tensor is $\Omega(n^2)$, since the rigidity rank is at least as large as the contact rank.

This piece of information is quite important, since it shows that certain direct approaches to the problem of superlinear lower bounds for boolean circuits do not work. We have to use more refined notions, like the rigidity of a tensor instead of its contact and rigidity ranks, or special kinds of protocols in the case of communication complexity.

We generalize this upper bound in two ways. First, we consider all permutations instead of just shifts. Using probabilistic methods, we prove that the upper bounds of $o(n)$ on the communication complexity and $o(n^2)$ on the rigidity rank of the corresponding tensor hold even for this much harder problem. Second, for the communication complexity we consider any constant number of players instead of just three, computing the composition of several shifts. In such a case we prove that the communication complexity decreases exponentially with the number of players. These results do not exclude the possibility of applying the communication complexity and related combinatorial tools to the problem of proving a superlinear lower bound for boolean circuits. However, they show that the usual intuition that for simple functions the natural protocols are essentially the best possible, can be wrong.

Furthermore, we prove a *lower bound* on the size of circuits computing all shifts with common bits. In the algebraic language, this gives a lower bound on the rigidity of the diagonal tensor. As a consequence, we improve the current best lower bound on boolean circuits of depth two with arbitrary gates computing multiplication of two integers (written in binary). We also give a slightly larger lower bound on the rigidity rank of the tensor of multiplication of polynomials than is known for contact rank. However, this bound is not strong enough to prove a lower bound for circuits of logarithmic depth.

Another restricted model of computation used for computing permutations of input bits are networks in which the inputs are routed through the network to the corresponding outputs along vertex disjoint paths [23, 21]. This can be viewed as a special case of semilinear circuits, if we allow only projections as the gates instead of general linear functions. We consider the gap between the complexity of *conservative computation* by such networks and *nonconservative computation* by general semilinear circuits. We prove that it is possible to compute all permutations by a semilinear circuit which can be divided into two parts, one with sublinear number of vertices and the other with $o(n^{3/2})$ edges. Using only routing along vertex disjoint paths, such networks cannot compute even all shifts. It is not surprising that circuits are more efficient, but as far as we know this is the first proof of such a fact for computation of a set of permutations.

One of our technical tools is Theorem 4.4 about *coloring random graphs*, which extends a famous estimate of the chromatic number of the random graph due to Bollobás [10] and may be of independent interest. We also use other powerful probabilistic techniques as *martingales* and *Janson's inequality*.

Even though our tools are combinatorial and we obtain some lower bounds on the complexity of certain restricted computations, we think that the main contributions of this paper are the refinement of the algebraic concepts used in this area (tensor ranks and rigidity) and results obtained about them, together with the upper bounds on the multiparty communication complexity and on the size of depth two circuits.

This paper expands the results of the extended abstract [25] and the note [28] and contains some new results.

Independently of us, Babai, Kimmel and Lokam [6] proved a $o(n)$ upper bound on communication complexity of a game corresponding to computing certain set of n permutations (instead of n shifts). Very recently (during the refereeing process) Ambainis [4] improved some of our results on communication complexity (Theorems 4.2 and 4.9). In another recent paper Damm, Jukna, and Sgall [11] prove bounds on the k-party communication complexity of the pointer jumping function in a restricted model not directly corresponding to the models considered here.

Here is an *overview of the paper*. In §2 we give the basic definitions and facts. Section 3 summarizes the relations between ranks of tensors, size of circuits and communication games. In §4 we prove the upper bounds on communication complexity of computing all permutations and iterated shift function, and the corresponding bound on tensor rigidity. The lower bounds on computation of shifts with common bits and the corresponding bounds on the rigidity and rigidity rank of the diagonal tensor are proved in §5. Section 6 proves the gap between conservative and nonconservative computation of shifts. The proof of the estimate on the chromatic number of a random graph is given in the §7.

2. Basic definitions and facts. We index all vectors and tensors starting from 0, as this is convenient for the modular arithmetic. An exception are some tensors that are indexed by a general parameter, which is not necessarily a number, in one coordinate.

2.1. Boolean functions. For circuit bounds we consider *multi-output functions*, whose values are vectors of n bits. The *shift function*, $\mathit{Shift}^n : \{0,\ldots,n-1\}\times\{0,1\}^n \to \{0,1\}^n$ is defined by

$$\mathit{Shift}^n(s,\vec{x}) = \vec{y} \quad \text{where} \quad y_i = x_{(i+s) \bmod n}.$$

This is a very important function, since it can be reduced to many naturally occurring functions, e.g., multiplication of binary numbers, convolution, etc. We believe that linear size circuits of logarithmic depth cannot compute the shift function, and hence also none of the functions to which it can be reduced.

If we use general permutations in place of shifts, we obtain the *permutation function*,

$$\mathit{Perm}^n(\pi,\vec{x}) = \vec{y} \quad \text{where} \quad y_i = x_{\pi(i)},$$

and π is a permutation of the set $\{0,\ldots,n-1\}$. Note that the input size for Perm^n is of the order of $n\log n$, while it is only $n+\lceil\log n\rceil$ for Shift^n.

Any multi-output function can be transformed into a *one-output function* with one extra argument that indexes the output bits. For any function $F(p,\vec{x}) : Y \times X \to \{0,1\}^n$ we define its one-output variant $bit_F : \{0,\dots,n-1\} \times Y \times X \to \{0,1\}$ by

$$bit_F(i,p,\vec{x}) = (F(p,\vec{x}))_i.$$

We use small letters for one-output functions and capitalize the names of corresponding multi-output functions. It is easily seen that we can get a circuit for bit_F by adding only $O(n)$ new gates to the circuit for F so that the depth increases only by $O(\log n)$. As we will see later, the communication complexity of one-output functions is related to the circuit complexity of the corresponding multi-output function.

We define

$$\mathit{shift}^n = bit_{\mathit{Shift}^n}, \text{ i.e., } \mathit{shift}^n(i,j,\vec{x}) = x_{(i+j) \bmod n};$$

$$\mathit{perm}^n = bit_{\mathit{Perm}^n}, \text{ i.e., } \mathit{perm}^n(i,\pi,\vec{x}) = x_{\pi(i)}.$$

These two functions can be generalized to several shifts or permutations. We define

$$\mathit{shift}^n_k(s_1,\dots,s_{k+1},\vec{x}) = x_{(s_1+\dots+s_{k+1}) \bmod n};$$

$$\mathit{perm}^n_k(i,\pi_1,\dots,\pi_k,\vec{x}) = x_{\pi_k\dots\pi_1(i)}.$$

Note that $\mathit{shift}^n(i,j,\vec{x}) = \mathit{shift}^n_1(i,j,\vec{x})$ and $\mathit{perm}^n(i,\pi,\vec{x}) = \mathit{perm}^n_1(i,\pi,\vec{x})$.

All function introduced above are linear in $\vec{x}$. We call such functions semilinear. More precisely, a function $F(p,\vec{x})$ is *semilinear* in $\vec{x}$, if for every fixed parameter p_0 the function $F(p_0,\vec{x})$ is a linear function of $\vec{x}$.

2.2. Circuits. The main motivation for our work is to study methods for proving superlinear lower bounds on the size of circuits with depth $O(\log n)$, where n is the number of inputs and all gates have fan-in two. No such bounds are known even for the functions with n outputs. In a boolean circuit, the gates are arbitrary binary functions. We also consider algebraic circuits, in which case the gates are arbitrary polynomials over the given field.

Algebraic circuits over the field GF_2 and boolean circuits are closely related, but there is a difference. In an algebraic circuit we actually compute in $GF_2(x_1,\dots,x_n)$, i.e., in GF_2 extended by indeterminates $x_1, \dots, x_n$, while in a boolean circuit we use only the elements of GF_2. A boolean function $f : \{0,1\}^n \to \{0,1\}$ can be identified with a multilinear polynomial p of $GF_2[x_1,\dots,x_n]$. Then an algebraic circuit for p is a boolean circuit for f. The converse is, however, not true. A polynomial p' obtained from a boolean circuit C for f is equal to p only after factorizing by the ideal defined by the equations $x_1^2 = x_1, \dots, x_n^2 = x_n$.

We are interested mainly in linear and semilinear functions. For that purpose it is natural to consider circuits in which the gates are linear functions. In the case of boolean circuits this means that the only gates allowed are parity, projections, and their negations. For linear circuits the distinction between boolean and algebraic circuits disappears.

In a circuit for a semilinear function $F(p,\vec{x})$ we consider p to be a parameter of each gate, therefore it is natural to use the following definition. A *semilinear circuit* for a function $\vec{y} = F(p,\vec{x})$ is a directed acyclic graph with sources labeled by the variables $\vec{x}$ and sinks labeled by the variables $\vec{y}$ such that for every fixed p, we can assign linear gates to the nodes so that the resulting circuit computes $F(p, \vec{x})$. We measure the dependence of the circuit size on $n = |x|$.

Every one-output semilinear function has a semilinear circuit of size n and depth $\log n$. However, for most multi-output linear functions $F : \{0,1\}^n \to \{0,1\}^n$ the size of a circuit computing F is $\Omega(n^2/\log n)$, and the same is true for semilinear functions and semilinear circuits. It is an open problem to prove that some explicitly defined boolean semilinear function has no semilinear circuit of size $O(n)$ and depth $O(\log n)$.

An important example of semilinear circuits are algebraic circuits for bilinear functions, which include matrix multiplication, multiplication of polynomials and convolution. Let $F(\vec{y}, \vec{x}) = \sum a_{ij} y_i x_j$ be a bilinear function, let C be an algebraic circuit for $F(\vec{y}, \vec{x})$. If we substitute arbitrary constants for $\vec{y}$, the circuit C computes a linear function. It is well-known and easy to prove that it is possible to convert a general algebraic circuit for a linear function into a linear circuit computing the same function in such a way that the underlying graph is unchanged. (Let us stress that this does not hold for boolean circuits, see [24].) Thus C, or more precisely the underlying graph, is a semilinear circuit for $F(\vec{y}, \vec{x})$.

An important tool for studying the circuits of size $O(n)$ and depth $O(\log n)$ is a reduction discovered by Valiant [32]. We use the terminology he introduced in [33].

Let F be a multi-output boolean function with input variables $\vec{x} = (x_0, \dots, x_{n-1})$ and outputs $\vec{y} = (y_0, \dots, y_{n-1})$. Let G be a bipartite graph with nodes $x_0, \dots, x_{n-1}$ and $y_0, \dots, y_{n-1}$. We say that F *can be computed by the graph G with r common bits,* if there exist boolean functions $h_1, \dots, h_r$ and $g_0, \dots, g_{n-1}$ such that

$$y_i = (F(\vec{x}))_i = g_i(h_1(\vec{x}), \dots, h_r(\vec{x}), \vec{x}^{(i)}),$$

where $\vec{x}^{(i)}$ is the substring of input variables adjacent to y_i in G. In other words, F can be computed by a circuit of depth 2 where there are some direct connections from inputs to outputs given by the graph G and some connections through an intermediate level of r gates of unbounded fan-in. By the *degree* of the graph G we mean the maximal degree of nodes y_i's in the graph G. By the *common bits* we mean the functions $h_1, \dots, h_r$. We are interested in bounding the degree of G and the number of common bits.

In the analogy with semilinear circuits, we are especially interested in the case where the common bits and the functions g_i are semilinear; we then say that the computation by the graph *uses only semilinear functions.* In this restricted case we can also use the same notion for algebraic circuits over a general field $\mathbf{F}$; the functions $h_1, \dots, h_r$ and $g_0, \dots, g_{n-1}$ are then required to be semilinear functions with values in $\mathbf{F}$, instead of boolean functions.

The reduction is based on the following graph-theoretic fact proved in [32, Theorem 5.3].

THEOREM 2.1 (VALIANT [32]). *For every $\varepsilon > 0$, c, and d there exists K such that for any directed acyclic graph C with cn nodes and depth $d \log n$ there exists a set S of $Kn/\log\log n$ edges such that every directed path of length $\varepsilon \log n$ in C contains an edge from S.*

The consequence that is important for us is this.

THEOREM 2.2 (VALIANT [32, 33]). *For every $\varepsilon > 0$, c, and d there exists K such that if a function F can be computed by a circuit of size cn and depth $d \log n$, then it can be computed by a graph G of degree at most n^ε with $Kn/\log\log n$ common bits. Moreover, if the original circuit is semilinear then computation by the graph G uses only semilinear functions.*

Proof. Let K and S be as in Theorem 2.1. Define the common bits to be the functions computed at the edges from S by the circuit C. From Theorem 2.1 it follows that each output can be expressed as a circuit of depth at most $\varepsilon \log n$ with inputs from S and from the original inputs. Construct the graph G by connecting each output to these inputs. Because the gates have fan-in two, this circuit depends on at most n^{ε} original inputs, and hence the degree of G is at most n^{ε}. The semilinearity is preserved, since any function computed in a semilinear circuit is semilinear. ∎

Let us state explicitly what this reduction means for the shift function $Shift^n$. No superlinear lower bounds are known for computing explicitly given functions by semilinear circuits. In particular, it is an open problem, if it is possible to compute shifts by a graph G of degree n^{ε}, $\varepsilon < 1$, with $o(n)$ common bits using only semilinear functions. We show that it is possible to compute by a graph of degree $o(n)$ with $o(n)$ common bits not only all n shifts but in fact all $n!$ permutations. Since the degree of our graph is $\Theta(n \log\log n/ \log n)$ which is much larger than n^{ε}, this does not solve the above open problem. However, it shows that it is possible to compute all permutations by a significantly smaller graph than it was expected.

Valiant's reduction suggests that we consider boolean circuits with gates of unbounded fan-in and constant depth, where the gates are arbitrary boolean functions, or arbitrary linear functions. The size of such a circuit is defined to be the number of edges. (This is a trivial notion for one-output functions.) For this case, some superlinear lower bounds for explicitly given boolean functions are known, see [13, 21, 22]. These bounds are based on some graph properties of circuits computing particular boolean functions. For instance a function can easily be defined so that every circuit for the function is a superconcentrator. Similar graph-theoretic arguments can also be used for the circuits of unbounded fan-in and small depth computing the function $Shift^n$ [26]. For example, for depth 2 this gives the bound of $\Omega(n \log n)$. However, these bounds are very small, and we do not get anything at all for depth $O(\log n)$.

An even more restricted model for computing $Shift^n$ that has been considered requires that the bits are routed along vertex disjoint paths. A graph with n sources $\{x_0, \dots, x_{n-1}\}$ and n sinks $\{y_0, \dots, y_{n-1}\}$ is an *n-shifter* if for every s there exist n vertex disjoint paths $x_{0+s} \to y_0$, $x_{1+s} \to y_1$, $\dots$, $x_{n-1+s} \to y_{n-1}$, where all indices are computed modulo n, see [23]. Obviously, any shifter is a semilinear circuit for $Shift^n$, but this condition seems to be much stronger. The lower bounds known for the shifters are much larger than for general semilinear circuits, e.g., shifters of depth two have size $\Omega(n^{3/2})$, and general shifters have size $\Omega(n \log n)$ (actually, these bounds are tight). These bounds can be applied to monotone circuits, as any circuit with only monotone gates that computes the shift function is a shifter, see [22]. However, for the monotone basis there are well-known even exponential lower bounds for other explicit functions. The reader interested in further applications of these methods to circuit complexity should consult [21, 24].

It is well-known that a general circuit for the shift function is not necessarily a shifter [22]. It is an open problem, if the shift function can be computed more efficiently using general circuits than using circuits that contain shifter graphs. We give a partial result in this direction below, by proving that certain circuits of depth 2 are more efficient than corresponding shifters, see Theorems 6.1 and 6.2.

2.3. Multiparty Communication Complexity. In the most common model of multiparty communication a function $f(\vec{x}_1, \dots, \vec{x}_k)$ is computed in the following way. There are k players; we denote them by Player 1, $\dots$, Player k. Player i knows $\vec{x}_1$, $\dots$, $\vec{x}_{i-1}, \vec{x}_{i+1}, \dots, \vec{x}_k$. They send messages consisting of binary strings, and the game ends when one of the players knows the answer $y = f(\vec{x}_1, \ldots, \vec{x}_k)$. The communication complexity is the minimal number of bits of communication needed in any protocol which computes $f(\vec{x}_1, \ldots, \vec{x}_k)$ correctly. For more background see e.g. [19]. We omit the general definition of a protocol, because in this paper we consider only some restricted models explained below. In particular, we require that each player sends only one message, and that all players send their messages simultaneously. This means that the message can depend only on the input available to the player, not on the previous communication. A lower bound for such a restricted model should be easier to prove, while it would still imply the circuit lower bound.

Now we define our models more precisely. In the *simultaneous* model for computing $f(\vec{x}_1, \ldots, \vec{x}_k)$ there are $k+1$ players, Player 0, Player 1, ..., Player k. The additional Player 0 does not have access to any input, while the others have the same access as in the general model. There is only one round of communication, when each of the Players 1, ..., k sends a piece of information to Player 0. Then Player 0 produces the answer. A slight modification is an *almost simultaneous protocol*: There are k Players, and the communication has one round, in which Players 1, ..., $k-1$ send independently a message to Player k who produces the answer. The *communication complexity* is the total number of bits that has to be communicated for the inputs of given length. By $SCC(f)$ and $ASCC(f)$ we denote the simultaneous and almost simultaneous communication complexity of f, respectively. Let us observe that $ASCC(f)$ differs from $SCC(f)$ by at most the sum of the sizes of inputs $\vec{x}_1$, ..., $\vec{x}_{k-1}$, since Player k can just send all the inputs he has access to Player 0, which changes an almost simultaneous protocol into a simultaneous protocol. Therefore, if the size of the first inputs is small, the difference can be disregarded. This is true for example for $shift_k^n$ where the size of the first two inputs is only $\log n$.

A *semilinear protocol* is an almost simultaneous protocol such that for every fixed $\vec{x}_1$, ..., $\vec{x}_{k-1}$ the message send by each Player i consists of a vector of fixed linear functions of $\vec{x}_k$. A *restricted protocol* is an almost simultaneous protocol for three players such that for any fixed $\vec{x}_1$ the message sent by Player 2 consists of a fixed substring of $\vec{x}_3$. A *restricted semilinear protocol* satisfies both conditions, i.e., Player 1 sends a vector determined by linear functions of $\vec{x}_3$ and Player 2 sends a substring of $\vec{x}_3$.

We study the communication complexity of the functions $shift_k^n$ and $perm_k^n$.

Let us first examine our intuition about restricted protocols for $shift^n(i, j, \vec{x})$. If Player 2 does not send anything, the only knowledge of Player 3 about $\vec{x}$ comes from Player 1. Therefore Player 1 has to send n bits about $\vec{x}$, because without knowledge of j he cannot determine which of the bits is relevant. (More precisely, for any two $\vec{x} \neq \vec{x}'$ the message sent by Player 1 has to be different.) Player 2 has to select a subset of bits without the knowledge of i, therefore if he sends only a small number bits, in most cases he does not send the bit x_{i+j}. It would appear that then his message can be hardly of any significant help in determining the output, and therefore Player 1 has to send a long message even in this case. A natural conjecture is that the total number of communicated bits for $shift^n$ must be $\Omega(n)$. As we shall see, this would yield a proof that there is no circuit of size $O(n)$ and depth $O(\log n)$ computing $Shift^n$. However, we disprove this conjecture by showing that even for the more general function $perm^n$ there exists a restricted semilinear protocol with the total communication of only $O(n \log \log n / \log n)$ bits, see Corollary 4.8.

Clearly the multiparty communication complexity of these functions can only decrease with increasing k in any of the above models. It is a major open problem to study the growth rate of communication complexity when k increases. It seems beyond the present means to prove any good bounds for increasing k. For the function $shift_k^n$ we have a partial result in that direction. We construct a restricted semilinear protocol with the total communication of only $O(n(\log\log n/\log n)^k)$ bits for any constant k.

The best known lower bound for $shift_k^n$ is given in the following proposition. It is a straightforward generalization of the bound for $k=1$ by Nisan and Wigderson [20].

PROPOSITION 2.3. *For every constant k, $SCC(shift_k^n) \geq \Omega(n^{\frac{1}{k+1}})$.*

Proof. Assume that the numbers $s_1,\dots,s_{k+1}$ are written as $(k+1)$-digit numbers in the basis of $b=\lceil n^{\frac{1}{k+1}}\rceil$. We restrict the domain of $shift_k^n(s_1,\dots,s_{k+1},\vec{x})$ so that in s_1 only the first digit may be non-zero, in s_2 only the second digit may be non-zero, etc. (Thus, for $k=1$, s_1 ranges over $0,\sqrt{n},2\sqrt{n},\dots,n$ and s_2 ranges over $0,1,\dots,\sqrt{n}$.) Each s_i can have only b different values, but $s_1+\cdots+s_{k+1}$ can be any number between 0 and $b^{k+1}\geq n$.

Now consider any simultaneous protocol for $shift_k^n$ and for a fixed x write down the information communicated by each player on all inputs $s_1,\dots,s_{k+1}$ from our restricted range. The total amount of communication is at most $O(nSCC(shift_k^n))$ bits. Because the protocol is simultaneous, the information communicated by each player depends only on the inputs available to him. For every input of a given player there are b different values of the input that he does not see, and therefore the same output appears at b predetermined positions. Thus the information contained in all communication is only $O(nSCC(shift_k^n)/b)$ bits. Given this information, it is possible to reconstruct the whole vector $\vec{x}$, because we can reconstruct any bit x_i by choosing inputs such that $s_1+\cdots+s_k=i$, and following the protocol. Thus for any two different vectors $\vec{x}$ this information must be different, and hence it has at least n bits for some $\vec{x}$. This gives us $SCC(shift_k^n)\geq\Omega(b)\geq\Omega(n^{\frac{1}{k+1}})$. $\square$

The best known lower bound on multiparty communication complexity is $\Omega(n/c^k)$ for the generalized inner product [7]. This means that we have no lower bounds at all for $k=\Omega(\log n)$. If we could extend the lower bounds on the almost simultaneous communication complexity to $k=\mathrm{polylog}(n)$, it would yield a lower bound on ACC circuits, which is a major open problem, see [34, 15, 8].

One function often considered in this context is pointer jumping in a directed acyclic graph with one source and k additional levels with n vertices on each level (the out-degree of each vertex is 1, except for the last level). The inputs are divided between the players so that every player sees everything except for one level. Intuitively the most difficult instances should be those in which all vertices in the graph are reachable, which means that on each level, the mapping given by the pointers is one-to-one. This restriction of pointer jumping is essentially a variation of our function $perm_k^n$. We define

$$jump_k^n(i,\pi_1,\dots,\pi_k)=\pi_k\dots\pi_1(i).$$

We can use our bound for the function $perm_k^n$, Corollary 4.8, to improve the obvious upper bound of $O(n\log n)$ on the communication complexity of $perm_k^n$ by a factor of $\log n/\log\log n$.

PROPOSITION 2.4. *For all $k\geq 2$, $ASCC(jump_k^n)\leq O(n\log\log n)$.*

Proof. We can compute the $\lceil\log n\rceil$ bits of $jump_k^n(i,\pi_1,\dots,\pi_k)$ by computing the value of $perm_{k-1}^n(i,\pi_1,\dots,\pi_{k-1},Perm^n(\pi_k,\vec{x}))$ for $\log n$ vectors $\vec{x}$ as follows. Define vectors $\vec{x}^{(j)}$, $1\leq j\leq\lfloor\log n\rfloor$, so that $\vec{x}_i^{(j)}$, $0\leq i<n$, is the jth digit in the binary

representation of i. By the definitions of $Perm^n$, $perm^n_{k-1}$, and $jump^n_k$, the jth bit of $jump^n_k(i,\pi_1,\dots,\pi_k)$ is $perm^n_{k-1}(i,\pi_1,\dots,\pi_{k-1},Perm^n(\pi_k,\vec{x}^{(j)}))$. We can run the $\lceil\log n\rceil$ protocols for $perm^n_{k-1}$ in parallel, as they are completely independent; note that all players have the necessary information, since the input is divided in the same way for $jump^n_k$ as for all the instances of $perm^n_{k-1}$. The total communication is $\lceil\log n\rceil ASCC(perm^n_{k-1}) = O(n\log\log n)$, by Corollary 4.8. □

Our results do not exclude the function $jump^n_k$ as a candidate for lower bounds. But we think that they give some insight into the difficulties that are encountered in the attempted proofs.

2.4. Tensors. By a *tensor* over a field F we simply mean a three-dimensional matrix or, equivalently, a finite sequence of matrices of the same size with entries from F. We use the following notation to denote the *slices* of tensors in different directions. For a tensor T the symbol $T_{i,*,*}$ denotes the matrix consisting of all entries of T with the first coordinate i; the matrix is indexed in the same way as the remaining two coordinates of the tensor. Similarly, $T_{i,j,*}$ denotes the vector of all entries with the first coordinates i and j, i.e., the j-th row of the matrix $T_{i,*,*}$. We define $T_{*,j,*}$, $T_{*,j,k}$, etc. similarly.

Tensors can be naturally associated with semilinear functions. Given a tensor T, for each parameter p the matrix $T_{*,p,*}$ determines a linear function. Thus the function defined by $F(p,\vec{x}) = T_{*,p,*}\vec{x}$ is a semilinear function naturally corresponding to the tensor T. Conversely, let $F(p,\vec{x})$ be a function linear in $\vec{x}$. Then there exists a tensor T^F and a vector $\vec{c}^F$, such that for all p,

$$F(p,\vec{x}) = T^F_{*,p,*}\vec{x} + \vec{c}^F_p .$$

The constants $\vec{c}^F_p$ have almost no influence on the complexity of computing F. The main information is contained in the tensor T^F, which we call a *tensor corresponding to F*.

We study the tensors corresponding to the functions introduced in §2.1. The tensor corresponding to $Shift^n$ is defined by $D^n = T^{Shift^n}$, i.e.,

$$\begin{aligned} D^n_{i,j,k} &= 1 \quad \text{if} \quad i+j \equiv k \pmod{n},\\ &= 0 \quad \text{otherwise.} \end{aligned}$$

D^n is called a *diagonal tensor*, or the *tensor of multiplying polynomials* modulo x^n-1, since it is connected with algebraic circuits computing this bilinear operation [12].

The tensor corresponding to $Perm^n$ is defined by $P^n = T^{Perm^n}$, i.e.,

$$\begin{aligned} P^n_{i,\pi,k} &= 1 \quad \text{if} \quad k = \pi(i),\\ &= 0 \quad \text{otherwise.} \end{aligned}$$

It turns out that the size of a circuit computing a semilinear function is related to some variants of the algebraic concept of rank.

Let F be some fixed field. For a positive integer k, let $e^k_0, \dots, e^k_{k-1}$ denote the standard basis of the vector space F^{k}, i.e., $e^k_i(j) = 0$ for $i \neq j$, and $e^k_i(i) = 1$. For $u \in \mathrm{F}^{\mathrm{l}}$, $v \in \mathrm{F}^{\mathrm{m}}$, $w \in \mathrm{F}^{\mathrm{n}}$, the *tensor product* of u, v, w denoted by $u \otimes v \otimes w$ is the $l\times m\times n$ tensor T defined by $T_{i,j,k} = u_i v_j w_k$ for $0 \le i < l$, $0 \le j < m$, $0 \le k < n$.

We define three different ranks of a tensor. In all three cases the rank of T is defined to be the minimal number of rank 1 tensors T^i such that $T = \sum_i T^i$; it is 0 if all entries of T are zeros. Note that the matrix rank can be defined in a similar way, since matrices of rank 1 are the matrices of the form $u \otimes v$, where u, v are nonzero vectors.

The usual *tensor rank*, $\mathrm{rank}(T)$, was introduced by Strassen [31] and is determined by

$$\mathrm{rank}(T) = 1 \quad \text{iff} \quad \begin{array}{l} T = u \otimes v \otimes w, \\ \text{for some nonzero vectors } u, v, w. \end{array}$$

The *contact rank*, $\mathrm{rank}^{2,2}(T)$, was introduced by Razborov [27] and is determined by

$$\mathrm{rank}^{2,2}(T) = 1 \quad \text{iff} \quad \begin{array}{l} T = e_i^l \otimes v \otimes w \quad \text{or} \quad T = u \otimes e_j^m \otimes q, \\ \text{for some nonzero vectors } u,\ v,\ w,\ q \text{ and some } i,\ j. \end{array}$$

In other words $\mathrm{rank}^{2,2}(T) = 1$ if all nonzero entries of T are either in the slice $T_{i,*,*}$ or in the slice $T_{*,j,*}$, for some i or j, and the matrix rank of that slice is 1.

We introduce the *rigidity rank* and denote it by $\mathrm{rank}^{2,1}(T)$; it is determined by

$$\mathrm{rank}^{2,1}(T) = 1 \quad \text{iff} \quad \begin{array}{l} T = u \otimes e_j^m \otimes w \quad \text{or} \quad T = e_i^l \otimes v \otimes e_k^n, \\ \text{for some nonzero vectors } u,\ v,\ w \text{ and some } i,\ j,\ k. \end{array}$$

Thus $\mathrm{rank}^{2,1}(T) = 1$ if

(i) either all nonzero entries of T are in the slice $T_{*,j,*}$ for some j, and the matrix rank of $T_{*,j,*}$ is 1,

(ii) or all nonzero entries of T are in the column $T_{i,*,k}$ for some i and k and at least one entry is not zero.

From these definitions it is easy to see that for every tensor T,

$$\mathrm{rank}(T) \le \mathrm{rank}^{2,2}(T) \le \mathrm{rank}^{2,1}(T).$$

To give yet another equivalent definition of $\mathrm{rank}^{2,1}$, we define $\mathrm{diff}(T, U)$ to be the *set of different columns* for T and U, i.e.,

$$\mathrm{diff}(T, U) = \{(i, k); \exists j \ T_{i,j,k} \neq U_{i,j,k}\}.$$

Using the fact that the rank of a matrix M is the minimal number of matrices of rank 1 that add to M, it is easy to see that

$$\mathrm{rank}^{2,1}(T) = \min_U \left(\sum_j \mathrm{rank}(U_{*,j,*}) + |\mathrm{diff}(T, U)| \right). \tag{1}$$

where the minimum is taken over all tensors U of the appropriate dimensions.

Now we extend the concept of rigidity from matrices to tensors. The *rigidity of a matrix* M is the function $R_M(r)$ equal to the minimal number of changes needed to reduce the rank of M to r or less [32]. If we want to work with a set of matrices, we can represent them as slices $T_{*,j,*}$ of a tensor T. The *rigidity of a tensor* T is the function $R_T(r)$ which for each r gives the minimal number of columns in which we have to change the tensor in order to reduce the rank of each slice to r or less. More precisely,

$$R_T(r) = \min\{|\mathrm{diff}(T, U)|; U \text{ is a tensor such that } \forall j \ \mathrm{rank}(U_{*,j,*}) \le r\}. \tag{2}$$

Note that in the special case of a single matrix we just get Valiant's rigidity.

It is known that the rank of the diagonal tensor D^n is $\Theta(n)$ for any field; for small fields it was proved only in 1987 by D.V. Chudnovsky and G.V. Chudnovsky using methods from algebraic geometry [17], also see [29] for a more detailed presentation. The growth rate of $\mathrm{rank}^{2,2}(D^n)$ and $\mathrm{rank}^{2,1}(D^n)$ is not known.

Razborov proved that $\mathrm{rank}^{2,2}(D^n) = \Omega(n^{3/2})$, which shows that the gap between rank and $\mathrm{rank}^{2,2}$ can be big [27]. He conjectured that $\mathrm{rank}^{2,2}(D^n) = \Omega(n^2)$.

We disprove this conjecture by proving an upper bound on $\mathrm{rank}^{2,1}(D^n)$. In fact, we prove a stronger result, an upper bound on rigidity of P^n, the tensor of computing all permutations,

$$R_{P^n}(r) = n^{2-\Omega(r/n)}.$$

This implies an upper bound on the rigidity rank of the same tensor,

$$\mathrm{rank}^{2,1}(P^n) = O\left(n^2 \frac{\log\log n}{\log n}\right).$$

The diagonal tensor D^n is a subtensor of P^n, hence the same upper bounds hold for D^n. The rigidity rank is always greater than or equal to the contact rank, so we get the upper bound

$$\mathrm{rank}^{2,2}(D^n) = O\left(n^2 \frac{\log\log n}{\log n}\right).$$

Rigidity and rigidity rank are closely related. From the equations (1) and (2) it follows easily that

$$\mathrm{rank}^{2,1}(T) \leq R_T(r) + nr. \tag{3}$$

Using this fact it is theoretically possible to get a lower bound on the rigidity of a tensor from a lower bound on the rigidity rank of it. However, the lower bounds needed for this approach to work are far beyond what we are able to prove nowadays.

In this paper we prove a lower bound on rigidity of D^n directly. We get

$$R_{D^n}(r) = \Omega\left(\frac{n^2}{r}\log\frac{n}{r}\right).$$

The same bound was proved even for a single matrix of J. Friedman [14]. However because we prove our bound for a special set of matrices, it enables us to prove the following lower bound on rigidity rank

$$\mathrm{rank}^{2,1}(D^n) = \Omega\left(n^{3/2}(\log n)^{1/2}\right).$$

This bound in turn implies some improvement of the lower bounds on the size of certain circuits of depth two.

3. Mutual relations. The concepts of computing a function by a graph with common bits, rigidity of the corresponding tensor, and communication complexity of the corresponding one-output function are closely related and can be all potentially useful for proving lower bounds on the size of circuits. We survey these relations in this section. Some parts of this material has been known and has appeared either explicitly or implicitly in the literature, see [19, 20, 32, 33]. We quantify the connections as precisely as possible, because our upper bounds show that this might be necessary for proving lower bounds.

We first demonstrate the connections on an example. We show a restricted semilinear protocol for the function $shift^n(i, j, \vec{x})$ in which Player 2 sends only 1 bit and Player 1 sends $\lfloor n/2 \rfloor$ bits. This protocol also demonstrates some of the ideas used in our more complicated upper bounds. According to the intuition discussed before Proposition 2.3, Player 2 almost never sends the bit that is to be computed. Before reading on, the reader can try to imagine how this single bit could possibly be used so that it saves half of the communication of the other player.

All indices in the protocol are computed modulo n. The protocol works as follows:

Player 1 sends the bits $x_0 \oplus x_j$, $x_1 \oplus x_{j-1}$, $\ldots$, i.e., parities of all pairs $x_k \oplus x_{j-k}$;
Player 2 sends the bit x_{n-i};
Player 3 computes x_{i+j} as the parity of x_{n-i} (from Player 2) and $x_{n-i} \oplus x_{i+j}$ (from Player 1).

The corresponding bound on computation by a graph with common bits is for the function $Shift^n(j, \vec{x})$. We construct a graph of degree 1 with $\lfloor n/2 \rfloor$ common bits that computes $Shift^n(j, \vec{x})$. Every y_i is adjacent to x_{n-i}, which corresponds to the bit sent by Player 2. For every j, the common bits are the same as the bits sent by Player 1 in the protocol. The output level computes x_{i+j} in the same way as Player 3 in the protocol.

The corresponding bound on the rigidity says that $R_{D^n}(\lfloor n/2 \rfloor) \leq n$ for the diagonal tensor D^n. To prove that, we flip all the entries $D_{i,j,n-i}$. It is easy to verify that the rank of each matrix $D_{*,j,*}$ is at most $\lfloor n/2 \rfloor$, because the matrix is essentially a matrix with ones only along the main diagonal and along the diagonal running from the lower left corner to the upper right corner (and zero in the intersection of the diagonals).

Now we state these relationships formally.

PROPOSITION 3.1. *A function $F(p, \vec{x})$ can be computed by a graph of maximal degree d with r common bits if and only if there exists a restricted protocol for computing $bit_F(i, p, \vec{x})$ in which Player 1 sends r bits and Player 2 sends d bits.*

The computation by the graph uses only semilinear functions if and only if the protocol is a restricted semilinear protocol.

Proof. Suppose that we have a graph that computes the function F. The protocol for f is as follows. Player 1 sends the value of the common bits for the given input $(p, \vec{x})$. If the first input is i, Player 2 sends the value of all inputs x_k that are adjacent to the output y_i in the graph. Player 3 can compute the output because he knows all of the inputs of the output gate. Clearly this protocol is a restricted protocol with the required number of bits sent by each player.

Given a protocol, we construct a graph that computes F as follows. The output y_i is adjacent to all inputs x_k such that Player 2 sends the value of input x_k if the first input is i. This graph computes F if the common bits compute the values communicated by Player 1 on the given input. Clearly, the degree of the graph and the number of the common bits are as required.

For both directions, the semilinearity is preserved, as the functions computed in the protocol and used in the computation by the graph are identical. ∎

Computing F by a graph with a bounded number of edges and a bounded number of common bits using only semilinear functions is equivalent to a bound on rigidity of the tensor T.

PROPOSITION 3.2. *Let $F(p, \vec{x})$ be a semilinear function. Then the rigidity of the* *corresponding tensor T^F satisfies $R_{T^F}(r) \leq R$ if and only if there exists a graph G with at most R edges such that for every p, $F(p, \vec{x})$ can be computed by G with r common bits using only semilinear functions.*

Proof. First assume that the condition on the rigidity is satisfied, which means that we have a tensor U such that $|\mathrm{diff}(T^F, U)| \leq R$ and $\forall p\ \mathrm{rank}(U_{*,p,*}) \leq r$. We choose the graph G so that the input x_k and the output y_i are adjacent if $(i,k) \in \mathrm{diff}(T^F, U)$. Obviously, there are at most R edges.

Now we show how this graph can compute F for fixed p with only r common bits. Let us denote the ith row of the matrix $U_{*,p,*}$ by $\vec{u}^{(i)}$. The ith output function is then given by

$$(F(p,\vec{x}))_i = T^F_{i,p,*} \cdot \vec{x} + c_i = \vec{u}^{(i)} \cdot \vec{x} + \sum_{(i,k)\in \mathrm{diff}(T^F,U)} a_{i,k} x_k + c_i,$$

where $a_{i,k}$ and c_i are some scalar constants from the given field. It follows that it suffices to set the common bits so that $\vec{u}^{(i)} \cdot \vec{x}$ can be computed from them for all i, as the rest of the expression can be computed based on the inputs adjacent to the given output in G.

From the condition on the rank, it follows that there exist r vectors $\vec{v}^{(1)}, \ldots, \vec{v}^{(r)}$ such that any $\vec{u}^{(i)}$ is their linear combination. We set the common bits to $\vec{v}^{(1)} \cdot \vec{x}, \ldots, \vec{v}^{(r)} \cdot \vec{x}$, hence any $\vec{u}^{(i)} \cdot \vec{x}$ can be computed as their linear combination.

For the other direction, we assume that F can be computed by a graph G. We want to find a tensor U which proves that the rigidity of T^F is small. For a fixed p we know that

$$(F(p,\vec{x}))_i = g_i(h_1(\vec{x}), \ldots, h_r(\vec{x}), \vec{x}^{(i)}),$$

where $\vec{x}^{(i)}$ is the substring of input variables adjacent to y_i in G, and both g_i and the common bits $h_1, \ldots, h_r$ are linear functions. So the function g_i can be written as a linear combination of the common bits, the extra inputs and a constant. Similarly to the other direction we set $U_{i,p,*}$ to be a vector which corresponds to the linear combination of the common bits used for the given output. The rank of the tensor is bounded by the number of common bits, and it differs from T^F only in columns corresponding to the edges of G. □

The relation between the rigidity of tensors and the communication complexity of the corresponding semilinear function now follows easily. We just have to examine the proof of Proposition 3.1, and verify that if we replace the maximal degree of the graph by the number of edges, the bound on the number of bits communicated by Player 2 is replaced by the average number of bits.

PROPOSITION 3.3. *Let $F(p, \vec{x})$ be a semilinear function. Then the rigidity of the corresponding tensor T^F satisfies $R_{T^F}(r) \leq dn$ if and only if there exists a restricted semilinear protocol for the associated one-output function $bit_F(i, p, \vec{x})$ in which Player 1 always sends r bits and Player 2 sends d bits on the average, where the average is taken over all values of i.*

Using Valiant's reduction, Theorem 2.2, we get the following two theorems.

THEOREM 3.4. *If $F(p, \vec{x})$ can be computed by a circuit of size $O(n)$ and depth $O(\log n)$, then for any $\varepsilon > 0$ there exists a constant K such that*

(i) *F can be computed by a graph G of degree at most n^ε and with $Kn/\log\log n$ common bits;*

(ii) *there exists a restricted protocol for $bit_F(i, p, \vec{x})$ such that Player 1 sends $O(n/\log\log n)$ bits and Player 2 sends n^ε bits.*

THEOREM 3.5. *If a semilinear function $F(p, \vec{x})$ can be computed by a semilinear circuit of size $O(n)$ and depth $O(\log n)$, then for any $\varepsilon > 0$ there exists a constant K such that*

(i) *F can be computed by a graph G of degree at most n^ε and with $Kn/\log\log n$ common bits using only semilinear functions*;

(ii) *there exists a restricted semilinear protocol for $bit_F(i, p, \vec{x})$ in which Player 1 sends $O(n/\log\log n)$ bits and Player 2 sends n^ε bits*;

(iii) *the rigidity of the corresponding tensor T^F satisfies $R_{T^F}(Kn/\log\log n) \leq n^{1+\varepsilon}$*

Originally several researchers believed that a superlinear lower bound for circuits of depth $O(\log n)$ can be proved by showing that

(i) *Shift*^{n} cannot be computed by a graph of degree $o(n)$ with $o(n)$ common bits, or

(ii) $SCC(\mathit{shift}^n) = \Omega(n)$, or

(iii) $\mathrm{rank}^{2,1}(D^n) = \Omega(n^2)$ (which would give bounds for semilinear circuits).

We prove that all these statements are false by exhibiting an almost simultaneous protocol for *shift*^{n} in which both players send $O(n \log\log n/\log n)$ bits. This does not mean that these approaches cannot be used at all—to demonstrate that we would need a protocol in which Player 2 sends only n^ε which is less than our bound. However, it shows that it is not sufficient to consider the total communication—to prove a lower bound that way, it would be necessary to prove that more than $\omega(n/\log\log n)$ bits are needed, and our bound shows that this is not the case. Hence it is necessary to consider more precise information, namely to estimate the number of bits sent by each of the two players instead of the total amount of communication, to use rigidity instead of the rigidity rank, or to estimate the degree of the graph in a different way than the number of common bits.

4. Upper bounds. In §4.1 we present the constructive proof of the upper bound on the communication complexity of the shift function *shift*^{n}. Then we generalize it in two different ways. In §4.2 we extend the upper bound to the permutation function *perm*^{n} using probabilistic method. These bounds also apply to the rigidity of the corresponding functions. In §4.3 we extend the upper bound to the function *shift*^n_k.

4.1. The shift function. The constructive proof of the upper bound for the function *shift*^{n} is based on a suggestion of Avi Wigderson to use arithmetic progressions.

The idea of the protocol is to divide the input $\vec{x}$ into groups. Player 1 sends the parity of each group and Player 2 sends some substring of $\vec{x}$ such that all but one elements of the group of x_{i+j} are sent, similarly as in the simple protocol at the beginning of §3. For the function *shift*^{n} we can do it constructively, based on the next lemma. It says that there exists a sparse subset B of $[0, n-1]$ such that every $a \in [0, n-1]$ can be surrounded by an arithmetic progression with all elements in B except, possibly, a itself.

LEMMA 4.1. *For every n and every $l < \log n/(2 \log\log n)$ there exists a set $B \subseteq [0, n-1]$ of size $O(ln^{1-1/(2l-1)})$ such that for every $a \in [0, n-1]$ there exist $a_1, \ldots, a_{2l-2} \in B$ such that $a_1, \ldots, a_{l-1}$, a, $a_l, \ldots, a_{2l-2}$ is an arithmetic progression (computing modulo n) with modulus $m = m(a)$ bounded by $m \leq O(n^{1-1/(2l-1)})$.*

Proof. From the well-known bounds on the distribution of primes it is easy to prove that there exist $2l-2$ primes $p_1, \ldots, p_{2l-2}$ larger than $\Omega(n^{1/(2l-1)})$ whose product is at most $n^{1-1/(2l-1)} < n/l$. Define B to be the set of all integer multiples of these primes between $-n$ and $2n$, taken modulo n, i.e.,

$$B = \{b \bmod n; b \in [-n, 2n] \wedge (p_1|b \vee p_2|b \vee \ldots \vee p_{2l-2}|b)\}.$$

For a given $a \in [0, n-1]$ consider the following system of linear congruences

$$\begin{aligned}
a-(l-1)m &\equiv 0 \pmod{p_1} \\
a-(l-2)m &\equiv 0 \pmod{p_2} \\
&\vdots \\
a-m &\equiv 0 \pmod{p_{l-1}} \\
a+m &\equiv 0 \pmod{p_l} \\
a+2m &\equiv 0 \pmod{p_{l+1}} \\
&\vdots \\
a+(l-1)m &\equiv 0 \pmod{p_{2l-2}}
\end{aligned}$$

Since $0 \leq l < p_1, \ldots, p_{2l-2}$, we can divide the congruences by $l-1, l-2, \ldots, 1, 1, 2, \ldots, l-1$ and apply Chinese remainder theorem to solve for m. We obtain a solution $0 < m \leq p_1 p_2 \cdots p_{2l-2}$, which is a modulus satisfying the requirements, because all the numbers $a_1 = a-(l-1)m$, $a_2 = a-(l-2)m$, $\ldots$, $a_{l-1} = a-m$, $a_l = a+m$, $\ldots$, $a_{2l-2} = a+(l-1)m$ taken modulo n are in B. ∎

THEOREM 4.2. *There exists a restricted semilinear protocol for the shift function* $shift^n$ *which requires only* $O(n \log\log n/\log n)$ *bits of communication.*

Proof. Let B be a set with the properties as in Lemma 4.1, for an l which we choose later. For a set $C \subseteq [0, n-1]$, let $x[C]$ denote the subsequence of the bits of x indexed by C. All indices of x are taken modulo n. The protocol is as follows.

Player 1 partitions the interval $[0, n-1]$ into $n' = O(n/l)$ arithmetic progressions $C_0, \ldots, C_{n'-1}$ of length at most l with modulus $m(j)$. Player 1 communicates $\bigoplus x[C_\alpha]$ for all $\alpha = 0, \ldots, l$, i.e., parity of bits of x indexed by each of the progressions, total of $O(n/l)$ bits.

Player 2 sends $x[B+i]$, i.e., bits x_{b+i} for all $b \in B$.

Player 3 computes x_{i+j} as described below.

Let C be the C_α containing $i+j$, let $c \in C$, $c \neq i+j$. By the definition of C, $c-i-j$ is a small nonzero multiple of $m(j)$. By the definition of B and $m(j)$ it follows that $j+(c-i-j) = c-i \in B$, and hence $c \in B+i$. This means that Player 2 communicated all bits of $x[C]$ except possibly x_{i+j}. Hence Player 3 can compute x_{i+j} by subtracting these bits sent by Player 2 from $\bigoplus x[C]$ sent by Player 1, using the fact that he knows i and j and therefore he knows the meaning of the bits send by Players 1 and 2.

For $l = \log n/(4 \log\log n)$, the size of B is $O(n/\log n) = O(n \log\log n/\log n)$, and each player communicates $O(n \log\log n/\log n)$ bits. ∎

COROLLARY 4.3. $\mathrm{rank}^{2,1}(D^n) = O(n^2 \log\log n/\log n)$.

Proof. By Proposition 3.3 we get $R_{D^n}(n \log\log n/\log n) = O(n^2 \log\log n/\log n)$; the result follows using the relation of the rigidity and the rigidity rank (3). ∎

4.2. The permutation function. We prove an upper bound on the rigidity of the tensor $Perm^n$ using the following bound on the chromatic number χ of a random graph $\mathbf{G}(n, q)$, which is an undirected random graph on n vertices with each edge chosen independently at random with probability q.

THEOREM 4.4. *For every $\varepsilon > 0$ there exists $\delta > 0$ and n_0 such that*

$$Pr\left[\chi(\mathbf{G}(n,q)) \le \left(\frac{1}{2}+\varepsilon\right)\frac{-n\log(1-q)}{\log n}\right] > 1-\exp\left(-n^{1+\delta}\right),$$

as long as $7/8 < q < 1 - 1/n^{\frac{1}{2}-\varepsilon}$ and $n \ge n_0$.

This is essentially the theorem which was proved by B. Bollobás for every *constant* q [9]. However for our application we are interested in cases when q approaches 1 as n increases which have not been considered in the literature. Therefore we give the proof of Theorem 4.4 in §7.

Let $\mathbf{S}(n,p)$ be the sum of independent Bernoulli random variables with mean p.

LEMMA 4.5. *Let $n, r, d, 0 < p < 1$ be numbers such that*

$$n!\cdot Pr\left[\chi(\mathbf{G}(n,1-p^2)) > r\right] + n\cdot Pr\left[\mathbf{S}(n,p) > d\right] < 1. \tag{4}$$

Then $perm^n$ can be computed by a restricted semilinear protocol where Player 1 sends r bits and Player 2 sends d bits.

Proof. Let $\{x_0,\dots,x_{n-1}\}$ and $\{y_0,\dots,y_{n-1}\}$ be disjoint sets of distinct vertices. Take a random bipartite graph $\mathbf{H}$ with each edge (x_k,y_i) chosen independently with probability p. Let π be a permutation on $\{0,\dots,n-1\}$. Let $\mathbf{G}_\pi$ be an undirected graph on $\{0,\dots,n-1\}$ such that for $i \ne j$, (i,j) is and edge of $\mathbf{G}_\pi$ if both edges $(x_i, y_{\pi^{-1}(j)})$ and $(x_j, y_{\pi^{-1}(i)})$ are in $\mathbf{H}$. Then $\mathbf{G}_\pi$ is a random graph with the same distribution as $\mathbf{G}(n,p^2)$, and its complement $\overline{\mathbf{G}}_\pi$ is a random graph from $\mathbf{G}(n,1-p^2)$. By the assumption (4) there exists a graph H such that the degree of H is at most d and for each π, $\chi(\overline{\mathbf{G}}_\pi) \le r$. This means that for each π the graph $\mathbf{G}_\pi$ can be covered by at most r cliques.

Using such a graph H, we construct a protocol for $perm^n(i,\pi,\vec{x})$ (in fact, this protocol demonstrates $Perm(\pi,\vec{x})$ can be computed by H with r common bits). For a fixed π, let $C_1,\dots,C_r$ be the cliques covering $\mathbf{G}_\pi$.

- **Player 1** sends the vector $(\bigoplus x[C_1], \dots, \bigoplus x[C_r])$ (for notation see Theorem 4.2).
- **Player 2** sends all inputs x_k adjacent to y_i in H.
- **Player 3** takes the clique C_s containing $\pi(i)$. Since for every $k \in C_s - \{\pi(i)\}$ the edge $(k,\pi(i))$ is in $\mathbf{G}_\pi$, it follows that x_k is adjacent to $y_{\pi^{-1}\pi(i)} = y_i$ in H. Thus the message of Player 2 contains the bit x_k for every $t \in C_s - \{\pi(i)\}$. Hence Player 3 can compute $x_{\pi(i)}$ by subtracting these bits from $\bigoplus x[C_s]$ sent by Player 1. ∎

THEOREM 4.6. *For every $\varepsilon > 0$ and $1/n^{\frac{1}{4}-\varepsilon} < p < 1/5$ there exists n_0 such that for every $n \ge n_0$, $perm^n$ can be computed by a restricted semilinear protocol where Player 1 sends $(1+\varepsilon)\frac{-n\log p}{\log n}$ bits and Player 2 sends $(1+\varepsilon)pn$ bits.*

Proof. Let ε and p satisfying the condition above be given, let n be sufficiently large. In order to apply Lemma 4.5, we only need to estimate the probabilities for the chromatic number and independent Bernoulli variables. The first one is proved by Theorem 4.4,

$$n!\cdot Pr\left[\chi(\mathbf{G}(n,1-p^2)) > (1+\varepsilon)\frac{-n\log p}{\log n}\right] < \exp(n\log n - n^{1+\delta}) = o(1).$$

The second one is a direct consequence of Chernoff-Hoeffding bounds,

$$n\cdot Pr\left[\mathbf{S}(n,p) > (1+\varepsilon)pn\right] \le n\cdot 2\exp(-\Omega(pn)) = o(1). \qquad \square$$

COROLLARY 4.7. *There exist constants $c, \delta > 0$ such that for $c\frac{n}{\log n} < r < \delta n$,*

$$R_{P^n}(r) \leq n^{2-(1-o(1))\frac{r}{n}}.$$

Proof. Follows from Theorem 4.6 and Proposition 3.3 by a computation. □

COROLLARY 4.8.

(i) *There exists a restricted semilinear protocol for $perm^n$ which uses at most $O(n \log\log n/\log n)$ bits of communication. Thus for every k, $ASCC(perm^n_k) = O(n \log\log n/\log n)$.*

(ii) $\mathrm{rank}^{2,1}(R^n) = O(n^2 \log\log n/\log n)$.

Proof. (i) Apply Theorem 4.6 with $p = \log\log n/\log n$. For larger k the communication complexity can only be smaller.

(ii) This follows from (i) using Proposition 3.3. □

4.3. The iterated shift function. In this section we generalize the upper bound from §4.1 to the iterated shift function. This generalization is possible, because the protocol for the function *shift* is given explicitly. We are not able to generalize the nonconstructive protocol of the previous section, thus we are not able to prove a bound on $ASCC(perm^n_k)$ which decreases with increasing k.

THEOREM 4.9. *There exists a semilinear protocol for the k-times iterated shift function $shift^n_k(s_1, \ldots, s_{k+1}, \vec{x})$ such that if k is an arbitrary constant, each player sends at most $O(n(\log\log n/\log n)^k)$ bits, and if $k \geq c\log n$, for some constant c, each player sends at most $O(n^{6/7})$ bits.*

Proof. Remember that in the construction for $k = 1$, Player 1 divides the input x into several groups and communicates the parity of all of them. However, only one of these bits is really used by the last player. In our generalized construction, we compute this one bit recursively, using the first k players. It turns out that if we are careful, this is very similar to computing the function $shift^n_{k-1}$.

The length l of the arithmetic sequences we use will be chosen later to balance the number of bits communicated by individual players. Let $B \subseteq [0, n-1]$ be the set constructed in Lemma 4.1.

Player $k+1$ acts as Player 2 in the construction for $k = 1$, he sends $x[B + s_1 + \cdots + s_k]$, i.e., bits $x_{b+s_1+\cdots+s_k}$ for all $b \in B$. (All indices of x are taken modulo n.)

Players 1 to k all know the input s_{k+1}, hence they can compute the modulus $m = m(s_{k+1})$ as in Lemma 4.1. First they pad the input x by zeros so that its length is the smallest $n' \geq n$ divisible by ml. Then they divide the interval $[0, n'-1]$ into the sequences $C_0, \ldots, C_{n''-1}$ of length l and modulus m as follows. The sequence C_0 starts with 0, C_1 with 1, …, C_{m-1} with $m-1$ (this covers the subinterval $[0, ml-1]$), C_m starts with x_{ml} and so on.

Let $y_j = \bigoplus x[C_j]$ be the parity of the bits of x indexed by the jth sequence. Let $f(j)$ be a number such that $j \in C_{f(j)}$. The goal of Players 1 to k is to compute $y_{f((s_1+\cdots+s_k) \bmod n')}$. If it were true that $f((s_1 + \cdots + s_{k+1}) \bmod n') = (f(s_1) + \cdots + f(s_k) + t) \bmod n''$ for some constant t, we would just use the protocol for $shift^n_{k-1}(f(s_1), \ldots, f(s_k), y')$, where y' is y shifted by t. This is not exactly the case, but we show that from the point of view of any player there are only constantly many possible values of t.

Every player can compute t under the assumption that the input he does not see is 0. This value differs from the correct value by $f(r+s) - f(s) - f(r)$, where s is the input he does not see and r is the sum of all other inputs s_i (taken modulo n').

We now show that the only possible values of this difference are $-m$, 0, or m. Let $a, a', a'' < m$, $b, b', b'' < l$, and c, c', c'' be such that $s = cml+bm+a$, $r = cml+bm+a$, and $r+s = cml+bm+a$. Then $f(s) = cm+a$, $f(r) = c'm+a'$, and $f(r+s) = c''m+a''$. The claim is proved by observing that the value of a'' is either $a+a'$ or $a+a'-m$ and the value of c'' is either $c+c'$ or $c+c'+1$. This works even for $r+s \geq n'$, since we have chosen n' divisible by ml.[1]

If each player communicates information according to the protocol for all three possible values of t, he certainly communicates the information for the correct t. Player $k+2$ knows all $s_1, \ldots, s_{k+1}$, hence he can determine which t is correct and recover the result. Here we use the fact that the protocol is almost simultaneous, hence a player can follow a protocol even if other players do not, as he is not dependent on them.

It is obvious that the iterated protocol is semilinear. It remains to compute the amount of communication. Let $F(n,k)$ denote the maximal number of bits sent by an individual player in our protocol for $shift_k^n$. As a base case, we know by Theorem 4.2 that $F(n,1) = O(n \log\log n/\log n)$. From the previous analysis we know that Players 1 to k send at most $3F(n'', k-1)$ bits. As long as Player $k+1$ does not send more bits, we have $F(n,k) \leq 3F(n'', k-1)$.

If k is constant, we choose $l = \log n/((k+2)\log\log n)$ for each level of recursion. Since $n'' \leq 2n/l$, the recurrence gives $F(n,k) = O(n(\log\log n/\log n)^k)$. By Lemma 4.1, the size of B is $O(n(\log n)^{-k})$, hence the number of bits communicated by the last player is small, and our use of the recurrence is correct.

If k is not constant, we choose $l = 4$. In this case the bound on the modulus is $o(n)$ and hence $n'' = (1+o(1))n/l = n(3/4+o(1))$. By induction we get $F(n,k) \leq n(3/4+o(1))^k$. By Lemma 4.1, the size of B is $O(n^{6/7})$. Therefore we can iterate only as long as $cn^{6/7} \leq n(3/4+o(1))^k$, i.e., up to some $k = \Theta(\log n)$. At that point $F(n,k) = O(n^{6/7})$. ∎

5. Lower bounds.

In this section we prove a trade-off between the size of graphs and common bits which are needed to compute the shift function. This implies lower bounds on the rigidity function and the rigidity rank of the tensor D^n. As corollaries we obtain best known lower bounds for depth 2 circuits computing the shift function and multiplication. The same technique also shows that the protocols based on disjoint parities used in our upper bound cannot be extended so that Player 2 sends less than $n^{1/3}$ bits.

Let H^n be defined as $H^n = \{(x_i, y_j); 0 \leq j < i < n\}$. We first show that this graph cannot compute identity very well.

LEMMA 5.1. *Suppose that the graph H^n computes the identity function with r common bits, either in the boolean case or using only linear functions over an arbitrary field. Then $r \geq n$.*

Proof. In the linear case, the computed function is represented by a matrix which is a sum of a matrix of rank r (the part with the common bits) and an upper triangular matrix with zeros on the diagonal (the edges). Such a matrix can be a diagonal matrix only if $r \geq n$.

In the boolean case we proceed by induction. The basis $n = 1$ is trivial. For the step from n to $n+1$, let $z_1, \ldots, z_r$ be the common bits for H^{n+1}. Since y_n is isolated in

[1] It might be clearer to look at special examples. If $s = ml$ then $f(s) = m$ and it is easy to check that $f(r+s) = f(r)+m = f(r)+f(s)$ for all r. If $s = m$ then $f(s) = 0$ and either $f(r+s) = f(r)$ or $f(r+s) = f(r)+m$ (if $r+s$ is a multiple of ml). If $s = 1$ then $f(s) = 1$ and either $f(r+s) = f(r)+1$ or $f(r+s) = f(r)+1-m$ (if $r+s$ is a multiple of m but not a multiple of ml).

H^{n+1}, $y_n = x_n = g(z_1, \ldots, z_r)$ for some function g. Let $a \in \{0,1\}$ be such that there are at most 2^{r-1} elements $\vec{z} \in \{0,1\}^r$ with $g(\vec{z}) = a$. The set $\{\vec{z} \in \{0,1\}^r;\ g(\vec{z}) = a\}$ can be represented as a subset of $\{0,1\}^{r-1}$, and thus we can compute Id^n with these $r-1$ common bits with the graph H^n. By the inductive assumption $r - 1 \geq n$, hence $r \geq n+1$. ☐

THEOREM 5.2. *For every $\alpha > 0$ there exists $\delta > 0$ such that for every n and $r \geq n^\alpha$, if Shift$^n(s, \vec{x})$ can be computed by a graph G with r common bits for any $n/2$ values of $s \in \{0, \ldots, n-1\}$, either in the boolean case or using only semilinear functions over an arbitrary field, then the size of G satisfies*

$$|G| \geq \delta \frac{n^2}{r} \log \frac{n}{r}. \tag{5}$$

Proof. Suppose that a graph $G \subseteq \{x_0, \ldots, x_{n-1}\} \times \{y_0, \ldots, y_{n-1}\}$ computes $n/2$ shifts with r common bits. Let $d = 4|G|/n$, and let $A = \{i; \mathrm{degree}(x_i) \leq d\}$ and $B = \{i; \mathrm{degree}(y_i) \leq d\}$ be the sets of vertices with small degrees in G. Clearly $|A|, |B| \geq \frac{3}{4}n$, as d is four times the average degree of G.

Now consider a particular shift s computed by G. Let $A' = A \cap (B + s)$, where $B + s = \{(i + s) \bmod n; i \in B\}$. Clearly $|A'| \geq n/2$. Let G_s be a graph with vertices A' such that (a, b) is an edge if $(x_{a+s}, y_b) \in G$ and $a \leq b$. Let S be the set of all shifts s such that G_s has at most $n/4$ loops (edges of type (a, a)). If $|S| < n/4$, there are at least $n^2/16$ loops in all graphs G_s, and (5) is satisfied, since each edge of G corresponds to at most one loop. Hence we assume that $|S| \geq n/4$ and restrict ourselves to shifts $s \in S$.

Suppose that for some $s \in S$ the graph G_s has less than $n/24$ triangles. Then there is a set $A'' \subseteq A'$, $|A''| \geq n/8$ which induces a triangle-free graph with no loops. Ajtai, Komlós, Szemerédi [1] proved that a triangle-free graph of degree at most d contains an independent set of size at least

$$t \geq \frac{|A''| \log d}{100d} = \frac{n \log d}{800d}.$$

If there is an independent set K in G_s, it follows that $a > b$ whenever $(x_{a+s}, y_b) \in G$ for $a, b \in K$. Hence the graph G restricted to the nodes $\{x_{a+s}; a \in K\}$ and $\{y_b; b \in K\}$ is essentially a subgraph of $H^{|K|}$ and it computes the identity function, assuming G computes the shift s correctly. By Lemma 5.1 the number of common bits r is at least $|K| \geq t$. It follows that

$$d = \Omega\left(\frac{n}{r} \log \frac{n}{r}\right)$$

and (5) is satisfied.

The remaining case is that each of the $n/4$ graphs G_s, $s \in S$, contains at least $n/24$ triangles. Let $c < a < b$ be vertices of a triangle in G_s. From the definition of G_s it follows that $(x_{a+s}, y_b, x_{c+s}, y_a)$ is a path in G. Such a path determines s (as the difference of the indices of the first and the last nodes), hence there are at least $|S|n/24 \geq n^2/96$ such paths. But since the degree of nodes in G indexed by A and B is bounded by d, the number of paths is at most nd^3. This gives $nd^3 \geq n^2/96$, hence $d = \Omega(n^{1/3})$, $|G| = \Omega(n^{4/3})$ and (5) is satisfied if $r \geq n^{2/3} \log n$.

Now suppose $n^\alpha < r < n^{2/3} \log n$. We reduce this case to the previous part of the proof. Take an ε such that

$$n^{\frac{2}{3}\varepsilon} \log n^\varepsilon < r < n^{\frac{5}{6}\varepsilon}.$$

Thus $\frac{6}{5}\alpha < \varepsilon < 1$. Divide each of the sets $\{x_0, \ldots, x_{n-1}\}$ and $\{y_0, \ldots, y_{n-1}\}$ into $n^{1-\varepsilon}$ disjoint intervals of size n^ε. Let $U \subseteq \{x_0, \ldots, x_{n-1}\}$ and $V \subseteq \{y_0, \ldots, y_{n-1}\}$ be two such intervals. Then $G \cap (U \times V)$ realizes $n^\varepsilon/2$ noncyclic shifts with r common bits. A simple modification of this graph ("wrapping around"), gives a graph with the same number of edges which computes $n^\varepsilon/2$ cyclic shifts with r common bits. Thus for each of the $n^{2-2\varepsilon}$ disjoint sections of G we have a lower bound $\Omega\left(\frac{n^{2\varepsilon}}{r}\log\frac{n^\varepsilon}{r}\right)$ and thus

$$|G| = n^{2-2\varepsilon}\Omega\left(\frac{n^{2\varepsilon}}{r}\log\frac{n^\varepsilon}{r}\right) = \Omega\left(\varepsilon\frac{n^2}{r}\log\frac{n^\varepsilon}{r}\right) = \Omega\left(\frac{n^2}{r}\log\frac{n}{r}\right),$$

because ε is bounded by the constant α. ◻

COROLLARY 5.3.

(i) *Let $\alpha > 0$ be fixed. Then $R_{D^n}(r) = \Omega\left(\frac{n^2}{r}\log\frac{n}{r}\right)$, for $r \geq n^\alpha$.*

(ii) *Let $\alpha > 0$ be fixed. In any restricted protocol that computes $shift^n$ in which Player 1 sends at most r bits, $r \geq n^\alpha$, Player 2 sends at least $\Omega\left(\frac{n}{r}\log\frac{n}{r}\right)$ bits. Thus the total communication is at least $\Omega(\sqrt{n\log n})$ bits.*

(iii) $\mathrm{rank}^{2,1}(D^n) = \Omega\left(n^{3/2}(\log n)^{1/2}\right)$.

Proof. (i) and (ii) follows from Theorem 5.2 using Propositions 3.2 and 3.1.

(iii) Let $r = n^{1/2}(\log n)^{1/2}$. Let a decomposition of D^n into tensors of the form $u \otimes e_j^n \otimes w$ and $e_i^n \otimes v \otimes e_k^n$ be given. Let S be the sum of the tensors of the first type. If $\mathrm{rank}(S_{*,j,*}) \geq r$ for at least $\frac{n}{2}$ of the slices, we are done. So suppose the converse. Using Theorem 5.2 for the set of shifts j such that $\mathrm{rank}(S_{*,j,*}) < r$ and the transformation from Proposition 3.2, it follows that the number of tensors of type $e_i^n \otimes v \otimes e_k^n$ is at least

$$\Omega\left(\frac{n^2}{r}\log\frac{n}{r}\right) = \Omega\left(n^{3/2}(\log n)^{1/2}\right). \qquad ◻$$

COROLLARY 5.4.

(i) *For any field $\mathbf{F}$, any depth two semilinear circuit for $Shift^n$ has size at least $\Omega\left(n(\log n)^{3/2}\right)$.*

(ii) *Any depth two circuit with arbitrary boolean functions as gates which computes $Shift^n$ has size at least $\Omega\left(n(\log n)^{3/2}\right)$.*

Proof. In both cases let a circuit for $Shift^n$ be given. Fix $\alpha > 0$. Let $d_1 \geq d_2 \geq \ldots$ be the degrees of the vertices on the middle level. For an arbitrary $r \geq n^\alpha$ we construct a graph with r common bits that computes $Shift^n$ as follows. We take the common bits to be the functions computed at r vertices with maximal degree. The graph is constructed by connecting an input to an output if they are connected by a path going through one of remaining vertices. This graph has at most $\sum_{j>r} d_j^2$ edges. By Theorem 5.2 it follows that

$$\sum_{j>r} d_j^2 \geq \Omega\left(\frac{n^2}{r}\log\frac{n}{r}\right).$$

This bound (for $n^\alpha \leq r \leq n$) implies

$$\sum_{j\geq 1} d_j = \Omega\left(n(\log n)^{3/2}\right),$$

see [24, Lemma 4] for a proof of this implication. ◻

COROLLARY 5.5.

(i) *Every depth two algebraic circuit for multiplying two polynomials has size* $\Omega\left(n(\log n)^{3/2}\right)$.

(ii) *Every depth two boolean circuit for multiplying two n bit numbers has size* $\Omega\left(n(\log n)^{3/2}\right)$.

Proof. Reduce the function $Shift^n$ to these functions. ◻

These are the best lower bounds for these functions for depth 2. Also the bound $\Omega\left(n(\log n)^{3/2}\right)$ is the asymptotically largest lower bound for any explicitly given boolean function. Previously such a bound for general depth 2 circuits was known only for functions which contain a superconcentrator [2]. For linear circuits such a bound can also be proved using the bound on rigidity of the parity check matrix of a good code, see [14, 24].

The technique of counting triangles from Theorem 5.2 can be used to prove a stronger result for a more restricted model of computation by graphs.

THEOREM 5.6. *Suppose that $Shift^n(s, \vec{x})$ can be computed by a graph G with $n/3$ common bits using only semilinear functions, with an additional condition that for every s, the common bits compute parities of* pairwise disjoint *subsets of bits of $\vec{x}$. Then the degree of G is at least $\Omega(n^{1/3})$.*

Proof. Suppose that the degree of G is $d = o(n^{1/3})$. Similarly as in Theorem 5.2, we consider the graph G_s for each shift. By counting the edges, for most values of s the graph G_s has at most $o(n^{1/3})$ loops. This means that for such a value of s the only way how to compute the value of other outputs is to use one of the common bits and the other input bits used by that common bit. As each input is used only by a single common bit, the corresponding vertices in G_s have to induce a complete graph (more precisely, a tournament). Because there are only $n/3$ common bits which have to use all of $n - o(n^{1/3})$ input bits not in loops, these complete graphs contain $\Omega(n)$ triangles for each s, a total of $\Omega(n^2)$ triangles. By the same argument as in Theorem 5.2, the total number of triangles is at most $nd^3 = o(n^2)$, a contradiction. ◻

Similarly as in Theorem 5.2, we can extend this result to all graphs computing some constant fraction of shifts and prove that the number of edges has to be at least $n^{4/3}$, which is slightly stronger.

If we could prove a similar result without the additional restriction on the computation, it would follow that there are no linear circuits of logarithmic depth for the function $Shift^n$. Or, conversely, this result says that the protocols from §4, which are all based on disjoint parities, are not powerful enough to break the bound of Theorem 3.5.

6. Conservative and nonconservative computation. Now we show that a conservative model of computations of shifts, which is based on sending information along vertex disjoint paths, is less efficient than a boolean circuit (with parity gates), or a linear circuit over any field. We show this for circuits of depth 2 when the complexity is measured as the number of edges, with a modification that we do not count edges incident to small set of vertices (of size $o(n)$). In fact, we show a stronger result, namely that in this way we can compute even all permutations more efficiently than one can compute only shifts using shifters. Ideally, we would like to prove the result when counting all edges, however we are not able to do so at present time.

THEOREM 6.1. *There exists a semilinear circuit of depth two for $Perm^n$ and a set X of vertices on the middle level such that $|X| = o(n)$ and there are only $o(n^{3/2})$ edges disjoint with X.*

Before we prove Theorem 6.1, we prove the complementary lower bound for shifters, which is quite simple.

THEOREM 6.2. *If G is an n-shifter of depth two, then G has $\Omega(n^{3/2})$ edges, even if we remove any $o(n)$ vertices and the edges incident with them.*

Proof. Suppose G is an n-shifter of depth two. Let A be an arbitrary set of vertices of size $o(n)$. Suppose that there are only $o(n^{3/2})$ edges of G which are not incident with A. Let B be the set of vertices incident with $n^{1/2}$ such edges. By the assumption $|B| = o(n)$. Since G is an n-shifter, there are $n - |A \cup B|$ paths from inputs to outputs disjoint with $A \cup B$ for each shift, total of $n^2(1 - o(1))$ paths disjoint with $A \cup B$. But each edge which is not incident with $A \cup B$ can belong only to $n^{1/2}$ paths, hence there are $n^{3/2}(1 - o(1))$ edges not incident with $A \cup B$, a contradiction. □

Bounds on the size of bounded depth shifters have been proved in [23], including a lower bound $\Omega(n^{3/2})$ for depth two. Let us also mention a related unpublished result of W. Maass which gives a bound $\Omega(n^{3/2})$ on the size of the circuits of depth two that compute $Shift^n$ with the restriction that there is a constant size set of boolean functions assigned to each vertex and for every shift we can only use as a gate assigned to this vertex either one of these functions or an arbitrary projection.

The proof of Theorem 6.1 is again probabilistic. A constructive proof for $Shift^n$ seems to be also possible. The basic idea is the same as in the upper bounds of §4. Instead of sending the bits directly, we partition the inputs into $o(n)$ blocks, send the parities of the blocks through the extra $o(n)$ vertices, and then compute the individual bits from the parities of the blocks using direct connections realized by vertex disjoint paths. The difficult part is to realize these direct connections with only $o(n^{3/2})$ edges. We prove that a random graph with suitable parameters satisfies this condition. We show that for each permutation it is possible to choose one block of inputs with the necessary vertex disjoint paths with a very large probability. This enables us to choose the blocks one by one until only $o(n)$ inputs remain. The values of these remaining inputs are also sent using the extra $o(n)$ vertices.

We need to make some preliminary considerations before we start the proof. For the rest of this section, a graph means a graph of a circuit of depth two, formally a quadruple (V_1, V_2, V_3, E) such that V_1, V_2 and V_3 are disjoint sets of vertices and the edges are $E \subseteq (V_1 \times V_2) \cup (V_2 \times V_3)$. An *embedding* of (V_1, V_2, V_3, E) into (V_1', V_2', V_3', E') is a one-to-one mapping g that maps vertices to vertices in the corresponding set and edges to edges. For technical reasons we assume that the sets V_1 and V_1' are ordered and any embedding has to preserve this ordering as well, i.e., if $V_1 = \{x_1, \dots, x_n\}$, $V_1' = \{x_1', \dots, x_{n'}'\}$, and $g(x_1) = x_{i_1}', \dots, g(x_n) = x_{i_n}'$, then $i_1 < i_2 < \dots < i_n$.

Let $G_k = (U_1, U_2, U_3, E)$ be the graph with k inputs and k outputs which realizes by vertex disjoint paths all connection between all pairs of inputs and outputs, except for the corresponding pairs, i.e.,

$$\begin{array}{rcl} U_1 & = & \{u_1, \dots, u_k\}, \\ U_2 & = & \{w_{i,j};\ 1 \le i, j \le k, i \ne j\}, \\ U_3 & = & \{v_1, \dots, v_k\} \\ E & = & \{(u_i, w_{i,j}), (w_{i,j}, v_j);\ \text{for } w_{i,j} \in U_2\}. \end{array}$$

The proof of the theorem is based on the following lemma, which we prove later.

LEMMA 6.3. *Let $k > 2$ be an arbitrary constant. Then for sufficiently large n there exists a graph $G = (V_1, V_2, V_3, E)$, $|V_1| = |V_3| = n$, $|V_2| = nk^2$, with $n^{3/2}/k$ edges such that for every m, $n/k \le m \le n$, every $W_1 \subseteq V_1$, $W_2 \subseteq V_2$, $W_3 \subseteq V_3$, $|W_1| = |W_3| = m$, $|W_2| = mk^2$, and every bijection $f : W_3 \to W_1$, there exists an embedding g of G_k into subgraph of G induced by $W_1 \cup W_2 \cup W_3$ such that if $g(v_j) = x$ then $g(u_j) = f(x)$.*

Proof of Theorem 6.1. For given n, pick the largest k such that the condition of Lemma 6.3 holds. Because k was an arbitrary constant, we have $k = \omega(1)$. Take the graph G from Lemma 6.3. Add a set X of $2n/k = o(n)$ extra vertices connected to all inputs $V_1 = \{x_1, \dots, x_n\}$ and all outputs $V_3 = \{y_1, \dots, y_n\}$. By Lemma 6.3 the graph has at most $n^{3/2}/k = o(n^{3/2})$ edges. Hence it only remains to prove that this circuit computes $Perm^n(\pi, \vec{x})$.

Fix a permutation π. By repeated applications of Lemma 6.3 we can choose disjoint embeddings of G_k into the graph G such that at most n/k inputs of G are left uncovered, and whenever an embedding maps an output to y_i, the corresponding input is mapped to $x_{\pi(i)}$.

For each of the chosen embeddings of G_k take one of the extra vertices in X and connect inputs of G_k with outputs of G_k through it. Thus for an input x_j of G_k and an output y_i of G_k we have just one path $x \to y$ through X if $j = \pi(i)$, and one path through X and one path in G_k if $j \neq \pi(i)$. Connect the inputs and the outputs not covered by the embeddings of G_k paths through the remaining vertices in X. This is possible, since we need at most n/k vertices in X for G_k's and at most n/k vertices in X for inputs and outputs not covered by G_k's.

Assign values to the edges as follows. All edges of the selected paths through X have value 1. For each embedding of G_k, the edges going from V_1 to V_2 have value 1 and the edges going from V_2 to V_3 have value -1. The remaining edges have value 0. We set the linear function at each vertex to be the sum of the values on its predecessors, each multiplied by the value of the connecting edge. Thus for $j = \pi(i)$ there is only one path from x_j to y_i with nonzero value and its value is 1. For $j \neq \pi(j)$ there are either no nonzero paths, or one with value 1 and one with value -1. Hence the circuit computes $Perm^n(\pi, \vec{x})$. □

For the rest of this section we set

$$p = n^{\frac{5/4-k^2}{2(k^2-k)}} = n^{-\frac{1}{2}-\frac{4k-5}{8(k^2-k)}}.$$

Let $\mathbf{R}_{p,m}$ denote the random graph with vertices

$$\begin{aligned} V_1 &= \{x_1, \dots, x_m\}, \\ V_2 &= \{z_1, \dots, z_{mk^2}\}, \\ V_3 &= \{y_1, \dots, y_m\}, \end{aligned}$$

and each edge from $V_1 \times V_2$ and $V_2 \times V_3$ chosen independently at random with probability p.

Proof of Lemma 6.3. We prove that for sufficiently large n, the graph $\mathbf{R}_{p,n}$ satisfies the conditions of Lemma 6.3 with high probability. The expected number of edges of $\mathbf{R}_{p,n}$ is

$$2n^2k^2p = 2k^2n^{\frac{3}{2}-\frac{4k-5}{8(k^2-k)}} = o(n^{3/2}/k),$$

hence the condition on the number of edges is satisfied with high probability.

The number of possible quadruples W_1, W_2, W_3, f is bounded by $e^{O(n \log n)}$. For a fixed quadruple W_1, W_2, W_3, f the probability that an appropriate embedding of G_k exists is the same as the probability that there exists an embedding g of G_k into $\mathbf{R}_{p,m}$ satisfying

$$g(u_i) = x_j \text{ iff } g(v_i) = y_j, \tag{6}$$

where $m = |W_1|$. Hence the proof is completed by the following lemma. $\square$

LEMMA 6.4. *Let $k > 2$ be an arbitrary constant. Then for n sufficiently large and any m, $n/k \le m \le n$,*

$$Pr[\text{there exists no embedding of } G_k \text{ into } \mathbf{R}_{p,m} \text{ satisfying } (6)] \le e^{-n^{5/4+o(1)}}.$$

Proof. Let $X(H,G)$ denote the number of embeddings of H into G satisfying (6). We want to prove an upper bound on $Pr[X(G_k, \mathbf{R}_{p,m}) = 0]$. We use Janson's inequality, see [3], Theorem 1.1, Chapter 8, page 96.

Let $G_k(1), \dots, G_k(i), \dots$ denote all possible occurrences of G_k in $\mathbf{R}_{p,m}$ satisfying the condition (6). Let

$$\begin{aligned} \varepsilon &= Pr[G_k(i) \subseteq \mathbf{R}_{p,m}] \\ \mu &= E\big(X(G_k, \mathbf{R}_{p,m})\big) \\ \Delta &= \sum_{i \sim j} Pr\left[G_k(i) \subseteq \mathbf{R}_{p,m} \wedge G_k(j) \subseteq \mathbf{R}_{p,m}\right], \end{aligned}$$

where we sum over the pairs of distinct occurrences of G_k which have at least one common edge. By Janson's inequality we have

$$Pr\left[X(G_k, \mathbf{R}_{p,m}) = 0\right] \le e^{-\mu + \frac{1}{1-\varepsilon}\frac{\Delta}{2}}.$$

Clearly $\varepsilon = p^{2(k^2-k)} = o(1)$. We prove that $\mu = \Theta(n^{5/4})$ and $\Delta = o(\mu)$, which gives the desired bound.

First we compute μ as the number of all possible occurrences times the probability of one fixed occurrence (remember that k is a constant).

$$\mu = \binom{m}{k}\binom{mk^2}{k^2-k} p^{2(k^2-k)} = \Theta(n^k n^{k^2-k} n^{\frac{5/4-k^2}{2(k^2-k)} \cdot 2(k^2-k)}) = \Theta(n^{5/4}).$$

We can bound Δ as in [18] by

$$\Delta \le \sum_{\emptyset \ne H \subset G_k} \frac{\mu^2}{E(X(H, \mathbf{R}_{p,m}))} X(H, G_k)^2,$$

where the sum is over all graphs H with at least one edge which may occur as intersections $G_k(i) \cap G_k(j)$ for $i \ne j$. Since k is a constant, the number of such graphs is bounded and the values of $X(H, G_k)$ are bounded as well, and therefore

$$\Delta \le O\left(\frac{\mu^2}{\min E(X(H, \mathbf{R}_{p,m}))}\right).$$

For a given graph H, let r, $1 \le r \le k$, be the number of its vertices from V_1. The vertices on the middle level of H have degree at most 2. Let s_0, s_1, and s_2 denote the number of vertices of H on the middle level whose degree in H is 0, 1, and 2, respectively. Now we estimate the growth rate of $E(X(H, \mathbf{R}_{p,m}))$ and prove that it is at least $\omega(n^{5/4}) = \omega(\mu)$ for every H.

$$\begin{aligned} E(X(H, \mathbf{R}_{p,m})) &= \binom{m}{r}\binom{mk^2}{s_2}\binom{mk^2 - s_2}{s_1}\binom{mk^2 - s_2 - s_1}{s_0} p^{2s_2+s_1} \\ &\approx n^{r+s_2+s_1+s_0} p^{2s_2+s_1} \ge n^r (np^2)^{s_2} (np)^{s_1} \\ &= n^{r - s_2\left(\frac{4k-5}{4(k^2-k)}\right) + s_1\left(\frac{1}{2} - \frac{4k-5}{8(k^2-k)}\right)}. \end{aligned}$$

We estimate the exponent separately for the following three cases.

(i) Let $r = 1$. Then $s_2 = 0$, since in G_k there is no path from u_i to v_i for any i. Since H is has at least one edge, $s_1 \geq 1$ and the exponent is at least $1 + 1/2 - (4k - 5)/(8(k^2 - k)) > 5/4$.

(ii) Suppose $2 \leq r \leq k - 1$. Since there are at most $r^2 - r$ vertices of degree 2 on the middle level of H, the exponent is minimized for $s_1 = 0$ and $s_2 = r^2 - r$, and then it is

$$\alpha = r - (r^2 - r)\frac{4k - 5}{4(k^2 - k)}.$$

This is a concave down function in r, thus it is minimized at one of the endpoints of the interval, i.e., for $r = 2$ or $r = k-1$. For $r = 2$, $\alpha = 2 + (4k-5)/(2(k^2-k)) > 5/4$. For $r = k - 1$ a short calculation gives $\alpha = 9/4 - 5/(2k) > 5/4$.

(iii) Finally, let $r = k$. Then $s_2 < k^2 - k$, since H is required to be a proper subgraph of G_k. Thus the exponent is minimized at $s_2 = k^2 - k - 1$ and $s_1 = 0$, when the value is $5/4 + (4k - 5)/(4(k^2 - k)) > 5/4$.

Hence $\Delta \leq \mu^2/\omega(\mu) = o(\mu)$ and we can apply Janson's inequality. $\square$

7. The chromatic number of a random graph. In this section we prove the estimate on the chromatic number of a random graph for a large probability of an edge, Theorem 4.4. In our presentation of the proof we follow the lines of [3, Chapter 7]. The proof is based on the following lemma, where clique(G) denotes the size of a maximal clique in a graph G.

LEMMA 7.1. *For every $\varepsilon > 0$ there exists $\delta > 0$ and n_0 such that*

$$Pr\left[clique(\mathbf{G}(n,p)) > (2-\varepsilon)\frac{\log n}{-\log p}\right] > 1 - \exp(-n^{1+\delta})$$

as long as $1/n^{\frac{1}{2}-\varepsilon} < p < 1/8$ and $n \geq n_0$.

First, we deduce Theorem 4.4 from Lemma 7.1. Set $m = n/(\log n)^2$, $\varepsilon' = \varepsilon/2$ and $p = 1 - q$. Let A be the event that every induced subgraph H of $\mathbf{G}(n,q)$ with m vertices has an independent set of size $\alpha(H) > (2-\varepsilon)\frac{\log m}{-\log(1-q)}$. Note that $(2-\varepsilon')\frac{\log m}{-\log(1-q)} \geq (2-\varepsilon)\frac{\log n}{-\log(1-q)}$ for n sufficiently large. Then by Lemma 7.1, applied with $\varepsilon' = \varepsilon/2$,

$$Pr(A) > 1 - \binom{n}{m}\exp(-n^{1+\delta}) > 1 - \exp\left(-n^{\delta}\right).$$

This means that with probability at least $1 - \exp\left(-n^{\delta}\right)$ the random graph $\mathbf{G}(n,q)$ can be colored by

$$\frac{n-m}{(2-\varepsilon)\frac{\log m}{-\log(1-q)}} + m \leq \left(\frac{1}{2} + \varepsilon\right)\frac{-n\log(1-q)}{\log n}$$

colors. This finishes the proof of Theorem 4.4.

Before we give a proof of Lemma 7.1 we prove another lemma. Let $p = p(n)$ be given, $1/n^{\frac{1}{2}-\varepsilon} < p < 1/8$, $\varepsilon > 0$ constant. Let $k = k(n)$ be the largest integer such that

$$\binom{n}{k}p^{\binom{k}{2}} \geq n^3; \tag{7}$$

note that $(1-\varepsilon)\frac{\log n}{-\log p} \leq k \leq 2\frac{\log n}{-\log p}+1$ for every sufficiently large n. Let $Y = Y(\mathbf{G}(n,p))$ be the maximal size of a family of edge disjoint k-cliques in $\mathbf{G}(n,p)$.

Lemma 7.2. $E(Y) \geq n^2p/(2k^5)$.

Proof. Let K denote the family of all k-cliques so that $E(|K|) = \binom{n}{k}p^{\binom{k}{2}} \geq n^3$. Let W be the set of ordered pairs $\{S,T\}$ of k-cliques of $\mathbf{G}(n,p)$ with $2 \leq |S \cap T| < k$. Then

$$E(|W|) = \binom{n}{k}\sum_{i=2}^{k-1}\binom{k}{i}\binom{n-i}{k-i}p^{2\binom{k}{2}-\binom{i}{2}}$$

Set

$$A_i = \binom{k}{i}\binom{n-i}{k-i}p^{2\binom{k}{2}-\binom{i}{2}}.$$

As for $p < \frac{1}{8}$ the sequence

$$B_i = \frac{A_i}{A_{i+1}} = \frac{(n-i)(i+1)}{(k-i)^2}p^i$$

decreases, we infer that for some i_0, $A_i \geq A_{i+1}$ for $i < i_0$ and $A_i \leq A_{i+1}$ for $i \geq i_0$. A straightforward calculation using (7) shows that $A_{k-1} \leq A_2$. Hence $A_i \leq A_2$ for all $i = 2,\ldots,k-1$ and thus

$$E(|W|) \leq \binom{n}{k}k\binom{k}{2}\binom{n-2}{k-2}p^{2\binom{k}{2}-1}.$$

Let K' be a random subfamily of K, where every $S \in K$ is chosen independently with

$$Pr[S \in K'] = \gamma = \frac{1}{4p^{\binom{k}{2}-1}k\binom{k}{2}\binom{n-2}{k-2}}.$$

Now construct a family $L \subseteq K'$ by removing each pair $S,T \in K'$ such that $\{S,T\} \in W$. No two cliques in L intersect, hence $|L| \leq Y$. Thus

$$\begin{aligned} E(Y) &\geq E(|L|) \geq \gamma|K| - 2\gamma^2E(|W|) \\ &= \gamma\binom{n}{k}p^{\binom{k}{2}} - 2\gamma^2\binom{n}{k}k\binom{k}{2}\binom{n-2}{k-2}p^{2\binom{k}{2}-1} = \frac{1}{2}\gamma\binom{n}{k}p^{\binom{k}{2}} \geq \frac{n^2p}{2k^5}. \end{aligned}$$ ◻

Furthermore we need the following definition. A *martingale* is a sequence $Y_0, Y_1, \ldots, Y_m$ of random variables, such that for $0 \leq i < m$,

$$E(Y_{i+1}|Y_i) = Y_i$$

holds. We use the following estimate.

Theorem 7.3 (Azuma's Inequality, [5, 3]). *Let $0 = X_0, X_1, \ldots X_m$ be a martingale with $|X_{i+1} - X_i| \leq 1$ for all $0 \leq i < m$. Let $\lambda > 0$ be arbitrary. Then*

$$Pr[X_m > \lambda] < e^{-\frac{\lambda^2}{2m}}.$$

Proof of Lemma 7.1. Let $Y_0, Y_1, \ldots, Y_m$, $m = \binom{n}{2}$ be the edge exposure martingale on $\mathbf{G}(n,p)$ with function Y defined above (see [3] for the definitions). Y_i is the conditional expectation of Y when we know the first i edges (for a fixed arbitrary ordering of the edges); thus $y_0 = E(Y)$ and $Y_m = Y$. As Y is the cardinality of a family of edge disjoint cliques, $|Y_{i+1} - Y_i| \leq 1$ holds, and hence by Azuma's inequality, taking $X_i = Y_i - E(Y)$,

$$\begin{aligned} Pr[\text{clique}(\mathbf{G}) < k] &= Pr[Y = 0] \leq Pr[Y - E(Y) \leq -E(Y)] \leq \\ &\leq \exp\left(\frac{-E(Y)^2}{2\binom{n}{2}}\right) = \exp\left(-\frac{n^2p^2}{4k^{10}}(1+o(1))\right) \\ &\leq \exp(-n^{1+\delta}). \quad \square \end{aligned}$$

8. Conclusions and open problems. We have shown relations between circuit complexity, multiparty complexity and the algebraic characteristics of rigidity and rigidity rank. The particular versions of these concepts that we have considered are related to the problem of proving superlinear lower bound on circuit complexity and on the size of bounded depth circuits with arbitrary boolean gates.

We have proved some upper bounds. Though it is only a small improvement, it disproves some earlier conjectures. The conclusion is that the problems are apparently harder than it was expected.

We have also improved some lower bounds. This is an example of how nontrivial results in combinatorics may help in the complexity theory. Still the gaps between upper and lower bounds remain very large.

There many open problems in this area, several of them are implicit in the above text. Here we state only two which we consider to be the most challenging.

Problem 1. Does the function Shift^n have circuits of size $O(n)$ and depth $O(\log n)$? This is open for semilinear circuits as well.

Problem 2. Improve the easy lower bound $\Omega(\sqrt{n})$ for the simultaneous communication complexity of shift^n.

We believe that $SCC(\mathit{shift}^n)$ is large, however even our slightly larger lower bound $\Omega(\sqrt{n \log n})$ from Corollary 5.3 works only under the restriction that Player 2 sends always a substring of the input string $\vec{x}$.

Acknowledgments. The first author would like to thank to Wolfgang Maass from whom he learned problems on circuits with arbitrary boolean functions as gates and to Avi Wigderson for an idea which eventually led to the constructive upper bound in Theorem 4.2. We also thank Steven Rudich for valuable comments.

REFERENCES

[1] M. Ajtai, J. Komlós, and E. Szemerédi, *A note on ramsey numbers*, J. Comb. Theor. (A), 29 (1980), pp. 354–360.

[2] N. Alon and P. Pudlák, *Superconcentrators of depth 2 and 3; odd levels help (rarely)*, J. Comput. Syst. Sci., 48 (1994), pp. 194–202.

[3] N. Alon and J. H. Spencer, *The Probabilistic Method*, Wiley, 1992.

[4] A. Ambainis, *Upper bounds on multiparty communication complexity of shifts*, in Proc. of the 13th Ann. Symp. on Theoretical Aspects of Computer Sci., Lecture Notes in Computer Sci. 1046, Springer-Verlag, 1996, pp. 631–642.

[5] K. Azuma, *Weighted sums of certain dependent random variables*, Tôhoku Math. J., 3 (1967), pp. 357–367.

[6] L. Babai, P. Kimmel, and S. V. Lokam, *Simultaneous messages vs. communication*, in Proc. of the 12th Ann. Symp. on Theoretical Aspects of Computer Sci., Lecture Notes in Computer Sci. 900, Springer-Verlag, 1995, pp. 361–372.

[7] L. Babai, N. Nisan, and M. Szegedy, *Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs*, J. Comput. Syst. Sci., 45 (1992), pp. 204–232.
[8] R. Beigel and J. Tarui, *On ACC*, Comput. Complexity, 4 (1994), pp. 350–366.
[9] B. Bollobás, *The chromatic number of random graphs*, Combinatorica, 8 (1988), pp. 49–55.
[10] ———, *Random Graphs*, Academic Press, 1985.
[11] C. Damm, S. Jukna, and J. Sgall, *Some bounds for multiparty communication complexity of pointer jumping*, in Proc. of the 13th Ann. Symp. on Theoretical Aspects of Computer Sci., Lecture Notes in Computer Sci. 1046, Springer-Verlag, 1996.
[12] H. F. de Groote, *Lectures on the Complexity of Bilinear Problems*, Lecture Notes in Computer Sci. 245, Springer-Verlag, 1987.
[13] D. Dolev, C. Dwork, N. Pippenger, and A. Wigderson, *Superconcentrators, generalizers and generalized connectors with limited depth*, in Proc. of the 15th Ann. ACM Symp. on Theory of Computing, ACM, 1983.
[14] J. Friedman, *A note on matrix rigidity*, Combinatorica, 13 (1993), pp. 235–239.
[15] J. Håstad and M. Goldmann, *On the power of small-depth threshold circuits*, Comput. Complexity, 1 (1991), pp. 113–129.
[16] A. K. Chandra, M. L. Furst, and R. J. Lipton, *Multi-party protocols*, in Proc. of the 15th Ann. ACM Symp. on Theory of Computing, ACM, 1983, pp. 94–99.
[17] D. V. Chudnovsky and G. V. Chudnovsky, *Algebraic complexities and algebraic curves over finite fields*, Proc. Nat. Acad. Sci. USA, 84 (1987), pp. 1739–1743.
[18] S. Janson, T. Luczak, and A. Ruciński, *An exponential bound for the probability of nonexistence of a specified subgraph in a random graph*, in Random Graphs '87, M. K. et al., ed., Wilew, 1990, pp. 73–87.
[19] E. Kushilevitz and N. Nisan, *Communication Complexity*, Cambridge University Press, to appear.
[20] N. Nisan and A. Wigderson, *Rounds in communication complexity revisited*, SIAM J. Comput., 22 (1993), pp. 211–219.
[21] N. Pippenger, *The complexity of computations by networks*, IBM Journal of Research and Development, 31 (1987), pp. 235–243.
[22] N. Pippenger and L. G. Valiant, *Shifting graphs and their applications*, J. ACM, 23 (1976), pp. 423–432.
[23] N. Pippenger and A. C.-C. Yao, *Rearrangeable networks with limited depth*, SIAM J. Alg. Discr. Methods, 3 (1982), pp. 411–417.
[24] P. Pudlák, *Communication in bounded depth circuits*, Combinatorica, 14 (1994), pp. 203–216.
[25] P. Pudlák and V. Rödl, *Modified ranks of tensors and the size of circuits*, in Proc. of the 25th Ann. ACM Symp. on Theory of Computing, ACM, 1993, pp. 523–531.
[26] P. Pudlák and P. Savický, *On shifting networks*, Theoretical Comput. Sci., 116 (1993), pp. 415–419.
[27] A. A. Razborov, *On rigid matrices.* preprint of Math. Inst. Acad. of Sciences of USSR (in Russian), 1989.
[28] J. Sgall, *A note on multiparty communication complexity of shifts.* Manuscript, 1992.
[29] M. A. Shokrolahi, *Beiträge zur codierungs- und komplexitätstheorie mittels algebraischer funktionkörper*, Bayreuther Mathematische Schriften, 39 (1991).
[30] V. Shoup and R. Smolensky, *Lower bounds for polynomial evaluation and interpolation*, in Proc. of the 32nd Ann. IEEE Symp. on Foundations of Computer Sci., IEEE, 1991, pp. 378–383.
[31] V. Strassen, *Vermeidung von divisionen*, Crelles J. Reine Angew. Math., 264 (1973), pp. 184–202.
[32] L. G. Valiant, *Graph-theoretic arguments in low level complexity*, in Proc. of the 6th Mathematical Foundations of Computer Sci., Lecture Notes in Computer Sci. 53, Springer-Verlag, 1977, pp. 162–176.
[33] ———, *Why is boolean complexity theory difficult?*, in Boolean Function Complexity, M. S. Paterson, ed., Cambridge Univ. Press, 1992, pp. 84–94.
[34] A. C.-C. Yao, *On ACC and threshold circuits*, in Proc. of the 31st Ann. IEEE Symp. on Foundations of Computer Sci., IEEE, 1990, pp. 619–627.