The Power of Super-logarithmic Number of Players*

Arkadev Chattopadhyay[1] and Michael E. Saks[2]

1 **School of Technology and Computer Science, Tata Institute of Fundamental Research, Mumbai, India**
`arkadev.c@tifr.res.in`
2 **Department of Mathematics, Rutgers University, Piscataway, NJ, USA**
`msaks30@gmail.com`

Abstract

In the 'Number-on-Forehead' (NOF) model of multiparty communication, the input is a $k \times m$ boolean matrix A (where k is the number of players) and Player i sees all bits except those in the i-th row, and the players communicate by broadcast in order to evaluate a specified function f at A. We discover new computational power when k exceeds $\log m$. We give a protocol with communication cost poly-logarithmic in m, for block composed functions with limited block width. These are functions of the form $f \circ g$ where f is a symmetric b-variate function, and g is a kr-variate function and $(f \circ g)(A)$ is defined, for a $k \times br$ matrix to be $f(g(A^1), \ldots, g(A^b))$ where A^i is the i-th $k \times r$ block of A. Our protocol works provided that $k > 1 + \ln b + 2^r$. Ada et al. [2] previously obtained *simultaneous* and deterministic efficient protocols for composed functions of block-width $r = 1$. The new protocol is the first to work for block composed functions with $r > 1$. Moreover, it is simultaneous, with vanishingly small error probability, if public coin randomness is allowed. The deterministic and zero-error version barely uses interaction.

1998 ACM Subject Classification F. Theory of Computation

Keywords and phrases Communication complexity, Number-On-Forehead model, composed functions

Digital Object Identifier 10.4230/LIPIcs.APPROX-RANDOM.2014.596

1 Introduction

In the *Number-on-Forehead* (NOF) model of communication, k players collaborate to evaluate a function f on a $k \times m$ boolean matrix $X = (x_{i,j})$. Player i knows all input bits except those in row i which is represented metaphorically by saying that row i is on the forehead of Player i, who sees all foreheads except her own. The players communicate by broadcast. The goal is to design a communication protocol for evaluating f that minimizes the number of bits of communication. Every such function can be evaluated with $m + 1$ bits of communication by having the kth player broadcast the first row of the matrix; the first player (who then knows the entire matrix) evaluates the function and announces the result.

Since it was introduced by Chandra, Furst and Lipton [8], the model has been studied extensively (e. g., [5, 12, 3, 6, 10, 18, 9, 14, 11, 16]), in part because it captures a communication bottleneck relevant to several models of computation such as branching programs, boolean circuits, SAT refutation via polynomial calculus etc. For each of these models, proving lower

* The first author is partially supported by a Ramanujan Fellowship of the DST and research grants of DAE. The second author's work was supported in part by NSF Grants CCF-0832787 and CCF-1218711.

bounds for computing some function f reduces to proving communication lower bounds for a function related to f in the NOF model.

For example, the complexity class ACC^0 is believed to be rather weak.[1] This belief is based on the famous Razborov-Smolensky Theorem [15, 17] stating that AC^0 circuits augmented with MOD_p gates, for any fixed prime p, cannot even compute efficiently the majority function MAJ (which outputs 1 if at least half the input bits are 1). A widely held conjecture says that ACC^0 does not contain MAJ, but the only known non-trivial separation is $NEXP \not\subset \mathrm{ACC}^0$ [19]. Combining results of [13, 7] gives that for any $f : \{0,1\}^n \to \{0,1\}$ in ACC^0 there is a constant C such that for $k = (\log n)^C$ if the variables of f are arranged (arbitrarily) in a matrix with k rows (padding rows with dummy inputs as needed) there is a k-player NOF protocol for evaluating f that is efficient (uses $\log(n)^{O(1)}$ bits of communication). This has inspired researchers to seek an explicit function f on $n = mk$ bits for which there is provably no efficient NOF k-party protocol as long as $k = (\log n)^{O(1)}$. This would separate ACC^0 from any complexity class containing f.

For the generalized inner product function GIP_k^m which outputs 1 if the input matrix has an odd number of all 1 columns, [5] proves a bound of $\Omega(m/4^k)$. This lower bound is $m^{\Omega(1)}$ if the number of players is less than $(1-\varepsilon)\log m$ but becomes trivial if $k \geq \log m$. Similarly all known NOF lower bounds become trivial for $k \geq \log m$.

One might guess that GIP remains hard for NOF when $k \geq \log(m)$, but surprisingly Grolmusz [12] found a protocol for $k \geq \log(m)$ with cost $O((\log m)^2)$. His protocol relies on the structure of GIP. For a k-variate boolean function g and an m variate function f the composition $\big(f \circ g\big)(M)$ has output $f(g(A^1), \ldots, g(A^m))$ where A^i is the i-th column of A. We call f the *outer function* and g the *inner function*. For GIP, f is sum modulo 2, and g is AND. In fact, Grolmusz's protocol works if f is symmetric (invariant under any permutation of the variables).

Babai et al. [4] suggested that the composed function with outer function MAJ_m (the m bit majority function) and inner function MAJ_k might be hard for NOF, however Babai, Gál, Kimmel and Lokam [3] refuted this by giving an efficient simultaneous protocol [2] that works for a composed function with symmetric outer function and an inner function that is both symmetric and *compressible*, provided that the number of players is a sufficiently large poly-logarithmic function of m. We won't define compressible here, but we note that MAJ is compressible and so their protocol applies to $\mathrm{MAJ}_m \circ \mathrm{MAJ}_k$.

Babai et al. [3] then suggested that $\mathrm{MAJ}_m \circ Q$ where Q is not compressible, might be hard for the NOF model. Recently, however, Ada et al.[2] showed that for k slightly larger than $\log m$, the composition of any symmetric function with *any* k-variate inner function, has a very efficient deterministic simultaneous NOF protocol.

Babai et al. also suggested considering composed functions whose inner function depends on more than 1 bit from each player. More precisely, let b and r be integers and let $m = br$. Split the $k \times m$ matrix A into b blocks, $A^1, \ldots, A^b$, where each block is a $k \times r$ matrix. Consider the composition $f \circ g$ where f has b variables and g has kr variables. We call r the *block width* of g. Specifically, they suggested looking at the function $\mathrm{MAJ}_b \circ T_t^{k,r}$, where $T_t^{k,r}$ takes as input a $k \times r$ matrix and interprets each row i as an r-bit integer z_i, and outputs 1 if $z_1 + \cdots + z_k > t$. They suggested $b = r$ as a case of special interest, but noted that even the case $r = 2$ is open.

[1] ACC^0 is the class of boolean functions computable by circuits of polynomial size and constant depth using AND gates, OR gates and MOD_w gates for some fixed positive integer w. A MOD_w gate outputs 1 iff the sum of the input values is divisible by w.

[2] In a simultaneous protocol, all processors simultaneously send one message to a *referee* who computes $f(A)$ from the messages.

Here we give the first efficient NOF protocol for composed functions having block width above 1. Corollary 2 implies that $\mathrm{MAJ} \circ T_t^{k,r}$ has an efficient NOF protocol of only poly-logarithmic (i. e. $\log(m)^{O(1)}$) cost, when the number of players k is $\Omega(log(m))^2$ and the block width r is at most $\log\log(m)$. While our primary interest is in boolean functions, our result is naturally stated for polynomial functions over a finite field. The set up we work with is:

- $\mathbb{F}$ is a finite field.
- $D \subseteq \mathbb{F}$.
- $p^1, \ldots, p^b$ are polynomial functions of the entries of a $k \times m$ matrix each of which depends on at most r variables per row.
- $p = \sum_{i=1}^{b} p^i$.
- A is an assignment to the variables whose entries are all in D.
- $n = mk$.

We consider the k-party NOF complexity of evaluating $p(A)$. A key observation that was used previously in making the connection between ACC^0 lower bounds and NOF-complexity, is that if p is a polynomial of degree strictly less than k, then p has a very efficient k-party simultaneous protocol: for any monomial of degree less than k there is some player who sees all the variables of that monomial and so the polynomial p can be decomposed as a sum of polynomials $p^1 + \cdots + p^k$ where Player j sees all of the variables needed to evaluate p^j, and so can simply announce $p^j(A)$. However, if the degree of p exceeds k there are no general methods known. In the above set up, the degree of p is rk. Our main result shows that if r is not too big then we can get efficient protocols.

▶ **Theorem 1.** **1.** *Let $\gamma > 0$ and suppose $k \geq 1 + |D|^r \ln(b/\gamma)$. There is a randomized simultaneous message NOF protocol that outputs either $p(A)$ or "failure", where the probability that it outputs "failure" is at most γ. The total communication cost of the protocol is at most $\left(1 + |D|^r \ln(b/\gamma)\right)\lceil\log(1 + |\mathbb{F}|)\rceil$.*

2. *Suppose $k \geq (1 + |D|^r \ln(2b))$. There is a 2 round deterministic NOF protocol that outputs $p(A)$ having total communication cost $(1 + |D|^r \ln(2b))(\lceil r \log|D|\rceil + \lceil\log|\mathbb{F}|\rceil)$.*

▶ Remark. As in the work of Babai et al. [3], in public-coin simultaneous message protocols, all coin-tosses are visible to all players and the referee.

For boolean functions we get:

▶ **Corollary 2.** *Let g be a boolean function whose variable set is a $k \times r$ matrix and let f be a symmetric b-variate boolean function.*

- *Suppose $\gamma > 0$ and $k \geq 1 + 2^r \ln(b/\gamma)$. There is a public-coin randomized simultaneous message protocol that outputs either $(f \circ g)(A)$ or "failure", where the probability that it outputs failure is at most γ. The total communication is at most $\left(1 + 2^r \ln(b/\gamma)\right)\lceil\log(1 + 2b)\rceil$.*
- *If $k \geq 1 + 2^r \ln(2b)$, there is a 2 round deterministic NOF protocol for $f \circ g$ with communication $(1 + 2^r \ln(2b))(r + \lceil\log(2b)\rceil)$.*

To deduce the corollary, let q be the smallest prime that is greater than b (so $b \leq q \leq 2b$) and let $\mathbb{F}$ be the field of integers mod q. For any boolean function there is a polynomial λ over field $\mathbb{F}$ that agrees with g on every 0-1 input. Let λ be the kr-variate polynomial over $\mathbb{F}$ that represents the given boolean function g. Let X be a $k \times rb$ matrix of variables. For $i \in [b]$, let X^i be the ith $k \times r$ block of variables and define the polynomial $p^i(X)$ by $\lambda(X^i)$. The polynomial $p(X) = \sum_{i=1}^{b} p^i(X)$ counts the number of X^i for which $g(X^i) = 1$ and since f is a symmetric function, $p(X)$ determines $(f \circ g)(X)$. Now apply Theorem 1 to p with $D = \{0, 1\}$.

Main Idea for our Protocol

As mentioned earlier, a polynomial p of degree less than k can be evaluated by k players in the NOF model by decomposing p as a sum of k polynomials, where the i-th polynomial can be evaluated privately by Player i. For a polynomial of degree k or more we can't do this. Still every polynomial p can be decomposed as a sum of polynomials $q_0 + q_1 + \cdots + q_k$ where q_0 consists of monomials that depend on every row of A (and thus can't be evaluated by any one player) and q_i consists of all monomials that contain at least one variable from each of the rows $1, \ldots, i-1$ and no variable from row i, and can thus be evaluated by Player i. So the problematic part is q_0, which is identically 0 if p has degree less than k. The first (simple) idea is that we don't need q_0 to be identically 0, we only need that $q_0(A) = 0$. The second idea is to consider alternative bases (rather than the standard monomial basis) for writing polynomials. A natural set of bases to consider are *shifted* monomial bases, where we fix a matrix B and consider the B-shifted basis consisting of products of terms of the form $x_{i,j} - B_{i,j}$. Each such B gives rise to an alternative decomposition $q_0^B + \cdots + q_k^B$. A simple but key observation is that the polynomial q_0^B varies depending on B, and it suffices for the players to agree on B so that $q_0^B(A) = 0$. Furthermore for our set up, the polynomial p is initially given as a sum of polynomials p^u each depending on only a few variables per row. The players can choose a different shift B^u for each polynomial p^u and decompose p^u with respect to that basis. Hence, the problem becomes to find a way for the players to identify and agree upon a sequence $(B^u : u \in [b])$ of shift matrices such that for every $u \in [b]$ when p^u is decomposed with respect to B^u the associated polynomial q_0^u evaluated at A is 0. It turns out that, using the fact that each p^u depends on only a few variables per row, this is easy to do.

To give a simple illustration of this idea, we give a short reformulation of the proof of Gromulsz' result that $GIP_k^m(A) = \sum_{i=1}^m \prod_{j=1}^k A_{i,j}$ has a very efficient protocol if $k > 1 + \log(m)$. We work over the field $\mathbb{F}_p$ where p is a prime larger than m (which we can assume is less than $2m$), which does not change the answer.

Here's the protocol: Since $k > 1 + \log(m)$ the number of columns is less than 2^{k-1} and Player k (who sees all rows but the last row) can identify a vector v of length $k-1$ that disagrees with every column he sees. He announces this vector. The players define the $k \times m$ matrix B to have all columns equal to the vector $\bar{v}$ (the vector obtained by complementing v) followed by a 0. All players can rewrite GIP_k^m in terms of the decomposition $q_0^B + \cdots + q_k^B$ described above. For $1 \leq i \leq k$, Player i is able to evaluate $q_i^B(A)$ and announce the result, and the output of the protocol is then $\sum_{i=1}^k q_i^B(A)$. The correctness of the protocol follows from the observation that $q_0^B(A) = 0$ since the monomials appearing in q_0^B are of the form $\prod_{i=1}^k (A_i^u - B_i^u)$ and for each column of A, there is an entry that agrees with the corresponding entry of B. The total cost of the protocol is $k - 1 + k\lceil \log(p) \rceil$.

The previous protocols for composed functions by Grolmusz [12], Babai et al. [3] and Ada et al. [2] did not use this polynomial view. In his Ph. D. thesis, Ada ([1]) gave an interpretation of the protocol of [2] in terms of polynomials, but not in terms of shifted bases. The use of shifted bases is the main technical innovation of this paper.

2 Some definitions

$\mathbb{F}[x_1, \ldots, x_n]$ denotes the ring of polynomials over field $\mathbb{F}$. The set of monomials $x_1^{j_1} \ldots x_n^{j_n}$ where $(j_1, \ldots, j_n) \in \mathbb{N}^n$ is a basis. More generally, for $c = (c_1, \ldots, c_n) \in \mathbb{F}^n$ the set of *c-shifted monomials* $(x_1 - c_1)^{j_1} \ldots (x_n - c_n)^{j_n}$ comprise a basis, called the *c-shifted basis.* A polynomial p is *independent* of x_i if no monomial in the monomial expansion of p includes x_i.

In the NOF setting, the variables are $(x_{i,j} : 1 \le i \le k, 1 \le j \le m)$. An *assignment* is a $k \times m$ matrix A. A polynomial p which contains no variable of row i is said to be *independent of row* i. The *row-by-row decomposition of* p *relative to assignment* B expresses p as the sum $q_0^B + q_1^B + \cdots + q_k^B$, as follows. Expand p in the B-shifted basis and let q_0^B be the sum of those (shifted) monomials in the expansion (with coefficients) that depend on every row, and for $i \ge 1$ let q_i^B be the sum of all monomials that are independent of row i and dependent on rows $1, \ldots, i-1$. Note each monomial is included in one and only one of the polynomials.

3 Proof of Theorem 1

The goal is to evaluate $p(A)$. Suppose the players are all given some fixed auxiliary assignment B. All of them can compute the row-by-row decomposition $q_0^B + \cdots + q_k^B$. Player i can evaluate $q_i^B(A)$ and announce the result with total cost $k\lceil \log |\mathbb{F}| \rceil$. If it happens that $q_0^B(A) = 0$ then this is enough to determine $p(A)$. It therefore suffices to show how the players agree on a matrix B such that $q_0^B(A) = 0$.

To do this, we use the hypothesis of the theorem that $p = p^1 + \cdots + p^b$ where p^j depends on at most r variables per row. We define a simultaneous protocol Π_C which depends on a $k \times r$ matrix C. We'll show that this protocol works provided that C satisfies certain properties. We will also show that the players can agree on a C to satisfy these properties (either using shared randomness, or deterministically by having Player k choose C).

The matrix C is used to define $k \times m$ matrices $B^1(C), \ldots, B^b(C)$ as follows: For each $u \in [b]$, let X_i^u be the sequence of (at most r) variables in row i on which p^u depends. In $B^u(C)$, assign the variables of X_i^u from left to right according to row i of C. Other variables in row i are set to 0. Let $q_0^u + q_1^u + \cdots + q_k^u$ be the row-by-row decomposition of p^u relative to $B^u(C)$. Given C, the matrices $B^u(C)$ and the decomposition of p^u can be computed privately by each player.

Now in Π_C each player i announces $\alpha_i = \sum_{u=1}^{b} q_i^u(A)$ and the output of the protocol is $\sum_i \alpha_i$. The cost is $k\lceil \log |\mathbb{F}| \rceil$.

The difference of the output of the protocol and the correct answer is $p(A) - \sum_{u=1}^{b} q_0^u(A)$, so it suffices to select C so that $q_0^u(A) = 0$ for all u. The following definitions will be helpful to achieve this.

- For polynomial p^u and row index i, and for matrices A and B we write $B \equiv_{p^u,i} A$ if A and B agree on all variables of row i on which p^u depends.
- For $u \in [b]$ and $j \in [k]$ we say that C satisfies property $Q^u(j)$ if there is an index $i^u \ne j$ such that $B^u(C) \equiv_{p^u,i^u} A$.
- For $j \in [k]$ we say that C satisfies property $Q(j)$ if it satisfies $Q^u(j)$ for every $u \in [b]$.

Observe that if C satisfies property $Q(k)$, then for each $u \in [b]$ there is an index $i^u < k$ such that $B^u(C)$ agrees with A on all variables of row i^u that appear in p^u. Each $B^u(C)$-shifted monomial of q_0^u contains a variable from each row so in particular it contains a variable from row i^u and thus the monomial vanishes at A. Thus $q_0^u(A) = 0$ for all u and so Π_C will give the correct answer. Observe also that Player k is able to privately check whether a matrix C satisfies $Q(k)$.

▶ **Claim 3.** *Let $\gamma > 0$, $1 \le j \le k$ and $k \ge \ln\big(b/\gamma\big)|D|^r + 1$. If C is chosen uniformly at random from among $k \times r$ matrices with entries in D, the probability that the matrix C does not satisfy $Q(j)$ is at most γ.*

Proof. By hypothesis, p^u depends on at most r variables from row i. Thus, for $i \neq j$, the probability that $B^u(C) \equiv_{p^u,i} A$ is at least $1/|D|^r$. Hence, the probability that $Q^u(j)$ does not hold (which is the probability that for all $i \neq j$, $B^u \not\equiv_{(p^u,i)} A$), is at most $(1 - 1/|D|^r)^{k-1} \leq e^{-(k-1)/|D|^r}$. Taking a union bound over $u \in [b]$ gives that the probability that $Q(j)$ fails is at most $be^{-(k-1)/|D|^r} \leq be^{-(k-1)/|D|^r} \leq \gamma$ using the hypothesized lower bound on k. ◀

We now state our randomized simultaneous message protocol: players use public coins to uniformly sample C. Every player other than player k runs Π_C. Player k first checks whether C satisfies $Q(k)$ (which can be done privately). If it does then he runs Π_C and makes the appropriate announcement. If C does not satisfy $Q(k)$, Player k announces "failure". By Claim 3, assuming that $k \geq 1 + \ln(b/\gamma)|D|^r$, this happens with probability at most γ. Each player sends at most $\lceil \log |\mathbb{F}| + 1 \rceil$ bits (where the "+1" includes the possibility of failure), for a total of $k\lceil \log |\mathbb{F} + 1| \rceil$ bits.

For the deterministic protocol, if we take $\gamma = 1/2$ in the Claim, then for $k \geq 1 + \ln(2b)|D|^r$, there is a matrix C satisfying $Q(k)$. Player k can select such a C privately satisfying $Q(k)$ and announce it (using $kr\lceil \log |D| \rceil$ bits). The players then run Π_C. The total communication is at most $k(r\lceil \log |D| \rceil + \lceil \log |\mathbb{F}| \rceil)$.

The communication costs of these randomized and deterministic protocols grow linearly with k. To reduce the communication cost to the cost claimed in the theorem, let $k' = \lceil 1 + |D|^r \ln(b/\gamma) \rceil$. Without any communication, each player $1, \ldots, k'$ can simplify the polynomial p by substituting in the variables appearing in rows after k'. This gives a polynomial p' that depends only on the first k' rows. The polynomial p' and the number k' satisfy the hypotheses for the above arguments for both the randomized and deterministic protocols. So players $1, \ldots k'$ can evaluate p' with the rest of the players remaining silent. Thus, replacing k by k' in the cost of the protocols above, completely establishes Theorem 1.

4 Conclusion and Open Problems

We give the first efficient NOF protocol for composed functions of block width greater than 1. Some further questions suggested by our work are stated below:

- To de-randomize our simultaneous message protocol, we used interaction in a very limited way. Can it be made a simultaneous deterministic protocol? The protocol Π_C is simultaneous, so the non-simultaneity only comes from having to choose C satisfying Claim 3. In our protocol this is done by Player k but it seems possible that this can be done simultaneously. Player j can privately determine the set of all matrices C that satisfy $Q(j)$. Claim 3 can be easily modified to show that (for k a bit larger than $2^r + \ln(b)$) there are several matrices C that satisfy $Q(i)$ for all i. Consider the simultaneous protocol in which each player j announces every C that satisfies $Q(j)$ together with his announcement for the protocol Π_C. For C that satisfies $Q(j)$ for all j, the players will have all run Π_C from which $p(A)$ can be deduced. The problem with this protocol is that if there are many matrices that satisfy $Q(j)$ for some j then it may be very costly. This gives rise to the following problem: is it possible for each player j to (privately) select a small subset $\mathcal{C}_j$ of matrices satisfying $Q(j)$ in such a way that $\cap_j \mathcal{C}_j$ is non-empty. If so, then Player j can announce only those matrices in $\mathcal{C}_j$, thereby giving an efficient simultaneous NOF protocol.
- Our protocol works for all inner functions of block width r. The number of players and the communication needed is exponential in r. Can the dependence on r be improved? The only lower bound on the communication we know is linear in r, which comes from a

simple counting argument (which is essentially the same argument which shows that for general functions on mk variables there is a function that requires communication $\Omega(m)$.)

- If we restrict the inner function to a specific interesting function, such as $T_t^{k,r}$, then the counting lower bounds don't work. Are there protocols that handle larger block width for this function?

Acknowledgements. We are grateful to anonymous referees for their many valuable comments and suggestions that helped improve readability of the paper. In particular, it has resulted in a simplification of the proof of Theorem 1.

References

1 A. Ada. *Communication complexity.* PhD thesis, McGill University, 2013.

2 A. Ada, A. Chattopadhyay, O. Fawzi, and P. Nguyen. The NOF multiparty communication complexity of composed functions. In *International Colloquium on Automata, Languages and Programming (ICALP)*, pages 13–24, 2012.

3 L. Babai, A. Gál, P. G. Kimmel, and S. V. Lokam. Communication complexity of simultaneous messages. *SIAM J. of Computing*, 33:137–166, 2003.

4 L. Babai, P. G. Kimmel, and S. V. Lokam. Simultaneous messages vs. communication. In *12th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 361–372. Springer, 1995.

5 L. Babai, N. Nisan, and M. Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45(2):204–232, 1992.

6 P. Beame, T. Pitassi, N. Segerlind, and A. Wigderson. A strong direct product theorem for corruption and the multiparty communication complexity of disjointness. *Computational Complexity*, 15(4):391–432, 2006.

7 Richard Beigel and Jun Tarui. On ACC. *Computational Complexity*, 4:350–366, 1994.

8 A. K. Chandra, M. L. Furst, and R. J. Lipton. Multiparty protocols. In *15th ACM Symposium on Theory of Computing (STOC)*, pages 94–99, 1983.

9 A. Chattopadhyay and A. Ada. Multiparty communication complexity of disjointness. Technical Report TR08-002, Electronic Colloquium on Computational Complexity (ECCC), 2008.

10 A. Chattopadhyay, A. Krebs, M. Koucký, M. Szegedy, P. Tesson, and D. Thérien. Languages with bounded multiparty communication complexity. In *Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 500–511, 2007.

11 M. David, T. Pitassi, and E. Viola. Improved separations between nondeterministic and randomized multiparty communication. *ACM Transactions on Computation Theory (TOCT)*, 1(2), 2009.

12 V. Grolmusz. The BNS lower bound for multi-party protocols is nearly optimal. *Information and Computation*, 112:51–54, 1994.

13 Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Computational Complexity*, 1:113–129, 1991.

14 T. Lee and A. Shraibman. Disjointness is hard in the multiparty number-on-the-forehead model. *Computational Complexity*, 18(2):309–336, 2009.

15 A. A. Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition. *Math. Notes of the Acad. of Sci. of USSR*, 41(3):333–338, 1987.

16 A. A. Sherstov. The multiparty communication complexity of set disjointness. In *44th Symposium on Theory of Computing (STOC)*, 2012.

17 Roman Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *19th Annual ACM Symposium on Theory of Computing*, pages 77–82. ACM Press, 1987.

18 E. Viola and A. Wigderson. Norms, XOR lemmas, and lower bounds for polynomials and protocols. *Theory of Computing*, 4(1):137–168, 2008.

19 Ryan Williams. Non-uniform ACC circuit lower bounds. In *IEEE Conference on Computational Complexity*, pages 115–125, 2011.