The Power of Super-logarithmic Number of Players

Arkadev Chattopadhyay∗ Michael E. Saks†

April 24, 2014

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 size. 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 $f(g(A^1),\ldots,g(A^k))$ 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 [ACFN12] 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.

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 $k$th 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 [CFL83], the model has been studied extensively (e.g., [BNS92, Gro94, BGKL03, BPSW06, CKK+07, VW08, CA08, LS09, DPV09, She12]), 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 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 $\text{ACC}^0$ is believed to be rather weak.\(^1\) This belief is based on the famous Razborov-Smolensky Theorem [Raz87, Smo87] stating that $\text{AC}^0$ circuits augmented with $\text{MOD}_p$ gates, for any fixed prime $p$, cannot even compute efficiently the majority function $\text{MAJ}$ (which outputs 1 if at least half the input bits are 1). A widely held conjecture says that $\text{ACC}^0$ does not contain $\text{MAJ}$, but the only known non-trivial separation is $\text{NEXP} \not\subseteq \text{ACC}^0$ [Wil11]. Combining results of [HG91, BT94] gives that for any $f : \{0,1\}^n \rightarrow \{0,1\}$ in $\text{ACC}^0$ 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 $\text{ACC}^0$ from any complexity class containing $f$.

The best lower bound known in the NOF model is $\Omega(m/\log k)$ [BNS92] for the generalized inner product function $\text{GIP}^m_k$, which outputs 1 if the input matrix has an odd number of all 1 columns. This lower bound is $n^{\Omega(1)}$ 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 [Gro94] 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 $(f \circ g)(M)$ 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. [BKL95] suggested that the composed function with outer function $\text{MAJ}_m$ (the $m$ bit majority function) and inner function $\text{MAJ}_k$ might be hard for NOF, however Babai, Gál, Kimmel and Lokam [BGKL03] 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 $\text{MAJ}_m \circ \text{MAJ}_k$.

Babai et al. [BGKL03] then suggested that $\text{MAJ}_m \circ Q$ where $Q$ is not compressible, might be hard for the NOF model. Recently, however, Ada et al. [ACFN12] showed that for $k$ slightly larger than $\log m$, the composition of any symmetric function with any 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

\(^1\) $\text{ACC}^0$ is the class of boolean functions computable by circuits of polynomial size and constant depth using AND gates, OR gates and $\text{MOD}_w$ gates for some fixed positive integer $w$. A $\text{MOD}_w$ 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.
the composition \( f \circ g \) where \( f \) has \( b \) variables and \( g \) has \( kr \) variables. We call \( r \) the block length of \( g \). Specifically, they suggested looking at the function \( \text{MAJ}_b \circ T^{k,r}_t \), where \( T^{k,r}_t \) 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.

Here we give the first efficient NOF protocol for composed functions having block length above 1. Corollary 1.2 implies that \( \text{MAJ} \circ T^{k,r}_t \) has an efficient NOF protocol of only polylogarithmic (i.e. \( \log(m)^{O(1)} \)) cost, when the number of players \( k \) is \( \Omega(\log(m))^2 \) and the block length \( r \) 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 \( \text{ACC}^0 \) 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.** 1. Let \( \gamma > 0 \) and suppose \( k \geq 1 + |D|^r \ln(bn/\gamma) \). There is a randomized simultaneous message NOF protocol which outputs either \( p(A) \) or ”failure”, where the probability that it outputs ”failure” is at most \( \gamma \). The total communication cost of the protocol is at most \( (1 + |D|^r \ln(bn/\gamma)) \lfloor \log(1 + |\mathbb{F}|) \rfloor \).

2. Suppose \( k \geq (1 + |D|^r \ln(2bn)) \). There is a deterministic NOF protocol that outputs \( p(A) \) having total communication cost \( (1 + |D|^r \ln(2bn))(|r \log |D|| + \lfloor \log |\mathbb{F}| \rfloor) \).

**Remark 1.** As in the work of Babai et.al [BGKL03], in public-coin simultaneous message protocols, all coin-tosses are visible to all players and the referee.

For boolean functions we get:

**Corollary 1.2.** 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.
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 for 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 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 $q^0_B$ depends on $B$, and so 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 when $p^u$ is decomposed with respect to $B^u$ the associated polynomial $q^0_B$ 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.

We point out that the previous works on protocols for composed functions by Grochowski [Gro94], Babai et.al [BGKL03] and Ada et.al [ACFN12] did not use this polynomial view.
monomials \((x_1 - c_1)^{j_1} \cdots (x_n - c_n)^{j_n}\) 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 \((x_{i,j} : 1 \leq i \leq k, 1 \leq j \leq m)\). 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 \(i \geq 1\) 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 1.1.

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 \(p = p_1^1 + \cdots + p_l^b\) where \(p_j^i\) depends in at most \(r\) variables per row. We define a simultaneous protocol \(\Pi_C\) 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 \(X_u^n\) be the sequence of (at most \(r\)) variables in row \(i\) on which \(p_u^i\) depends. In \(B_u^i(C)\), assign the variables of \(X_u^n\) 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 \(p_u^i\) relative to \(B_u^i(C)\). Given \(C\), the matrices \(B_u^i(C)\) and the decomposition of \(p_u^i\) can be computed privately by each player.

Now in \(\Pi_C\) each player \(i\) announces \(\alpha_i = \sum_{u=1}^b q_u^i(A)\) 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 from the correct answer is equal to \(p(A) - \sum_{u=1}^b q_0^u(A)\), so it suffices that \(q_0^u(A) = 0\) for all \(u\). The following definitions will be helpful to achieve this.

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

Observe that if \(C\) satisfies property \(Q(j)\) for some \(j\), then for each \(u \in [b]\) there is an index \(i_u^j \neq j\) such that \(B_u^i(C)\) agrees with \(A\) on all variables of row \(i_u^j\) that appear in \(p_u^i\). Each
variables appearing in rows after any communication, each player 1 on kr there is a matrix Ck most
the rest of the players remaining silent. Thus, replacing kr,...,k for both the randomized and deterministic protocols. So players 1
γ > 0 and k ≥ ln (bn/γ)|D|r + 1. If C is chosen uniformly at random from among k × r matrices with entries in D, the probability that the matrix C does not satisfy Q is at most γ.

Proof. Let j be any row. By hypothesis, p\textsuperscript{u} depends on at most r variables from row i. Thus, for i ≠ j, the probability that B\textsuperscript{u}(C) ≡ \textit{p}_{\textit{r},i} A is at least 1/|D|\textsuperscript{r}. Hence, the probability that Q\textsuperscript{u}(j) does not hold, which is the probability that for all i ≠ j, B\textsuperscript{u} ≠ (p\textsuperscript{u},i) A, is at most (1 − 1/|D|\textsuperscript{r})\textsuperscript{k−1} ≤ e−(k−1)/|D|\textsuperscript{r}. Taking a union bound over \textit{u} ∈ [b] and j ∈ [k] gives that the probability that Q fails is at most bk\textsuperscript{e}−(k−1)/|D|\textsuperscript{r} ≤ bne−(k−1)/|D|\textsuperscript{r} ≤ γ using the hypothesized lower bound on k.

We now state our randomized simultaneous message protocol: players use public coins to uniformly sample C. Each player j checks whether C satisfies Q(j) (which can be done privately). If it does then he runs Π\textsubscript{C} and makes the appropriate announcement. If C does not satisfy Q(j), player j announces “failure”. If no player says failure then C satisfies property Q and so Π\textsubscript{C} provides the correct answer. If any player announces “failure” then the referee announces “failure”. By Claim 3.1, assuming that k ≥ 1 + ln(bn/γ)|D|\textsuperscript{r}, this happens with probability at most γ. Each player sends at most \lceil log |F| + 1 \rceil bits (where the “+1” includes the possibility of failure), for a total of k\lceil log |F| + 1 \rceil bits.

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

Note that both our randomized and deterministic protocols have an explicit dependence on k which becomes unaffordable for large values of k. To reduce the communication cost of these protocols to the amount claimed in the theorem, let k′ = \lceil 1 + |D|\textsuperscript{r} ln(nb/γ) \rceil. Without any communication, each player 1,...,k′ can simplify the polynomial \textit{p} by substituting in the variables appearing in rows after k′. This gives a polynomial \textit{p′} that depends only on the first k′ rows. The polynomial \textit{p′} and the number k′ satisfy the hypotheses for the above arguments for both the randomized and deterministic protocols. So players 1,...,k′ can evaluate \textit{p′} with the rest of the players remaining silent. Thus, replacing k by k′ in the cost of the protocols above, completely establishes Theorem 1.1.

4 Conclusion and Open Problems

We give the first efficient NOF protocol for composed functions of block length 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 \( \Pi_C \) is simultaneous, so the non-simultaneity only comes from having to choose \( C \) satisfying Claim 3.1. 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 3.1 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 \( \Pi_C \). For \( C \) that satisfies \( Q(j) \) for all \( j \), the players will have all run \( \Pi_C \) 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 \( C_j \) of matrices satisfying \( Q_j \) in such a way that \( \bigcap_j C_j \) is nonempty. If so, then player \( j \) can announce only those matrices in \( C_j \), thereby giving an efficient NOF protocol.

Our protocol works for all inner functions of block length \( r \). 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 \( T_{k,r}^t \), then the counting lower bounds don’t work. Are there protocols that handle larger block length for this function?

References


