CIRCUITS AND MULTI–PARTY PROTOCOLS

VINCE GROLMUSZ

Abstract.

Using multi–party communication techniques, we prove that depth–3 circuits with a threshold gate at the top, arbitrary symmetric gates at the next, and fan–in k MOD m gates at the bottom, need exponential size to compute the k-wise inner product function of *Babai, Nisan* and *Szegedy*, where m is odd positive integer, satisfying $m \equiv k \pmod{2m}$. This is one of the rare lower–bound results involving MOD m gates with non–prime power moduli.

Exponential gap–theorems are also presented between the multi–party communication complexities of closely related functions.

Key words. lower bounds, threshold circuits, ACC–circuits, communication protocols

Subject classifications. Primary: 68Q05, Secondary: 68Q15, 68Q22

1. Introduction

1.1. Circuit Complexity: MOD m vs. MOD p. Yao (1985) and Håstad (1986) proved, that any Boolean circuit with gates AND, OR, and NOT, and with depth less than

$$O\Big(\frac{\log n}{\log\log n}\Big),$$

needs exponential size to compute the PARITY function. After this result, the following question emerged: if PARITY is so hard, then what happens to the power of the circuit if PARITY gates are also allowed? Or, more generally, if

MOD m gates are allowed in the circuit, where a MOD m gate outputs 1 if the sum of its input–bits is divisible by m, and 0 otherwise. This question was first asked by Barrington (1986).
Razborov (1987) proved that the MAJORITY function needs exponential size if it is computed by bounded–depth circuits with AND, OR, NOT and MOD 2 (i.e., PARITY) gates.

Smolensky (1987) generalized this result to circuits with MOD p gates instead of MOD 2 ones, where p is a prime or prime–power. The case, where the modulus is a non-prime–power composite number, remained widely open. No lower bound was known even for depth–2 circuits with MOD 6 gates only.

For the depth–2 case Krause & Waack (1991) proved that any circuit with a MOD m gate at the top and arbitrary symmetric gates (e.g. MOD m gates) at the bottom needs exponential size to compute the $ID(x,y)$ function, where ID is defined as

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

Allender & Gore (1994) – using the result of Beigel & Tarui (1991) – proved that any *uniform* sequence of circuits of AND, OR, NOT, and MOD m gates needs exponential size to compute the *permanent* function. Using the uniformity assumption is *essential* here, since without it, it is unknown whether there exists any language in **NP**, or, even in **NEXP**, which cannot be computed with polynomial–size, bounded–depth circuits of AND, OR, NOT, and MOD m gates, where m is a non–prime–power positive integer.

1.2. Our Results – Circuit Complexity. Let A be a 0–1 matrix with n rows and k columns, that is, $A \in \{0,1\}^{n\times k}$. Let $GIP(A)$ denote the number of the all–1 rows of matrix A, modulo 2. If $k = 2$, we got the inner product function mod 2. Function GIP is called the *generalized inner product function* (Babai *et al.* (1992)).

Let f be a Boolean function of h variables. Function f is a *symmetric function* if there exists a $g : \mathbf{Z}_0^+ \to \{0,1\}$, such that

$$f(x_1, x_2, ..., x_h) = g\Big(\sum_{i=1}^{h} x_i\Big),$$

where $\mathbf{Z}_0^+$ denotes the set of all non-negative integers. A gate is a *symmetric gate* if it computes a symmetric function. The AND, OR, MOD m, MAJORITY, PARITY gates are all symmetric gates.

Our main result is the following theorem:

THEOREM 1. *Let $\mathcal{C} = \{C_{n,k}\}$ be a sequence of circuits, computing GIP(A) for $A \in \{0,1\}^{n\times k}$, where circuit $C_{n,k}$ has an unweighted threshold gate at the top, arbitrary symmetric gates at the second, and MOD m gates of fan–in k on the first level, where m is odd and*

$$k \equiv m \pmod{2m}$$

is satisfied. Then the size of $C_{n,k}$ is

$$\exp\left(\Omega\left(\frac{n}{4^k mk}\right)\right).$$

REMARKS.

(i) Since the MOD m gates are symmetric gates, Theorem 1 implies that any circuit, with a threshold gate at the top, MOD m gates of arbitrary fan–in on the next, and MOD m gates of fan–in k on the first level, needs exponential size for computing GIP(A), if m is odd and $k \equiv m \pmod{2m}$. The result of Krause & Waack (1991) does not need the fan–in bound, and the constraint for m, but works only for depth–2 circuits.

(ii) Håstad & Goldmann (1991) proved that a depth–3 circuit with a threshold gate at the top, symmetric gates at the next, and arbitrary gates with fan–in $k-1$ on the first level needs exponential size to compute GIP. This theorem cannot be generalized to lower fan–in of k, since with a MOD 2 gate (a symmetric gate) at the top, and n copies of fan–in k AND–gates at the bottom one can compute GIP(A). So the fan–in bound of k in Theorem 1 is not unreasonably restrictive. In the next section we survey the proof of *Håstad* and *Goldmann*, and highlight the difference between the fan–in bounds k and $k-1$ for matrices $A \in \{0,1\}^{n\times k}$.

(iii) One can allow that the depth–2 subcircuits of circuit $C_{n,k}$ have different odd moduli m_i, if they all satisfy $k \equiv m_i \pmod{2m_i}$, and have a common upper bound, independently from n.

The k–fan–in EXACT_ℓ gate outputs 1 iff exactly ℓ of its k inputs are 1. The following theorem applies for circuits with EXACT gates.

THEOREM 2. *Let $\mathcal{C}'$ denote the family of depth–3 circuits $C'_{n,k}$ computing GIP(A) for any $A \in \{0,1\}^{n\times k}$, where $k = p^c$ for some prime p and positive integer c. $C'_{n,k}$ has an unweighted threshold gate at the top, MOD p gates on* *the second, and EXACT$_\ell$ gates of fan–in k on the first level, where $1 \le \ell \le k-1$, and the entries of A with their negations on level 0. If $C'_{n,k}$ computes GIP(A), then its size is at least*

$$\exp\left(\frac{n}{4^k} - O(k^2 \log p)\right).$$

REMARK 3. Obviously, a k–fan–in AND is an EXACT$_k$ gate. Choosing $p = 2$, GIP(A) can be computed by a depth–2 circuit with a MOD 2 gate at the top, and n EXACT$_k$ gates on the next level, but, as Theorem 2 shows, no such circuit of a subexponential size can compute GIP, if EXACT$_\ell$ gates are used with $1 \le \ell \le k-1$.

We give the proofs of Theorems 1 and 2 in Section 3. The main tool in their proof is a multi–player communication game.

1.3. Multi–Party Communication Complexity. The notion of the two–party communication complexity was introduced by Yao (1979). Due to the algebraic characterization of the communication complexity, several strong lower bounds were proved for this model (see Lovász (1989) for a survey).

The *multi–party communication game*, first examined by Chandra *et al.* (1983), is a generalization of the 2–party communication game. In this game, k players: $P_1, P_2 ..., P_k$ intend to compute the value of $g(A_1, A_2, ..., A_k)$, where $g : \{0,1\}^{kn} \to \mathbf{Z}_0^+$ where $\mathbf{Z}_0^+$ denotes the set of non–negative integers, and $A_i \in \{0,1\}^n$, for $i = 1, 2, ..., k$. Player P_i knows every variable, *except* A_i, for $i = 1, 2, ..., k$. The players have unlimited computational power, and they communicate with the help of a blackboard, viewed by all players. Only one player may write on the blackboard at a time. The goal is to compute $g(A_1, A_2, ..., A_k)$, such that at the end of the computation, every player knows this value. The cost of the computation is the number of bits written on the blackboard for the given $A = (A_1, A_2, ..., A_k) \in \{0,1\}^{nk}$. The cost of a multi–party protocol is the maximum number of bits communicated for any A from $\{0,1\}^{nk}$. The k-party communication complexity, $C^{(k)}(g)$, of a function g, is the minimum of costs of those k-party protocols which compute g.

In contrast with the rich theory of the two–party communication games, there are only few results known about the multi–party communication complexity of functions. Communicating n bits, P_1 can compute any function of A: P_2 writes down the n bits of A_1 on the blackboard, P_1 reads it, and computes the value $g(A)$ at no cost. The additional cost of diffusing the result $g(A)$ to other players is the binary length of $g(A)$.

Babai *et al.* (1992) examined the multi–party communication complexity of the *Generalized Inner Product* (GIP) function:

DEFINITION 4. *(Babai* et al. *(1992)) The k–party ε–distributional communicational complexity of a function g, denoted by $C_\varepsilon^{(k)}(g)$, is the minimum number of bits that needed to be exchanged in the worst case, by any k–party protocol which computes g correctly on $1/2+\varepsilon$ fraction of the inputs.*

THEOREM 5. *(Babai* et al. *(1992), Theorem 2)*

$$C_\varepsilon^{(k)}(GIP) = \Omega\left(\frac{n}{4^k} + \log\varepsilon\right).$$

□

Substituting $\varepsilon = 1/2$ in Theorem 5, we get that the multi– party communication complexity of GIP is

$$\Omega\left(\frac{n}{4^k}\right).$$

A protocol of Grolmusz (1994) communicates

$$O\left(\frac{n}{2^k}k\right)$$

bits to compute GIP, which shows that the lower bound in Theorem 5 cannot be improved significantly.

Håstad & Goldmann (1991) found a surprising application of Theorem 5 to circuit–complexity. Håstad & Goldmann (1991) considered depth–3 threshold circuits with fan–in on the lowest level bounded by $k-1$, and they have shown, that the size of those circuits, computing GIP(A), should be exponential in n. The strategy of their proof is the following: it is assumed that the circuit of a given type and size M computes GIP(A). Then they show a k–party protocol, where all the players know the circuit, and which computes the output of the circuit (i.e. GIP(A)), with communicating about $O(\log M)$ bits. From Theorem 5, $O(\log M) \geq n/4^k$, which yields an exponential lower bound to M. The same proof applies to depth–3 circuits with a threshold gate at the top, arbitrary SYMMETRIC gates on the next and arbitrary gates of fan–in at most $k-1$ on the lowest level.

For the significance of this result it is worthwhile to mention, that no superpolynomial lower bound is known for the sizes of the depth–3 threshold circuits (without fan–in constraint), which compute a function in **NP**.

However, the $k-1$ bound on the fan–in on the lowest level is essential in the proof in Håstad & Goldmann (1991), with k players this facilitates the $O(\log M)$–communication protocol: since every gate at the bottom has fan–in at most $k-1$, for every gate there exists a player who knows all the inputs of that gate, and, consequently, does know its output. This method, however, cannot be applied, when the lower fan–in is k instead of $k-1$, since it may happen that no player knows all the inputs of a gate with fan–in k.

1.4. Our Results – Communication Complexity. First we give a definition of easy and hard functions in multi–party communication complexity:

DEFINITION 6. *Let* $G = \{g_{n,k} \mid n,k \in \mathbf{Z}_0^+,\ g_{n,k} : \{0,1\}^{n\times k} \to \mathbf{Z}_0^+\}$, *where* $\mathbf{Z}_0^+$ *denotes the set of non–negative integers. We say that a* G *is multi–party easy if* $\exists c > 0$ *such that for all* $g_{n,k} \in G$, $C^{(k)}(g_{n,k}) \leq 2^{ck}\log n$. *Let* **ME** *denote the family of all multi–party easy sets. We say that* G *is multi–party hard, if* $\exists c' > 0$ *such that for all* $g_{n,k} \in G$, $C^{(k)}(g_{n,k}) \geq n2^{-c'k}$. *Let* **MH** *denote the family of all multi–party hard functions.*

REMARK. Usually k is thought to be much smaller than $\log n$, say $o(\log n)$, or constant. When k is constant, then the membership in **ME** implies a logarithmic communication complexity, while members of **MH** have a linear communication complexity, so in this case the gap between these two classes is exponential.

Theorem 5 shows that GIP is in **MH**. In Section 2 we show several surprising theorems about the membership in the classes **MH** and **ME**, and these theorems form the basis for proving the circuit results:

THEOREM 7. *Let* m *be an odd, positive integer, let* $0 \leq \ell \leq m-1$, *and* $k \equiv m + 2\ell \pmod{2m}$. *Let* $A \in \{0,1\}^{n\times k}$. *Then the number of those rows of* A *which are congruent to* $\ell \pmod{m}$, *is in* **ME**.

With $\ell = 0$ we get that the number of rows divisible by m is in **ME**. However, not every congruence–class can be counted easily, even with the assumptions of Theorem 7:

COROLLARY 8. *Let* $m = k = 3$. *Then* $k \equiv m \pmod{2m}$ *is satisfied, but the number of rows congruent to 1* $\pmod{m}$ *is in* **MH**.

For even m, congruence–class counting is hard:

THEOREM 9. *Let* $A \in \{0,1\}^{n\times k}$, *and let* m *be an even positive integer. Then to compute the number of that rows of* A, *which are congruent to* $\ell \pmod m$ *is in* **MH**, *for any integer* ℓ.

If $m = 2$, at least a modular result is easy:

THEOREM 10. *The function, which is defined to be the number of even rows of* A, $\bmod 2^{k-1}$, *is in* **ME**.

From Theorem 5, the number of the all–1 rows is in **MH**.

COROLLARY 11. *Let* k *be an odd positive integer. The function which gives the number of the all–0 rows plus the number of the all–1 rows of* A *is in* **ME**.

2. The Protocol

In this section we describe a multi–party protocol which plays a main role in this paper.

DEFINITION 12. *Let* $A \in \{0,1\}^{n\times k}$. *We shall denote the rows of* A *by* A^j, $j = 1,2,...,n$, *and the columns of* A *by* A_i, *for* $i = 1,2,...,k$. *Let* A_i^j *denote the entry in row* j *and column* i *of* A. *Let* $m, z \in \mathbf{Z}_0^+$. *Suppose that* $1 \le j \le n$. *We say that row* A^j *is congruent to* $z \pmod m$, *iff*

$$\sum_{i=1}^{k} A_i^j \equiv z \pmod m.$$

We say that row A^j *is divisible by* m *if it is congruent to 0* $\pmod m$.

The goal of the players in protocol **MOD m** is to compute the number of the rows of A in every congruency–class, mod m.

NOTATION 13. *We denote the elements of set* ${\mathbf{Z}_0^+}^m$ *by small–case greek letters, and we index their coordinates from* 0 *through* $m-1$.

Definition 14. *Let* $A \in \{0,1\}^{n\times k}$ *and* $m \in \mathbf{Z}_0^+$. *Let*

$$\delta^{(m)}(A) = (\delta_0, \delta_1, ..., \delta_{m-1})$$

denote a vector where δ_i *is the number of that rows of* A*, which are congruent to* $i \pmod m$. *Let* $v \in \{0,1\}^k$*, then* $CT(v, A)$ *denotes the number of that rows of* A*, which are equal to* v. *Let* $\mathbf{0} = (0, 0, ..., 0) \in \{0,1\}^k$*, and* $\mathbf{1} = (1, 1, ..., 1) \in \{0,1\}^k$.

The fundamental strategy of the players in protocol **MOD m** is the following: Player P_i $(1 \le i \le k)$ assumes that column i of A, A_i is the all–1 vector. P_1 communicates the number of rows in separate congruency– classes, and then P_2 corrects him in case of that rows, which begin with 0, instead of the assumed 1. Then P_3 corrects P_2 and P_1 in case of that rows, which begins with two zeros, and so on, until P_k comes. Then P_k corrects P_1, P_2, ..., P_{k-1} in case of those rows which begin with $k-1$ zeros. The protocol makes errors only in the case of that rows, for which *neither of the assumptions* were satisfied: the rows with k 0's. Every other row will be counted correctly: since at least one player's assumption was right, he saw the row entirely, and counted it to the proper congruency–class, corrected the errors of the others.

Now we present a more detailed description of the protocol, together with its analysis. (The protocol itself is typesetted in a different (sans-serif) font, while the analytical remarks are in roman)

Protocol MOD m

P_1 begins the communication. Since P_1 assumes that the first column of A is the all–1 vector, P_1 is assumed to know the entire input, so he can communicate any function of it. P_1 first communicates α_0, the number of those rows, which are congruent to 0 (mod m), second α_1, the number of rows, congruent to 1 (mod m),..., and last α_{m-1}, the number of rows, congruent to m-1 (mod m). So P_1 communicates vector

$$\alpha = (\alpha_0, \alpha_1, ..., \alpha_{m-1})$$

of length $m\lceil\log(n+1)\rceil$. Let us note that

$$\sum_{\ell=0}^{m-1} \alpha_\ell = n.$$

P_1 correctly counts that rows, which begins with a 1, but if a row begins with a 0, and P_1 counted it to α_ℓ then correctly it should have been counted to $\alpha_{(\ell-1) \bmod m}$. P_2 communicates next. Since P_1 already advertised vector α, the task of P_2 is only to correct the errors made by P_1. P_2 knows where P_1 made an error: those rows begin with 0. Suppose that row A^j begins with a 0, and P_2 — using his assumption that A_2 is the all–1 vector — sees that A^j is congruent to ℓ mod m. P_2 knows, that P_1 assumed that the first entry of A^j is 1, and assumes that the second entry in A^j is also 1, so P_2 assumes that P_1 counted erroneously A^j to that rows, which are congruent to $\ell + 1 \pmod{m}$. P_2 subtracts 1 from the number $\alpha_{\ell+1 \pmod{m}}$ and adds 1 to α_ℓ. P_2 repeats this for all rows, beginning with 0, but communicates only the vector–sum of the corrections:

$$\beta^{(2)} = (\beta_0^{(2)}, \beta_1^{(2)}, ..., \beta_{m-1}^{(2)}),$$

where $\beta_i^{(2)}$ the number of those rows which begin with 0 and P_2 sees them to be congruent to i, minus the number of those rows, which begin with 0 and P_2 sees them to be congruent to sl i–1 mod m. Note that

$$\sum_{\ell=0}^{m-1} \beta_\ell^{(2)} = 0,$$

and $\beta^{(2)}$ can be communicated with $m\lceil\log(2n+1)\rceil$ bits, since every $\beta_i^{(2)}$ is a number of absolute value of at most $2n$. P_3, after that $P_4, ..., P_{i-1}$ communicates i $\leq$ k, and P_i communicates next. The task of P_i is to correct errors, committed by $P_1, P_2, ..., P_{i-1}$. Until now, all of the rows were counted correctly, which contain at least one bit 1 in the first $i-1$ positions. P_i deals only with rows which begin with i–1 zeros. Suppose that a row, A^j, begins with i-1 zeros, and P_i sees it to be congruent to ℓ mod m. Let $z = 1, 2, ..., i-1$. Then P_i assumes that P_z has seen A^j to be congruent with $\ell + 1$, so he corrects P_z. However, so far P_z have corrected $P_{z-1}, P_{z-2}, ..., P_1$ with an assumption that $A_z^j = 1$, but P_i knows that $A_z^j = 0$, so P_i should also *correct the corrections* of P_z. Let P_i communicate

$$\beta^{(i)} = (\beta_0^{(i)}, \beta_1^{(i)}, ..., \beta_{m-1}^{(i)}),$$

the vector–sum of the correction vectors, for $z = 1, 2, ..., i-1$. Since P_i knows the strategy of the other players, and assumes to know the whole input, he can simulate their computation, and can correct their errors. So P_i computes $\beta^{(i)}$, and can communicate it with $m\lceil\log(2^{i-1}n+1)\rceil$ bits. Let us note again, that

$$\sum_{\ell=0}^{m-1} \beta_\ell^{(i)} = 0.$$

When P_k has communicated $\beta^{(k)}$, all players compute – privately – the vector–sum

$$\gamma = \alpha + \sum_{i=2}^{k} \beta^{(i)}.$$

End of protocol MOD m.

The players of this protocol uses $O(mk\log n)$ bits of communication.

Let us observe that if no row of A is equal to $\mathbf{0}$, then

$$\gamma = \delta^{(m)}(A),$$

since every row is correctly counted by one player, and that player corrected all the previous errors, for that row.

NOTATION 15. *Let*

$$\Pi = \begin{pmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 1 & 0 & 0 & \dots & 0 & 0 \end{pmatrix}$$

the $m \times m$ circular–right–shift permutation–matrix.

LEMMA 16.

$$\gamma = \delta^{(m)}(A) + CT(\mathbf{0}, A)(\mu - \nu) \tag{1}$$

where $\nu = (1, 0, 0, ..., 0)$, and $\mu = \nu - \nu(I - \Pi)^k$.

PROOF. In protocol **MOD m** players count correctly all the rows, except those, which are equal to **0**. In fact, they never count the **0**–rows, since no player's assumption is compatible with **0**. Player P_i for each row **0** compute some vector $\mu^{(i)}$, which they add up to μ at the end:

$$\mu = \sum_{i=1}^{k} \mu^{(i)},$$

instead of the correct $\nu = (1, 0, 0, ..., 0)$, this shows the correctness of equation (1).

Our remaining task is to compute μ.

P_1 counts **0** to rows, congruent to $1 \pmod m$, so he adds the following $\mu^{(1)}$ to its communicated vector α, for each row **0**:

$$\mu^{(1)} = (0, 1, 0, ..., 0).$$

P_2 also counts **0** to rows, congruent to $1 \pmod m$, and he assumes, that P_1 counted the row to the rows, congruent to $2 \pmod m$. So P_2 adds

$$\mu^{(2)} = (0, 1, 0, ..., 0) - (0, 0, 1, 0, ..., 0) = \mu^{(1)} - \mu^{(1)}\Pi = \mu^{(1)}(I - \Pi)$$

to its $\beta^{(2)}$, where I denotes the $m \times m$ unit–matrix.

Now let $2 \le i \le k - 1$, and suppose that

$$\mu^{(i)} = \mu^{(1)}(I - \Pi)^{i-1}. \tag{2}$$

We state that P_{i+1} communicates $\mu^{(i)}$, the same corrections to $P_1, P_2, ..., P_{i-1}$ as P_i has communicated, since P_i assumes that bit i is the only 1–bit in the row, while P_{i+1} assumes that bit $i+1$ is the only 1–bit in the row, and these assumptions are equivalent, from the viewpoints of $P_1, P_2, ..., P_{i-1}$, so when P_i and P_{i+1} correct them, they must communicate the same number.

However, P_{i+1} corrects P_i, too. P_{i+1} assumes that P_i sees one more bit than himself, so P_{i+1} assumes that P_i has computed the correction–vectors for $P_1, P_2, ..., P_{i-1}$ as himself, but with a circular right–shift. So to correct P_i, P_{i+1} should subtract $\mu^{(i)}\Pi$ from $\mu^{(i)}$:

$$\mu^{(i+1)} = \mu^{(i)} - \mu^{(i)}\Pi = \mu^{(1)}(I - \Pi)^i.$$

We have got that

$$\mu = \sum_{i=1}^{k} \mu^{(i)} = \mu^{(1)}((I - \Pi)^0 + (I - \Pi)^1 + ... + (I - \Pi)^{k-1}).$$

Using that $\mu^{(1)} = \nu\Pi$,

$$\mu = \nu\Pi((I-\Pi)^0 + (I-\Pi)^1 + ... + (I-\Pi)^{k-1}) \tag{3}$$

Multiplying both sides of (3) from right by $(I-\Pi) - I = -\Pi$:

$$-\mu\Pi = \nu\Pi((I-\Pi)^k - I),$$

since Π commutes with its powers,

$$-\mu\Pi = \nu((I-\Pi)^k - I)\Pi. \tag{4}$$

Multiplying both sides of (4) with $-\Pi^{-1}$, from right:

$$\mu = \nu - \nu(I-\Pi)^k,$$

and this equation proves the theorem. □

Lemma 17.

$$\delta^{(m)}(A) = \gamma - CT(\mathbf{0}, A)\theta,$$

where $\theta = (\theta_0, \theta_1, ..., \theta_{m-1})$, *and*

$$\theta_j = \sum_{\substack{0 \le i \le k \\ i \equiv j \pmod m}} (-1)^i \binom{k}{i}.$$

Proof. From the binomial theorem,

$$(I-\Pi)^k = \binom{k}{0} I - \binom{k}{1}\Pi + \binom{k}{2}\Pi^2 - ... + (-1)^k \binom{k}{k}\Pi^k.$$

Since $\Pi^m = I$, we can write

$$(I-\Pi)^k = \sum_{\ell=0}^{m-1} \Pi^\ell \left(\sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m}} (-1)^i \binom{k}{i} \right), \tag{5}$$

It is easy to see, if a matrix is multiplied by ν from the left, the result is the first row of the matrix. When a row-vector is multiplied by Π the effect is the circular right–shift of the coordinates; this also holds for the first rows of the powers of Π: the first row of I is $1, 0, ..., 0$, the first row of Π is $0, 1, 0, ..., 0$, the first row of Π^2 is $0, 0, 1, 0, ..., 0$,..., the first row of Π^{m-1} is $0, ..., 0, 1$.

From (5) we got:

$$\nu(I-\Pi)^k = (\theta_0, \theta_1, ..., \theta_{m-1}) = \theta, \tag{6}$$

where

$$\theta_j = \sum_{\substack{0 \le i \le k \\ i \equiv j \pmod m}} (-1)^i \binom{k}{i}.$$

Lemma 16 together with (6) imply Lemma 17.□

Now we are ready to prove Theorems 7, 9, 10 and Corollaries 8 and 11:
PROOF OF THEOREM 7. By Lemma 17,

$$\theta_\ell = \sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m}} (-1)^i \binom{k}{i} = \sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m \\ i \text{ even}}} \binom{k}{i} - \sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m \\ i \text{ odd}}} \binom{k}{i} =$$

$$= \sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m \\ i \text{ even}}} \binom{k}{i} - \sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m \\ i \text{ odd}}} \binom{k}{k-i} =$$

$$= \sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m \\ i \text{ even}}} \binom{k}{i} - \sum_{\substack{0 \le j \le k \\ j \equiv \ell \pmod m \\ j \text{ even}}} \binom{k}{j} = 0,$$

since k is odd, and $k - i \equiv \ell \pmod m$.

So, $\gamma_\ell = \delta_\ell^{(m)}(A)$, and since protocol **MOD m** computes γ in **ME**, we are done.□

PROOF OF THEOREM 9. We may assume that $0 \le \ell \le m-1$. From Lemma 17,

$$\delta_\ell^{(m)} = \gamma_\ell - CT(\mathbf{0}, A)\theta_\ell, \tag{7}$$

and

$$\theta_\ell = \sum_{\substack{0 \le i \le k \\ i \equiv \ell \pmod m}} (-1)^i \binom{k}{i} \ne 0$$

since every summand is of the same sign. k players, who compute $\delta_\ell^{(m)}$ with communicating c bits can compute $CT(\mathbf{0}, A)$ with communicating $c+$

$O(km \log n)$ bits, using protocol **MOD m**, and equation (7). However, Theorem 5 shows (interchanging the roles of bits 1 and 0 in its proof), that computing $CT(\mathbf{0}, A)$ needs $\Omega(n/4^k)$ bits to communicate, and since any player can compute θ without any communication, we are done.□

PROOF OF COROLLARY 8. As in the proof of Theorem 9, we need to prove that $\theta_1 \neq 0$. Since

$$\theta_1 = -\binom{k}{1} \neq 0,$$

we are done.□

Let $A \in \{0,1\}^{n\times k}$. A row of A is called *even*, if it is divisible by 2. Theorem 9 shows, that the number of even rows of A is in **MH**.

PROOF OF THEOREM 10. Protocol **MOD m**, with $m = 2$, computes vector

$$\gamma = \delta^{(2)}(A) + CT(\mathbf{0}, A)\left(\sum_{\substack{0\le i\le k\\ i \text{ even}}} \binom{k}{i}, -\sum_{\substack{0\le i\le k\\ i \text{ odd}}} \binom{k}{i}\right) =$$

$$= \delta^{(2)}(A) + CT(\mathbf{0}, A)(2^{k-1}, -2^{k-1}).$$

The first coordinate of γ is congruent to $\delta_0^{(2)} \pmod{2^{k-1}}$, and this proves the statement. □

PROOF OF COROLLARY 11. Let $m = k$ and $\ell = 0$ in Theorem 7.□

3. Circuits with MOD m gates

DEFINITION 18. *Let $\mathcal{C}^*$ be a family of depth–2 circuits $C^*_{n,k}$, where n and k are positive integers, m is odd and positive, and $k \equiv m \pmod{2m}$ is also satisfied. Moreover*

- *the input of $C^*_{n,k}$ is A for $A \in \{0,1\}^{n\times k}$,*
- *on the bottom level (level 0) situated the variables A^i_j, with their negations;*
- *on the top (level 2), there is a symmetric gate,*

◦ *there are MOD_m gates of fan–in k on the first level.*

THEOREM 19. *Suppose that members of the circuit family $\mathcal{C}^*$ computes GIP(A). Then the size of $C^*_{n,k}$ is exponential in n.*

PROOF. Let us consider circuit $C^*_{n,k}$, computing GIP(A), $A \in \{0,1\}^{n\times k}$, and k players, such that player i knows every column of A, except column i, for $i = 1, 2, ..., k$, and suppose that all the players know circuit $C^*_{n,k}$. On the top of the circuit there is a symmetric gate, and the output of that gate depends only on the number of MOD_m gates, evaluated to 1, on level 1.

Players will collectively compute the number of MOD_m gates, evaluated to 1. Every MOD_m gate has at most k input wires. Let us call a MOD_m gate *easy*, if it has no input from a column of A. The easy gates can be evaluated as follows (Håstad & Goldmann (1991)): Suppose that an easy gate has no input from A_i, then P_i knows every variable of it, so he knows its output. Before the computation, the players agree in a scheme, which partitions the easy gates between the players, who know their inputs. These players simply communicate the numbers of those easy–gates in their classes, which are evaluated to 1. This can be done with $O(k \log N)$ bits of communication, where N is the size of circuit $C^*_{n,k}$.

Next, the players evaluate the non–easy gates. To do this, first they – individually, without any communication – build a matrix B. B has k columns, and each row of it corresponds to one of the non–easy MOD_m gates of the circuit; suppose that row B^i corresponds to a MOD_m gate G, and G has k input–variables, one from each column of A. Let B^i_j be equal to the input of G in A_j.
Note, that player j knows all the columns of B, except column j, B_j. Let us observe that B^i is divisible by m exactly when G is evaluated to 1. Since the size of $C^*_{n,k}$ is N, B has at most N rows. From Theorem 7, protocol **MOD m** computes the number of rows B, divisible by m, with communicating

$$O(mk \log N)$$

bits. To compute the number of easy–gates, evaluated to 1, the players used $O(k \log N)$ bits, so $O(mk \log N)$ bits in total. Theorem 5 shows that to compute GIP(A) the players should communicate

$$\Omega\left(\frac{n}{4^k}\right)$$

bits, so

$$O(mk\log N) = \Omega\left(\frac{n}{4^k}\right)$$

or

$$N \geq \exp\left(\Omega\left(\frac{n}{4^k mk}\right)\right).$$

□

THEOREM 20. *The family $\mathcal{C}^{**}$ of depth–2 circuits $C^{**}_{n,k}$ cannot compute GIP(A) for all $A \in \{0,1\}^{n\times k}$, where $k = p^c$ for some prime p and positive integer c. $C^{**}_{n,k}$ has a MOD p gate on the top and EXACT$_\ell$ gates of fan–in–k on the first level, where $1 \leq \ell \leq k-1$, and variables A_i^j with their negations on level 0.*

PROOF. Let us consider circuit $C^{**}_{n,k}$, computing GIP(A), $A \in \{0,1\}^{n\times k}$, and k players, such that player i knows every column of A, except column i, for $i = 1,2,...,k$, and suppose that all the players know circuit $C^{**}_{n,k}$. On the top of the circuit there is a MOD p, and the output of that gate depends only on the number of the EXACT$_\ell$ gates, evaluated to 1, on level 1.

Players will collectively compute the number of EXACT$_\ell$ gates, evaluated to 1. Every EXACT$_\ell$ gate has at most k input wires. Let us call an EXACT$_\ell$ gate *easy*, if it has no input from a column of A. The easy gates can be evaluated exactly as in the proof of Theorem 19, with $O(k\log p)$ bits of communication, since the number of the EXACT$_\ell$ gates, evaluated to 1, is needed only mod p.

Next, the players evaluate the non–easy gates. To do this, first they – individually, without any communication – build a matrix B. B has k columns, and each row of it corresponds to one of the non–easy EXACT$_\ell$ gates of the circuit; suppose that row B^i corresponds to an EXACT$_\ell$ gate G, and G has k input–variables, one from each column of A. Let B_j^i be equal to the input of G in A_j.

Note, that player j knows all the columns of B, except column j, B_j. Let us observe that B^i is congruent to ℓ mod k, exactly when G is evaluated to 1. Let the players play the **MOD m** protocol with $m = k = p^c$. Then for any ℓ, for $1 \leq \ell \leq k-1$, the error, made by the protocol is 0 mod p, since from Lemma 17:

$$\theta_\ell = (-1)^\ell \binom{p^c}{\ell} \equiv 0 \pmod{p}.$$

To compute the number of easy–gates, evaluated to 1, the players used $O(k\log p)$ bits, the **MOD m** protocol used so $O(k^2\log p)$ bits, since it is enough to communicate every number mod p only. Theorem 5 shows that to compute GIP(A) the players should communicate

$$\Omega\left(\frac{n}{4^k}\right)$$

bits, but the players can evaluate circuit $C^{**}_{n,k}$ with constant number of bits in n, so we have got that circuits in class $\mathcal{C}^{**}$ cannot compute GIP(A) at all. □

With standard techniques of Hajnal *et al.* (1987) and Håstad & Goldmann (1991), we can generalize Theorem 19 and Theorem 20, getting Theorems 1 and 2.

Proof of Theorem 1. If $C_{n,k}$ of size N computes GIP(A) then – by Lemma 2 of Håstad & Goldmann (1991) or Lemma 3.3. of Hajnal *et al.* (1987) – at least one of the depth–2 subcircuits computes GIP(A) or 1–GIP(A) correctly on at least

$$\frac{1}{2}+\frac{1}{2N}$$

fraction of the inputs. Theorem 19 shows that the output of that depth–2 subcircuit can be computed with $O(mk \log N)$ communication. From Theorem 5, with $\varepsilon = 1/2N$:

$$O(m_i k \log N) = \Omega\left(\frac{n}{4^k} - \log N\right),$$

and this completes the proof.□

Proof of Theorem 2. As in the proof of Theorem 1, at least one of the depth– 2 subcircuits of $C'_{n,k}$ computes GIP(A) or 1–GIP(A) correctly on at least

$$\frac{1}{2}+\frac{1}{2N}$$

fraction of the inputs. From Theorem 20, the players communicate

$$O(k^2 \log p)$$

bits for evaluating this circuit, while, from Theorem 5,

$$\Omega\left(\frac{n}{4^k} - \log N\right)$$

bits is needed for this, and the statement follows. □

Acknowledgements

Part of this work was done while visiting Max Planck Institute for Computer Science in Saarbrücken, Germany. The author acknowledges the support of grants OTKA F014919, and T017580.

References

ERIC ALLENDER AND VIVEK GORE, A uniform circuit lower bound for the permanent. *SIAM Journal on Computing* **23**(5) (1994), 1026–1049.

LÁSZLÓ BABAI, NOAM NISAN, AND MÁRIÓ SZEGEDY, Multiparty protocols, pseudorandom generators for logspace, and time–space trade-offs. *Journal of Computer and System Sciences* **45** (1992), 204–232.

DAVID A. MIX BARRINGTON, Bounded-width polynomial size branching programs recognize exactly those languages in NC^1. In *Proc. 18th ACM STOC*, 1986, 1–5. Appeared also in *JCSS*, Vol. 38. (1989).

RICHARD BEIGEL AND JUN TARUI, On ACC. In *Proc. 32nd IEEE FOCS*, 1991, 783–792.

ASHOK K. CHANDRA, MERRICK L. FURST, AND RICHARD J. LIPTON, Multi-party protocols. In *Proc. 15th ACM STOC*, 1983, 94–99.

VINCE GROLMUSZ, The BNS lower bound for multi–party protocols is nearly optimal. *Information and Computation* **112**(1) (1994), 51–54.

ANDRÁS HAJNAL, WOLFGANG MAASS, PAVEL PUDLAK, MÁRIÓ SZEGEDY, AND GYÖRGY TURÁN, Threshold circuits of bounded depth. In *Proc. 28th IEEE FOCS*, 1987, 99–110. Appeared also in *JCSS* Vol. 46, 1993.

JOHAN HÅSTAD, Almost optimal lower bounds for small depth circuits. In *Proc. 18th ACM STOC*, 1986, 6–20.

JOHAN HÅSTAD AND MIKAEL GOLDMANN, On the power of the small-depth threshold circuits. *Computational Complexity* **1** (1991), 113–129.

MATTHIAS KRAUSE AND STEPHAN WAACK, Variation ranks of communication matrices and lower bounds for depth two circuits having symmetric gates with unbounded fan–in. In *Proc. 32nd IEEE FOCS*, 1991, 777–782.

LÁSZLÓ LOVÁSZ, Communication complexity: a survey. In *Paths, Flows, and VLSI-Layout*, ed. B. KORTE, L. LOVÁSZ, H.J. PRÖMEL, AND A. SCHRIJVER, 235–265. Springer, 1989.

ALEXANDER RAZBOROV, Lower bounds on the size of bounded depth networks over a complete basis with logical addition, (in Russian). *Mat. Zametki* **41** (1987), 598–607.

ROMAN SMOLENSKY, Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proc. 19th ACM STOC*, 1987, 77–82.

ANDREW C. YAO, Some complexity questions related to distributed computing. In *Proc. 11th ACM STOC*, 1979, 209–213.

ANDREW C. YAO, Separating the polynomial–time hierarchy by oracles. In *Proc. 26th IEEE FOCS*, 1985, 1–10.

Manuscript received 5 January 1994

VINCE GROLMUSZ
Department of Computer Science
Eötvös University
Múzem krt. 6-8.
H-1088 BUDAPEST
HUNGARY
grolmusz@cs.elte.hu