CSE 291: Communication Complexity, Winter 2019
Multi-party protocols

Shachar Lovett
March 22, 2019

1 Overview

We have studied two-party protocols so far. In this section we consider multi-party protocols with more than two players. Let $P_1, \ldots, P_k$ be the players, where we typically think of $k$ as constant or as growing slowly with the input length (for example $k = O(\log n)$).

Number in hand. The most natural extension of the two-player model is that each player $P_i$ holds an input $x_i \in X_i$, and their goal it to compute some function $f : X_1 \times \ldots \times X_k \to Z$. This is called the number-in-hand (NIH) model. This model is in many senses weaker than the two-player model, as the input is spread across more players. Proving lower bounds in this model often reduces to two-player lower bounds, and the usefulness of such lower bounds is quite limited.

Example 1.1. Let $N = 2^n$, and interpret inputs $x \in \{0,1\}^n$ as numbers in $\{1, \ldots, N\}$. Let $T \in [N]$. The exact-$T$ function is defined as

$$EXACT_T(x_1, \ldots, x_k) = \begin{cases} 1 & \text{if } x_1 + \ldots + x_k = T \\ 0 & \text{otherwise} \end{cases}$$

It is easy to see that for $T = N$, the NIH deterministic communication complexity of exact-$N$ is $\Omega(\log N) = \Omega(n)$, as we can embed the $n$-bit identity function inside it: for $x, y \in \{0,1\}^n$ set $x_1 = x, x_2 = N - y, x_3 = \ldots = x_k = 0$.

Number on the forehead. Lower bounds for multi-party protocols become much more useful when we allow for inputs to be somewhat shared between the players. An extreme case of this is the number-on-the-forehead (NOF) model, defined by Chandra, Furst and Lipton [3]. Let $P_1, \ldots, P_k$ be $k$ players and let $x_i \in X_i$ be inputs for $i = 1, \ldots, k$. We assume that the $i$-th player $P_i$ can see all the inputs except $x_i$ (the name comes from thinking of the inputs as cards, where the players cards are on their foreheads, so that they can see all cards except their own). We assume that the players broadcast their messages, so that
all the other players can see the message (this is also called the “blackboard model” in the literature). As we will see, lower bounds for this model are useful in applications, and both upper and lower bounds have surprising connections to Ramsey theory.

2 Deterministic protocols for NOF

Let \( f : X_1 \times \ldots \times X_k \to Z \) be a \( k \)-partite function. Let \( \pi \) be a deterministic protocol computing it. Given inputs \( x = (x_1, \ldots, x_k) \) we denote the transcript by \( \pi(x) \). The cost of \( \pi \) is the maximal length of a transcript. We denote by \( D(f) \) the minimal cost of a protocol computing \( f \). Some complicated-looking functions have surprisingly efficient protocols in this model, as the following example shows.

Example 2.1 (XOR-Majority). Let \( k = 3 \), and denote the inputs as \( x, y, z \in \{0, 1\}^n \) for simplicity of notation. Consider the following function:

\[
\text{XOR-MAJ}(x, y, z) = \sum_{i=1}^{n} \text{MAJ}(x_i, y_i, z_i) \mod 2.
\]

We claim that \( D(f) = O(1) \) in the NOF model. This follows since we can re-write the 3-bit majority function as

\[
\text{MAJ}(a, b, c) = ab + ac + bc \mod 2.
\]

Thus

\[
\text{XOR-MAJ}(x, y, z) = \sum_{i=1}^{n} x_i y_i + x_i z_i + y_i z_i \mod 2.
\]

Each of the terms \( \sum_{i=1}^{n} x_i y_i, \sum_{i=1}^{n} x_i z_i, \sum_{i=1}^{n} y_i z_i \) can be computed by one of the players given their input. Thus, each of them only needs to send the sum modulo 2 to compute the function; this protocol has cost 3.

Next, we move the studying general NOF protocols. We saw that rectangles are central to understanding two-player deterministic protocols. A similar role for deterministic NOF protocols is played by cylinder intersections.

2.1 Cylinder intersections

Definition 2.2 (Cylinders). Let \( X_1, \ldots, X_k \) be sets and \( i \in [k] \). A subset \( S \subseteq X_1 \times \ldots \times X_k \) is an \( i \)-cylinder if for any \( x, x' \in X_1 \times \ldots \times X_k \), if \( x_j = x'_j \) for all \( j \neq i \), then \( x \in S \iff x' \in S \). In other words, the indicator function \( 1_S(x_1, \ldots, x_k) \) does not depend on \( x_i \).

Note that if \( S, S' \) are \( i \)-cylinder intersections then so is \( S \cap S' \).
Definition 2.3 (Cylinder intersection). Let $X_1, \ldots, X_k$. A subset $S \subset X_1 \times \ldots \times X_k$ is a cylinder intersection if $S = S_1 \cap \ldots \cap S_k$ where $S_i$ is an i-cylinder. Equivalently, the indicator $1_S$ can be factored as

$$1_S(x_1, \ldots, x_k) = \prod_{i=1}^k f_i((x_j)_{j \neq i}).$$

That is, each $f_i$ does not depend on $x_i$.

The following claim shows why cylinder intersections arise in the study of NOF protocols.

Claim 2.4. Let $\pi$ be $k$-player deterministic NOF protocol. For any transcript $b$, the set of inputs $x = (x_1, \ldots, x_k)$ for which $\pi(x) = b$ is a cylinder intersection.

Proof. An input $x$ generates a transcript $b \in \{0, 1\}^c$ if $\pi(x)_1 = b_1, \pi(x)_2 = b_2, \ldots, \pi(x)_c = b_c$. Consider a specific bit $b_i$ in the transcript. Let $S_i$ denote the set of inputs for which, given that the first $i - 1$ bits in the transcript are $b_1, \ldots, b_{i-1}$, the next bit send will be $b_i$. Then

$$\pi^{-1}(b) = S_1 \cap \ldots \cap S_c.$$ 

We next analyze $S_i$. Given the assumption that the first $i - 1$ bits of the transcript are $b_1, \ldots, b_{i-1}$, the next player to speak is fixed, say it is $P_j$ for some $j = j(b_1, \ldots, b_{i-1})$. The next bit that $P_j$ sends is a function of his input, which means that it does not depend on $x_j$. Thus $S_i$ is a $j$-cylinder. We thus conclude that $\pi^{-1}(b)$ is a cylinder intersection.

Let $f : X_1 \times \ldots \times X_k \rightarrow Z$. Similar to the two-player setting, we define its associated order-$k$ tensor $M_f$ as

$$(M_f)_{x_1, \ldots, x_k} = f(x_1, \ldots, x_k).$$

Corollary 2.5. If $D(f) = c$ then $M_f$ can be partitioned into $2^c$ monochromatic cylinder intersections.

The following structural result on cylinder intersections will be useful later.

Definition 2.6 (Star). Let $x = (x_1, \ldots, x_k)$. A star centered at $x$ is a set of $k$ points $x^i = (x_1, \ldots, x_{i-1}, x'_i, x_{i+1}, \ldots, x_k)$ where $x'_i \neq x_i$, for $i = 1, \ldots, k$. We call $x^1, \ldots, x^k$ the star and $x$ the center. Note that the center is not part of the star.

Lemma 2.7. Let $S \subset X_1 \times \ldots \times X_k$. The following are equivalent:

1. $S$ is a cylinder intersection.
2. For any $x^1, \ldots, x^k \in S$ which are a star, its center also belongs to $S$.

Proof. (1) $\Rightarrow$ (2): Assume that $S$ is a cylinder intersection. As we saw, we can decompose the indicator of $S$ as

$$1_S(x_1, \ldots, x_k) = \prod_{i=1}^k f_i((x_j)_{j \neq i}),$$

3
where \( f_i \) does not depend on \( x_i \). Let \( x^1, \ldots, x^k \) be a star centered at \( x \) with \( x^1, \ldots, x^k \in S \). As \( x^i \in S \) we must have \( f_i(x^i) = 1 \). Since \( x^i, x \) only differ in the \( i \)-th coordinate, and since \( f_i \) does not depend on the \( i \)-coordinate of its inputs, we also have \( f_i(x) = 1 \). As this is true for all \( i \in [k] \) we have \( 1_S(x) = 1 \), namely \( x \in S \).

(2) \( \Rightarrow \) (1): For each \( i \in [k] \), let \( S_i \) be the set of points that would belong to \( S \) if we allow to change their \( i \)-th coordinate, namely

\[
S_i = \{ x : \exists x' \in S, x_j = x'_j \forall j \neq i \}.
\]

Observe that \( S_i \) is an \( i \)-cylinder and \( S \subseteq S_i \). Thus \( S \subseteq S_1 \cap \ldots \cap S_k \). We will show that also \( S \supseteq S_1 \cap \ldots \cap S_k \), and hence \( S = S_1 \cap \ldots \cap S_k \) is a cylinder intersection. To see that, let \( x \in S_i \), there exist \( x^i \in S \) such that \( x_j = x'_j \) for all \( j \neq i \). In other words, \( x^1, \ldots, x^k \in S \) is a star centered at \( x \). By our assumption this implies that \( x \in S \). \( \square \)

### 2.2 The complexity of exact-\( N \) and Ramsey theory

The deterministic NOF complexity of exact-\( N \) turns out to have surprising connections to Ramsey theory. To simplify notations we focus on the case of \( k = 3 \) players. We recall the setup: \( N = 2^n \); the inputs are \( x, y, z \in [N] \), where each player sees two out of the three inputs; the goal is to check whether \( x + y + z = N \).

We start with some definitions. Let \( A \) denote the inputs on which the protocol should accept, namely

\[
A = \{ (x, y, z) : x, y, z \in [N], x + y + z = N \}.
\]

We say that a set \( S \subseteq [N]^3 \) is star-free if it does not contain a star.

**Claim 2.8.** Let \( S \subseteq A \) be a cylinder intersection. Then \( S \) is star-free.

**Proof.** Assume there exists a star \( x^1, x^2, x^3 \in S \) centered as \( x \). As \( S \) is a cylinder intersection, Lemma 2.7 shows that also \( x \in S \), and in particular \( x \in A \). However this is impossible as \( x, x^1 \) agree in all coordinates except the first, and hence they cannot both be in \( A \). \( \square \)

A coloring \( \chi : A \to \{1, \ldots, C\} \) of \( A \) is star-free if each color class \( \chi^{-1}(c) \) does not contain a star. We denote by \( \xi(N) \) the minimal number of colors for which exists a star-free coloring of \( A \).

**Lemma 2.9.** \( \log \xi(N) \leq D(\text{EXACT}_N) \leq \log \xi(N) + O(1) \).

**Proof.** We shorthand \( f = \text{EXACT}_N \) for ease of notation.

We first prove the lower bound. Assume that \( D(f) = c \). Corollary 2.5 then shows that \( M_f \) can be partitioned into \( 2^c \) monochromatic cylinder intersections. In particular, \( A \) can be partitioned into at most \( 2^c \) cylinder intersections. Claim 2.8 gives that each of these cylinder intersections is star-free. Hence \( \xi(N) \leq 2^c \).

We next prove the upper bound. Assume there exists a coloring \( \chi : A \to [C] \) where each color class is star-free. We will use it to design a protocol for \( f \). Let \( x, y, z \in [N] \) be the inputs of the players. The first step is that each player checks if there is a possible input for
himself that would make the sum \( N \). That is, the first player checks if \( y + z < N \); the second player if \( x + z < N \); and the third player if \( x + y < N \). If for one of them the answer is no, then they reject and abort. This part can be implemented in \( O(1) \) deterministic communication.

So, we assume from now on that \( x + y, y + z, x + z < N \). Define

\[
x' = N - y - z, \quad y' = N - x - z, \quad z' = N - x - y
\]

and note that \((x', y, z), (x, y', z), (x, y, z') \in A\). The first player computes \( c_1 = \chi(x', y, z) \), the second player computes \( c_2 = \chi(x, y', z) \) and the third player computes \( c_3 = \chi(x, y, z') \). We claim that \( x + y + z = N \) iff \( c_1 = c_2 = c_3 \).

To see that, assume first that \( x + y + z = N \). In this case \( x' = x, y' = y, z' = z \) and hence \( c_1 = c_2 = c_3 \). Otherwise, assume \( x + y + z \neq N \). Assume towards a contradiction that \( c_1 = c_2 = c_3 = c \). Then we have a star \((x', y, z), (x, y', z), (x, y, z') \in \chi^{-1}(c)\), a contradiction to the assumption that the coloring \( \chi \) is star-free.

To conclude, we need to see how the players can efficiently check if \( c_1 = c_2 = c_3 \). This is simple: the first player sends \( c_1 \) using \( \log C \) bits, and then players two and three send one bit indicating if \( c_1 = c_i \) or not for \( i = 2, 3 \). The total cost of this protocol is thus \( \log C + O(1) \). \( \square \)

Next we prove lower and upper bounds on \( \xi(N) \), by relating it to more well known quantities in Ramsey theory. We will show that

\[
(\log \log N)^{\Omega(1)} \leq \xi(N) \leq 2^{O(\sqrt{\log N})}.
\]

Translating back to deterministic communication complexity, and recalling that \( N = 2^n \), gives

\[
\Omega(\log \log n) \leq D(\text{EXACT}_N) \leq O(\sqrt{n}).
\]

**Open problem 2.10.** What is the correct asymptotics of \( D(\text{EXACT}_N) \)?

### 2.2.1 Lower bound on \( \xi(N) \) via the corner problem

**Definition 2.11 (Corner).** A corner is a triple \((x, y), (x, y + \lambda), (x + \lambda, y) \in \mathbb{Z}^2 \) for some \( \lambda \neq 0 \).

Let \( \alpha(N) \) denote the smallest number of colors, such that \([N]^2\) can be colored with each color class not containing a corner. Shkredov \cite{shkredov} proved that

\[
\alpha(N) \geq (\log \log N)^{\Omega(1)}
\]

We use this to derive lower bounds on \( \xi(N) \).

**Lemma 2.12.** \( \xi(N) \geq \alpha(N/3) \).
Proof. Assume for simplicity that $N$ is divisible by 3. Assume there exists a star-free coloring $\chi : A \to [C]$. We will use it to give a coloring of $[N/3]^2$ with $C$ colors, that does not contain a monochromatic corner.

Given $(x, y) \in [N/3]^2$ let $\sigma(x, y) = (x, y, N - x - y) \in A$. Define the color of $(x, y)$ to be $\chi(\sigma(x, y))$. Assume towards a contradiction that there exists a monochromatic corner $(x, y), (x + \lambda, y), (x, y + \lambda)$. Then the following points in $A$ also have the same color:

$$(x, y, N - x - y), (x + \lambda, y, N - x - y - \lambda), (x, y + \lambda, N - x - y - \lambda).$$

This is however a star centered at $(x, y, N - x - y - \lambda)$, a contradiction.

2.2.2 Upper bound on $\xi(N)$ via the Van der Waerden numbers

We next prove an upper bound on $\xi(N)$ based on the Van der Waerden numbers. Let $W_3(N)$ be the smallest number of colors, such that $[N]$ can be colored with each color class not containing a 3-term arithmetic progression. The asymptotic of $W_3(N)$ has been extensively studied. For our application we need upper bounds. Behrend [2] proved in 1946 that

$$W_3(N) \leq 2^{O(\sqrt{\log N})},$$

and this has not been improved since (except for the unspecified constants in the $O(\cdot)$).

Lemma 2.13. $\xi(N) \leq W_3(3N)$.

Proof. Let $\phi$ be a coloring of $[3N]$ such that each color class does not contain a 3-term arithmetic progression. Consider the following coloring of $A$:

$$\chi(x, y, z) = \phi(x + 2y).$$

We will show that each color class of $\chi$ does not contain a star. Assume towards a contradiction that some star $(x', y, z), (x, y', z), (x, y, z')$ is in the same color class. Then

$$\phi(x' + 2y) = \phi(x + 2y') = \phi(x + 2y).$$

Note that as $(x', y, z), (x, y', z) \in A$ we have $x' + y = x + y' = N - z$, which implies that $y' - y = x' - x$. Let $a = x + 2y, d = x' - x$. Then

$$x + 2y = a, \quad x' + 2y = a + d, \quad x + 2y' = a + 2d.$$

Thus $\phi$ contains a monochromatic 3-term arithmetic progression, a contradiction.

2.2.3 An explicit protocol

We present a more direct protocol for EXACT$_T$ by [5], which is implicitly using the construction of Behrend.

We assume that $x, y, z, T \in \{0, \ldots, N - 1\}$, and the goal is to check if $x + y + z = T$. Assume that $N = m^d$ for $m, d$ to be optimized later, the main idea is to express the numbers in base $m$, and then to verify that the digits add up given the correct carry bits.
**Base-m representation.** For each $x \in [N]$ we denote by $\bar{x}$ its base-$m$ representation. That is, $\bar{x} = (x_0, \ldots, x_{d-1})$ where $x_i \in \{0, \ldots, m - 1\}$. Given $x, y, z \in [N]$, let $\bar{c} = \bar{c}(x, y, z) \in \{0, 1, 2\}^{d+1}$ denote the carry vector in the addition of $x, y, z$ in base $m$.

Formally, denote by $x^{(\ell)}$ the integer represented by the first $\ell$ digits of $x$, namely $x^{(\ell)} = \sum_{j=0}^{\ell-1} x_j m^j$. Note that $x^{(\ell)} \in \{0, \ldots, m^\ell - 1\}$. Then $c_0 = 0$ and for $\ell = 1, \ldots, d$, if we let $s^{(\ell)} = x^{(\ell)} + y^{(\ell)} + z^{(\ell)}$ then

$$c_\ell = \begin{cases} 0 & \text{if } s^{(\ell)} < m^\ell \\ 1 & \text{if } m^\ell \leq s^{(\ell)} < 2m^\ell \\ 2 & \text{if } 2m^\ell \leq s^{(\ell)} \end{cases}.$$

Assume for a minute that we know the carry vector $\bar{c}$. Then the problem of verifying whether $x + y + z = T$ is equivalent to verifying the vector equation $\bar{c} + \bar{x} + \bar{y} + \bar{z} = \bar{T}$ (formally, we encode vectors in $d+1$ coordinates, with the last coordinate of $\bar{x}, \bar{y}, \bar{z}, \bar{T}$ being 0). This motivates defining the vector-exact problem.

**Vector-exact problem** The vector-exact-$\bar{T}$ problem is to decide, given inputs $\bar{x}, \bar{y}, \bar{z} \in [m]^d$ and a target $\bar{T} \in [m]^d$, whether $\bar{x} + \bar{y} + \bar{z} = \bar{T}$. This problem turns out to have a surprisingly efficient solution in the NOF model.

**Lemma 2.14.** Let $\bar{x}, \bar{y}, \bar{z}, \bar{T} \in [m]^d$. The NOF deterministic communication complexity of the vector-exact-$\bar{T}$ problem is $O(\log md)$.

**Proof.** Let $\bar{x}' = \bar{T} - \bar{y} - \bar{z}, \bar{y}' = \bar{T} - \bar{x} - \bar{z}, \bar{z}' = \bar{T} - \bar{x} - \bar{y}$. The players should accept only if $\bar{x} = \bar{x}', \bar{y} = \bar{y}', \bar{z} = \bar{z}'$. Consider the following expressions:

$$\bar{a} = \bar{x}' + 2\bar{y} + 3\bar{z} = \bar{T} + \bar{y} + 2\bar{z}$$
$$\bar{b} = \bar{x} + 2\bar{y}' + 3\bar{z} = 2\bar{T} - \bar{x} + \bar{z}$$
$$\bar{c} = \bar{x} + 2\bar{y} + 3\bar{z}' = 3\bar{T} - 2\bar{x} - \bar{y}$$

The first observation is that $\bar{a} = \bar{b} = \bar{c}$ if and only if the players should accept. A second observation is that the points $\bar{a}, \bar{b}, \bar{c}$ lie on a line. To see that, let $\bar{\lambda} = \bar{T} - \bar{x} - \bar{y} - \bar{z}$. Then

$$\bar{b} = \bar{a} + \bar{\lambda}, \quad \bar{c} = \bar{a} + 2\bar{\lambda}.$$

The crux of the algorithm (which is also the crux of Behrend’s construction) is that if we additionally know that $\bar{a}, \bar{b}, \bar{c}$ are on a sphere (that is, $\|\bar{a}\|_2 = \|\bar{b}\|_2 = \|\bar{c}\|_2$), then the only way that they would be both on a line and on a sphere is if they are all equal.

To realize this in the protocol, the first player (who can compute $\bar{a}$) computes $\|\bar{a}\|_2^2$, the second player computes $\|\bar{b}\|_2^2$ and the third player computes $\|\bar{c}\|_2^2$. Note that $\|\bar{a}\|_2^2, \|\bar{b}\|_2^2, \|\bar{c}\|_2^2 = O(m^2d)$. They then exchange these values, and except only if they are all equal. This protocol has communication complexity $O(\log md)$.

We now use our observations so far to deduce an efficient protocol for the integer version of exact-$\bar{T}$.
Lemma 2.15. Let $x, y, z, T \in [N]$ where $N = m^d$. The NOF deterministic communication complexity of the exact-$T$ problem is $O(d + \log md)$.

Proof. The main idea is to compute the carry vector $c = c(x, y, z)$ assuming that $x + y + z = T$, and delegate the verification of that to a second stage which uses the vector-exact problem.

Let consider the first player, who knows $y, z$ but do not know $x$. His goal is to compute the carry. If he assumes that $x + y + z = T$ then he can compute $x$ and hence his “local” version of the carry vector $\overline{c}^1 \in \{0, 1, 2\}^{d+1}$. The other two players do the same, computing $\overline{c}^2, \overline{c}^3$. The players first verify that $\overline{c}^1 = \overline{c}^2 = \overline{c}^3$ and otherwise this reject. This part requires communication $O(d)$.

Assume now that $\overline{c}^1 = \overline{c}^2 = \overline{c}^3 = \overline{c}$. Then $x + y + z = T$ if and only if $\overline{c} + \overline{x} + \overline{y} + \overline{z} = \overline{T}$. The players can verify this using the vector-exact problem, which require by Lemma 2.14 communication of $O(\log (md))$.

Optimizing the parameters in Lemma 2.15 gives $d = O(\sqrt{\log n}, m = 2^{O(\sqrt{\log n})}$ which results in an NOF protocol with communication cost $O(\sqrt{\log n})$.

2.3 Application: lower bounds for branching programs

Branching programs are a non-uniform model of computation, which captures computations which query the input a bounded number of times, and are also bounded in space. Informally speaking, a branching program $\mathcal{B}$ has query access to an input $x \in \{0, 1\}^n$, and in addition it has $s$ bits of memory it can use and a counter $t$ for the current step. In every step, the program chooses some bit location $i \in [n]$ to query, based on the current counter $t$ and the memory contents, it queries some bit $x_i$, and updates the memory. At the end, it should output an answer.

Equivalently, we can model a branching program as a layered graph with at most $S = 2^s$ nodes per layer. For every node $v$ in all but the last layer, there is an associated variable index $i(v) \in [n]$, and two outgoing edges to nodes in the next layer, one corresponding to $x_{i(v)} = 0$ and one to $x_{i(v)} = 1$. There is a single node in the first layer (the starting node). Every input $x \in \{0, 1\}^n$ thus defines a unique path from the starting node to a node in the last layer, on which the output is given. The width of a branching program is the maximal number of nodes per layer (that is, $S$) and its length is the number of layers. Width corresponds to memory and length to runtime.

We would consider a very simple function: HAM-EXACT$_{n,w}: \{0, 1\}^n \to \{0, 1\}$ defined as

$$\text{HAM-EXACT}_{n,w}(x) = \begin{cases} 1 & \text{if } |x| = w \\ 0 & \text{otherwise} \end{cases}.$$  

There is a simple branching program with length $n$ and width $n + 1$, which computes $|x|$ by computing prefix sums, and then at the end checks if this equals $n/2$ or not. We show that if we require the length to be linear, then the space must be super-constant.

Theorem 2.16. HAM-EXACT$_{n,w}$ cannot be computed by branching programs of length $O(n)$ and width $O(1)$. 


Proof. Assume towards a contradiction, that there exists $c \geq 1$ for which HAM-EXACT$_{n,w}$ can be computed by a branching program of length $cn$ and width $c$ for all $n, w$. We will use this to show that EXACT$_N$ problem on $k$ players (for some $k = k(c)$) can be solved with deterministic NOF communication complexity $O(1)$. On the other hand, similar Ramsey results to the ones we saw for $k = 3$ show that this is false (however the bounds deteriorate very fast as a function of $k$).

Let $k = 2c^2$. Fix $N$, and let $n = 2kN, w = N$ and $B$ be a branching program of length $cn$ and width $c$ which computes HAM-EXACT$_{n,w}$. Partition $B$ into $k$ consecutive chunks $B_1, \ldots, B_k$, each of which with $cn/k$ consecutive layers. Note that each $B_i$ queries at most $(cn/k) \cdot c \leq n/2$ variables. Thus, we can find disjoint subsets $S_1, \ldots, S_k \subset [n]$ of inputs of size $|S_i| = n/2k = N$ such that $B_i$ does not query $S_i$. We fix all inputs outside $S_1 \cup \ldots \cup S_k$ to zero.

Let now $x_1, \ldots, x_k \in [N]$ be the input to the $k$-party exact-$N$ problem. We map them to an input $y \in \{0, 1\}^n$ for the branching program $B$ as follows. We encode each input $x_i$ as an assignment for the variables $\{y_j : j \in S_i\}$ such that $\sum_{j \in S_i} y_j = x_i$ in some canonical way. Thus $x_1 + \ldots + x_k = N$ iff $\sum y_i = N$. The main point is that the players can simulate running $B(y)$ efficiently. The first player simulates running $B_1(y)$ which he can do, as the inputs to $B_1$ do not depend on $x_1$: he then sends the memory state (which is $O(\log c) = O(1)$ bits) to the second player, which continues to simulate $B_2$, and so on. So the total communication is $O(k \log c) = O(1)$.

3 Randomized protocols

Randomized NOF protocols are defined analogously to the two-player setup, where we may consider public or private protocols or specific distributions. We adopt the definitions of $R(f), R^{\text{pub}}(f), D_{\mu}(f)$ which have the same meaning as in the two-party world.

As an example to the power of randomness in NOF protocols, exact-$N$ becomes much easier when randomness is allowed.

**Lemma 3.1.** EXACT$_N$ has a private randomness NOF protocol of cost $O(\log \log N)$ and a public randomness NOF protocol of cost $O(1)$.

**Proof.** The problem reduces to two-player equality: the first player computes $x' = T - x - y$ and needs to check if it equals to $x$, which the second player knows. □

We focus in this section on lower bound techniques, in particular an extension of the discrepancy method for NOF protocols.

3.1 Discrepancy

Recall the definition of discrepancy for two-party protocols. If $f : X \times Y \to \{-1, 1\}$ is a two-party boolean function, then the discrepancy of $f$, with respect to an input distribution
\( \mu \), is

\[
\text{disc}_\mu(f) = \max_R \left| \sum_{(x,y) \in R} \mu(x,y)f(x,y) \right|,
\]

where the maximum is over all rectangles \( R \subset X \times Y \). Discrepancy for NOF protocols is defined analogously, except that the role of rectangles is replaced by cylinder intersections.

**Definition 3.2 (Discrepancy for NOF protocols).** Let \( f : X_1 \times \ldots \times X_k \to \{-1,1\} \). The discrepancy of \( f \), with respect to an input distribution \( \mu \), is

\[
\text{disc}_\mu(f) = \max_C \left| \sum_{x \in C} \mu(x)f(x) \right|,
\]

where the maximum is over all cylinder intersections \( C \subset X_1 \times \ldots \times X_k \).

Discrepancy gives lower bounds on randomized protocols in the NOF model in exactly the same way it does for two-party protocols.

**Lemma 3.3.** Let \( f : X_1 \times \ldots \times X_k \to \{-1,1\} \), and let \( \mu \) be an input distribution. Assume that \( D_\mu(f) = c \). Then \( \text{disc}_\mu(f) \geq 2^{-c}/3 \).

**Proof.** Fix a deterministic NOF protocol \( \pi \) of cost \( c \) that computes \( f \) under input distribution \( \mu \) with error \( \varepsilon = 1/3 \). Let \( g : X_1 \times \ldots \times X_k \to \{-1,1\} \) be the function computed by \( \pi \). Then

\[
\left| \sum_x \mu(x)f(x)g(x) \right| \geq 1 - 2\varepsilon = 1/3.
\]

Corollary 2.5 shows that \( M_g \) can be partitioned into \( 2^c \) monochromatic cylinder intersections \( C_1, \ldots, C_{2^c} \). Let \( v_i \in \{-1,1\} \) be the value that \( \pi \) outputs on \( C_i \). Then

\[
\sum_x \mu(x)f(x)g(x) = \sum_{i=1}^{2^c} v_i \sum_{x \in C_i} \mu(x)f(x).
\]

By the triangle inequality, there must exist \( C_i \) for which

\[
\left| \sum_{x \in C_i} \mu(x)f(x) \right| \geq 2^{-c}/3.
\]

This implies that \( \text{disc}_\mu(f) \geq 2^{-c}/3 \). \( \Box \)
3.2 Cube norm

Cylinder intersections are complicated, and hence it is not clear how to use Lemma 3.3 directly. Babai, Nisan and Szegedy [1] found a related measure which allows to bound the discrepancy.

**Definition 3.4 (Cube norm).** Let \( f : X_1 \times \ldots \times X_k \to \{-1,1\} \). Its cube norm is \( r(f) \) defined as follows. Let \( x^0, x^1 \in X_1 \times \ldots \times X_k \) be uniformly chosen. For each \( b \in \{0,1\}^k \) define a “mixed input” \( x^b \in X_1 \times \ldots \times X_k \) as

\[
x^b = (x^b_1, x^b_2, \ldots, x^b_k).
\]

Define

\[
r(f)^{2k} = \mathbb{E}_{x^0, x^1} \left[ \prod_{b \in \{0,1\}^k} f(x^b) \right].
\]

For example, for \( k = 3 \) we have

\[
r(f)^8 = \mathbb{E}_{x^0, x^1} f(x^0_1, x^0_2, x^0_3) f(x^0_1, x^0_2, x^1_3) \cdots f(x^1_1, x^1_2, x^1_3).
\]

It turns out that if we extend the definition of \( r(f) \) to all real-valued functions, then it is a norm (namely, \( r(f + g) \leq r(f) + r(g) \)); however, we would not need this. The main reason we introduced it is that it serves as an upper bound on the discrepancy under the uniform distribution.

**Theorem 3.5.** Let \( f : X_1 \times \ldots \times X_k \to \{-1,1\} \) and let \( U \) be the uniform distribution over inputs. Then

\[
\text{disc}_U(f) \leq r(f).
\]

**Proof.** We prove for \( k = 3 \), the general theorem follows in the same way, except with more indexes. To simplify notations denote the inputs by \( x, y, z \). Let \( f : X \times Y \times Z \to \{-1,1\} \). Let \( C(x, y, z) = a(x, y)b(y, z)c(x, z) \) be a cylinder intersection which maximizes correlation with \( f \). Then

\[
\text{disc}_U(f) = \mathbb{E}_{x, y, z} f(x, y, z)a(x, y)b(y, z)c(x, z).
\]

The main idea is to apply the Cauchy-Schwartz inequality repeatedly. In general, if \( g(u), h(u, v) \) are functions which map to \([-1,1]\), then we can always bound

\[
(\mathbb{E}_{u,v} g(u) h(u, v))^2 \leq (\mathbb{E}_{u,v} g(u))^2 \cdot \mathbb{E}_{u} (\mathbb{E}_{v} h(u, v))^2 \leq \mathbb{E}_{u,v,v'} h(u, v)h(u, v').
\]

In our case, \( u \) will correspond to a carefully chosen subset of the variables that we want to isolate, and \( v \) will be the remaining variables.

First, we isolate \( x, y \) and obtain

\[
(\text{disc}_U(f))^2 = (\mathbb{E}_{x,y} a(x, y) \mathbb{E}_{z} f(x, y, z)b(y, z)c(x, z))^2
\]

\[
\leq (\mathbb{E}_{x,y} a(x, y))^2 \cdot \mathbb{E}_{x,y} (\mathbb{E}_{z} f(x, y, z)b(y, z)c(x, z))^2
\]

\[
\leq \mathbb{E}_{x,y,z,z'} f(x, y, z)f(x, y, z')b(y, z)b(y, z')c(x, z)c(x, z').
\]
Next, we isolate \( y, z, z' \) and similarly obtain

\[
(disc_U(f))^4 \leq (\mathbb{E}_{y,z,z'} b(y, z) b(y, z') \mathbb{E}_x f(x, y, z) f(x, y, z') c(x, z) c(x, z'))^2
\]

\[
\leq (\mathbb{E}_{y,z,z'} b(y, z) b(y, z'))^2 \cdot (\mathbb{E}_x f(x, y, z) f(x, y, z') c(x, z) c(x, z'))^2
\]

\[
\leq \mathbb{E}_{x,x',y,y',z,z'} f(x, y, z) f(x, y', z) f(x', y, z') f(x', y', z)
\]

\[
f(x, y, z') f(x, y', z') f(x', y, z') f(x', y', z').
\]

In the last stage, we isolate \( x, x', z, z' \) and obtain

\[
(disc_U(f))^8 \leq \mathbb{E}_{x,x',y,y',z,z'} f(x, y, z) f(x, y', z) f(x', y, z') f(x', y', z)
\]

\[
f(x, y, z') f(x, y', z') f(x', y, z') f(x', y', z').
\]

In other words,

\[
disc_U(f) \leq r(f).
\]

\[\square\]

### 3.3 Example: Generalized Inner-Product

Let \( x_1, \ldots, x_k \in \mathbb{F}_2^n \). The Generalized Inner-Product (GIP) function is defined as

\[
\text{GIP}_{k,n}(x_1, \ldots, x_k) = \sum_{i=1}^{n} \prod_{j=1}^{k} (x_i)_j.
\]

That is, if we view the inputs as subsets of \([n]\), then it is the parity of the size of their intersection.

**Theorem 3.6.** \( R(\text{GIP}_{k,n}) = \Omega(n/4^k) \).

**Proof.** We prove this by bounding the discrepancy of GIP under the uniform distribution, which in turn we do by studying its \( r \)-norm. Shorthand \( f = (-1)^{\text{GIP}_{k,n}} \). Let \( x^0, x^1 \in (\mathbb{F}_2^n)^k \) and let \( x^b \in (\mathbb{F}_2^n)^k \) for \( b \in \{0, 1\}^k \) be their mixed inputs. We need to compute

\[
B(x^0, x^1) = \prod_{b \in \{0, 1\}^k} f(x^b) = (-1)^{\sum_{b \in \{0, 1\}^k} \sum_{i \in [n]} \prod_{j \in [k]} (x_i^b)_j}.
\]

Using the fact that GIP is computed by a multi-linear polynomial, we can simplify the expression as

\[
B(x^0, x^1) = \prod_{b \in \{0, 1\}^k} f(x^b) = (-1)^{\sum_{i \in [n]} \prod_{j \in [k]} (x_i^0 + x_i^1)_j} = f(x^0 + x^1).
\]

Thus we get that

\[
r(f)^{2^k} = \mathbb{E}_{x^0, x^1} B(x^0, x^1) = \mathbb{E}_x f(x).
\]
So we reduced the problem to computing the expected value of GIP. We have

\[ \mathbb{E}_x f(x) = \mathbb{E}_x (-1) \sum_{i=1}^{n} \prod_{j=1}^{k} (x_{ij}) = \prod_{i=1}^{n} \mathbb{E}_{x_i} \left((-1) \prod_{j=1}^{k} (x_{ij})\right). \]

Each term equals to \(-1\) only if \((x_1)_j = \ldots = (x_k)_j\), which happens with probability \(2^{-k}\), and otherwise equals 1. Thus

\[ \mathbb{E}_x f(x) = \left(1 - 2^{-k}\right)^n \leq \exp(-n/2^k). \]

We conclude that \(r(f) \leq 2^{-O(n/4^k)}\), and hence \(\text{disc}_U(f) \leq 2^{-O(n/4^k)}\) and \(R(f) = \Omega(n/4^k)\). □

So we see that the quality of the lower bound deteriorates exponentially with the number of players. This seems like an artifact of the Cauchy-Schwartz argument. However, as we shall soon see, Groth and Durfee [4] showed that this is indeed the case for GIP and many similar functions.

### 3.4 The Groth and Durfee protocol

Let \(f : \{0,1\}^n \rightarrow \{0,1\}\) be a symmetric function, namely where \(f(x)\) depends only on the hamming weight (number of ones) of \(x\). We consider the \(k\)-party function \(f(x_1 \land \ldots \land x_k)\). For example, GIP is such a function, with \(f\) being the PARITY function.

**Theorem 3.7.** Let \(x_1, \ldots, x_k \in \{0,1\}^n\). There is a deterministic NOF protocol which computes the hamming weight of \(x_1 \land \ldots \land x_k\) with cost \(O(k \log n \cdot (n/2^k))\). In particular, the player can compute the value of \(f(x_1 \land \ldots \land x_k)\) for any symmetric function \(f\).

The proof of Theorem 3.7 relies on the following lemma. For it, it will be convenient to consider the input as a \(k \times n\) matrix.

**Lemma 3.8.** Let \(x_1, \ldots, x_k \in \{0,1\}^n\). Assume that the players know of a pattern \(r \in \{0,1\}^k\) that does not appear as a column in the input matrix. Then there is a deterministic NOF protocol which computes the hamming weight of \(x_1 \land \ldots \land x_k\) with cost \(O(k \log n)\).

**Proof.** Let us assume be re-arranging the players’ order that \(r = 0^\ell 1^{k-\ell}\) for some \(\ell = 0, \ldots, k\). Let \(E_a\) denote the number of input columns equal to \(0^a 1^{k-a}\). The players goal is to compute \(E_0\), which equals to the hamming weight of \(x_1 \land \ldots \land x_k\).

The main observation is that the \(i\)-th player can compute \(E_{i-1} + E_i\). This is since this equals the number of input columns equal to \(0^{i-1} 1^{k-i}\), which the player can compute as it does not depend on \(x_i\). Assume that all the players compute this and share this information, which requires communication \(O(k \log n)\). Then, the knowledge that \(E_\ell = 0\) together with the value of \(E_{i-1} + E_i\) for all \(i = 1, \ldots, k\) is sufficient to compute \(E_0\). □
Proof of Theorem 3.7. Partition the inputs into \( n/m \) chunks of size \( m = 2^{k-1} - 1 \). Let \( x_{i,j} \in \{0,1\}^m \) denote the \( j \)-th chunk of \( x_i \), then

\[
|x_1 \land \ldots \land x_k| = \sum_{j=1}^{n/m} |x_{1,j} \land \ldots \land x_{k,j}|.
\]

For each \( j \in [n/m] \) the players compute \( |x_{1,j} \land \ldots \land x_{k,j}| \). We would like to apply Lemma 3.8, for which we need to identify a pattern \( r \) which does not appear in the corresponding \( k \times 2^{k-1} \) input matrix. The first player, which sees all rows except the first row, sees \( m \) different patterns on the last \( k-1 \) rows, which translate into \( 2^m \) possible patterns on all the \( k \) rows. As \( 2m < 2^k \) there is a pattern that never appears there. He sends this pattern to the remaining players, which use it in the protocol given in Lemma 3.8.

The total cost is: \( n/m = O(n/2^k) \) rounds, in each of which the first player sends \( k \) bits (the forbidden pattern) followed by the protocol of Lemma 3.8 which has cost \( O(k \log n) \). The total communication cost is thus \( O(k \log n \cdot (n/2^k)) \).

\( \square \)

3.5 Set disjointness

There has been considerable research on the \( k \)-party set disjointness function, as this function comes up in various applications. Here, we think of the inputs as subsets \( x_1, \ldots, x_k \subset [n] \) and wish to compute

\[
\text{DISJ}(x_1, \ldots, x_k) = \begin{cases} 1 & \text{if } |x_1 \cap \ldots \cap x_k| = 0 \\ 0 & \text{otherwise} \end{cases}.
\]

The best known bounds are by Sherstov [6], and give a randomized lower bound of \( \Omega(n/4^k)^{1/4} \). The best upper bounds are \( O(kn/2^k) \) by a variant of Theorem 3.7.

**Open problem 3.9.** Show that the 3-player NOF randomized communication complexity of set disjointness is \( \Omega(n) \).

References


