Symposium on Theoretical Aspects of Computer Science 2008 (Bordeaux), pp. 145-156
www.stacs-conf.org

SUBLINEAR COMMUNICATION PROTOCOLS FOR MULTI-PARTY POINTER JUMPING AND A RELATED LOWER BOUND

JOSHUA BRODY [1] AND AMIT CHAKRABARTI [1]

[1] Department of Computer Science
Dartmouth College
Hanover, NH 03755, USA

Abstract. We study the one-way number-on-the-forehead (NOF) communication complexity of the k-layer pointer jumping problem with n vertices per layer. This classic problem, which has connections to many aspects of complexity theory, has seen a recent burst of research activity, seemingly preparing the ground for an $\Omega(n)$ lower bound, for constant k. Our first result is a surprising sublinear — i.e., $o(n)$ — upper bound for the problem that holds for $k \geq 3$, dashing hopes for such a lower bound.

A closer look at the protocol achieving the upper bound shows that all but one of the players involved are *collapsing*, i.e., their messages depend only on the composition of the layers ahead of them. We consider protocols for the pointer jumping problem where *all* players are collapsing. Our second result shows that a strong $n - O(\log n)$ lower bound does hold in this case. Our third result is another upper bound showing that nontrivial protocols for (a non-Boolean version of) pointer jumping are possible even when all players are collapsing.

Our lower bound result uses a novel proof technique, different from those of earlier lower bounds that had an information-theoretic flavor. We hope this is useful in further study of the problem.

1. Introduction

Multi-party communication complexity in general, and the *pointer jumping* problem (also known as the *pointer chasing* problem) in particular, has been the subject of plenty of recent research. This is because the model, and sometimes the specific problem, bears on several aspects of computational complexity: among them, circuit complexity [Yao90, HG91, BT94], proof size lower bounds [BPS05] and space lower bounds for streaming algorithms [AMS99, GM07, CJP08]. The most impressive known consequence of a strong

1998 ACM Subject Classification: F.1.3, F.2.2.

Key words and phrases: Communication complexity, pointer jumping, number on the forehead.

Work supported in part by an NSF CAREER Award CCF-0448277, NSF grants CCF-0514870 and EIA-98-02068. Work partly done while the authors were visiting the University of Washington, Seattle, WA.
.

multi-party communication lower bound would be to exhibit non-membership in the complexity class ACC^0; details can be found in Beigel and Tarui [BT94] or in the textbook by Arora and Barak [AB07]. Vexingly, it is not even known whether or not $\mathsf{ACC}^0 = \mathsf{NEXP}$.

The setting of multi-party communication is as follows. There are k players (for some $k \geq 2$), whom we shall call $\text{PLR}_1, \text{PLR}_2, \ldots, \text{PLR}_k$, who share an input k-tuple $(x_1, x_2, \ldots, x_k)$. The goal of the players is to compute some function $f(x_1, x_2, \ldots, x_k)$. There are two well-studied sharing models: the *number-in-hand* model, where PLR_i sees x_i, and the *number-on-the-forehead* (NOF) model, where PLR_i sees all x_js such that $j \neq i$. Our focus in this paper will be on the latter model, which was first introduced by Chandra, Furst and Lipton [CFL83]. It is in this model that communication lower bounds imply lower bounds against ACC^0. We shall use $C(f)$ to denote the deterministic communication complexity of f in this model. Also of interest are randomized protocols that only compute $f(x)$ correctly with high probability: we let $R_\varepsilon(f)$ denote the ε-error randomized communication complexity of f. Our work here will stick to deterministic protocols, which is a strength for our upper bounds. Moreover, it is not a serious weakness for our lower bound, because the ACC^0 connection only calls for a deterministic lower bound.

Notice that the NOF model has a feature not seen elsewhere in communication complexity: the players *share* plenty of information. In fact, for large k, each individual player already has "almost" all of the input. This intuitively makes lower bounds especially hard to prove and indeed, to this day, no nontrivial lower bound is known in the NOF model for any explicit function with $k = \omega(\log n)$ players, where n is the total input size. The pointer jumping problem is widely considered to be a good candidate for such a lower bound. As noted by Damm, Jukna and Sgall [DJS98], it has many natural special cases, such as shifting, addressing, multiplication and convolution. This motivates our study.

1.1. The Pointer Jumping Problem and Previous Results

There are a number of variants of the pointer jumping problem. Here we study two variants: a Boolean problem, MPJ_k^n, and a non-Boolean problem, $\widehat{\text{MPJ}}_k^n$ (henceforth, we shall drop the superscript n). In both variants, the input is a subgraph of a fixed layered graph that has $k+1$ layers of vertices, with layer 0 consisting of a single vertex, v_0, and layers 1 through $k-1$ consisting of n vertices each (we assume $k \geq 2$). Layer k consists of 2 vertices in the case of MPJ_k and n vertices in the case of $\widehat{\text{MPJ}}_k$. The input graph is a subgraph of the fixed layered graph in which every vertex (except those in layer k) has outdegree 1. The desired output is the name of the unique vertex in layer k reachable from v_0, i.e., the final result of "following the pointers" starting at v_0. The output is therefore a single bit in the case of MPJ_k or a $\lceil \log n \rceil$-bit string in the case of $\widehat{\text{MPJ}}_k$.[1]

The functions MPJ_k and $\widehat{\text{MPJ}}_k$ are made into NOF communication problems as follows: for each $i \in [k]$, a description of the ith layer of edges (i.e., the edges pointing into the ith layer of vertices) is written on PLR_i's forehead. In other words, PLR_i sees every layer of edges except the ith. The players are allowed to write one message each on a public *blackboard* and must do so in the fixed order $\text{PLR}_1, \text{PLR}_2, \ldots, \text{PLR}_k$. The final player's message must be the desired output. Notice that the specific order of speaking — $\text{PLR}_1, \text{PLR}_2, \ldots, \text{PLR}_k$ — is important to make the problem nontrivial. Any other order of speaking allows an easy deterministic protocol with only $O(\log n)$ communication.

[1] Throughout this paper we use "log" to denote logarithm to the base 2.

Consider the case $k = 2$. The problem MPJ_2 is equivalent to the two-party communication problem INDEX, where Alice holds a bit-vector $x \in \{0,1\}^n$, Bob holds an index $i \in [n]$, and Alice must send Bob a message that enables him to output x_i. It is easy to show that $C(\text{MPJ}_2) = n$. In fact, Ablayev [Abl96] shows the tight tradeoff $R_\varepsilon(\text{MPJ}_2) = (1 - H(\varepsilon))n$, where H is the binary entropy function. It is tempting to conjecture that this lower bound generalizes as follows.

Conjecture 1.1. There is a nondecreasing function $\xi : \mathbb{Z}^+ \to \mathbb{R}^+$ such that, $\forall\, k :\ C(\text{MPJ}_k) = \Omega(n/\xi(k))$.

Note that, by the results of Beigel and Tarui [BT94], in order to show that $\text{MPJ}_k \notin \mathsf{ACC}^0$ it would suffice, for instance, to prove the following (possibly weaker) conjecture.

Conjecture 1.2. There exist constants $\alpha, \beta > 0$ such that, for $k = n^\alpha$, $C(\text{MPJ}_k) = \Omega(n^\beta)$.

Conjecture 1.1 is consistent with (and to an extent motivated by) research prior to this work. In weaker models of information sharing than the NOF model, an equivalent statement is known to be true, even for randomized protocols. For instance, Damm, Jukna and Sgall [DJS98] show an $\Omega(n/k^2)$ communication lower bound in the so-called *conservative* model, where PLR_i has only a limited view of the layers of the graph behind her: she only sees the result of following the first $i-1$ pointers. Chakrabarti [Cha07] extends this bound to randomized protocols and also shows an $\Omega(n/k)$ lower bound in the so-called *myopic* model, where PLR_i has only a limited view of the layers ahead of her: she cannot see layers $i+2, \ldots, k$.

For the full NOF model, Wigderson, building on the work of Nisan and Wigderson [NW93], showed that $C(\text{MPJ}_3) = \Omega(\sqrt{n})$. This result is unpublished, but an exposition can be found in Babai, Hayes and Kimmel [BHK01]. Very recently, Viola and Wigderson [VW07] generalized this result and extended it to randomized protocols, showing that $R_{1/3}(\text{MPJ}_k) = \Omega(n^{1/(k-1)}/k^{O(k)})$. Of course, this bound falls far short of that in Conjecture 1.1 and does nothing for Conjecture 1.2. However, it is worth noting that the Viola-Wigderson bound in fact applies to the much smaller subproblem of *tree pointer jumping* (denoted TPJ_k), where the underlying layered graph is a height-k tree, with every vertex in layers 0 through $k-2$ having $n^{1/(k-1)}$ children and every vertex in layer $k-1$ having two children. It is easy to see that $C(\text{TPJ}_k) = O(n^{1/(k-1)})$. Thus, one might hope that the more general problem MPJ_k has a much stronger lower bound, as in Conjecture 1.1.

On the upper bound side, Damm et al. [DJS98] show that $C(\widehat{\text{MPJ}}_k) = O(n \log^{(k-1)} n)$, where $\log^{(i)} n$ is the ith iterated logarithm of n. This improves on the trivial upper bound of $O(n \log n)$. Their technique does not yield anything nontrivial for the Boolean problem MPJ_k, though. However, Pudlak, Rödl and Sgall [PRS97] obtain a sublinear upper bound of $O(n \log\log n / \log n)$ for a special case of MPJ_3. Their protocol works only when every vertex in layer 2 has *indegree* 1, or equivalently, when the middle layer of edges in the input describes a *permutation* of $[n]$.

1.2. Our Results

The protocol of Pudlak et al. [PRS97] did not rule out Conjecture 1.1, but it did suggest caution. Our first result is the following upper bound — in fact the first nontrivial upper bound on $C(\text{MPJ}_k)$ — that falsifies the conjecture.

Theorem 1.3. *For $k \geq 3$, we have*

$$C(\text{MPJ}_k) \;=\; O\left(n\left(\frac{k\log\log n}{\log n}\right)^{(k-2)/(k-1)}\right).$$

In particular, $C(\text{MPJ}_3) = O(n\sqrt{\log\log n/\log n})$.

A closer look at the protocol that achieves the upper bound above reveals that all players except for PLR_1 behave in the following way: the message sent by PLR_i depends only on layers 1 through $i-1$ and the composition of layers $i+1$ through k. We say that PLR_i is *collapsing*. This notion is akin to that of the aforementioned conservative protocols considered by Damm et al. Whereas a conservative player composes the layers behind hers, a collapsing player does so for layers ahead of hers.

We consider what happens if we require *all* players in the protocol to be collapsing. We prove a strong linear lower bound, showing that even a single non-collapsing player makes an asymptotic difference in the communication complexity.

Theorem 1.4. *In a protocol for MPJ_k where every player is collapsing, some player must communicate at least $n - \frac{1}{2}\log n - 2 = n - O(\log n)$ bits.*

Finally, one might wonder whether the collapsing requirement is so strong that nothing nontrivial is possible anyway. The same question can be raised for the conservative and myopic models where $\Omega(n/k^2)$ and $\Omega(n/k)$ lower bounds were proven in past work. It turns out that the upper bound on $C(\widehat{\text{MPJ}}_k)$ due to Damm et al. [DJS98] (see Section 1.1) is achievable by a protocol that is both conservative and myopic. We can show a similar upper bound via a different protocol where every player is collapsing.

Theorem 1.5. *For $k \geq 3$, there is an $O(n\log^{(k-1)} n)$-communication protocol for $\widehat{\text{MPJ}}_k^{\text{perm}}$ in which every player is collapsing. Here $\widehat{\text{MPJ}}_k^{\text{perm}}$ denotes the subproblem of $\widehat{\text{MPJ}}_k$ in which layers 2 through k of the input graph are permutations of $[n]$.*

The requirement that layers be permutations is a natural one and is not new. The protocol of Pudlak et al. also had this requirement; i.e., it gave an upper bound on $C(\text{MPJ}_3^{\text{perm}})$. Theorem 1.5 can in fact be strengthened slightly by allowing one of the layers from 2 through k to be arbitrary; we formulate and prove this stronger version in Section 4.

1.3. Organization

The rest of the paper is organized as follows. Theorems 1.3, 1.4 and 1.5 are proven in Sections 2, 3 and 4 respectively. Section 2.1 introduces some notation that is used in subsequent sections.

2. A Sublinear Upper Bound

2.1. Preliminaries, Notation and Overall Plan

For the rest of the paper, "protocols" will be assumed to be deterministic one-way NOF protocols unless otherwise qualified. We shall use $\text{cost}(P)$ to denote the total number of bits communicated in P, for a worst case input.

Let us formally define the problems MPJ_k and $\widehat{\text{MPJ}}_k$. We shall typically write the input k-tuple for MPJ_k as $(i, f_2, \ldots, f_{k-1}, x)$ and that for $\widehat{\text{MPJ}}_k$ as $(i, f_2, \ldots, f_k)$, where $i \in [n]$, each $f_j \in [n]^{[n]}$ and $x \in \{0,1\}^n$. We then define $\text{MPJ}_k : [n] \times \left([n]^{[n]}\right)^{k-2} \times \{0,1\}^n \to \{0,1\}$ and $\widehat{\text{MPJ}}_k : [n] \times \left([n]^{[n]}\right)^{k-1} \to [n]$ as follows.

$$\text{MPJ}_2(i,x) := x_i\,; \qquad \text{MPJ}_k(i, f_2, f_3, \ldots, f_{k-1}, x) := \text{MPJ}_{k-1}(f_2(i), f_3, \ldots, f_{k-1}, x)\,, \text{ for } k \geq 3$$

$$\widehat{\text{MPJ}}_2(i,f) := f(i)\,; \qquad \widehat{\text{MPJ}}_k(i, f_2, f_3, \ldots, f_k) := \widehat{\text{MPJ}}_{k-1}(f_2(i), f_3, \ldots, f_k)\,, \text{ for } k \geq 3\,.$$

Here, x_i denotes the ith bit of the string x. It will be helpful, at times, to view strings in $\{0,1\}^n$ as functions from $[n]$ to $\{0,1\}$ and use functional notation accordingly. It is often useful to discuss the composition of certain subsets of the inputs. Let $\hat{i}_2 := i$, and for $3 \leq j \leq k$, let $\hat{i}_j := f_{j-1} \circ \cdots \circ f_2(i)$. Similarly, let $\hat{x}_{k-1} := x$, and for $1 \leq j \leq k-2$, let $\hat{x}_j := x \circ f_{k-1} \circ \cdots \circ f_{j+1}$. Unrolling the recursion in the definitions, we see that, for $k \geq 2$,

$$\text{MPJ}_k(i, f_2, \ldots, f_{k-1}, x) \;=\; x \circ f_{k-1} \circ \cdots \circ f_2(i) \;=\; \hat{x}_1(i) \;=\; x_{\hat{i}_k}\,; \tag{2.1}$$

$$\widehat{\text{MPJ}}_k(i, f_2, \ldots, f_k) \;=\; f_k \circ \cdots \circ f_2(i) \;=\; f_k(\hat{i}_k)\,. \tag{2.2}$$

We also consider the subproblems $\text{MPJ}_k^{\text{perm}}$ and $\widehat{\text{MPJ}}_k^{\text{perm}}$ where each f_j above is a bijection from $[n]$ to $[n]$ (equivalently, a permutation of $[n]$). We let $\mathcal{S}_n$ denote the set of all permutations of $[n]$.

Here is a rough plan of the proof of our sublinear upper bound. We leverage the fact that a protocol P for $\text{MPJ}_3^{\text{perm}}$ with sublinear communication is known. To be precise:

Fact 2.1 (Pudlak, Rödl and Sgall [PRS97, Corollary 4.8])**.** $C(\text{MPJ}_3^{\text{perm}}) = O(n \log\log n / \log n)$.

The exact structure of P will not matter; we shall only use P as a black box. To get a sense for why P might be useful for, say, MPJ_3, note that the players could replace f_2 with a permutation π and just simulate P, and this would work if $\pi(i) = f(i)$. Of course, there is no way for PLR_1 and PLR_3 to agree on a suitable π without communication. However, as we shall see below, it is possible for them to agree on a small enough *set* of permutations such that either some permutation in the set is suitable, or else only a small amount of side information conveys the desired output bit to PLR_3.

This idea eventually gives us a sublinear protocol for MPJ_3. Clearly, whatever upper bound we obtain for MPJ_3 applies to MPJ_k for all $k \geq 3$. However, we can decrease the upper bound as k increases, by embedding several instances of MPJ_3 into MPJ_k. For clarity, we first give a complete proof of Theorem 1.3 for the case $k = 3$.

2.2. A 3-Player Protocol

Following the plan outlined above, we prove Theorem 1.3 for the case $k = 3$ by plugging Fact 2.1 into the following lemma, whose proof is the topic of this section.

Lemma 2.2. *Suppose $\phi : \mathbb{Z}^+ \to (0,1]$ is a function such that $C(\text{MPJ}_3^{\text{perm}}) = O(n\phi(n))$. Then $C(\text{MPJ}_3) = O(n\sqrt{\phi(n)})$.*

Definition 2.3. A set $\mathcal{A} \subseteq \mathcal{S}_n$ of permutations is said to d-cover a function $f : [n] \to [n]$ if, for each $r \in [n]$, at least one of the following conditions holds:

(i) $\exists\, \pi \in \mathcal{A}$ such that $\pi(r) = f(r)$, or
(ii) $|f^{-1}(f(r))| > d$.

Lemma 2.4. *Let $f : [n] \to [n]$ be a function and d be a positive integer. There exists a set $\mathcal{A}_d(f) \subseteq \mathcal{S}_n$, with $|\mathcal{A}_d(f)| \leq d$, that d-covers f.*

Proof. We give an explicit algorithm to construct $\mathcal{A}_d(f)$. Our strategy is to partition the domain and codomain of f (both of which equal $[n]$) into parts of matching sizes and then define bijections between the corresponding parts. To be precise, suppose $\text{Range}(f) = \{s_1, s_2, \ldots, s_t\}$. Let $A_i = f^{-1}(s_i)$ be the corresponding fibers of f. Clearly, $\{A_i\}_{i=1}^t$ is a partition of $[n]$. It is also clear that there exists a partition $\{B_i\}_{i=1}^t$ of $[n]$ such that, for all $i \in [t]$, $B_i \cap \text{Range}(f) = \{s_i\}$ and $|B_i| = |A_i|$. We shall now define certain bijections $\pi_{i,\ell} : A_i \to B_i$, for each $i \in [t]$ and $\ell \in [d]$.

Let $a_{i,1} < a_{i,2} < \cdots < a_{i,|A_i|}$ be the elements of A_i arranged in ascending order. Similarly, let $b_{i,1} < \cdots < b_{i,|B_i|}$ be those of B_i. We define

$$\pi_{i,\ell}(a_{i,j}) \;:=\; b_{i,(j-\ell) \bmod |B_i|}\,, \quad \text{for } i \in [t], \ell \in [d]\,,$$

where, for convenience, we require "α mod β" to return values in $[\beta]$, rather than $\{0, 1, \ldots, \beta - 1\}$. It is routine to verify that $\pi_{i,\ell}$ is a bijection. Notice that this construction ensures that for all $i \in [t]$ and $j \in [|A_i|]$ we have

$$|\{\pi_{i,\ell}(a_{i,j}) : \ell \in [d]\}| \;=\; \min\{d, |B_i|\}\,. \tag{2.3}$$

Let $\pi_\ell : [n] \to [n]$ be the bijection given by taking the "disjoint union" of $\pi_{1,\ell}, \ldots, \pi_{t,\ell}$. We claim that $\mathcal{A}_d(f) = \{\pi_1, \ldots, \pi_d\}$ satisfies the conditions of the lemma.

It suffices to verify that this choice of $\mathcal{A}_d(f)$ d-covers f, i.e., to verify that every $r \in [n]$ satisfies at least one of the two conditions in Definition 2.3. Pick any $r \in [n]$. Suppose $r \in A_i$, so that $f(r) \in B_i$ and $\pi_\ell(r) = \pi_{i,\ell}(r)$. If $|B_i| > d$, then $|f^{-1}(f(r))| = |A_i| = |B_i| > d$, so condition (ii) holds. Otherwise, from Eq. (2.3), we conclude that $\{\pi_{i,\ell}(r) : \ell \in [d]\} = B_i$. Therefore, for each $s \in B_i$ — in particular, for $s = f(r)$ — there exists an $\ell \in [d]$ such that $\pi_\ell(r) = \pi_{i,\ell}(r) = s$, so condition (i) holds. ∎

Proof of Lemma 2.2. Let $(i, \pi, x) \in [n] \times \mathcal{S}_n \times \{0,1\}^n$ denote an input for the problem $\text{MPJ}_3^{\text{perm}}$. Then the desired output is $x_{\pi(i)}$. The existence of a protocol P for $\text{MPJ}_3^{\text{perm}}$ with $\text{cost}(P) = O(n\phi(n))$ means that there exist functions

$$\alpha : \mathcal{S}_n \times \{0,1\}^n \to \{0,1\}^m\,,\ \beta : [n] \times \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^m\,,\ \text{and}$$
$$\gamma : [n] \times \mathcal{S}_n \times \{0,1\}^m \times \{0,1\}^m \to \{0,1\}\,,$$

where $m = O(n\phi(n))$, such that $\gamma(i, \pi, \alpha(\pi, x), \beta(i, x, \alpha(\pi, x))) = x_{\pi(i)}$. The functions α, β and γ yield the messages in P of $\text{PLR}_1, \text{PLR}_2$ and PLR_3 respectively.

To design a protocol for MPJ_3, we first let PLR_1 and PLR_3 agree on a parameter d, to be fixed below, and a choice of $\mathcal{A}_d(f)$ for each $f : [n] \to [n]$, as guaranteed by Lemma 2.4. Now, let $(i, f, x) \in [n] \times [n]^{[n]} \times \{0,1\}^n$ be an input for MPJ_3. Our protocol works as follows.

- PLR_1 sends a two-part message. The first part consists of the strings $\{\alpha(\pi, x)\}_\pi$ for all $\pi \in \mathcal{A}_d(f)$. The second part consists of the bits x_s for $s \in [n]$ such that $|f^{-1}(s)| > d$.
- PLR_2 sends the strings $\{\beta(i, x, \alpha)\}_\alpha$ for all strings α in the first part of PLR_1's message.
- PLR_3 can now output $x_{f(i)}$ as follows. If $|f^{-1}(f(i))| > d$, then she reads $x_{f(i)}$ off from the second part of PLR_1's message. Otherwise, since $\mathcal{A}_d(f)$ d-covers f, there exists a $\pi_0 \in \mathcal{A}_d(f)$ such that $f(i) = \pi_0(i)$. She uses the string $\alpha_0 := \alpha(\pi_0, x)$ from

the first part of PLR_1's message and the string $\beta_0 := \beta(i, x, \alpha_0)$ from PLR_2's message to output $\gamma(i, \pi_0, \alpha_0, \beta_0)$.

To verify correctness, we only need to check that PLR_3's output in the "otherwise" case indeed equals $x_{f(i)}$. By the correctness of P, the output equals $x_{\pi_0(i)}$ and we are done, since $f(i) = \pi_0(i)$.

We now turn to the communication cost of the protocol. By the guarantees in Lemma 2.4, $|\mathcal{A}_d(f)| \leq d$, so the first part of PLR_1's message is at most dm bits long, as is PLR_2's message. Since there can be at most n/d values $s \in [n]$ such that $|f^{-1}(s)| > d$, the second part of PLR_2's message is at most n/d bits long. Therefore the communication cost is at most $2dm + n/d = O(dn\phi(n) + n/d)$. Setting $d = \lceil 1/\sqrt{\phi(n)} \rceil$ gives us a bound of $O(n\sqrt{\phi(n)})$, as desired. ∎

2.3. A k-Player Protocol

We now show how to prove Theorem 1.3 by generalizing the protocol from Lemma 2.2 into a protocol for k players. It will help to view an instance of MPJ_k as incorporating several "embedded" instances of MPJ_3. The following lemma makes this precise.

Lemma 2.5. *Let $(i, f_2, \ldots, f_{k-1}, x)$ be input for MPJ_k. Then, for all $1 < j < k$,*

$$\text{MPJ}_k(i, f_2, \ldots, x) = \text{MPJ}_3(f_{j-1} \circ \cdots \circ f_2(i), f_j, x \circ f_{k-1} \circ \cdots \circ f_{j+1}).$$

In our protocol for MPJ_k, for $2 \leq j \leq k-1$, the players $\text{PLR}_1, \text{PLR}_j$, and PLR_k will use a modified version of the protocol from Lemma 2.2 for MPJ_3 on input $(f_{j-1} \circ \cdots \circ f_2(i), f_j, x \circ \cdots \circ f_{j+1})$. Before we get to the protocol, we need to generalize the technical definition and lemma from the previous subsection.

Definition 2.6. Let $S \subseteq [n]$ and let d be a positive integer. A set $\mathcal{A} \subseteq \mathcal{S}_n$ of permutations is said to (S, d)-cover a function $f : [n] \to [n]$ if, for each $r \in S$, at least one of the following conditions holds:

(i) $\exists\, \pi \in \mathcal{A}$ such that $\pi(r) = f(r)$, or
(ii) $|S \cap f^{-1}(f(r))| > d$.

Lemma 2.7. *Let $f : [n] \to [n]$ be a function, $S \subseteq [n]$, and d be a positive integer. There exists a set $\mathcal{A}_{S,d}(f) \subseteq \mathcal{S}_n$, with $|\mathcal{A}_{S,d}(f)| \leq d$, that (S, d)-covers f.*

Proof. This proof closely follows that of Lemma 2.4. As before, we give an explicit algorithm to construct $\mathcal{A}_{S,d}(f)$. Suppose $\text{Range}(f) = \{s_1, s_2, \ldots, s_t\}$, and let $\{A_i\}$ and $\{B_i\}$ be defined as in Lemma 2.4. Let $a_{i,1} < \cdots < a_{i,z}$ be the elements of $A_i \cap S$ arranged in ascending order, and let $a_{i,z+1} < \cdots < a_{i,|A_i|}$ be the elements of $A_i \setminus S$ arranged in ascending order. Similarly, let $b_{i,1} < \cdots < b_{i,|B_i|-1}$ be the elements of $B_i \setminus \{s_i\}$ arranged in ascending order, and let $b_{i,|B_i|} = s_i$. For $i \in [t], \ell \in [d]$, we define $\pi_{i,\ell}(a_{i,j}) := b_{i,(j-\ell) \bmod |B_i|}$. As before, it is routine to verify that $\pi_{i,\ell}$ is a bijection. Let $\pi_\ell : [n] \to [n]$ be the bijection given by taking the "disjoint union" of $\pi_{1,\ell}, \ldots, \pi_{t,\ell}$. We claim that $\mathcal{A}_{S,d}(f) = \{\pi_1, \ldots, \pi_d\}$ satisfies the conditions of the lemma.

It suffices to verify that this choice of $\mathcal{A}_{S,d}(f)$ (S, d)-covers f, i.e., to verify that every $r \in S$ satisfies at least one of the two conditions in Definition 2.6. Pick any $r \in S$. Suppose $r \in A_i$, and fix j such that $r = a_{i,j}$. If $|S \cap f^{-1}(f(r))| > d$, then condition (ii) holds. Otherwise, setting $\ell = j < |S \cap f^{-1}(f(i))| \leq d$, we conclude that $\pi_\ell(r) = \pi_{i,\ell}(r) = \pi_{i,\ell}(a_{i,j}) = b_{i,|B_i|} = s_i = f(r)$, so condition (i) holds. ∎

Proof of Theorem 1.3. To design a protocol for MPJ_k, we first let PLR_1 and PLR_k agree on a parameter d, to be fixed below. They also agree on a choice of $\mathcal{A}_{S,d}(f)$ for all $S \subseteq [n]$ and $f : [n] \to [n]$. Let $(i, f_2, \ldots, f_{k-1}, x)$ denote an input for MPJ_k. Also, let $S_1 = [n]$, and for all $2 \le j \le k-1$, let $S_j = \{s \in [n] : |S_{j-1} \cap f_j^{-1}(s)| > d\}$. Our protocol works as follows:

- PLR_1 sends a $(k-1)$-part message. For $1 \le j \le k-2$, the jth part of PLR_1's message consists of the strings $\{\alpha(\pi, \hat{x}_{j+1})\}_\pi$ for each $\pi \in \mathcal{A}_{S_j,d}(f_{j+1})$. The remaining part consists of the bits x_s for $s \in S_{k-1}$.
- For $2 \le j \le k-1$, PLR_j sends the strings $\{\beta(\hat{i}_j, \hat{x}_j, \alpha)\}_\alpha$ for all strings α in the $(j-1)$th part of PLR_1's message.
- PLR_k can now output $x_{\hat{i}_k}$ as follows. If $|S_1 \cap f_2^{-1}(f_2(i))| \le d$, then, because $\mathcal{A}_{S_1,d}(f_2)$ (S_1, d)-covers f_2, there exists $\pi_0 \in \mathcal{A}_{S_1,d}(f_2)$ such that $f_2(i) = \pi_0(i)$. She uses the string $\alpha_0 = \alpha(\pi_0, \hat{x}_2)$ from the first part of PLR_1's message and the string $\beta_0 = \beta(i, \hat{x}_2, \alpha_0)$ from PLR_2's message to output $\gamma_0 = \gamma(i, \pi_0, \alpha_0, \beta_0)$. Similarly, if there is a j such that $2 \le j \le k-2$ and $|S_j \cap f_{j+1}^{-1}(f_{j+1}(\hat{i}_{j+1}))| \le d$, then since $\mathcal{A}_{S_j,d}(f_{j+1})$ (S_j, d)-covers f_{j+1}, there exists a $\pi_0 \in \mathcal{A}_{S_j,d}(f_{j+1})$ such that $f_{j+1}(\hat{i}_{j+1}) = \pi_0(\hat{i}_{j+1})$. She uses the string $\alpha_0 = \alpha(\pi_0, \hat{x}_{j+1})$ from the jth part of PLR_1's message and the string $\beta_0 = \beta(\hat{i}_{j+1}, \hat{x}_{j+1}, \alpha_0)$ from PLR_{j+1}'s message to output $\gamma_0 = \gamma(\hat{i}_{j+1}, \pi_0, \alpha_0, \beta_0)$. Otherwise, $|S_{k-2} \cap f_{k-1}^{-1}(f_{k-1}(\hat{i}_{k-1}))| > d$, hence $\hat{i}_k \in S_{k-1}$, and she reads $x_{\hat{i}_k}$ off from the last part of PLR_1's message.

To verify correctness, we need to ensure that PLR_k always outputs $x \circ f_{k-1} \circ \cdots \circ f_2(i)$. In the following argument, we repeatedly use Lemma 2.5. We proceed inductively. If $|S_1 \cap f_2^{-1}(f_2(i))| \le d$ then there exists $\pi_0 \in \mathcal{A}_{S_1,d}(f_2)$ such that $f_2(i) = \pi_0(i)$, $\alpha_0 = \alpha(\pi_0, \hat{x}_2)$, and $\beta_0 = \beta(i, \hat{x}_2, \alpha_0)$, and PLR_k outputs $\gamma_0 = \gamma(i, \pi_0, \alpha_0, \beta_0) = \hat{x}_2(\pi_0(i)) = x \circ f_{k-1} \circ \cdots \circ f_2(i)$. Otherwise, $|S_1 \cap f_2^{-1}(f_2(i))| > d$, hence $f_2(i) \in S_2$. Inductively, if $\hat{i}_j \in S_{j-1}$, then either $|S_{j-1} \cap f_j^{-1}(f_j(\hat{i}_j))| \le d$, or $|S_{j-1} \cap f_j^{-1}(f_j(\hat{i}_j))| > d$. In the former case, there is $\pi_0 \in \mathcal{A}_{S_{j-1},d}(f_j)$ such that $f_j(\hat{i}_j) = \pi_0(\hat{i}_j)$; $\alpha_0(\pi_0, \hat{x}_j)$, and $\beta_0 = \beta(\hat{i}_j, \hat{x}_j, \alpha_0)$, and PLR_k outputs $\gamma_0 = \gamma(\hat{i}_j, \pi_0, \alpha_0, \beta_0) = \hat{x}_j(f_j(\hat{i}_j)) = x \circ f_{k-1} \circ \cdots \circ f_2(i)$. In the latter case, $f_j(\hat{i}_j) \in S_j$. By induction, we have that either PLR_k outputs $x \circ f_{k-1} \circ \cdots \circ f_2(i)$, or $\hat{i}_k \in S_{k-1}$. But in this case, PLR_k outputs $x(\hat{i}_k) = x \circ f_{k-1} \circ \cdots \circ f_2(i)$ directly from the last part of PLR_1's message. Therefore, PLR_k always outputs $x \circ f_{k-1} \circ \cdots \circ f_2(i)$ correctly.

We now turn to the communication cost of the protocol. By Lemma 2.7, $|\mathcal{A}_{S_j,d}(f_j)| \le d$ for each $2 \le j \le k-1$, hence the first $k-2$ parts of PLR_1's message each are at most dm bits long, as is PLR_j's message for all $2 \le j \le k-1$. Also, since for all $2 \le j \le k-1$, there are at most $|S_{j-1}|/d$ elements $s \in S_j$ such that $|S_{j-1} \cap f_j^{-1}(s)| > d$, we must have that $|S_2| \le |S_1|/d = n/d, |S_3| \le |S_2|/d \le n/d^2$, etc., and $|S_{k-1}| \le n/d^{k-2}$. Therefore, the final part of PLR_1's message is at most n/d^{k-2} bits long, and the total communication cost is at most $2(k-2)dm + n/d^{k-2} = O((k-2)dn\phi(n) + n/d^{k-2})$. Setting $d = \lceil 1/((k-2)\phi(n))^{1/(k-1)} \rceil$ gives us a bound of $O(n(k\phi(n))^{(k-2)/(k-1)})$ as desired. ∎

Note that, in the above protocol, except for the first and last players, the remaining players access very limited information about their input. Specifically, for all $2 \le j \le k-1$, PLR_j needs to see only $\hat{i}_j$ and $\hat{x}_j$, i.e., PLR_j is both *conservative* and *collapsing*. Despite this severe restriction, we have a sublinear protocol for MPJ_k. As we shall see in the next section, further restricting the input such that PLR_1 is also collapsing yields very strong lower bounds.

3. Collapsing Protocols: A Lower Bound

Let $F : \mathscr{A}_1 \times \mathscr{A}_2 \times \cdots \times \mathscr{A}_k \to \mathscr{B}$ be a k-player NOF communication problem and P be a protocol for F. We say that PLR_j is *collapsing* in P if her message depends only on $x_1, \ldots, x_{j-1}$ and the function $g_{x,j} : \mathscr{A}_1 \times \mathscr{A}_2 \times \cdots \times \mathscr{A}_j \to \mathscr{B}$ given by $g_{x,j}(z_1, \ldots, z_j) = F(z_1, \ldots, z_j, x_{j+1}, \ldots, x_k)$. For pointer jumping, this amounts to saying that PLR_j sees all layers $1, \ldots, j-1$ of edges (i.e., the layers preceding the one on her forehead), but not layers $j+1, \ldots, k$; however, she does see the result of following the pointers from each vertex in layer j. Still more precisely, if the input to MPJ_k (or $\widehat{\text{MPJ}}_k$) is $(i, f_2, \ldots, f_k)$, then the only information PLR_j gets is $i, f_2, \ldots, f_{j-1}$ and the composition $f_k \circ f_{k-1} \circ \cdots \circ f_{j+1}$.

We say that a protocol is collapsing if every player involved is collapsing. We shall prove Theorem 1.4 by contradiction. Assume that there is a collapsing protocol P for MPJ_k in which every player sends less than $n - \frac{1}{2}\log n - 2$ bits. We shall construct a pair of inputs that differ only in the last layer (i.e., the Boolean string on PLR_k's forehead) and that cause players 1 through $k-1$ to send the exact same sequence of messages. This will cause PLR_k to give the same output for both these inputs. But our construction will ensure that the desired outputs are unequal, a contradiction. To aid our construction, we need some definitions and preliminary lemmas.

Definition 3.1. A string $x \in \{0,1\}^n$ is said to be *consistent* with $(f_1, \ldots, f_j, \alpha_1, \ldots, \alpha_j)$ if, in protocol P, for all $h \le j$, PLR_h sends the message α_h on seeing input $(i = f_1, \ldots, f_{h-1}, x \circ f_j \circ f_{j-1} \circ \cdots \circ f_{h+1})$ and previous messages $\alpha_1, \ldots, \alpha_{h-1}$.[2] A subset $T \subseteq \{0,1\}^n$ is said to be consistent with $(f_1, \ldots, f_j, \alpha_1, \ldots, \alpha_j)$ if x is consistent with $(f_1, \ldots, f_j, \alpha_1, \ldots, \alpha_j)$ for all $x \in T$.

Definition 3.2. For strings $x, x' \in \{0,1\}^n$ and $a, b \in \{0,1\}$, define the sets

$$I_{ab}(x, x') \;:=\; \{j \in [n] : (x_j, x'_j) = (a, b)\}\,.$$

A pair of strings (x, x') is said to be a *crossing pair* if for all $a, b \in \{0,1\}$, $I_{ab}(x, x') \neq \emptyset$. A set $T \subseteq \{0,1\}^n$ is said to be *crossed* if it contains a crossing pair and *uncrossed* otherwise. The *weight* of a string $x \in \{0,1\}^n$ is defined to be the number of 1s in x, and denoted $|x|$.

For the rest of this section, we assume (without loss of generality) that n is large enough and even.

Lemma 3.3. *If $T \subseteq \{0,1\}^n$ is uncrossed, then $|\{x \in T : |x| = n/2\}| \le 2$.*

Proof. Let x and x' be distinct elements of T with $|x| = |x'| = n/2$. For $a, b \in \{0,1\}$, define $t_{ab} = |I_{ab}(x, x')|$. Since $x \neq x'$, we must have $t_{01} + t_{10} > 0$. An easy counting argument shows that $t_{01} = t_{10}$ and $t_{00} = t_{11}$. Since T is uncrossed, (x, x') is not a crossing pair, so at least one of the numbers t_{ab} must be zero. It follows that $t_{00} = t_{11} = 0$, so x and x' are bitwise complements of each other. Since this holds for any two strings in $\{x \in T : |x| = n/2\}$, that set can have size at most 2. ∎

Lemma 3.4. *Suppose $t \le n - \frac{1}{2}\log n - 2$. If $\{0,1\}^n$ is partitioned into 2^t disjoint sets, then one of those sets must be crossed.*

[2] It is worth noting that, in Definition 3.1, x is not to be thought of as an input on PLR_k's forehead. Instead, in general, it is the composition of the rightmost $k - j$ layers of the input graph.

Proof. Let $\{0,1\}^n = T_1 \sqcup T_2 \sqcup \cdots \sqcup T_m$ be a partition of $\{0,1\}^n$ into m uncrossed sets. Define $X := \{x \in \{0,1\}^n : |x| = n/2\}$. Then $X = \bigcup_{i=1}^m (T_i \cap X)$. By Lemma 3.3,

$$|X| \;\leq\; \sum_{i=1}^{m} |T_i \cap X| \;\leq\; 2m\,.$$

Using Stirling's approximation, we can bound $|X| > 2^n/(2\sqrt{n})$. Therefore, $m > 2^{n-\frac{1}{2}\log n - 2}$. ∎

Proof of Theorem 1.4. Set $t = n - \frac{1}{2}\log n - 2$. Recall that we have assumed that there is a collapsing protocol P for MPJ_k in which every player sends at most t bits. We shall prove the following statement by induction on j, for $j \in [k-1]$.

(*) There exists a partial input $(i = f_1, f_2, \ldots, f_j) \in [n] \times \left([n]^{[n]}\right)^{j-1}$, a sequence of messages $(\alpha_1, \ldots, \alpha_j)$ and a crossing pair of strings $(x, x') \in (\{0,1\}^n)^2$ such that both x and x' are consistent with $(f_1, \ldots, f_j, \alpha_1, \ldots, \alpha_j)$, whereas $x \circ f_j \circ \cdots \circ f_2(i) = 0$ and $x' \circ f_j \circ \cdots \circ f_2(i) = 1$.

Considering (*) for $j = k-1$, we see that PLR_k must behave identically on the two inputs $(i, f_2, \ldots, f_{k-1}, x)$ and $(i, f_2, \ldots, f_{k-1}, x')$. Therefore, she must err on one of these two inputs. This will give us the desired contradiction.

To prove (*) for $j = 1$, note that PLR_1's message, being at most t bits long, partitions $\{0,1\}^n$ into at most 2^t disjoint sets. By Lemma 3.4, one of these sets, say T, must be crossed. Let (x, x') be a crossing pair in T and let α_1 be the message that PLR_1 sends on seeing a string in T. Fix $i = f_1$ such that $i \in I_{01}(x, x')$. These choices are easily seen to satisfy the conditions in (*). Now, suppose (*) holds for a particular $j \geq 1$. Fix the partial input $(f_1, \ldots, f_j)$ and the message sequence $(\alpha_1, \ldots, \alpha_j)$ as given by (*). We shall come up with appropriate choices for f_{j+1}, α_{j+1} and a new crossing pair (y, y') to replace (x, x'), so that (*) is satisfied for $j+1$. Since PLR_{j+1} sends at most t bits, she partitions $\{0,1\}^n$ into at most 2^t subsets (the partition might depend on the choice of $(f_1, \ldots, f_j, \alpha_1, \ldots, \alpha_j)$).

As above, by Lemma 3.4, she sends a message α_{j+1} on some crossing pair (y, y'). Choose f_{j+1} so that it maps $I_{ab}(x, x')$ to $I_{ab}(y, y')$ for all $a, b \in \{0,1\}$; this is possible because $I_{ab}(y, y') \neq \emptyset$. Then, for all $i \in [n]$, $x_i = y_{f_{j+1}(i)}$ and $x'_i = y'_{f_{j+1}(i)}$. Hence, $x = y \circ f_{j+1}$ and $x' = y' \circ f_{j+1}$. Applying the inductive hypothesis and the definition of consistency, it is straightforward to verify the conditions of (*) with these choices for f_{j+1}, α_{j+1}, y and y'. This completes the proof. ∎

4. Collapsing Protocols: An Upper Bound

We now turn to proving Theorem 1.5 by constructing an appropriate collapsing protocol for $\widehat{\text{MPJ}}_k^{\text{perm}}$. Our protocol uses what we call *bucketing schemes*, which have the flavor of the conservative protocol of Damm et al. [DJS98]. For any function $f \in [n]^{[n]}$ and any $S \subseteq [n]$, let $\mathbf{1}_S$ denote the indicator function for S; that is, $\mathbf{1}_S(i) = 1 \Leftrightarrow i \in S$. Also, let $f|_S$ denote the function f restricted to S; this can be seen as a list of numbers $\{i_s\}$, one for each $s \in S$. Players will often need to send $\mathbf{1}_S$ and $f|_S$ together in a single message. This is because later players might not know S, and will therefore be unable to interpret $f|_S$ without $\mathbf{1}_S$. Let $\langle m_1, \ldots, m_t \rangle$ denote the concatenation of messages $m_1, \ldots, m_t$.

Definition 4.1. A *bucketing scheme* on a set X is an ordered partition $\mathcal{B} = (B_1, \ldots, B_t)$ of X into *buckets*. For $x \in X$, we write $\mathcal{B}[x]$ to denote the unique integer j such that $B_j \ni x$.

We actually prove our upper bound for problems slightly more general than $\widehat{\text{MPJ}}_k^{\text{perm}}$. To be precise, for an instance $(i, f_2, \ldots, f_k)$ of $\widehat{\text{MPJ}}_k$, we allow any one of $f_2, \ldots, f_k$ to be an arbitrary function in $[n]^{[n]}$. The rest of the f_js are required to be permutations, i.e., in $\mathcal{S}_n$.

Theorem 4.2 (Slight generalization of Theorem 1.5)**.** *There is an $O(n \log^{(k-1)} n)$ collapsing protocol for instance $(i, f_2, \ldots, f_k)$ of $\widehat{\text{MPJ}}_k$ when all but one of $f_2, \ldots, f_k$ are permutations. In particular, there is such a protocol for $\widehat{\text{MPJ}}_k^{\text{perm}}$.*

Proof. We prove this for $\widehat{\text{MPJ}}_k^{\text{perm}}$ only. For $1 \le t \le \lceil \log n \rceil$, define the bucketing scheme $\mathcal{B}_t = (B_1, \ldots, B_{2^t})$ on $[n]$ by $B_j := \{r \in [n] : \lceil 2^t r/n \rceil = j\}$. Note that each $|B_j| \le \lceil n/2^t \rceil$ and that a bucket can be described using t bits. For $1 \le j \le k$, let $b_j = \lceil \log^{(k-j)} n \rceil$. In the protocol, most players will use two bucketing schemes, $\mathcal{B}$ and $\mathcal{B}'$. On input $(i, f_2, \ldots, f_k)$:

- PLR_1 sees $\hat{f}_1$, computes $\mathcal{B}' := \mathcal{B}_{b_1}$, and sends $\langle \mathcal{B}'[\hat{f}_1(1)], \ldots, \mathcal{B}'[\hat{f}_1(n)] \rangle$.
- PLR_2 sees $\hat{i}_2, \hat{f}_2$, and PLR_1's message. PLR_2 computes $\mathcal{B} := \mathcal{B}_{b_1}$ and $\mathcal{B}' := \mathcal{B}_{b_2}$. She recovers $b := \mathcal{B}[\hat{f}_2(f_2(\hat{i}_2))]$ and hence B_b. Let $S_2 := \{s \in [n] : \hat{f}_2(s) \in B_b\}$. Note that $f_2(\hat{i}_2) \in S_2$. PLR_2 sends $\langle \mathbf{1}_{S_2}, \{\mathcal{B}'[\hat{f}_2(s)] : s \in S_2\} \rangle$.

  $\vdots$
- PLR_j sees $\hat{i}_j, \hat{f}_j$, and PLR_{j-1}'s message. PLR_j computes $\mathcal{B} := \mathcal{B}_{b_{j-1}}$ and $\mathcal{B}' := \mathcal{B}_{b_j}$. She recovers $b := \mathcal{B}[\hat{f}_j(f_j(\hat{i}_j))]$ and hence B_b. Let $S_j := \{s \in [n] : \hat{f}_j(s) \in B_b\}$. Note that the definitions guarantee that $f_j(\hat{i}_j) \in S_j$. PLR_j sends $\langle \mathbf{1}_{S_j}, \{\mathcal{B}'[\hat{f}_j(s)] : s \in S_j\} \rangle$.

  $\vdots$
- PLR_k sees $\hat{i}_k$ and PLR_{k-1}'s message and outputs $f_k(\hat{i}_k)$.

We claim that this protocol costs $O(n \log^{(k-1)} n)$ and correctly outputs $\widehat{\text{MPJ}}_k(i, f_2, \ldots, f_k)$. For each $2 \le j \le k-1$, PLR_j uses bucketing scheme $\mathcal{B}_{b_{j-1}}$ to recover the bucket B_b containing $\hat{f}_j(f_j(\hat{i}_j))$. She then encodes each element in B_b in the bucketing scheme $\mathcal{B}_{b_j}$. Each bucket in $\mathcal{B}_{b_j}$ has size at most $\lceil n/b_{j+1} \rceil$. In particular, each bucket in scheme $\mathcal{B}_{k-1}$ has size at most $\lceil n/b_k \rceil = 1$, and the unique element in the bucket (if present) is precisely $f_k(\hat{i}_k)$. Turning to the communication cost, PLR_1 sends $b_1 = \lceil \log^{(k-1)} n \rceil$ bits to identify the bucket for each $i \in [n]$, giving a total of $n\lceil \log^{(k-1)} n \rceil$ bits. For $1 < j < k$, PLR_j uses $n + b_j(n/b_j) = O(n)$ bits. Thus, the total cost is $O(n \log^{(k-1)} n + kn)$ bits.

For $k \le \log^* n$ players, we are done. For larger k, we can get an $O(n)$ protocol by doubling the size of each b_j and stopping the protocol when the buckets have size ≤ 1. ∎

5. Concluding Remarks

We have presented the first nontrivial upper bound on the NOF communication complexity of the Boolean problem MPJ_k, showing that $C(\text{MPJ}_k) = o(n)$. A lower bound of $\Omega(n)$ had seemed *a priori* reasonable, but we show that this is not the case. One plausible line of attack on lower bounds for MPJ_k is to treat it as a *direct sum* problem: at each player's turn, it seems that n different paths need to be followed in the input graph, so it seems that an information theoretic approach (as in Bar-Yossef et al. [BJKS02] or Chakrabarti [Cha07]) could lower bound $C(\text{MPJ}_k)$ by n times the complexity of some simpler problem. However, it appears that such an approach would naturally yield a lower bound of the form $\Omega(n/\xi(k))$, as in Conjecture 1.1, which we have explicitly falsified.

The most outstanding open problem regarding MPJ_k is to resolve Conjecture 1.2. A less ambitious, but seemingly difficult, goal is to get tight bounds on $C(\text{MPJ}_3)$, closing the gap between our $O(n\sqrt{\log\log n/\log n})$ upper bound and Wigderson's $\Omega(\sqrt{n})$ lower bound. A still less ambitious question is prove that MPJ_3 is harder than its very special subproblem TPJ_3 (defined in Section 1.1). Our $n - O(\log n)$ lower bound for collapsing protocols is a step in the direction of improving the known lower bounds. We hope our technique provides some insight about the more general problem.

References

[AB07] Sanjeev Arora and Boaz Barak. *Complexity Theory: A Modern Approach*. Available online at ⟨http://www.cs.princeton.edu/theory/complexity/⟩, 2007.

[Abl96] Farid Ablayev. Lower bounds for one-way probabilistic communication complexity and their application to space complexity. *Theoretical Computer Science*, 175(2):139–159, 1996.

[AMS99] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. Preliminary version in *Proc. 28th Annu. ACM Symp. Theory Comput.*., pages 20–29, 1996.

[BHK01] László Babai, Thomas P. Hayes, and Peter G. Kimmel. The cost of the missing bit: Communication complexity with help. *Combinatorica*, 21(4):455–488, 2001.

[BJKS02] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. In *Proc. 43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 209–218, 2002.

[BPS05] Paul Beame, Toniann Pitassi, and Nathan Segerlind. Lower bounds for Lovász-Schrijver systems and beyond follow from multiparty communication complexity. In *Proc. 32nd International Colloquium on Automata, Languages and Programming*, pages 1176–1188, 2005.

[BT94] Richard Beigel and Jun Tarui. On ACC. *Comput. Complexity*, 4:350–366, 1994.

[CFL83] Ashok K. Chandra, Merrick L. Furst, and Richard J. Lipton. Multi-party protocols. In *Proc. 15th Annual ACM Symposium on the Theory of Computing*, pages 94–99, 1983.

[Cha07] Amit Chakrabarti. Lower bounds for multi-player pointer jumping. In *Proc. 22nd Annual IEEE Conference on Computational Complexity*, pages 33–45, 2007.

[CJP08] Amit Chakrabarti, T. S. Jayram, and Mihai Pătraşcu. Tight lower bounds for selection in randomly ordered streams. In *Proc. 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2008. to appear.

[DJS98] Carsten Damm, Stasys Jukna, and Jiří Sgall. Some bounds on multiparty communication complexity of pointer jumping. *Comput. Complexity*, 7(2):109–127, 1998. Preliminary version in *Proc. 13th International Symposium on Theoretical Aspects of Computer Science*, pages 643–654, 1996.

[GM07] Sudipto Guha and Andrew McGregor. Lower bounds for quantile estimation in random-order and multi-pass streaming. In *Proc. 34th International Colloquium on Automata, Languages and Programming*, pages 704–715, 2007.

[HG91] Johan Håstad and Mikael Goldmann. On the power of small-depth threshold circuits. *Comput. Complexity*, 1:113–129, 1991.

[NW93] Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SICOMP*, 22(1):211–219, 1993. Preliminary version in *Proc. 23rd Annu. ACM Symp. Theory Comput.*, pages 419–429, 1991.

[PRS97] Pavel Pudlák, Vojtěch Rödl, and Jiří Sgall. Boolean circuits, tensor ranks and communication complexity. *SIAM J. Comput.*, 26(3):605–633, 1997.

[VW07] Emanuele Viola and Avi Wigderson. One-way multi-party communication lower bound for pointer jumping with applications. In *Proc. 48th Annual IEEE Symposium on Foundations of Computer Science*, pages 427–437, 2007.

[Yao90] Andrew C. Yao. On ACC and threshold circuits. In *Proc. 31st Annual IEEE Symposium on Foundations of Computer Science*, pages 619–627, 1990.