Full version of an abstract appearing in
13th Conference on the Foundations of Software Technology and Computer Science, Dec. 1993

On some Communication Complexity problems related to Threshold Functions

Magnús Halldórsson* Jaikumar Radhakrishnan*† K. V. Subrahmanyam*†

* School of Information Science,
Japan Advanced Institute of Science and Technology – Hokuriku,
Tatsunokuchi, Ishikawa 923-12, JAPAN.
email: {magnus,jaikumar,mani}@jaist-east.ac.jp

† Theoretical Computer Science Group,
Tata Institute of Fundamental Research,
Bombay, INDIA 400 005.

Abstract

We study the computation of threshold functions using formulas over the basis {AND, OR, NOT}, with the aim of unifying the lower bounds of Hansel, Krichevskii, and Khrapchenko. For this we consider communication complexity problems related to threshold function computation.

- We obtain an upper bound for a communication problem used by Karchmer and Wigderson to derive the depth version of the Khrapchenko bound. This shows that their method, as it is, cannot give better lower bounds.
- We show how better lower bounds can be obtained if the *referee* (who was ignored in the Karchmer-Wigderson method) is involved in the argument.
- We show that the difficulty of the communication task persists even if the parties are required to operate correctly only for certain special inputs. Such restrictions on inputs do not have any natural formula complexity analogs; this makes the communication complexity approach more suited for our purposes.

As byproducts, we strengthen a result of Fredman and Komlós on graph covering, and obtain a Hodes-Specker like theorem for the basis {AND, OR, NOT}.

1 Introduction

1.1 Background

Threshold functions: The kth threshold function $\mathrm{TH}_{k,n}$ is the Boolean function with n variables that takes the value 1 precisely when at least k of the variables are assigned 1. In this paper, we study problems related to the computation of threshold functions using formulas over the basis {AND, OR, NOT}.

Formulas: Formulas are circuits whose underlying graph is a tree. The computation of threshold functions by formulas has been widely studied. Over the complete binary basis, Paterson, Pippenger, and Zwick [13] showed that all threshold functions can be computed by formulas of size $O(n^{3.13})$. For this basis, improving the results of Hodes and Specker, Pudlák [16] showed that all symmetric functions on n variables, except the sixteen having linear size formulas, need formula size $\Omega(n \log\log n)$; Pudlák's result applies to $\mathrm{TH}_{k,n}$, $2 \le k \le n-1$. Fischer, Meyer, and Paterson [5] showed a lower bound of $\Omega(n \log n)$ for the *majority* function $\mathrm{TH}_{\lceil n/2 \rceil,n}$.

Over the basis {AND,OR,NOT}, Paterson, Pippenger, and Zwick [13] showed that $\mathrm{TH}_{k,n}$ can be computed by formulas of size $O(n^{4.57})$. Lower bounds on the size of such formulas were shown by Hansel [6], Krichevskii [12], and Khrapchenko [9]. Hansel and Krichevskii showed a lower bound of $\Omega(n \log n)$ for computing $\mathrm{TH}_{2,n}$. This implies an $\Omega(n \log n)$ lower bound for all threshold functions $\mathrm{TH}_{k,n}$, $2 \le k \le n-1$. Khrapchenko showed that any such formula computing $\mathrm{TH}_{k,n}$ has size at least $k(n-k+1)$.

For very small or very large thresholds, the Hansel-Krichevskii lower bound is stronger; for other thresholds, the Khrapchenko lower bound is stronger. Furthermore, these lower bounds have been obtained by using different and unrelated considerations. A unified lower bound of $\left\lfloor \frac{k}{2} \right\rfloor n \log(\frac{n}{k-1})$ for computing $\mathrm{TH}_{k,n}$, $2 \le k \le n-1$, has been shown in [17]. The proof there used the following two step process.

STEP 1: Every monotone formula that accepts all inputs with k 1's and rejects all inputs with one 1 has size at least $n \log(\frac{n}{k-1})$.

STEP 2: Every formula that computes $\mathrm{TH}_{k,n}$ can be decomposed into $\lfloor k/2 \rfloor$ formulas of the type considered in STEP 1.

Unfortunately, the argument used there break down when negations are allowed. In this paper, we study this problem using the communication complexity characterization of formula complexity.

1.2 The communication complexity approach

Karchmer and Wigderson [8], and, independently, Yannakakis [7, p. 19] discovered a characterization of formula complexity using communication complexity. They observed that the formula depth of a Boolean function $f : \{0,1\}^n \to \{0,1\}$ is equal to the complexity of a game between two parties A and B. Party A is given an input $X \in f^{-1}(0)$ and party B is given an input $Y \in f^{-1}(1)$; their task then is to determine a literal l (a variable or its negation) that is false for the input X and true for the input Y. Here 'determine' means that a third party, a *referee*, be able to know the answer based only on the messages exchanged and without access to the actual inputs; as we shall see presently, much depends on this interpretation. The complexity of any solution (a *protocol*) that accomplishes this task is defined to be the maximum, over all such pairs (X, Y), of the number of bits exchanged. A similar relationship exists between the number of distinct message histories needed in any protocol and the formula size of the function. We do not describe the precise rules of the communication game here; the reader should consult [7] for further information. We do remind the reader, however, that the messages passed should be self delimiting.

Karchmer and Wigderson point out that the communication complexity of threshold functions is not fully understood. For example, no natural protocol for the majority function on n variables using $O(\log n)$ bits of communication is known, although there exist $O(\log n)$ depth monotone formulas for this function. *Then, why use communication complexity?* The reason is that the communication complexity approach allows us, sometimes, to make statements that have no analogs in the world of formulas. For example, consider the depth version of Khrapchenko's bound shown in [8]. For threshold functions this implies that $\mathrm{TH}_{k,n}$ needs formulas of depth at least $\log(k(n-k+1)$. But their communication complexity based proof shows more. In fact, it shows that A and B need to communicate this many bits even if they are set an easier task. In this easier version, A and B need not necessarily reveal the answer to the referee; it suffices if they both know it. Secondly, their protocol for $\mathrm{TH}_{k,n}$ is required to perform correctly only for certain special inputs: X contains exactly $k-1$ 1's, all positions that have 1 in X are also have 1 in Y, and Y has one additional 1. That is, A and B operate under the *promise* that the inputs have this property. In this paper, we study the effect of the referee and the promise on the communication tasks related to threshold functions. The absence of the referee, or the notion of a promise, do not have any natural interpretation in the world of formulas. We believe that the communication complexity approach, since it admits such variations, is better suited for our purposes.

1.3 Statement of the problems

We first study the communication problem suggested by the depth version of the Khrapchenko bound described above.

> Problem: $P_{k-1,k}(n)$.
> A's input: $X \in \{0,1\}^n$ with $k-1$ 1's.
> B's input: $Y \in \{0,1\}^n$ with k 1's.
> Promise: $X_i = 1 \Rightarrow Y_i = 1$, for $i = 1, 2, \ldots, n$.
> Task: Determine i such that $X_i \neq Y_i$.

Note that the promise on the inputs makes this problem monotone, since X_i must be 0 and Y_i 1. We show that the bounds obtained by Karchmer and Wigderson are essentially optimal.

To obtain a unified Hansel-Krichevskii-Khrapchenko bound, we consider the fruitful two step process for the monotone case. The natural analog for STEP 1 is the following problem.

> Problem: $P_{1,k}(n)$, $k \geq 2$.
> A's input: $X \in \{0,1\}^n$ with exactly one 1.
> B's input: $Y \in \{0,1\}^n$ with exactly k 1's.
> Promise: $X_i = 1 \Rightarrow Y_i = 1$, for $i = 1, 2, \ldots, n$.
> Task: Determine i such that $X_i \neq Y_i$.

With this definition, we now have the following revised strategy.

Step 1: Show that every protocol for $P_{1,k}$ (with referee) needs $n \log(\frac{n}{k-1})$ distinct message histories.

Step 2: Show that every formula computing $T_{k,n}$ (or perhaps every protocol for $P_{k-1,k}$) can be decomposed into $\Omega(k)$ protocols for $P_{1,k}$.

In this paper, we consider the problem $P_{1,k}$ under both interpretations of 'determine', that is, with and without the referee.

The problem $P_{1,2}$ was studied by Orlitsky [11] under yet another interpretation of the word 'determine'. In this interpretation, only B, and not necessarily A, needs to know the answer. For this one-sided communication problem, Orlitsky determined that the complexity is precisely $\lceil \log \log n \rceil + 1$. Unfortunately, Orlitsky's framework does not apply to the problem $P_{1,k}$ in general. We study this problem using a different, but related, approach, and determine its precise complexity. The model of one-sided communication does not have any direct bearing on the computation of threshold functions. Yet, we feel justified in reporting this work here, because the method used is related to the other proofs in this paper.

1.4 Our results

<u>**No referee**</u>

Result 1 *There exists a protocol of $P_{k-1,k}$ (without referee) using $\log(k(n-k+1)) + O(1)$ bits.*

By the lower bound of Karchmer and Wigderson, discussed above, this result is essentially optimal. It shows that better lower bounds for $\mathrm{TH}_{k,n}$ cannot be obtained if we allow the protocol to operate simultaneously without the referee and with the promise. For completeness we study the problem $P_{1,k}$ without the referee.

Result 2 *For $k \geq (n+1)/2$, the complexity of $P_{1,k}$ (without referee) is $\lceil \log(n-k+1) \rceil + 1$; for $k < (n+1)/2$, the complexity is at most $\lceil \log n \rceil + 1$ and at least $\lceil \log(n-k+1) \rceil + 1$.*

<u>**With referee**</u>

To understand the roles of the referee and the promise, we study the problem $P_{1,2}$.

Result 3 *Any protocol for $P_{1,2}$ (with referee) must use $n \log n$ different message histories, and (hence) $\log n + \log \log n$ bits.*

Compare this to the $\log n + O(1)$ upper bound when no referee was present. This shows that the problem remains hard even with the promise, provided the parties are required to reveal the answer to the referee. It is easy to see that this result is tight. The proof of this result is obtained using ideas from the Hansel-Krichevskii lower bound for $\mathrm{TH}_{2,n}$. We observe that this result has the following interesting consequence in the spirit of the Hodes-Specker-Pudlák theorem.

Result 4 *Over the basis {AND, OR, NOT}, all symmetric functions on n variables, except the eight that have linear size formulas, need formula size $\Omega(n \log n)$.*

We attempt to extend Result 3 and show similar lower bounds for the problem $P_{1,k}$ in general.

Result 5 *Any protocol for $P_{1,3}$ (with referee) must use $(n/2)\log(n/2)$ different message histories.*

The best upper bound we know for this problem uses $n \log(n/2)$ message histories. We believe that this upper bound is optimal, but have not succeeded in proving it so far. To obtain Result 5, we needed to strengthen the results of Fredman and Komlós on graph covering. This extension of their result might be of interest on its own. We believe that Result 3 and Result 5, contribute towards determining the communication complexity of the task in STEP 1.

One-sided communication

We determine the exact complexity of $P_{1,k}$ in the one-sided communication game.

Result 6 *The one-sided complexity of the problem $P_{1,k}$ is precisely* $\lceil \log\log(n/(k-1)) \rceil + 1$.

2 Graph Covering

We shall use the following terminology. The size of the largest independent set in a graph G will be denoted by $\alpha(G)$; size(G) will denote the number of non-isolated vertices in G. A function f with domain $V(G)$ will be called a *coloring* of G if $f(i) \neq f(j)$ whenever $(i,j) \in E(G)$. For a graph G, G^N will denote the subgraph of G induced by the non-isolated vertices of G. For two graphs F and G on the same set of vertices V, we denote their union $(V, E(F) \cup E(G))$ by $F \cup G$. Let G be a graph on n vertices. For $v \in V(G)$, let $\alpha_v(G)$ be the size of the largest independent set in G containing the vertex v. Let

$$\hat{\alpha}(G) = [\prod_{v \in V(G)} \alpha_v(G)]^{1/n},$$

that is, $\hat{\alpha}(G)$ is the geometric mean of the $\alpha_v(G)$. Note that $\hat{\alpha}(G)$ never exceeds $\alpha(G)$.

We shall need the following standard definition from information theory. For a random variable X with finite support, its entropy is given by

$$H(X) = -\sum_x \Pr[X = x] \log \Pr[X = x].$$

The entropy of a function f will be the entropy of the random variable $f(X)$, where X assumes values in the domain of f with uniform distribution. The following information theoretic measure on graphs was introduced by Fredman and Komlós [4].

Definition 1 (Coloring Entropy) Let G be a graph. Let f be the coloring of the graph G^N with minimum entropy. The coloring entropy of G is given by

$$H(G) = \frac{\text{size}(G)}{|V(G)|} H(f).$$

(If $E(G)$ is empty, then $H(G) = 0$.)

The following lemma, due to Fredman and Komlós, relates the coloring entropy of graphs with the independence number of their union.

Lemma 1 *If $G_1, G_2, \ldots, G_r$ are graphs with vertex set [n] such that*

$$\bigcup_{i=1}^{r} G_i = G,$$

then

$$\sum_{i=1}^{r} H(G_i) \geq \log \frac{n}{\alpha(G)}.$$

For our purposes, we need a slightly stronger form of the inequality (with $\hat{\alpha}(G)$ instead of $\alpha(G)$) than proved them. This stronger version can also be proved using their method; alternatively, it can be proved using the methods used by Pippenger [14], Hansel [6] and Körner [10] to derive related inequalities. In fact, this result is a special case of the sub-additivity property of Körner's *graph entropy*. The proof given below uses Hansel's method.

Lemma 2 (Stronger version) *If $G_1, G_2, \ldots, G_r$ are graphs with vertex set [n] such that*

$$\bigcup_{i=1}^{r} G_i = G,$$

then

$$\sum_{i=1}^{r} H(G_i) \geq \log \frac{n}{\hat{\alpha}(G)}.$$

Proof. Fix f_i to be the coloring of G_i^N with minimum entropy. Let $X_1, X_2, \ldots, X_r$ be mutually independent random variables, such that X_i takes values in $V(G_i^N)$ with uniform distribution. Let $Y_i = f_i(X_i)$, for $i = 1, 2, \ldots, r$. For a vertex v of G, we say that $(Y_1, Y_2, \ldots, Y_r)$ is *compatible* with v if the following holds for $i = 1, 2, \ldots, r$:

$$v \textit{ is not isolated in } G_i \;\Rightarrow\; f_i(v) = Y_i.$$

Let $C(v)$ be the event "$(Y_1, Y_2, \ldots, Y_r)$ is compatible with v". Then we have

$$\Pr[C(v)] = \prod_{i:\, v \in V(G_i^N)} \Pr[f_i(v) = Y_i]. \tag{1}$$

Note that if a sequence of colors is compatible with two vertices then the two vertices are non-adjacent in G. Hence, any sequence of colors compatible with v can be compatible with at most $\alpha_v(G)$ vertices. It follows that

$$\sum_{v \in V(G)} \frac{\Pr[C(v)]}{\alpha_v(G)} \leq 1.$$

Using (1) we have that

$$\sum_{v \in V(G)} \frac{1}{\alpha_v(G)} \prod_{i:\, v \in V(G_i^N)} \Pr[f_i(v) = Y_i] \leq 1.$$

Since the arithmetic mean is always at least the geometric mean, we have

$$\left[\prod_{v \in V(G)} \frac{1}{\alpha_v(G)} \prod_{i:\, v \in V(G_i^N)} \Pr[f_i(v) = Y_i] \right]^{1/n} \leq \frac{1}{n}.$$

By interchanging the order of the products and collecting the contributions we obtain

$$\frac{1}{\hat{\alpha}(G)} \prod_{i=1}^{r} \prod_{j \in range(f_i)} (\Pr[Y_i = j])^{|f_i^{-1}(j)|/n} \leq \frac{1}{n}.$$

On taking logs this gives

$$-\log \hat{\alpha}(G) + \sum_{i=1}^{r} \sum_{j \in range(f_i)} \frac{|f_i^{-1}(j)|}{n} \log \Pr[Y_i = j] \leq -\log n.$$

Since $|f_i^{-1}(j)|/\text{size}(G_i) = \Pr[Y_i = j]$, we have

$$-\sum_{i=1}^{r} \frac{\text{size}(G_i)}{n} \sum_{j \in range(f_i)} \Pr[Y_i = j] \log \Pr[Y_i = j] \geq \log \frac{n}{\hat{\alpha}(G)},$$

that is,

$$\sum_{i=1}^{r} H(G_i) \geq \log \frac{n}{\hat{\alpha}(G)}.$$

∎

Corollary 3 *If $G_1, G_2, \ldots, G_r$ are bipartite graphs such that*

$$\bigcup_{i=1}^{r} G_i = G,$$

then

$$\sum_{i=1}^{r} \mathrm{size}(G_i) \geq n \log \frac{n}{\hat{\alpha}(G)}.$$

∎

3 Without Referee

In this section we show that the $\log(k(n-k+1)$ [8] lower bound for the problem $P_{k-1,k}$ is nearly optimal, if only the two parties, and not necessarily the referee, are required to know the answer. We consider the problem $P_{1,2}$ first. The main idea for the proof is already used in this special case.

Theorem 1 *The communication complexity of $P_{1,2}$ (without referee) is at most $1 + \lceil \log n \rceil$.*

Proof. We think of A's input as a vertex v of K_n and of B's input as an edge $e = (v, w)$. It is well known that K_n can be edge colored with n colors ($n-1$ colors suffice if n is even). Fix one such coloring χ. Consider the following protocol, where A is given an input v and B is given the input $e = (v, w)$.

B: Send color $\chi(e)$ to A.

A: [There is exactly one edge incident on v with color $\chi(e)$. Hence, A can determine the other vertex w.]

Sends 0 to B if $v < w$, and 1 otherwise.

It can be easily verified that this protocol solves $P_{1,2}$ and uses at most $1 + \lceil \log n \rceil$ colors. ∎

Theorem 2 *For n large enough, the communication complexity of $P_{k-1,k}$ is at most $\lceil \log k \rceil + \lceil \log(n-k+1) \rceil + 1$.*

Proof. Since the problems $P_{k-1,k}$ and $P_{n-k,n-k+1}$ have the same complexity, we shall restrict our attention to case when k is at most $(n+1)/2$. The natural setting in this case is the hypergraph G_k, defined as follows:

$$\begin{aligned} V(G_k) &= \binom{[n]}{k-1}; \\ E(G_k) &= \left\{ \binom{S}{k-1} : S \in \binom{[n]}{k} \right\}. \end{aligned}$$

Then, we think of A's input as a vertex $v \in V(G_k)$ and identify B's input, a set S of size k, with the edge $\binom{S}{k-1}$. Note that $v \in \binom{S}{k-1}$.

It follows from a theorem of Pippenger and Spencer [15] that G_k can be edge colored with $(n-k+1)(1+\mathrm{o}(1))$ colors. Fix such a coloring χ and consider the following protocol, where each A is given a vertex v and B an edge e incident on v.

B: Sends the color $\chi(e)$ to A.

A: [There is exactly one edge incident on vertex v with color $\chi(e)$. Hence A can determine the edge e and the set S such that $\binom{S}{k-1} = e$.] Then A sends the rank in S of the unique element $i \in S - v$.

It is easily verified that this protocol solves $P_{k-1,k}$ using at most $\lceil \log k \rceil + \lceil \log(n-k+1) \rceil + 1$ bits, for n large enough. ∎

We now investigate the complexity of the problem $P_{1,k}$ when no referee is present.

Theorem 3 *For $k \geq (n+1)/2$, the complexity of $P_{1,k}$ is $\lceil \log(n-k+1) \rceil + 1$; for $k < (n+1)/2$, the complexity is at most $\lceil \log n \rceil + 1$ and at least $\lceil \log(n-k+1) \rceil + 1$.*

Proof. We think of A's input as an element a of $[n]$ and B's input as a subset S of $[n]$ of size k.

Construct a binary tree, of the smallest height, with n leaves. The n leaves correspond to the n elements of $[n]$. Thus, in this setting A is given a leaf a of the tree and B is given a set S ($a \in S$) of k leaves.

Consider the following protocol.

B: Suppose v is the lowest common ancestor of the leaves in S. Let i be the level of v in the tree (level of the root is 0). Then, B sends $0^i 1$.

A: From B's message and its own input a, A can determine the lowest common ancestor v. Now, if a is in the left subtree of v, then A sends 0, otherwise A sends 1.

B: If A's message was 0, B sends the path from the right child of v to a leaf in S. If A's message was 1, B sends the path from the left child of v to a leaf in S.

It can be easily verified that this protocol solves $P_{1,k}$ correctly. The total number of bits exchanged is at most

$$(i+1) + 1 + (\lceil \log n \rceil - i + 1) = \lceil \log n \rceil + 1.$$

For $k \geq (n+1)/2$, a different protocol can be used to obtain some savings. Let T and T' be two disjoint subsets of $[n]$ of size $n-k+1$ each. Since $|S| = k$, we have $S \cap T, S \cap T' \neq \emptyset$.

A: If $a \in T$, then A sends 0; otherwise A sends 1.

B: If A sent 0 the B sends the encoding of an element in $S \cap T'$; otherwise, it sends the encoding of an element in $S \cap T$.

Since $|T|, |T'| = n-k+1$, the second step requires only $\lceil \log(n-k+1) \rceil$ bits, giving a total of $\lceil \log(n-k+1) \rceil + 1$ bits.

A lower bound of $\lceil \log(n-k+1) \rceil + 1$ is easy to obtain. To see this, first observe that on every valid input A must transmit at least one bit. Also, for any fixed input of A, at least $n-k+1$ of the remaining $n-1$ positions must appear as answers when we vary B's input. It follows that there is some input where B sends $\lceil \log(n-k+1) \rceil$ bits. ∎

4 With Referee

When the answer is to be determined by a third party with access only to the history of the messages exchanged, it is helpful to picture the computation as a communication tree. In this tree each internal node has a label of the form (P, f), where $P \in \{A, B\}$ and $f : \{0,1\}^n \to \{0,1\}$. The leaves are labelled by literals, that is, i or $\neg i$ for some $i \in [n]$. On input (X, Y), the computation on the communication tree proceeds as follows. We start at the root and compute a function to determine whether to go to the left child or the right child. The function we compute is given by the label (P, f) of the root. If P is A then we compute $f(X)$ and go left if $f(X) = 0$ and right if $f(X) = 1$; if P is B, then we go left or right based on the value of $f(Y)$. We continue this till we reach a leaf, at which point we call the label of the leaf our answer.

We say that a communication tree represents a function $F : \{0,1\}^n \to \{0,1\}$, if for all $(X, Y) \in F^{-1}(0) \times F^{-1}(1)$, the literal obtained as the answers is false for X and true for Y. The depth of a communication tree is the length of the longest path from the root to a leaf. The size of the communication tree is the number of leaves. It can be easily seen that the depth corresponds to the communication complexity of F and that the size corresponds to the formula size of F.

For any problem $\mathcal{P}$ we may similarly consider the complexity of computation using communication trees. In this note we shall show lower bounds for the problems $P_{1,2}$ and $P_{1,3}$. Since all valid answers for these problems are positive literals, we shall assume that our tree has no negated literals.

Theorem 4 *Any communication tree for $P_{1,2}$ has size at least $n \log n$.*

Proof. First we need some notation. An input in $P_{1,2}$ has the form $(\{i\}, \{i, j\})$. We shall write such inputs as (i, ij). Note that for each such input there is a complementary input (j, ij). Let h be the leaf reached on input (i, ij); we write $(i, ij) \to h$. Note that the label on h must be j. Suppose $(j, ij) \to h'$. Then we call the lowest common ancestor of h and h' the *join* of ij, that is, *join*(ij) is the lowest common ancestor of the leaves corresponding to the two inputs (i, ij) and (j, ij).

Claim 1:

1. For all ij, *join*(ij) has label of the form (A, f) for some f.

2. If $(i, ij) \to h$ and $(k, jk) \to h$ then *join*(ij) = *join*(jk).

Proof of claim:

1. Suppose that *join*(ij) has label of the form (B, f). Then the function f computes different values for the inputs (i, ij) and (j, ij). However, this is impossible, because f is evaluated on the same pair ij in both cases. Hence the label must be of the form (A, f).

2. Suppose *join*$(ij) \neq$ *join*(jk). Say $p =$ *join*(ij) and $q =$ *join*(jk). Since both p and q are ancestors of h, they must be related to each other; say q is an ancestor of p. Let the label on q be (A, f_q). Since $q =$ *join*(jk), we have

$$f_q(j) \neq f_q(k). \tag{2}$$

On the other hand, since $p =$ *join*(ij) is a proper descendant of q, the two leaves corresponding to ij lie in the same under q. That is

$$f_q(i) = f_q(j). \tag{3}$$

Further, since $(i, ij) \to h$ and $(k, kj) \to h$, we have

$$f_q(i) = f_q(k). \tag{4}$$

Combining (2), (3), and (4), we obtain

$$f_q(j) \neq f_q(j).$$

This contradiction establishes the second part of the claim.

(End of Claim.)

From the second part of Claim 1 we have that $join(ij)$ is the same for all ij such that $(i, ij) \to h$, that is, $join(ij)$ is completely determined if one of the leaves corresponding to the input ij is known. We therefore refer to this as $join(h)$. For each node q with label of the form (A, f_q), define

$$\begin{aligned} S_0(q) &= \{label(h) : q = join(h) \text{ and } f_q(label(h)) = 0\}; \\ S_1(q) &= \{label(h) : q = join(h) \text{ and } f_q(label(h)) = 1\}. \end{aligned}$$

Claim 2:

$$\sum_q |S_0(q)| + |S_1(q)| \leq size(T).$$

Proof of claim: Every leaf belongs to at most one set corresponding to its unique join. *(End of Claim.)*

Claim 3: For all q with labels of of the form (A, f_q), $S_0(q) \cap S_1(q) = \emptyset$.

Proof of claim: If $i \in S_0(q)$, then $f_q(i) = 0$; if $i \in S_1(q)$, then $f_q(i) = 1$. Hence $S_0(q) \cap S_1(q) = \emptyset$. *(End of Claim.)*

Claim 4: If $q = join(ij)$ then $ij \in S_0(q) \times S_1(q)$.

Proof of claim: Since $q = join(ij)$, we have that q is the lowest common ancestor of h and h', where $(i, ij) \to h$ and $(j, ij) \to h'$. Then $join(h), join(h') = q$, where $label(h) = j, label(h') = i$, and $f_q(i) \neq f_q(j)$. The claim follows from this. *(End of Claim.)*

We may now prove the theorem. For each internal node q with label of the form (A, f), let G_q be the graph with edge set $S_0(q) \times S_1(q)$ and vertex set $[n]$. By Claim 3, G_q is a bipartite graph. Since every communication tree that solves $P_{1,2}$ must have a join for every pair ij, we have from Claim 4 that

$$\bigcup_q G_q = K_n.$$

Using Corollary 3 we have

$$\sum_q |S_0(q)| + |S_1(q)| \geq n \log n.$$

Then, using Claim 2, we have

$$\text{size}(T) \geq n \log n.$$

∎

It now follows easily, using the communication complexity characterization of formula complexity, that any function that rejects all inputs with one 1 and accepts all inputs with two 1's needs formula size at least $n \log n$. This observation has an interesting consequence concerning the formula complexity of symmetric functions. (A function is *symmetric* if its value is determined by the number of 1's in the input.) This result can also be derived using the method of Krichevskii, and has probably been observed before, but we could not find it mentioned in any of the standard references on circuit complexity [2, 3, 18].

Note that if f is symmetric, then its negation $\neg f$ and its dual[1] f^* are also symmetric and have the same formula complexity as f. For a symmetric function $f : \{0,1\}^n \to \{0,1\}$, let $\hat{f} : \{0,1,\ldots,n\} \to \{0,1\}$ be defined by: $\hat{f}(i) = 1$ iff f evaluates to 1 on inputs with exactly i 1's.

Theorem 5 *Over the basis $\{AND, OR, NOT\}$, all symmetric functions on n variables, except*

$$0, \quad \bigvee x_i, \quad \bigwedge x_i, \quad \left(\neg \bigvee x_i\right) \vee \left(\bigwedge x_i\right),$$

and their four negations, need formula size $\Omega(n \log n)$.

Proof. Let f be a symmetric function on n variables. Consider the restriction of $\hat{f}$ to $\{1,2,\ldots,n-1\}$. There are exactly eight symmetric functions for which this restriction is a constant function. These are the ones listed above; clearly, they all have linear size formulas. Hence suppose that $\hat{f}$ to $\{1,2,\ldots,n-1\}$ is not a constant function. Then there exists an $i \in [n-2]$, such that $\hat{f}(i) \neq \hat{f}(i+1)$. Since f, $\neg f$, and f^* have the same complexity, we may assume that $i \le n/2$, $\hat{f}(i) = 0$, and $\hat{f}(i+1) = 1$. Consider the function g obtained from f by fixing some $i-1$ variables at the value 1. Now g is a function on at least $n/2$ variables that rejects all inputs with one 1 and accepts all inputs with two 1's. As observed above, g (and hence f) needs formula size at least $(n/2)\log(n/2)$. [We may instead invoke Khrapchenko's theorem [2, p. 795] and obtain a lower bound of $i(n-i+1)$, which is better than our bound for $i > \log n$. Hence our result is interesting only for small values of i.] ∎

Theorem 6 *If T solves $P_{1,3}$ then* $\mathrm{size}(T) = \Omega(n \log n)$.

Proof. The argument in this case has two parts. The first part is similar to the proof of Theorem 4; this we only sketch. The second part is new and we discuss it in detail.

First we need to introduce some notation. Consider an input (i,α). As before, we write $(i,ijk) \to h$ if the computation reaches leaf h on input (i,ijk). Suppose, further, that the label of the leaf h is j (it must be one of j and k). Then we write $(i,ijk) \to j$. We say that a triple α *flip-flops* if there exist $i,j \in \alpha$ such that $(i,\alpha) \to j$ and $(j,\alpha) \to i$. In that case, i and j are called the *pivots* of α. If α does not flip-flop then we say that α *cycles*, for, in this case, there exist i,j,k such that $\alpha = ijk$ and $(i,\alpha) \to j$, $(j,\alpha) \to k$ and $(k,\alpha) \to i$.

Suppose the triple ijk flip-flops with pivots i and j. Let $(i,ijk) \to h$ and $(j,ijk) \to h'$. Then we define $join(ijk)$ to be the lowest common ancestor of h and h'.

Claim 1:

1. If ijk flip-flops, then $join(ijk)$ has label of the form (A,f).
2. Suppose α and β are triples that flip-flop, and for some $i \in pivot(\alpha)$, $j \in pivot(\beta)$, and some leaf h, $(i,\alpha) \to h$ and $(j,\beta) \to h$. Then $join(\alpha) = join(\beta)$.

□

If h is a leaf and if ijk is a triple that flip-flops such that $i \in pivot(ijk)$ and $(i,ijk) \to h$, then define $join(h) = join(ijk)$. In light of part 2 of the above claim, this constitutes a valid definition.

For a node q with label of the form (A,f), let

$$\begin{aligned} S_0(q) &= \{label(h) : join(h) = q \text{ and } f_q(label(h)) = 0\}; \\ S_1(q) &= \{label(h) : join(h) = q \text{ and } f_q(label(h)) = 1\}; \end{aligned}$$

[1] The dual f^* is defined by $f^*(x) = \neg f(\neg x)$, where $\neg x$ is the component-wise negation of x.

Claim 2:
$$\sum_q |S_0(q)| + |S_1(q)| \le \text{size}(T).\Box$$

Claim 3: For all q, $S_0(q) \cap S_1(q) = \emptyset$. $\Box$
Claim 4: If $q = join(ijk)$, then one of ij, ik, and jk is in $S_0(q) \times S_1(q)$. $\Box$

As before, let G_q be the graph with edge set $S_0(q) \times S_1(q)$ and vertex set $[n]$. Let

$$G = \bigcup_q G_q.$$

The rest of the argument is new to this proof. Note that our definition ensures that a triple ijk that is independent in G does not flip-flop, that is, it cycles.

Claim 5: The number of leaves with label i is at least $\log \alpha_i(G) - 1$.

Proof of claim: Let I be the largest independent set containing i. Let h be a leaf with label i. Let

$$\begin{aligned} S_0(h) &= \{l \in I : \exists km \subseteq I \text{ such that } klm \text{ cycles and } (l, klm) \to h\}; \\ S_1(h) &= I - S_0(h) - \{i\}. \end{aligned}$$

Consider a pair $jk \subset I$. It is easy to see that if $(j, ijk) \to h$, then $jk \in S_0(h) \times S_1(h)$. Hence, of the $\binom{|I|-1}{2}$ pairs p for whom $p \cup \{i\}$ is a cycling triple, those that could lead to h form a bipartite graph. Since every such triple must lead to some leaf with label i, it follows that the number of leaves with label i is at least $\log(|I| - 1)$. Since $|I| \ge 1$, we have $\log(|I| - 1) \ge \log |I| - 1$. The claim follows from this. *(End of Claim.)*

Using Corollary 3 and Claim 2 we get

$$\text{size}(T) \ge n \log(\frac{n}{\hat{\alpha(G)}}). \tag{5}$$

Using Claim 5, we get that

$$\text{size}(T) \ge \sum_i (\log \alpha_i(G) - 1). \tag{6}$$

Combining (5) and (6), we get

$$\text{size}(T) \ge \frac{1}{2}(n \log n - n) \ge \frac{n}{2} \log \frac{n}{2}.$$

∎

5 One-sided communication

Theorem 7 *The one-sided communication complexity of $P_{1,k}$ is precisely* $\left\lceil \log \log \left(\frac{n}{k-1} \right) \right\rceil + 1$.

5.1 Upper bound

Orlitsky proposed the following protocol for $P_{1,2}$. A is given an input $x \in [n]$ and B a set $Y \subseteq [n]$, $|Y| = 2$, such that $x \in Y$.

B: Let $Y = \{i, j\}$. In the binary representation $i_1 i_2 \cdots i_{\lceil \log n \rceil}$ of i and the binary representation $j_1 j_2 \cdots j_{\lceil \log n \rceil}$ of j, let p be a position such that $i_p \ne j_p$. B transmits p using $\lceil \log \log n \rceil$ bits.

A: A transmits the p-th bit in the binary representation of x.

It is clear that B can then determine which of i and j A does not have. The protocol uses $\lceil \log\log n \rceil + 1$ bits. Observe that this protocol can be used even if $x \notin Y$, and that B always determines an element of Y different from x. Using this observation, we now describe a protocol for $P_{1,k}$. Both A and B assume a common partition of $[n]$ into $k-1$ blocks of size at most $\lceil \frac{n}{k-1} \rceil$ each. They also agree on a binary representation for the (at most) $\lceil \frac{n}{k-1} \rceil$ elements in each block; that is, each element in $[n]$ is given a code of length $\lceil \log \lceil n/(k-1) \rceil \rceil = \lceil \log(n/(k-1)) \rceil$ bits, so that no two elements in the same block get the same code.

B: Let $Y = \{i_1, i_2, \ldots, i_k\}$. Since there are only $k-1$ blocks, there exists a block that contains two elements, say i_1, i_2, from Y. The codes of i_1 and i_2 must differ; B transmits a position p, where they differ, using $\left\lceil \log\log \left(\frac{n}{k-1}\right) \right\rceil$ bits.

A: A transmits the p-th bit in the code of x.

It is easy to verify that this protocol is correct. It uses $\left\lceil \log\log \left(\frac{n}{k-1}\right) \right\rceil + 1$ bits.

5.2 Lower bound

We think of A's input as an element x of $[n]$, and of B's input as a subset Y of $[n]$ of size k. Note that the promise ensures that $x \in Y$. We require the parties to satisfy the following properties.

1. A message sent by any party depends only on its input and the messages it received previously.

2. Each party can determine when the message being received is over, based on its input and the messages received so far.

3. When the communication ends, B determines a $z \in Y - \{x\}$. We denote this by $(x, Y) \to z$.

Consider a protocol for $P_{1,k}$ and let l be the number of bit communicated by it in the worst case. Let $A : [n] \times \{0,1\}^* \to \{0,1\}^* \cup \{\perp\}$, be defined by $A(x, \sigma) = \beta$, where β is the message sent by A, when A has x and σ is sequence of bits that A has received from B; let $A(x, \sigma) = \perp$ if it is not possible for A to receive the sequence σ when it has input x.

For $\sigma \in \{0,1\}^*$ ($|\sigma| < l$) and $i = 1, 2, \ldots, l$ let

$$\begin{aligned} V_0(\sigma, i) &= \{x \in [n] : A(x, \sigma)[i] = 0\}; \\ V_1(\sigma, i) &= \{x \in [n] : A(x, \sigma)[i] = 1\}, \end{aligned}$$

and let the graph G^i_σ be defined by

$$\begin{aligned} V(G^i_\sigma) &= [n]; \\ E(G^i_\sigma) &= V_0(\sigma, i) \times V_1(\sigma, i). \end{aligned}$$

Since $V_0(\sigma_i) \cap V_1(\sigma, i) = \emptyset$, G^i_σ is bipartite. Let $G = \bigcup_{\sigma, i} G^i_\sigma$.

Lemma 4 *G has no independent set of size k.*

Proof. Let $Y \subset \{1, \ldots, n\}$, $|Y| = k$. We show that Y is not independent in G. Let $x \in Y$ and assume $(x, Y) \to z$. By definition, $z \in Y$ and $(z, Y) \not\to z$. Let $\langle \sigma_1, \ldots, \sigma_r \rangle$ be the sequence of messages sent by B for the input pair (x, Y) and let $\langle \sigma'_1, \ldots, \sigma'_{r'} \rangle$ be the sequence of messages sent by B for input pair (z, Y). Properties 1 and 2 of the protocol imply that for some j, $\sigma_j \neq$

σ'_j. For the minimum such j, property 1 implies that, if σ is the concatenation of $\sigma_1, \ldots, \sigma_{j-1}$, $A(x,\sigma) \neq A(z,\sigma)$. Moreover, from Property 2 of the protocol, neither $A(x,\sigma)$ nor $A(z,\sigma)$ is a proper prefix of the other, so there exists a bit position i where they differ. But then (x,z) is an edge in G^i_σ and hence in G. ∎

For $x \in [n]$, let $\mu(x)$ be the number of graphs G^i_σ in which x appears non-isolated. From Corollary 3, we have

$$\sum_x \mu(x) = \sum_{\sigma,i} \mathrm{size}(G^i_\sigma) \geq n \log\left(\frac{n}{k-1}\right)$$

We shall show (Lemma 5 below) that $\mu(x) \leq 2^{l-1}$ for all $x \in [n]$. It will follow that

$$n2^{l-1} \geq n \log\left(\frac{n}{k-1}\right),$$

and, since l is an integer, $l \geq \left\lceil \log\log\left(\frac{n}{k-1}\right)\right\rceil + 1$, completing the proof of Theorem 7.

Lemma 5 *For all $x \in [n]$, $\mu(x) \leq 2^{l-1}$.*

Proof. Let $s(x,Y)$ be the sequences of bits sent *by* B for the valid input (x,Y). Let

$$S(x) = \{s(x,Y)\}_Y,$$

where Y ranges over the sets of size k that contain x. It can be verified, using Properties 1 and 2, that $S(x)$ is prefix free, that is, if $\sigma, \sigma' \in S(x)$ and $\sigma \neq \sigma'$, then σ is not a prefix of σ'. We first observe that

$$\mu(x) \leq \sum_{\sigma \in S(x)} l - |\sigma'|. \tag{7}$$

To justify this, note that if x is not isolated in G^i_σ, then σ is a prefix of some $\sigma' \in S(x)$. For $\sigma' \in S(x)$, A sends at most $l - |\sigma'|$ bits. Hence, for all $\sigma' \in S(x)$,

$$|\{(i,\sigma) : \sigma \text{ is a prefix of } \sigma' \text{ and } x \text{ is not isolated in } G^i_\sigma\}| \leq l - |\sigma|.$$

To get (7), we sum this inequality for all $\sigma' \in S(x)$.

Next, we show that

$$\sum_{\sigma \in S(x)} l - |\sigma| \leq 2^{l-1}. \tag{8}$$

Since A must always communicate at least one bit, all sequences in $S(x)$ have length at most $l-1$. To prove (8) we show that the maximum of $\sum_{\sigma \in S} l - |\sigma|$, taken over prefix sets S of sequences of length at most $l-1$, is at most 2^{l-1}. Let S^* be a set achieving this maximum. If $\sigma \in S^*$ has length less than $l-1$, then we may replace it by $\sigma 0$ and $\sigma 1$, and obtain another prefix free set with sum at least $\sum_{\sigma \in S^*} l - |\sigma|$. Hence, we may assume that all $\sigma \in S^*$ have length exactly $l-1$. But then the sum is at most 2^{l-1} because each term contributes 1 and there are at most 2^{l-1} sequences in S^*.

We complete the proof of the lemma by combining inequalities (7) and (8). ∎

Acknowledgment

The upper bound for the case $k < (n+1)/2$ in Theorem 3 is due to Desh Ranjan and Shiva Chaudhuri; we thank them for their contribution.

References

[1] M. Ajtai, J. Komlós, and E. Szemerédi. *Sorting in* $c \log n$ *parallel steps.* Combinatorica, 3, 1983, pp. 1–19.

[2] R. B. Boppana and M. Sipser. *The Complexity of Finite Functions.* Chapter 14, The Handbook of Theoretical Computer Science, (J. van Leeuwen, ed.), Elsevier Science Publishers B. V., 1990, pp. 759–804.

[3] P. E. Dunne. *The Complexity of Boolean Networks.* Academic Press, London, 1988.

[4] M. Fredman and J. Komlós. *On the size of Separating Systems and Perfect Hash Functions.* Siam J. Alg. Disc. Meth., 5, 1984, pp. 61–68.

[5] M. J. Fischer, A. R. Meyer, and M. S. Paterson. $\Omega(n \log n)$ *Lower bounds on the length of Boolean formulas.* SIAM J. Comp., 11(3), 1982, pp. 416–427.

[6] G. Hansel. *Nombre minimal de contacts de fermature nécessaires pour réaliser une fonction booléenne symétrique de* n *variables.* C. R. Acad. Sci. Paris 258 (1964), pp. 6037–6040.

[7] M. Karchmer. *Communication Complexity: A New Approach to Circuit Depth.* The MIT Press, 1989.

[8] M. Karchmer and A. Wigderson.*Monotone circuits for connectivity require superlogarithmic depth.* Proceedings of the 20th ACM STOC, 1988, pp. 539–550.

[9] V. M. Khrapchenko. *A method of obtaining lower bounds for the complexity* π*-schemes.* Math. Notes Acad. Sci. USSR, 11 (1972), pp. 474–479.

[10] János Körner. *Fredman–Komlós Bound and Information Theory.* Siam J. Alg. Disc. Meth., vol. 7, 1986, pp. 560–570.

[11] A. Orlitsky. *Worst-case interactive communication I: Two messages are almost optimal.* IEEE Trans. Inform. Theory, vol. 36, no. 5 (1990), pp. 1111-1126.

[12] R. E. Krichevskii. *Complexity of contact circuits realizing a function of logical algebra.* Sov. Phys. Dokl., 8 (1964) pp. 770–772.

[13] M. S. Paterson, N. Pippenger, and U. Zwick. *Optimal carry save networks.* Boolean Function Complexity. LMS Lecture Note Series 169, Cambridge Univ. Press, 1992, pp. 174–201.

[14] N. Pippenger. *An Information-Theoretic Method in Combinatorial Theory.* Journal of Combinatorial Theory (A) 23, 1977, pp. 99-104.

[15] N. Pippenger and J. Spencer. *Asymptotic behaviour of the chromatic index of hypergraphs.* J. Combinatorial Theory, Series A 51, 24–42.

[16] P. Pudlák. *Bounds for Hodes-Specker theorem.* Logic and Machines: Decision Problems and Complexity, LNCS 171, (Springer, Berlin, 1984), pp. 421–445.

[17] J. Radhakrishnan. *Better Bounds for Threshold Formulas.* In the proceedings of the 32nd IEEE FOCS, 1991, pp. 314–323.

[18] I. Wegener. *The Complexity of Boolean Functions.* Wiley-Teubner Series in Computer Science, 1987.