comput complexity 1 (1991), 113–129 1016-3328/91/020113-17 $1.50+0.20

ON THE POWER OF SMALL-DEPTH THRESHOLD CIRCUITS

Johan Håstad and Mikael Goldmann

Abstract. We investigate the power of threshold circuits of small depth. In particular, we give functions that require exponential size unweighted threshold circuits of depth 3 when we restrict the bottom fanin. We also prove that there are monotone functions f_k that can be computed in depth k and linear size $\wedge, \vee$-circuits but require exponential size to compute by a depth $k-1$ monotone weighted threshold circuit.
Key words. Circuit complexity, monotone circuits, threshold circuits, lower bounds
Subject classifications. 68Q15, 68Q99

1. Introduction

The study of circuit complexity has in one sense been successful and in another not so successful. While there are still no non-linear lower bounds on circuit-size for any function in NP, several interesting results have been shown for restricted circuit classes e.g. monotone circuits [4, 16, 3, 13, 14, 15] and circuits of bounded depth [1, 9, 10, 17, 19, 21].

The smallest natural circuit-class that is not known to be strictly contained in NP is TC^0, the set of functions computable by constant-depth polynomial-size circuits containing threshold gates. Threshold gates are quite powerful and many fairly complicated functions (like division, implicit in [6]) are in TC^0.

It also seems like the techniques used for proving lower bounds for usual constant depth circuits (with or without modular gates) are not sufficient to prove lower bounds for threshold circuits. The best known results about small-depth threshold circuits are those by Hajnal *et al.* [12] where, among other results, it is established that depth 3 threshold circuits of polynomial size are more powerful than corresponding circuits of depth 2.

To further understand the nature of threshold circuits Yao [22] studied monotone threshold circuits. In particular, Yao was interested in the question whether it is true for monotone circuits that depth k and polynomial size is more powerful than depth $k-1$ and polynomial size. He proved that this is indeed the case by exhibiting a function f_{2k} computable by ordinary $\wedge, \vee$-circuits of depth $2k$ that requires exponential size when the depth is restricted to k.

In this paper we generalize both the above results. First we give an explicit function that cannot be computed by small depth 3 unweighted threshold circuits of small bottom fanin. The proof of this is based on a communication game analyzed by Babai, Nisan and Szegedy [5]. In this communication game $s+1$ players (which we call $P_j, j=1,\ldots,s+1$) participate and share some variables which are partitioned into $s+1$ groups (the j^{th} group being G_j). The player P_j knows all variables except the variables in G_j and the cost of evaluating a function is the number of bits the players have to exchange. In [5] they show that a function that is a generalization of the Boolean inner product has high complexity in this model. Our proof is based on the observation that in this model it is quite cheap to evaluate an unweighted threshold circuit of depth 2 and bottom fanin bounded by s. Applying the correlation lemma of [12] and the lower bound for the communication game [5] gives the desired lower bound.

We get nontrivial lower bounds as long as $s \leq c \log n$ for a constant c (which is about $\frac{1}{2}$). It is very interesting to note that if these results could be extended to hold for s as large as $(\log n)^k$ for any constant k, then by a recent result by Yao [23] it would be possible to prove that the given function cannot be computed by constant depth polynomial size circuits containing modular gates with composite modulus. This would be the first such lower bound. One possible approach to this problem is to try to improve the lower bounds for the multiparty protocol communication games. This could not happen for the functions we are studying in this paper, but the better bounds could be true for the other function studied in [5] (which is based on quadratic residuosity).

In the monotone case we sharpen Yao's result by proving an exponential difference between depth k and depth $k-1$. In this case we also extend the result by allowing arbitrary positive weights in the circuit. Let us outline the method of proof used. The function f_k that we consider is defined by a depth k tree with alternating levels of $\wedge$-gates and $\vee$-gates. We define a probability distribution p_k on minterms with a minimal number of 1's and a distribution q_k on maxterms with a maximal number of 1's. The key point is to prove that a threshold circuit of depth $k-2$ that has a good probability of outputting 1 on a random input picked according to the distribution p_k has a probability very close to 1 of outputting 1 on a random input picked according to the distribution q_k. This is done in Lemma 9.

This main lemma is proved by induction over k and uses the fact that f_k and the distributions p_k and q_k have the same recursive structure. The base case is an easy verification and the crux of the proof is the induction step. The induction uses in an essential way the monotonicity of the circuits as follows. The distribution p_k (q_k) consists of many copies of p_{k-1} (q_{k-1}) with some extra zeroes (ones) added. The output of a threshold circuit of depth $k-2$ is a threshold of outputs of threshold circuits of depth $k-3$. Applying the induction hypothesis on each pair (p_{k-1}, q_{k-1}) of subdistributions inside p_k and q_k plus the monotonicity of the circuit makes it possible to complete the proof.

The general outline of the proof is exactly the same as that used by Yao. We obtain sharper results by also arguing about the negation of f_k, $\bar{f}_k$ and the corresponding distributions $\bar{p}_k$ and $\bar{q}_k$.

It is interesting to note that while the functions f_k cannot be computed by depth $k-1$ monotone threshold circuits of size $\leq 2^{n^{\frac{1}{2k}}}$, by a result of Allender [2] (which is based on work by Toda [20]), they can be computed by depth 3 general threshold circuits of size $2^{O((\log n)^k)}$. This might be taken as another piece of evidence that monotonicity is a severe restriction, and that new techniques probably have to be developed to attack general threshold circuits of small depth.

A preliminary version of this paper appeared in [11].

2. Preliminaries

Let us recall the relevant definitions. The basic element of a our circuits will be a threshold gate. A threshold gate with inputs $y_1, y_2 \ldots y_m$, weights $w_1, w_2 \ldots w_m$ and threshold t will output 1 if $\sum_{i=1}^{m} y_i w_i$ is at least t and 0 otherwise. We will say that a gate is monotone if $w_i > 0$ for all i. A threshold circuit is a directed acyclic graph where each node is a threshold gate or an input variable. The only nodes of indegree 0 of the circuit are the input variables and there is only node of outdegree 0 is the output node and this output is computed by evaluating the gates. The circuit is monotone if each gate in the circuit is monotone and we have no negated variables. The *depth* of the circuit is the longest path from any input to the output. The *size* is the number of gates. Please note that this definition does not allow multiple edges between pairs of nodes, and thus the fanin of a gate is bounded by the size of the circuit.

TC^0 is the set of functions computable by constant depth polynomial size threshold circuits. Monotone TC^0 is the set of functions computable by by constant depth polynomial size threshold circuits consisting of monotone gates and having no negated variables on the input level.

A more restricted version of threshold circuits is obtained by having all weights equal to 1 (but allowing negated variables at the input level). This way we obtain *unweighted threshold circuits.* When we want to emphasize that we speak about the more general model we call them *weighted threshold circuits.* It is interesting to note that TC^0 remains the same even if we restrict the circuits to be unweighted [7].

3. Lower Bounds for Depth 3 Circuits

The function we will consider is the "generalized inner product function" considered in [5]. We use doubly indexed variables $x_{i,j}$ where i ranges from 1 to n and j ranges from 1 to $s+1$. Our function gip_s is defined by

$$gip_s(x) = \sum_{i=1}^{n} \prod_{j=1}^{s+1} x_{i,j}$$

where the sum is calculated modulo 2. We will be interested in a communication game among $s+1$ players P_j, $j = 1, \ldots s+1$ where P_{j_0} knows the value of all variables $x_{i,j}$ except those with $j_0 = j$. Collectively the players want to evaluate $gip_s(x)$ (or something close to it) and the measure of complexity is the number of bits exchanged. We call this game the s-communication game and we will say that a protocol ϵ-evaluates a function f if the output agrees with f with probability $\frac{1+\epsilon}{2}$. Here the probability is taken over a uniformly picked random input (and not a probabilistic protocol). The following very powerful result is proved by Babai, Nisan and Szegedy [5].

Lemma 1. **(Theorem 2 in [5])** *To ϵ-evaluate gip_s in an s-communication game requires $\Omega(n2^{-2s} + \log \epsilon)$ bits communication.*

We will be interested in evaluating gip_s by a depth 3 threshold circuit. In this section we assume that there are no weights in the circuit i.e. that each gate C in the circuit just counts the number of inputs to it that takes the value 1 and outputs 1 iff this number is at least t_C for a predetermined value t_C. We also assume that we have small bottom fanin i.e. that the number of inputs to any gate next to the input variables is small. The main result of the section is:

Theorem 2. *To evaluate gip_s by a depth 3 unweighted threshold circuit with bottom fanin at most s requires size $2^{\Omega(n/s4^s)}$.*

PROOF. Assuming that such a circuit exists, we will eventually obtain a contradiction to Lemma 1. We first need a lemma which is the same (although we phrase it differently and use that gip_s is almost unbiased) as Lemma 3.3 of Hajnal *et al.*.

Lemma 3. (Lemma 3.3 in [12]) *Let gip_s be computed by an unweighted threshold circuit where the top gate has fanin R. Then one of the inputs to the top gate $\frac{1}{R}$-evaluates gip_s.*

In our case this input to the top gate will correspond to a depth 2 threshold circuit with bottom fanin bounded by s. For these we have the following interesting lemma.

Lemma 4. *Suppose f is computed by a depth 2 unweighted threshold circuit of size R and bottom fanin bounded by s. Then f can be evaluated in the s-communication game with $1 + s \log R$ communicated bits.*

PROOF. Since the bottom fanin is at most s every such gate can be evaluated by one of the players. Thus partition the gates between the players in such a way that that each player can evaluate all the gates given to him. Now player 1 need only tell the other players how many of his gates evaluated to 1. This he can do with $\log R$ bits. Players 2 through s do the same, and finally player $s+1$ can evaluate the top gate and tell the one bit result to the others. The total number of bits used is $1 + s \log R$. □

To finish the proof of the theorem we now just need to collect the pieces. By Lemmas 3 and 4 we see that by the circuit assumed to exist there is a protocol that $\frac{1}{R}$-evaluates gip_s with $1 + s \log R$ bits of communication. Combining this with Lemma 1 finishes the proof. □

Remark: We have used very little of the properties of the threshold gates in the proof. We have used that the top gate was a threshold gate only to make sure that one of its inputs was correlated with gip_s. On the third level we only needed that each gate depended on at most s variables. The type of this dependence was unimportant. Thus only on level two did we use any real properties of the threshold gates.

If we instead limit the fanin on the middle level the situation gets even simpler.

Theorem 5. *Suppose an unweighted depth 3 threshold circuit computes the inner product (i.e. gip_1) and that the fanin on the middle level is bounded by s. Then the size of the circuit is $2^{\Omega(n/s)}$.*

PROOF. In this case we will use a two person communication game. Lemma 1 could be used also for this purpose but let us use an older more accurate result by Chor and Goldreich [8].

Lemma 6. **(Theorem 21 in [8])** *To ϵ-evaluate the inner product function (gip_1) in a two-person game requires at least at least $n - 3 - 3\log\epsilon^{-1}$ bits of communication.*

Now suppose there is a circuit of size R with fanin s on the middle level that computes the inner product. We conclude by Lemma 3 that there is a depth 2 threshold circuit with top fanin bounded by s that $\frac{1}{R}$-evaluates the inner product. Such a circuit can be evaluated by two players by one player telling the other player how many of his inputs to each of the various gates take the value 1. The total number of bits needed to specify this is bounded by $s\log R$. The other player then evaluates the top gate and tells the result to the first player. By Lemma 6 we get $1 + s\log R \geq n - 3 + 3\log R$ and the theorem follows. □

4. Monotone Threshold Circuits

Let us start by formally defining our functions f_k. For technical reasons the circuit defining f_k will not be defined by a regular tree.

Definition 7. *The function f_k is a function of N^{2k-2} variables. It is defined by a depth k circuit that is a tree. At the leaves of the tree there are unnegated variables. The i^{th} level from the bottom consists of $\wedge$-gates if i is even and otherwise it consists of $\vee$-gates. The fanin at the top and bottom levels is N and at all other levels it is N^2.*

This function was used by Sipser in [18] who showed that it requires superpolynomial size depth $k-1$ circuits over the basis $\{\wedge, \vee, \neg\}$.

It will be convenient to also consider the functions $\bar{f}_k$, the negations of f_k. Clearly $\bar{f}_k$ is computed by a circuit very similar to the circuit computing f_k. The only difference being that $\wedge$ and $\vee$ change places and that the inputs are negated variables. To make $\bar{f}_k$ formally a monotone function we set $y_i = \bar{x}_i$ and let these be the input variables of $\bar{f}_k$.

There is a nice inductive definition of f_k that will be very useful to us. By definition f_2 is a depth 2 circuit that is an $\wedge$ of size N where each input is an $\vee$ of size N. Now for even k, f_k is just f_2 with each input variable changed to an independent copy of f_{k-1}. Similarly for odd k, f_k is obtained from $\bar{f}_2$ by replacing each variable by an independent copy of f_{k-1}.

Using this construction it is natural to label each variable of f_k by a $2k-2$ tuple of numbers between 1 and N.

In this section we will be interested in monotone weighted threshold circuits i.e. circuits containing gates that are weighted threshold gates with all weights positive. We have:

Theorem 8. *A monotone weighted threshold circuit computing f_k that is of depth $k-1$ has size at least 2^{cN} for some constant $c > 0$ and $N > N_0$.*

We will look at inputs chosen according to two different probability distributions, p_k and q_k, for inputs in $f_k^{-1}(1)$ and inputs in $f_k^{-1}(0)$ respectively. They are the same distributions as those Yao constructed. Here p_k picks a random assignment to the variables that contains as many zeroes as possible under the condition that $f_k(x) = 1$, while q_k picks a random assignment to the variables that contains as many ones as possible under the condition that $f_k(x) = 0$.

The distributions p_k and q_k are defined inductively starting with $k = 2$. To get the feeling for the definition, please remember the inductive definition of f_k. Index the variables in f_2 by two indices running from 1 to N and let x_{ij} be the j^{th} variable in the i^{th} $\vee$ defining f_2. An element from p_2 is chosen as follows: For each i, randomly and uniformly pick a $j(i)$, set $x_{ij(i)} = 1$ and set all other variables to 0. An element from q_2 is chosen by randomly choosing i_0 and setting $x_{i_0 j} = 0$ for all j, while all other variables are set to 1.

Clearly for each x chosen by p_2, $f_2(x) = 1$, while for each x chosen by q_2, $f_2(x) = 0$. We will also need the corresponding distribution for $\bar{f}_2$. A random element from $\bar{p}_2$ ($\bar{q}_2$) is a random element from p_2 (q_2) which is changed by setting $y_{ij} = \bar{x}_{ij}$ for all i and j. Observe that the definition implies that if y is chosen according to $\bar{p}_2$ then $\bar{f}_2(y) = 0$ and if it chosen according to $\bar{q}_2$ then $\bar{f}_2(y) = 1$.

Now to chose an element from p_k proceed as follows:

1. For k even, choose a random element from p_2. Each variable now corresponds to an independent copy of f_{k-1} and the value of that variable will decide how to give values to the variables corresponding to that function. If a variable is given the value 1, then set the corresponding variables according to p_{k-1} and if the value is 0 then set all corresponding variables to 0.

2. For k odd, choose a random element from $\bar{q}_2$. If a variable is given the value 1, then set the corresponding variables according to p_{k-1} and if the value is 0 then set all corresponding variables to 0.

In a similar way an element from q_k is picked as follows:

1. For k even, choose a random element from q_2. If a variable is given the value 1, then set the corresponding variables to 1 and if the value is 0 then set the corresponding variables according to q_{k-1}.

2. For k odd, choose a random element from $\bar{p}_2$. If a variable is given the value 1, then set the corresponding variables to 1 and if the value is 0 then set the corresponding variables according to q_{k-1}.

Again we define $\bar{p}_k$ and $\bar{q}_k$ by negating variables.

To make formulas simple in the future, suppose that g is a Boolean function and let p be a probability distribution. Then we let $g(p)$ denote the probability that g takes the value 1 on a random input from the distribution p. Using this notation observe that the above definitions imply that:
$f_k(p_k) = 1,\ f_k(q_k) = 0,\ \bar{f}_k(\bar{p}_k) = 0,\ \bar{f}_k(\bar{q}_k) = 1.$
Now we are ready to state the main lemma. Let $\epsilon = 0.01$.

Lemma 9. *Let g be a function that is computed by a monotone weighted threshold circuit of depth $k-2$ and size $\leq 2^{\epsilon N}$, then for $N > N_0$,*

1. $g(p_k) \geq 2^{-2\epsilon N} \Rightarrow g(q_k) > \frac{3}{5}$,

2. $g(p_k) \geq \frac{2}{5} \Rightarrow g(q_k) > 1 - 2^{-2\epsilon N}$.

We postpone the proof of the lemma to Section 5. Before we prove that Theorem 8 follows from Lemma 9 let us just observe that this would have been trivial if we would have wanted a weaker theorem with $k-2$ instead of $k-1$. In fact a circuit g computing f_k has $g(p_k) = 1$ and $g(q_k) = 0$ and thus either part of Lemma 9 tells us that this circuit has to be large.

Let us see how the current Theorem 8 can be proved using Lemma 9. Suppose f_k was computed by a threshold circuit of size $2^{\epsilon N}$ and depth $k-1$. Suppose that the top gate has fanin R, weights w_i and threshold t. Let the function computed by the i^{th} input of the top gate be called g_i. This function is computed by a depth $k-2$ threshold circuit of size at most $2^{\epsilon N}$ and hence we can use Lemma 9.

Let

$$\begin{aligned} S_1 &= \{i \mid g_i(p_k) \leq 2^{-2\epsilon N}\}, \\ S_2 &= \{i \mid 2^{-2\epsilon N} < g_i(p_k) \leq 2/5\}, \\ S_3 &= \{i \mid g_i(p_k) > 2/5\}. \end{aligned}$$

Consider the following distributions on inputs. Let p be the distribution obtained by choosing an x according to p_k while requiring that $g_i(x) = 0$ for all $i \in S_1$. Similarly, we get q by choosing x according to q_k while requiring that $g_i(x) = 1$ for all $i \in S_3$.

The probabilities $g_i(p)$ and $g_i(q)$ are close to $g_i(p_k)$ and $g_i(q_k)$ respectively. We have for any function h:

$$\begin{aligned} h(p) = \Pr_{p_k}[h(x) = 1 \mid \forall_{i \in S_1} g_i(x) = 0] \leq \\ \frac{h(p_k)}{\Pr_{p_k}[\forall_{i \in S_1} g_i(x) = 0]} \leq h(p_k) + 2^{1-\epsilon N} \end{aligned} \tag{1}$$

and

$$\begin{aligned} h(q) = \Pr_{q_k}[h(x) = 1 \mid \forall_{i \in S_3} g_i(x) = 1] \geq \\ h(q_k) - \Pr_{q_k}[\exists_{i \in S_3} g_i(x) = 0] \geq h(q_k) - 2^{-\epsilon N}. \end{aligned} \tag{2}$$

Let P be the expected weight to the top gate when the input is chosen according to p and let Q be the corresponding weight when the input is chosen according to q. Then by the assumption that the circuit computes f_k we have $P \geq t$ and $Q \leq t - 1$. Now each input to the top gate is a function g_i that satisfies the hypothesis of Lemma 9. Then using (1) and (2) we get

$$P \leq \left(\frac{2}{5} + 2^{1-\epsilon N}\right) \sum_{i \in S_2} w_i + \sum_{i \in S_3} w_i$$

$$Q \geq \left(\frac{3}{5} - 2^{-\epsilon N}\right) \sum_{i \in S_2} w_i + \sum_{i \in S_3} w_i$$

which implies

$$P - Q \leq \left(3 \cdot 2^{-\epsilon N} - \frac{1}{5}\right) \sum_{i \in S_2} w_i \leq 0$$

since all $w_i \geq 0$ and we have reached a contradiction.

5. Proof of Lemma 9

Consider the following lemma:

Lemma 10. *Let g be a function that is computed by a monotone weighted threshold circuit of depth $k-2$ and size $\leq 2^{\epsilon N}$, then for $N > N_0$,*

1. $g(\bar{q}_k) \geq 2^{-2\epsilon N} \Rightarrow g(\bar{p}_k) > \frac{3}{5}$,
2. $g(\bar{q}_k) \geq \frac{2}{5} \Rightarrow g(\bar{p}_k) > 1 - 2^{-2\epsilon N}$.

Let us prove that this lemma is equivalent to Lemma 9. Suppose g is computed by a threshold circuit as described by the lemmas. We claim that its negation is computed by a circuit that is identical except that the inputs are negated and if a certain gate in the circuit computing g has fanin R, total weight $w = \sum_1^R w_i$ and threshold t then the corresponding gate in the circuit computing $\bar{g}$ also has the same weights w_i but threshold $w + 1 - t$. We leave it to the reader to verify this. The equivalence of the two lemmas is now obvious. In particular, we see that the first part of Lemma 9 is equivalent to the second part of Lemma 10 and the other way around.

Let us now prove the lemmas. We proceed by induction and since the four statements in the two lemmas are pairwise equivalent, we need only prove the first part of each lemma for each k. On the other hand clearly we can use both parts of the induction hypothesis.

We start by proving Lemma 9 for the base case, $k = 2$. A threshold circuit of depth 0 is just a variable and thus we need only compute the probability that a variable is 1 in the two distributions. The distribution p_2 gives the value 1 to a variable with probability $\frac{1}{N}$ while the distribution q_2 gives it with probability $1 - \frac{1}{N}$. Thus Lemma 9 is true for $k = 2$ and sufficiently large N.

Now for the induction case assume that k is even (we will later see that odd k is almost the same, but we postpone this point) and that $g(p_k) \geq 2^{-2\epsilon N}$. We want to prove that $g(q_k) \geq \frac{3}{5}$. Consider the depth $k-2$ circuit that computes g and look at its top gate. Assume that it has R inputs and threshold t. The i^{th} input of the circuit corresponds to a function g_i that is computed by a depth $k-3$ threshold circuit of size $\leq 2^{\epsilon N}$.

First observe that we can erase all inputs corresponding to an i such that $g_i(p_k) \leq 2^{-4\epsilon N}$ without decreasing the value of $g(p_k)$ by more than $2^{-3\epsilon N}$. For notational convenience we ignore this tiny error term.

Now define distributions p_k^l, $l = 1, 2 \ldots n$ which look like p_k on the variables corresponding to the l^{th} input node to the top node of f_k but assigns ones to all other variables. Using that each variable is labelled by a $2k-2$ tuple of numbers between 1 and N, then a random setting of the variables according to the distribution p_k^l can be described as follows.

Any variable whose label does not start with l is set to 1. A random value m_0 between 1 and N is picked. The variables $x_{l,m_0,*}$ (we use this notation for set of variables whose first two labels are l and m_0) are set according to p_{k-1} while $x_{l,m,*}$ are set to 0 for $m \neq m_0$.

We have a very fruitful relationship between $g_i(p_k)$ and $g_i(p_k^l)$. Since this only depends on the fact that g_i is monotone we state a general fact.

Lemma 11. *For any monotone function h we have $h(p_k) \leq \prod_{l=1}^N h(p_k^l)$.*

PROOF. An instance of p_k corresponds to an instance of each of p_k^l since p_k^l only really lives on variables with first label l. If any of these instance of p_k^l forces h to 0 so does p_k. The lemma now follows. □

Observe that from this lemma and the assumption that $g_i(p_k) \geq 2^{-4\epsilon N}$ it follows that there are at most $\frac{4\epsilon N}{\log 5 - 1} < 4\epsilon N$ different l such that $g_i(p_k^l) \leq \frac{2}{5}$.

Let q_k^l be the distribution that is similar to q_k but where we fix the random choice of where to put the copies of q_{k-1} to be the l^{th} branch out of the top $\wedge$. We have the following immediate observation:

Lemma 12. *For any function h we have $h(q_k) = \frac{1}{N}\sum_{l=1}^N h(q_k^l)$.*

PROOF. This just follows from the fact that choosing an instance from q_k can be viewed as choosing a random l and then choosing a random instance from q_k^l. □

Let us next see the relation between $g_i(p_k^l)$ and $g_i(q_k^l)$.

Lemma 13. *We have*

1. $g_i(p_k^l) \geq 2^{-2\epsilon N} \Rightarrow g_i(q_k^l) > \frac{3}{5}$,
2. $g_i(p_k^l) \geq \frac{2}{5} \Rightarrow g_i(q_k^l) > 1 - 2^{-2\epsilon N}$.

PROOF. Clearly we want to establish this by the induction hypothesis. First observe that g_i is computed by a depth $k-3$ threshold circuit of size $\leq 2^{\epsilon N}$ and thus the only problem to apply the induction hypothesis is that the probability distributions are not quite correct.

Observe first that both p_k^l and q_k^l give the value 1 to all variables whose first index is not l and thus we can disregard those variables. Now let m_0 be the random choice in the definition of p_k^l that maximizes $g_i(p_k^l)$. Define two distributions p_k^{l,m_0} and q_k^{l,m_0} where the first distribution is the part of p_k^l where the random choice is actually m_0 and the second distribution is obtained from q_k^l by setting $x_{l,m,*}$ to 0 for all $m \neq m_0$. Since g_i is monotone we get

$$g_i(q_k^l) \geq g_i(q_k^{l,m_0}).$$

Now clearly the induction hypothesis applies to the pair of distributions p_k^{l,m_0} and q_k^{l,m_0} (since they are just p_{k-1} and q_{k-1} on the variables with two first indices are l and m_0 respectively, and all other variables are constants) and finally by the choice of m_0 we have

$$g_i(p_k^l) \leq g_i(p_k^{l,m_0}).$$

Adding these facts together we have proved the lemma. □

Next, call i *small for* l if $g_i(p_k^l) < \frac{2}{5}$. Otherwise call i *large for* l. By the previous lemma we know that when i is large for l then $g_i(q_k^l) > 1 - 2^{-2\epsilon N}$. Let N_0 be some absolute constant and remember that the top gate of the circuit computing g has R inputs, weights w_i and threshold t.

We call l *strong* if

$$\sum_{\substack{i \text{ large} \\ \text{for } l}} w_i \geq t$$

otherwise l is *weak*.

Lemma 14. *If at least $2N/3$ different l are strong, then $g(q_k) > \frac{3}{5}$ for $N \geq N_0$.*

PROOF. Take any strong l. We claim that $g(q_k^l) \geq 1 - R \cdot 2^{-2\epsilon N} \geq 1 - 2^{-\epsilon N}$. The reason is that if for all i large for l, the value of g_i is 1, then also g is 1 (since the large i have combined weight at least t for a strong l). The probability that an individual g_i is not 1 is bounded by $2^{-2\epsilon N}$ for a large i, and the claim follows. The lemma now follows by Lemma 12 applied to g. □

To complement the above lemma we have:

Lemma 15. *If at least $N/3$ different l are weak then $g(p_k) < 2^{-2\epsilon N}$ for $N \geq N_0$.*

PROOF. We use Lemma 11 (mainly its proof). Let $w = \sum_1^R w_i$. By the remark after the lemma, for each i there are at most $4\epsilon N$ different l such that i is small for l.

We again use the characterization that an instance $x^{(k)}$ of p_k can be viewed as choosing independent copies $x^{(k),l}$ of p_k^l for $l = 1, 2 \ldots N$ and if one of the instances $x^{(k),l}$ forces g_i to 0, then so does $x^{(k)}$. We choose the pieces $x^{(k),l}$ in a specific order to facilitate the analysis of how many g_i are forced to 0. Say that an i is *alive* if g_i is not forced to 0 by the $x^{(k),l}$ chosen this far (otherwise it is *dead*). Let an l be *unused* if $x^{(k),l}$ is not chosen yet. Now consider the following procedure:

```
For $j = 1, 2 \ldots N/4$
Let $l_j$ be the unused $l$ with maximum combined weight on its small $i$ that
are alive. Choose $x^{(k),l_j}$.
```

Clearly $g(p_k)$ is bounded by the probability that the combined weight of the dead i is at most $w - t$ after the above procedure. We analyze this probability.

Claim: If the weight of the dead i is at most $w - t$ at stage j then the weight of live i that are small for l_j is at least $(w - t)/2$.

Just consider the unused part of the set of at least $N/3$ different l that initially were weak. Of these l, at least $N/3 - j$ are still unused and on these unused l we initially had weight at least $(N/3 - j)(w - t)$. We know that the weight of the dead i is at most $w - t$ and since each i is small for at most $4\epsilon N$ different l, at most weight $4\epsilon N(w - t)$ is lost by these casualties. Hence one of the unused l has weight at least

$$\frac{N/3 - j - 4\epsilon N}{N/3 - j}(w - t) \geq \frac{w - t}{2}$$

on its small i:s since $\epsilon < \frac{1}{96}$. This establishes the claim.

Let $s_j = \sum_{\substack{i \text{ live and} \\ \text{small for } l_j}} w_i.$

Consider the weight on the live i that are small for l_j and that are killed by $x^{(k),l_j}$. The expected weight killed is at least $3s_j/5$ and the maximum is s_j. Thus by standard reasoning, with probability at least 1/2, at least $s_j/10$ are killed. By the claim this means that for the weight of killed i to remain below $w - t$ an event that happens with probability 1/2 must happen at most 9 times in $N/4$ trials. The probability of this is 2^{-cN} for some $c > .02 = 2\epsilon$ and sufficiently large N. This completes the proof of Lemma 15. □

Obviously Lemmas 14 and 15 imply statement 1 of Lemma 9. Next we complete the proof when k is even by establishing 1 of Lemma 10. We need to prove that if $g(\bar{q}_k) \geq 2^{-2\epsilon N}$ then $g(\bar{p}_k) > \frac{3}{5}$. Remember that $\bar{q}_k$ is defined by first picking an element from $\bar{q}_2$ and then changing each 0 to all zeroes and each 1 to a copy of $\bar{q}_{k-1}$. The $\bar{q}_2$ distribution is chosen by first choosing an input, l, to the top $\vee$ gate and then letting each input to that gate take the value 1 while all other inputs are given the value 0. If we fix the first choice in $\bar{q}_k$ to a given l we get a distribution which we call $\bar{q}_k^l$. Choose l to maximize $g(\bar{q}_k^l)$ and call it l_0. Now let $\bar{p}_k^{l_0}$ be the distribution that looks like $\bar{p}_k$ except that all variables with first index $\neq l_0$ are set to 0. Since g is monotone we have $g(\bar{p}_k) \geq g(\bar{p}_k^{l_0})$ and by the choice of l_0 we have $g(\bar{q}_k) \leq g(\bar{q}_k^{l_0})$ and we thus only need to establish the desired relation between $g(\bar{p}_k^{l_0})$ and $g(\bar{q}_k^{l_0})$, and from now on we only look at variables with first index l_0.

Let $\bar{q}_k^{l_0,m}$ be the distribution that looks like $\bar{q}_k^{l_0}$ except that all variable with second index not equal to m are given the value 1. Now we are basically in the same situation as before. First we have

Lemma 16. *For any monotone function h we have $h(\bar{q}_k^{l_0}) \leq \prod_{m=1}^n h(\bar{q}_k^{l_0,m})$.*

PROOF. As Lemma 11. □

Now let $\bar{p}_k^{l_0,m}$ be the distribution $\bar{p}_k^{l_0}$ where the random choice on the second level is replaced by the special choice m. Corresponding to Lemma 12 we have

Lemma 17. *For any function h we have $h(\bar{p}_k^{l_0}) = \frac{1}{N}\sum_{m=1}^N h(\bar{p}_k^{l_0,m})$.*

As before, we have by induction

1. $g_i(\bar{q}_k^{l_0,m}) \geq 2^{-2\epsilon N} \Rightarrow g_i(\bar{p}_k^{l_0,m}) > \frac{3}{5}$,
2. $g_i(\bar{q}_k^{l_0,m}) \geq \frac{2}{5} \Rightarrow g_i(\bar{p}_k^{l_0,m}) > 1 - 2^{-2\epsilon N}$.

Using a proof with the same outlines as those for Lemmas 14 and 15, we get

Lemma 18.

$$g(\bar{q}_k^{l_0}) \geq 2^{-2\epsilon N} \Rightarrow g(\bar{p}_k^{l_0}) > \frac{3}{5}.$$

Now, by the choice of l_0 part 1 of Lemma 10 follows. This completes the proof for k even.

The proof when k is odd is now easy. We just need to observe that when k is odd the two cases just switch (i.e. the proof of 1 of Lemma 9 is now the proof of 1 of Lemma 10 and the other way around). The reason being that f_k now has a top $\vee$-gate and $\bar{f}_k$ has a top $\wedge$-gate while the recursive structure is the same. We have completed the proof of Lemma 9.

Acknowledgements

We would like to thank Andy Yao for sending an early draft of his paper and for encouraging us to publish the present paper. We are also grateful to Oded Goldreich and Richard Beigel for some helpful comments. This research was partly supported by the Swedish board for technical development (STU).

References

[1] M. Ajtai. $\sum_1^1$-formulae on finite structures. *Annals of Pure and Applied Logic*, 24:1–48, 1983.

[2] E. Allender. A note on the power of threshold circuits. *Proceedings 30'th Annual Symposium on Foundations of Computer Science*, pages 580–584, 1989.

[3] N. Alon and R. B. Boppana. The monotone circuit complexity of boolean functions. *Combinatorica*, 7:1–22, 1987.

[4] A. E. Andreev. On a method for obtaining lower bounds for the complexity of individual monotone functions. *Dokl. Ak. Nauk. SSSR 282*, pages 1033–1037, 1985. English translation in *Sov. Math. Dokl.*, 31:530–534, 1985.

[5] L. Babai, N. Nisan, and M. Szegedy. Multipary protocols and logspace-hard pseudorandom sequences. *Proceedings of 21st Annual ACM Symposium on Theory of Computing*, pages 1–11, 1989.

[6] P. W. Beame, S. A. Cook, and H.J. Hoover. Log depth circuits for division and related problems. *Proceedings 25'th Annual Symposium on Foundations of Computer Science*, pages 1–6, 1984.

[7] A. Chandra, L. Stockmeyer, and U. Vishkin. Constant depth reducibility. *SIAM J. on Computing*, 13:423–439, 1984.

[8] B. Chor and O. Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM J. on Computing*, 17:230-261, 1988.

[9] M. Furst, J. Saxe, and M. Sipser. Parity, circuits, and the polynomial time hierarchy. *Math. System Theory*, 17:13–27, 1984.

[10] J. Håstad. *Computational Limitations of Small-Depth Circuits.* MIT PRESS, 1986.

[11] J. Håstad and M. Goldmann. On the power of small-depth threshold circuits. *Proceedings 31st Annual IEEE Symposium on Foundations of Computer Science*, pages 610–618, 1990.

[12] A. Hajnal, W. Maass, P. Pudlak, M. Szegedy, and G. Turán. Threshold circuits of bounded depth. *Proceedings 28th Annual IEEE Symposium on Foundation of computer science*, pages 99–110, 1987.

[13] M. Karchmer and A. Wigderson. Monotone circuits for connectivity require super-logarithmic depth. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing*, 1988.

[14] R. Raz and A. Wigderson. Probabilistic communication complexity of boolean relations. In *Proceedings of the 30th Annual IEEE Symposium on Foundation of computer science*, pages 562–567, 1989.

[15] R. Raz and A. Wigderson. Monotone circuits for matching require linear depth. *22nd annual ACM Symposium on Theory of Computing*, pages 287-292, 1990.

[16] A. A. Razborov. Lower bounds on monotone network complexity of the logical permanent. *Matem. Zam.*, 37(6):887–900, 1985. English translation in *Math. Notes of the Academy of Sciences of the USSR*, 37:485–493, 1985.

[17] A. A. Razborov. Lower bounds on the size of bounded-depth networks over a complete basis with logical addition. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):598–607, 1987. English translation in 41:4, pages 333-338.

[18] M. Sipser. Borel sets and circuit complexity. In *Proceedings of 15th Annual ACM Symposium on Theory of Computing*, pages 61–69, 1983.

[19] R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. *Proceedings of 19th Annual ACM Symposium on Theory of Computing*, pages 77–82, 1987.

[20] S. Toda. On the computational power of PP and $\oplus P$. *Proceedings 30th Annual IEEE Symposium on Foundations of Computer Science*, pages 514–519, 1989.

[21] A. Yao. Separating the polynomial-time hierarchy by oracles. *Proceedings 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 1–10, 1985.

[22] A. Yao. Circuits and local computation. *Proceedings of 21st Annual ACM Symposium on Theory of Computing*, pages 186–196, 1989.

[23] A. Yao. On ACC and threshold circuits. *Proceedings 31st Annual IEEE Symposium on Foundations of Computer Science*, pages 619–627, 1990.

Manuscript received 14 June 1990

JOHAN HÅSTAD
Department of Numerical Analysis and Computing Science
Royal Institute of Technology
S-100 44 Stockholm
SWEDEN
`johanh@nada.kth.se`

MIKAEL GOLDMANN
Department of Numerical Analysis and Computing Science
Royal Institute of Technology
S-100 44 Stockholm
SWEDEN
`migo@nada.kth.se`