Polynomial Size Constant Depth Circuits with a Limited Number of Negations

M. Santha — Université Paris-Sud
C. Wilson — University of Oregon

CIS-TR-91-10
April 1991

Abstract

It follows from a theorem of Markov that the minimum number of negation gates in a circuit sufficient to compute any Boolean function on n variables is $l = \lfloor \log n \rfloor + 1$. It can be shown that, for functions computed by families of polynomial size, $O(\log n)$ depth and bounded fan-in circuits (NC^1), the same result holds: on such circuits l negations are necessary and sufficient. In this paper we prove that this situation changes when polynomial size circuit families of constant depth are considered: l negations are no longer sufficient. For threshold circuits we prove that there are Boolean functions computable in constant depth (TC^0) such that no such threshold circuit containing $o(n^\epsilon)$, for all $\epsilon > 0$, negations can compute them. We have a matching upper bound: for any $\epsilon > 0$, everything computed by constant depth threshold circuits can be so computed using n^ϵ negations asymptotically. We also have tight bounds for constant depth, unbounded fan-in circuits (AC^0): $n/\log^r n$, for any r, negations are sufficient, and $\Omega(n/\log^r n)$, for some r, are necessary.

Department of Computer and Information Science
University of Oregon

1 Introduction

Although extensively studied, not very much is known about the circuit complexity of Boolean functions. The results are especially few concerning lower bounds. While it is conjectured that NP-complete problems can not be computed with circuits of less than exponential size, the best known lower bounds are linear with small constants. In striking opposition to this situation, important progress has been made recently on monotone circuits. In his famous result Razborov [14] has proved a superpolynomial lower bound on the monotone circuit complexity of an appropriate clique function. Later this lower bound was strenghtened to exponential size by Alon and Boppana [5]. In another development, Tardos [15] pointed out that there are even problems in P whose monotone circuit complexity is exponential, thus proving that negation may be exponentially powerful.

Of course, one would like to extend Razborov's lower bound result to the general model. As this seems to be at the moment quite elusive, a natural intermediate step is the study of circuits with a limited number of negations. If negations are also permitted in the circuit, then we should not restrict the study just to monotone functions. But if we consider also non-monotone functions in our investigations, then before the study of lower bounds there is an even more basic question: can a given function be computed at all with a limited number of negations?

This question was answered by Markov [12] without any complexity theoretical considerations. He defined for any Boolean function $f : \{0,1\}^n \rightarrow \{0,1\}^m$, the *inversion* complexity $inv(f)$ of f as the minimum number of negation gates contained in a circuit which computes f. Let $f = (f_1, \ldots, f_m)$, and let x and y be two Boolean vectors in $\{0,1\}^n$. The ordered pair (x, y) is a *gap* for f if $x < y$ and for some j, $1 \leq j \leq m$, $f_j(x) > f_j(y)$. Let $x_1 < \ldots < x_r$ be an increasing sequence of Boolean vectors in $\{0,1\}^n$. The *decrease* of f on the sequence $x_1, \ldots, x_r$ is the number of indices i such that (x_i, x_{i+1}) is a gap for f. Finally the *decrease* $dec(f)$ of f is the maximum decrease over all increasing sequence of Boolean vectors. The result of Markov establishes a precise relationship between the inversion complexity of f and its decrease.

Theorem (Markov) *For every Boolean function f we have*

$$inv(f) = \lfloor \log(dec(f)) \rfloor + 1.$$

As the length of any increasing sequence of n-dimensional Boolean vectors is at most $n+1$, by Markov's Theorem $\lfloor \log n \rfloor + 1$ negations are sufficient to compute any Boolean function on n variables. On the other hand, it is easy to find a very simple function f for which there exists an increasing sequence of vectors with n gaps. Thus, for some functions $\lfloor \log n \rfloor + 1$ negations are also necessary.

In this paper we will study what remains true of this necessary and sufficient condition when restrictions are imposed on the size and depth of the circuits computing f. The restriction we will impose on the size of the circuits is polynomial size. Thus, the question we would like to answer is the following: Let $f : \{0,1\}^* \rightarrow \{0,1\}^*$ be a Boolean function which can be computed by a family of polynomial size circuits. Is it true that f can also be computed with a family of polynomial size circuits which contain at most $\lfloor \log n \rfloor + 1$ negations?

It turns out that the answer strongly depends on whether any restriction is put on the depth of the circuits. If f can be computed in depth $d(n)$, where $d(n) = \Omega(\log n)$, then the answer is *yes* (**Theorem 1**): f can be computed in the same order of depth with $\lfloor \log n \rfloor + 1$ negations, even if the underlying model has bounded fan-in. This result is implicitly contained in an early survey paper of Fischer [7], where he also considers circuits with limited negation.

Our results on the other hand show that the answer is *no* for constant depth circuits. In the case of threshold circuits we show that there exists a function computable in constant depth which can not be computed in constant depth on threshold circuits using $o(n^\epsilon)$, for all $\epsilon > 0$, negations (**Corollary 2**). We also establish a matching upper bound on the number of negations sufficient for constant depth (**Theorem 4**): For any $\epsilon > 0$, every function which can be computed in constant depth on a family of threshold circuits can be computed in constant depth by threshold circuits with n^ϵ negations asymptotically. This will give us a sublinear bound on the number of negations needed for AC^0 circuits (**Theorem 5**): For any r, every Boolean function computable in constant depth can also be computed in constant depth with at most $n/\log^r n$ negations. This is the best bound one can obtain (**Corollary 4**): There is a function computable in constant depth which cannot be computed in constant depth with $o(n/\log^r n)$, for all $r > 0$, negations.

The tight lower bounds of Corollary 2 and Corollary 4 are obtained from trade-off results between depth and number of negations in constant depth. **Theorem 2** says that depth d threshold circuits for NEG (see the definition in Section 2) require $d(n+1)^{1/d} - d$ negations, and **Theorem 6** claims that any circuit family computing NEG in depth d has $\Omega(n/\log^{d+3} n)$ negations. We can also prove that depth d threshold circuits for $PARITY$ have $d(\lceil n/2 \rceil)^{1/d} - d$ negations. However, we are not able to obtain a tight lower bound on the number of negations reguired by a constant depth AND/OR circuit for a *single-valued* function (NEG has n outputs). See Section 5 for more comments on this problem.

Let us mention at this point a result of Okolnishnikova [13] and Ajtai and Gurevich [2] related to our Theorem 2: There exists a monotone function which can be computed with polynomial size, constant depth circuits, but can not be computed with monotone, polynomial size, constant depth circuits.

The paper is organized as follows: After some preliminaries, Section 2 contains a short outline of the proof of the already known result about circuits with $\Omega(\log n)$ depth. Section 3 deals with upper and lower bounds for constant depth threshold circuits. Section 4 derives upper and lower bounds for unbounded fan-in *AND/OR* circuits. Finally, in Section 5 we conclude and mention some open problems.

2 Preliminaries

We will use standard notions from circuit complexity theory, for which the reader is referred *e.g.* to Wegener's book [16]. We will also use some conventions throughout the paper. When it is not otherwise specified, we will deal with circuits on n variables. Let x denote the Boolean vector $(x_1, \ldots, x_n)$, and $x - x_i$ the vector $(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$. We often identify the vector x with the word $x_1 \ldots x_n$. For $w \in \{0,1\}^*$, the *weight* of w is the number of ones in w, denoted by $|w|$. If f is a one-output Boolean function, then $\bar{f}$ is the negation of f.

Two circuits are *equivalent* if they compute the same function. A circuit is *monotone* if it does not contain any negation gates. Ignoring uniformity considerations, for $i \geq 0$, the classes NC^i and AC^i are defined to be the set of functions computable by polynomial size, $O(\log^i n)$ depth circuit families with bounded and unbounded fan-in, respectively.

Another important class of circuits we examine is that of threshold circuits. By definition, for $k = 0, \ldots, n$, the k^{th} *threshold* function $T_k(x) = 1$ if and only if $|x| \geq k$. The class TC^i is defined to be the class of functions computed by a family of polynomial size, $O(\log^i n)$ depth circuits consisting of negations and gates which compute threshold functions. It is known that $NC^i \subseteq AC^i \subseteq TC^i \subseteq NC^{i+1}$. An especially interesting problem recently has been that of separating these classes when $i = 0$. It is known that $AC^0 \subset TC^0$ ([9]), but $TC^0 \subseteq NC^1$ is still open. In [8] it is shown that depth 2 threshold circuits are weaker than depth 3 threshold circuits. In [17] it is shown that depth k monotone threshold circuits are weaker than depth $k+1$ monotone threshold circuits, for any k.

The *sorting* function $S(x)$, and the *exact* function $E(x)$ are closely related to threshold functions. By definition $S(x) = (T_1(x), \ldots, T_n(x))$, and $E(x) = (E_0(x), \ldots, E_n(x))$, where $E_k(x) = 1$ iff $|x| = k$. Indeed, $S(x)$ is the simultaneous computation of all the non trivial threshold functions, $E_n(x) = T_n(x)$, and $E_k(x) = T_k(x) \wedge \bar{T}_{k+1}(x)$ for $0 \leq k \leq n-1$. These functions will be extensively used as well as the function NEG, defined by $NEG(x_1, \ldots, x_n) = (\bar{x}_1, \ldots, \bar{x}_n)$.

Let C be the class of functions computed by a class of families of polynomial size circuits, and let $g(n)$ be a function from the natural numbers to the natural numbers. Then $C_{g(n)}$ is the set of functions which can be computed by a circuit family in the class which contains at most $g(n)$ negation gates. The class C_0 will be denoted *mon-C*, this is the set of functions computable by a monotone circuit family in the class. By the *type* of a circuit we mean bounded fan-in, unbounded fan-in or threshold. The following facts are well known.

Fact 1 (Ajtai, Komlós and Szemerédi [3]) *The sorting function $S(x)$ is in mon-NC^1.*

Fact 2 (Ajtai and Ben-Or [1]) *For every $t \geq 0$, the threshold function $T_{\log^t n}(x)$ is in mon-AC^0.*

The importance of the function NEG lies in the fact that it incorporates all the "non-monotone" information one needs to compute any function by a circuit. This is expressed in the following Completeness Lemma.

Lemma 1 (Completeness Lemma) *Let C be a class of functions computed by families of polynomial size circuits of the types described above, such that the allowable depth of the circuit families is closed under multiplication by a constant. Let $g(n)$ be a function on the natural numbers. Then we have*

$$C_{g(n)} = C \quad \text{if and only if} \quad NEG \in C_{g(n)}.$$

Proof The implication is straightforward from left to right. The other direction is implied by the following well known result (see *e.g.* Wegener [16]): For every circuit C of size s and depth d, there exists a monotone circuit C' of the same type, of size at most $2s$ and depth d which is equivalent to C, when the output of $NEG(x)$ is also given as input to C'. □

The function NEG can easily be computed by a monotone AC^0 circuit from the outputs of the sorting function and the exact functions. This is stated in the following lemma.

Lemma 2 *For $1 \leq i \leq n$, we have*

$$\bar{x}_i = \bigvee_{k=0}^{n} (T_k(x - x_i) \ \wedge \ E_k(x)).$$

Proof First we claim that for any $1 \leq i \leq n$, $\bar{x}_i = T_{|x|}(x - x_i)$. If $\bar{x}_i = 1$ then the number of ones in $x - x_i$ is $|x|$ and $T_{|x|}(x - x_i) = 1$. On the other hand, if $\bar{x}_i = 0$ then the number of ones in $x - x_i$ is $|x| - 1$ and $T_{|x|}(x - x_i) = 0$. The result follows since $E_k(x) = 1$ iff $k = |x|$. □

Fischer [7] constructed a circuit which contains only $\lfloor \log n \rfloor + 1$ negations, and computes the exact function, when the inputs are already sorted. The size of the circuit is polynomial, and its depth is $O(\log n)$. This enabled him to compute the exact functions by a polynomial size circuit family containing $\lfloor \log n \rfloor + 1$ negations. The depth of his circuit family depended on the circuit depth of the sorting function, which was at that time still an open problem. Today, it is known (Fact 1) that threshold functions can in fact be computed in *mon*-NC^1. This means that Fischer's result implicitly implies the following theorem.

Theorem 1 *For every circuit family of polynomial size and depth $d(n)$, there exists an equivalent circuit family of the same type, also of polynomial size and depth $d(n) + O(\log n)$ which contains only $\lfloor \log n \rfloor + 1$ negations.*

Corollary 1 *For all $k \geq 1$, we have*

$$\begin{aligned} &i) \quad NC^k_{\lfloor \log n \rfloor + 1} = NC^k, \\ &ii) \quad AC^k_{\lfloor \log n \rfloor + 1} = AC^k, \\ &iii) \quad TC^k_{\lfloor \log n \rfloor + 1} = TC^k. \end{aligned}$$

This method of computing NEG in $NC^1_{\lfloor \log n \rfloor + 1}$ can be viewed as a constructive (and efficient) implementation of Markov's result. This, to some extent, was foreshadowed by Akers [4]. An analysis of his method reveals that NEG can be computed using few negations in $O(\log n)$ depth using threshold circuits; that is, $TC^1_{\lfloor \log n \rfloor + 1}$.

3 Bounds for Threshold Circuits

3.1 Lower Bounds

Here we shall prove that it is impossible to compute NEG on a depth d threshold circuit which uses fewer than $d(n+1)^{1/d} - d$ negations. In the proof of this lower bound we will concentrate on inputs which are integers in unary notation. These inputs are sequences of n bits with the ones preceeding the zeroes. When we say that j is the input value, we mean that $1^j 0^{n-j}$ is the input string.

With each gate g in a circuit we associate a satisfying set $I_g \subseteq \{0, \ldots, n\}$ such that gate g outputs 1 if and only if input $j \in I_g$. For example, the satisfying set of x_i is $[i, n]$, that of $\bar{x}_5 \vee x_{10}$ is $[0, 5) \cup [10, n]$, and that of $x_5 \wedge \bar{x}_{10}$ is $[5, 10)$.

Let $I \subseteq \{0, \ldots, n\}$. We define j as a *right boundary* of I if $j \in I$ and $j + 1 \notin I$. The value j is a right boundary of a gate of a circuit if it is a right boundary of the satisfying set of the gate, and j is a right boundary of a circuit if it is the right boundary of some of its gates. For example, the unique right boundary of $\bar{x}_i$ is $i - 1$. In what follows, it is important to count the number of right boundaries of a circuit since these are the inputs where it behaves non-monotonically. First we will note that the only way to create new right boundaries is by the use of negations.

Lemma 3 *Let C be a circuit of any type whose gates are either negations or monotone functions. Suppose that gate g is a monotone function of its input gates. Then any right boundary of g is a right boundary of at least one of its input gates.*

Proof Let us suppose that $g = f(g_1, \ldots, g_r)$, where f is a monotone function and $g_1, \ldots, g_r$ are the input gates to g. We claim that if j is not a right boundary of g_i, for $1 \leq i \leq r$, then j is not a right boundary for g either. This is true since $g_i(j) \leq g_i(j+1)$, for $1 \leq i \leq r$, implies $g(j) \leq g(j+1)$ by the monotonicity of f. □

Theorem 2 *Let C be a circuit computing NEG on inputs of size n. Suppose that C has depth d, uses ν negations, and has gates which are either negations or arbitrary monotone functions. Then $\nu \geq d(n+1)^{1/d} - d$.*

Proof For $0 \leq i \leq d$, let level i of the circuit consist of all gates whose longest path to an input is of length i. Level 0 consists of inputs and constants, and thus only presents the single right boundary n. The circuit must eventually create n other right boundaries. Our bound will follow by showing that added depth can create only few right boundaries if insufficient negations are available.

Observe what happens when a node is negated. If gate g has k right boundaries and satisfying set $[i_1, i_2) \cup [i_3, i_4) \cup \cdots \cup [i_{2k-1}, i_{2k})$, then $\bar{g}$ has satisfying set $[0, i_1) \cup [i_2, i_3) \cup \cdots \cup [i_{2k}, n]$. This creates up to k new right boundaries. By the previous lemma, no other type of gate can create new right boundaries. Thus, if up to some level the gates present altogether t right boundaries and at the next level μ gates are negated, this next level creates at most $t\mu$ new right boundaries. This gives a total of $t(1+\mu)$ possible right boundaries up to the next level.

For $i = 1, \ldots d$ let ν_i be the number of negations at level i, where $\sum_{i=1}^{d} \nu_i = \nu$. By the above, the circuit can create at most $\prod_{i=1}^{d}(1+\nu_i)$ right boundaries. This product is maximized when $\nu_i = \nu/d$. Since we must have $(1+\nu/d)^d \geq n+1$, it follows that $\nu \geq d(n+1)^{1/d} - d$. □

Similarly we can prove that if C is a circuit computing $PARITY$ which satisfies the conditions of Theorem 2, then $\nu \geq d(\lceil n/2 \rceil)^{1/d} - d$. As threshold gates are monotone, the following corollary is immediate.

Corollary 2 *Let $g(n) = o(n^\epsilon)$ for all $\epsilon > 0$. Then*

1. $NEG,\ PARITY \notin TC^0_{g(n)}$
2. $TC^0 \neq TC^0_{g(n)}$

Notice that these results did not put any restrictions on the size of the circuit. Even exponentially many monotone gates are of little use without enough negations. We can state something strong about depth as well. For example, no family of threshold circuits of depth $(\log n)^{1-\epsilon}$, $\epsilon > 0$, with $(\log n)^r$ negations can compute NEG or $PARITY$. Furthermore, Corollary 2 immediately implies that $NC^1_{poly\text{-}log}$ properly contains $TC^0_{poly\text{-}log}$.

Corollary 3 *Let $g(n) \geq \log n$ and for all $\epsilon > 0$, $g(n) = o(n^\epsilon)$. Then $TC^0_{g(n)} \neq NC^1_{g(n)}$.*

It is interesting to compare Corollary 3 to the result in [17] showing that *mon-*TC^0 is properly contained in *mon-*NC^1. Corollary 3 can also be generalized to show a separation of NC^1 and TC^0 restricted to, say, $\log\log n$ negations. This generalization does not subsume the result in [17].

Theorem 3 *Let $0 < f(n) \leq \lfloor \log n \rfloor + 1$. If $\forall \epsilon > 0$ $g(n) = o(2^{\epsilon f(n)})$, then*

$$NC^1_{f(n)} - TC^0_{g(n)} \neq \emptyset.$$

Proof Let $k = n/2^{f(n)-1}$ and consider the function

$$e(x) = (x_0 \wedge \bar{x}_k) \vee (x_{2k} \wedge \bar{x}_{3k}) \vee \cdots \vee (x_{n-k} \wedge \bar{x}_n)$$

(x_0 by default is 1). This can be viewed as a function on $\frac{n}{k} = 2^{f(n)-1}$ inputs, so by Theorem 1 is in $NC^1_{f(n)}$. Following the proof of Theorem 2, $e(x)$ has $\frac{n}{2k}$ right boundaries. Thus, any depth d threshold circuit with ν negations computing $e(x)$ must have $(1 + \nu/d)^d \geq \frac{n}{2k} = 2^{f(n)-2}$ or $\nu \geq d2^{(f(n)-2)/d} - d$. This is not possible under the assumption on $\nu = g(n)$ in the statement of the theorem. □

3.2 Upper Bounds

We have seen that everything computable in TC^0 cannot be computed using $o(n^\epsilon)$, for all $\epsilon > 0$, negations. The question arises naturally: how many negations are sufficient to give full power to TC^0? We can show that the lower bounds derived above for threshold circuits are essentially optimal.

Our main tool will be the computation of the exact function in constant depth on a threshold circuit. Since this is just two levels away from NEG, it is evident that we cannot compute it in depth $d-2$ using less than $d(n+1)^{1/d} - d$ negations. We will show how to compute it in depth $3d + O(1)$ using no more $dn^{1/d} - d + 1$ negations. Hence, the upper and lower bounds are nearly tight.

Lemma 4 *Let $d \geq 1$ be an integer. There exists a depth $3d + O(1)$ family of threshold circuits with $dn^{1/d} - d + 1$ negations computing the exact function E.*

Proof Set $x_0 = 1$ for the sake of convenience. We will assume that the input has been sorted as $x_1 \geq x_2 \geq \cdots \geq x_n$. This can be done in depth 1 on a threshold circuit, and this is the only place that we need threshold gates. The rest of the circuit will consist of negations and unbounded fan-in *AND/OR* gates.

The circuit we describe will have d layers, each layer will consist of several levels. Let us define the functions F_i^k for $0 \leq k \leq d$ and $0 \leq i < n^{k/d}$:

$$F_i^k = 1 \Leftrightarrow in^{(d-k)/d} \leq |x| < (i+1)n^{(d-k)/d}.$$

One layer of the circuit will use $n^{1/d} - 1$ negations in transforming the $\{F_i^k\}_{i=0}^{n^{k/d}-1}$ into the $\{F_i^{k+1}\}_{i=0}^{n^{(k+1)/d}-1}$. Clearly, the F_i^d are the desired outputs E_i, so long as we add $E_n = x_n$. The base case is simple as there is only one possible i: $i = 0$. We let $F_0^0 = \bar{x}_n$.

Given the F_i^k for $0 \leq i < n^{k/d}$, the following describes how to construct the F_i^{k+1} for $0 \leq i < n^{(k+1)/d}$ in constant depth.

(k,1) For $0 \leq i < n^{k/d}$ and $0 \leq j < n^{1/d}$ compute $G_{i,j}^k = F_i^k \wedge x_{in^{(d-k)/d} + jn^{(d-k-1)/d}}$.

(k,2) For $0 \leq j < n^{1/d}$ compute $H_j^k = \bigvee_{i=0}^{n^{k/d}-1} G_{i,j}^k$.

(k,3) For $1 \leq j < n^{1/d}$ compute $\bar{H}_j^k$.

(k,4) For $0 \leq j \leq n^{1/d} - 2$ compute $I_j^k = H_j^k \wedge \bar{H}_{j+1}^k$. Let $I_{n^{1/d}-1}^k = H_{n^{1/d}-1}^k$.

(k,5) For $0 \leq i < n^{k/d}$ and $0 \leq j < n^{1/d}$ compute $F_{in^{1/d}+j}^{k+1} = I_j^k \wedge G_{i,0}^k$.

We can describe when the functions above are satisfied.

1. $G_{i,j}^k = 1$ if and only if $in^{(d-k)/d} + jn^{(d-k-1)/d} \leq |x| < (i+1)n^{(d-k)/d}$.
2. $H_j^k = 1$ if and only if $\exists i$ $(1 \leq i < n^{k/d})$ $in^{(d-k)/d} + jn^{(d-k-1)/d} \leq |x| < (i+1)n^{(d-k)/d}$.
3. $\bar{H}_j^k = 1$ if and only if $\exists i$ $(1 \leq i < n^{k/d})$ $in^{(d-k)/d} \leq |x| < in^{(d-k)/d} + jn^{(d-k-1)/d}$.
4. $I_j^k = 1$ if and only if $\exists i$ $(1 \leq i < n^{k/d})$ $in^{(d-k)/d} + jn^{(d-k-1)/d} \leq |x| < in^{(d-k)/d} + (j+1)n^{(d-k-1)/d}$.
5. $F_{in^{1/d}+j}^{k+1} = 1$ if and only if $(in^{1/d} + j)n^{(d-k-1)/d} \leq |x| < (in^{1/d} + j + 1)n^{(d-k-1)/d}$, as desired.

The steps (k,4), (k,5), and ($k+1$,1) are all computed by $\wedge$ gates, so they can be combined into one level. This yields a circuit for the $E_i = F_i^d$ of depth $3d + O(1)$. The number of negations needed is $1 + d(n^{1/d} - 1)$. $\square$

Theorem 4 *For every $\epsilon > 0$, we have asymptotically*

$$TC^0_{n^\epsilon} = TC^0.$$

Proof Choose d so that $1/d < \epsilon$. For this d we have that $dn^{1/d} \leq n^\epsilon$ asymptotically. From Lemma 4 we can compute E in constant depth using asymptotically less than n^ϵ negations. From Lemma 2, we see that $\bar{x}_i = \bigvee_{k=0}^{n}(T_k(x - x_i) \wedge E_k(x))$. The Completeness Lemma then implies the result. $\square$

4 Constant Depth AND/OR Circuits

In this section we show that AC^0 remains invariant under a restriction to some sublinear number of negations: $n/\log^r n$ negations, for any r, are sufficient. Furthermore, this bound is tight.

The upper bound follows from a construction used in the previous section.

Theorem 5 *Let $r \geq 0$ and $g(n) = n/\log^r n$. Then we have*

$$AC^0_{g(n)} = AC^0.$$

Proof We will break the input up into n/N groups, each of size $N = 4\log^{2r} n$. According to Fact 2, T_N is in mon-AC^0. Thus, by fixing some of the inputs to the threshold gate, we can sort any group $y_1, \ldots, y_N$ in monotone constant depth and polynomial (in n) size.

As we have seen in the proof of Lemma 4, we can compute E from $y_1 \geq \cdots \geq y_N$ in constant depth using AND/OR gates and $2\sqrt{N}$ negations. The thresholds can be applied again to find $\bar{y}_i = \bigvee_{k=0}^{N}(T_k(y - y_i) \wedge E_k(y))$, as per Lemma 2. The total number of negations used will be $2\sqrt{N}(n/N) = n/\log^r n$. The result then follows from the Completeness Lemma. $\square$

Definition The *sensitivity on* a string w of the single valued function f, $s(f, w)$, is the number of neighbors w' of w differing in exactly one bit such that $f(w) \neq f(w')$. The *sensitivity* of f, $s(f)$, is the average over $w \in \{0,1\}^n$ of $s(f, w)$.

There have been several works in recent years relating the sensitivity of a Boolean function to its Fourier transform [10, 11]. To show a matching lower bound to Theorem 5, we make use of the following application of these results to AC^0 functions.

Lemma 5 [11] *If f is computed by a circuit family of depth d, then $s(f) = O(\log^{d+3} n)$.*

Theorem 6 *Let $\{C_n\}$ be a depth d circuit family which computes $NEG(x)$ with $\nu(n)$ negations. Then $\nu(n) = \Omega(n/\log^{d+3} n)$.*

Proof Let us suppose on the contrary that there is depth d circuit family $\{C_n\}$ which computes $NEG(x)$ with $\nu(n) \neq \Omega(n/\log^{d+3} n)$ negations. By Lemma 5 there exists a constant $c > 0$ and n_0 such that for every $n \geq n_0$, for every function f which is computed at some gate of C_n,

$$s(f) < c\log^{d+3} n.$$

Our hypothesis implies that there exists $n > n_0$ such that

$$\nu(n) < n/c\log^{d+3} n.$$

Let n be such an integer and let $f_1, \ldots, f_k$, $k \leq \nu(n)$, be the outputs of the negation gates of C_n.

Given a string w, we say a bit of w is sensitive to f if changing that bit changes the output of f on w. Otherwise, that bit is insensitive to f. It follows that there is a string w which has a bit insensitive to all $f_1, \ldots, f_k$. This is because

$$\begin{aligned} E[s(f_1, w) + \cdots + s(f_k, w)] &= E[s(f_1, w)] + \cdots + E[s(f_k, w)] \\ &\leq k\,c\log^{d+3} n < n, \end{aligned}$$

where E indicates the expectation of an event over all strings w of length n uniformly distributed.

Let w be a string whose jth bit is insensitive to all $f_1, \ldots, f_k$. Obtain w' by changing this bit. We can suppose that $w_j = 0$ and $w_j{}' = 1$, which implies $w < w'$. Since the bit is insensitive, we have $\forall i$, $f_i(w) = f_i(w')$. Thus, between w and w' the outputs of all negation gates of C_n are constant. This implies that no gate of C_n can take a greater value on w than on w'. However, C_n computes $\bar{x}_j$, so on input w it outputs $\bar{w}_j = 1$ and on w' it outputs $\bar{w}'_j = 0$. This is impossible if no negation gate changes. □

Corollary 4 *If $g(n) = o(n/\log^r n)$ for all r, then $AC^0 \neq AC^0_{g(n)}$.*

5 Conclusions and Open Problems

Let us say a few words about the uniformity of our circuits. In fact all of them are uniform, except the sorting circuit of the first layer in Theorem 5: the constant depth circuits of Fact 2 are probabilistic. This construction can be made uniform by using

Theorem 4.1 of [6]. That indicates how to find any threshold on m variables in uniform constant depth if m is bounded by a poly-log function of n, which is what is needed in Theorem 5.

Although the upper bound of Theorem 5 and the lower bound of Theorem 6 were matching, the lower bound held only for a multi-valued function. An intriguing problem is to find out the exact number of negations necessary and sufficient to compute every single-valued function in AC^0. We can show a weaker lower bound for some single valued function. Let the problem *EXISTODD*, defined on $x_1, \ldots, x_n$, be

$$(x_1 \wedge \bar{x}_2) \vee (x_3 \wedge \bar{x}_4) \vee \cdots \vee (x_{n-1} \wedge \bar{x}_n)$$

(if n is odd, then the formula will be $\cdots \vee (x_{n-2} \wedge \bar{x}_{n-1}) \vee x_n$). Since *EXISTODD* on inputs of the form $1^j 0^{n-j}$ is equivalent to *PARITY*, the proof technique of Theorem 2 also works for this function. Clearly, *EXISTODD* is in AC^0 but it is not in $AC^0_{g(n)}$ if, for all $\epsilon > 0$, $g(n) = o(n^\epsilon)$.

Another direction is where we started our reasoning: can one find lower bounds with limited negations, without any restriction on the depth? For example, can one prove Razborov type results for circuits containing a few (say, a constant number of) negations?

Acknowledgements

The authors would like to thank Noam Nisan for pointing out the use of Lemma 5 in deriving the lower bound for AC^0.

References

[1] M. Ajtai and M. Ben-Or (1984), A theorem on probabilistic constant depth circuits, *Proceedings of 16th ACM STOC*, 471-474.

[2] M. Ajtai and Y. Gurevich (1987), Monotone versus positive, *JACM* 34:4, 1004-1015.

[3] M. Ajtai, J. Komlós and E. Szemerédi (1983), An $O(n \log n)$ sorting network, *Combinatorica* 3:1, 1-19.

[4] S. Akers (1968), On maximum inversion with minimum inverters, *IEEE Transactions on Computers*, 17:2, 134-135.

[5] N. Alon and R. Boppana (1987), The monotone circuit complexity of Boolean functions, *Combinatorica* 7:1, 1-22.

[6] R. Boppana (1986), Threshold functions and bounded depth monotone circuits, *JCSS* 32:2, 222-229.

[7] M. Fischer (1975), The complexity of negation-limited networks - a brief survey, *Automata Theory and Formal Languages, 2nd GI Conference*, ed. H. Brakhage, LNCS, vol. 33, Springer Verlag, 71-82.

[8] A. Hajnal, W. Maass, P. Pudlák, M. Szegedy, and G. Turán (1987), Threshold circuits of bounded depth, *Proceedings of the 28th IEEE FOCS*, 99-110.

[9] J. Hastad (1986), Almost optimal lower bounds for small depth circuits, *Proceedings of 18th ACM STOC*, 6-20.

[10] J. Kahn, G. Kalai, and N. Linial (1988), The influence of variables on Boolean functions, *Proceedings of the 29th IEEE FOCS*, 68-80.

[11] N. Linial, Y. Mansour, and N. Nisan (1989), Constant depth circuits, Fourier transform and learnability, *Proceedings of the 30th IEEE FOCS*, 574-579.

[12] A. Markov (1963), On the inversion complexity of systems of Boolean functions, *Soviet Math. Doklady* 4:3, 694-696.

[13] E. A. Okolnishnikova (1982), On the influence of negation on the complexity of a realization of monotone Boolean functions by formulas of bounded depth, *Metody Diskretnogo Analiza* 38, 74-80.

[14] A. A. Razborov (1985), Lower bounds on the monotone complexity of some Boolean functions, *Doklady Akademii Nauk SSSR* 281:4, 798-801.

[15] É. Tardos (1987), The gap between monotone and non-monotone circuit complexity is exponential, *Combinatorica* 7:4, 393-394.

[16] I. Wegener (1987), *The complexity of Boolean functions*, Wiley-Teubner series in computer science.

[17] A. Yao (1989), Circuits and local computation, *Proceedings of 20th ACM STOC*, 186-196.