Theoretical Computer Science 100 (1992) 253–265
Elsevier

Note

The complexity of computing symmetric functions using threshold circuits*

Paul Beame, Erik Brisson and Richard Ladner

Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195, USA

Communicated by M.S. Paterson
Received March 1990
Revised August 1991

Abstract

Beame, R., E. Brisson and R. Ladner, The complexity of computing symmetric functions using threshold circuits, Theoretical Computer Science 100 (1992) 253–265.

This paper considers size–depth tradeoffs for threshold circuits computing symmetric functions. The size measure used is the number of *connections* or edges in the threshold circuits as opposed to the number of gates in the circuits. The main result is that for all $d \geqslant 2$ and $n \geqslant 8^{2^d}$ there is a threshold circuit to compute any n-input symmetric function which has size

$$O\left(\sqrt{1+\frac{\log n}{2^d-1}} \cdot n^{1+1/(2^d-1)}\right)$$

and depth bounded by $6d+8$. As a consequence, there is a threshold circuit for any n-input symmetric function which has size $O(n)$ and depth bounded by $O(\log \log n)$. A somewhat simpler construction that contains many features of the general solution shows that for all $d \geqslant 1$ and $n \geqslant 2^{2^{d-1}}$ there is a threshold circuit for the n-input parity function which has size bounded by $(27/2\sqrt{2})n^{1+1/(2^d-1)}$ and depth bounded by $2d$.

1. Introduction

Threshold circuits are circuits whose gates are binary threshold units. Threshold circuits are an interesting class of circuits to study because of their relationship to

*This research was supported by NSF grants CCR-8714782 and CCR-8858799.

neural networks [15] and perceptrons [9]. Recent papers [2, 3, 6, 12, 14] have begun the study of functions solvable by threshold circuits of polynomial size and constant depth, commonly called TC^0. It is interesting that functions such as parity, counting, sorting, and multiplication are all in TC^0 when none of these functions are in AC^0, the class of functions computable in polynomial size and constant depth by unbounded fan-in and–or circuits. Superpolynomial lower bounds on the size of unbounded fan-in and–or circuits computing parity demonstrating this separation were obtained independently by Ajtai [1] and Furst et al. [5] and have since been significantly improved, culminating in the work of Håstad [7].

We should note that when we talk of the size of a circuit we mean the number of edges in the graph representing the circuit. Many authors define the size of a circuit to be the number of nodes in this graph rather than the number of edges. However, counting the number of edges gives us a more discriminating measure of the size of unbounded fan-in circuits. All the threshold circuits constructed in this paper have a linear number of threshold units.

It is not difficult to see that all the threshold functions can be computed in linear size and constant depth using just the majority function and negation. Thus, parity is computable in linear size and constant depth using majority functions and negation as a basis. By contrast, Razborov [13] has shown that the majority function cannot be computed in polynomial size and constant depth using parity and negation as a basis.

A *symmetric* Boolean function is one whose value depends only on the number of 1's in its input. The focus of this paper is the computation of symmetric functions using threshold circuits. There is a rich history of results about symmetric functions [16]. There is an elegant construction of a Boolean circuit (bounded fan-in, and–or circuit) of size $O(n)$ and depth $O(\log n)$ for any n-input symmetric function [10], and $\log n$ is optimal for such circuits. A technique of Chandra et al. [4] enables a construction of an unbounded fan-in and–or circuit of quadratic size and depth $O(\log n/\log\log n)$ for n-input symmetric functions. By a result of Håstad [7], $O(\log n/\log\log n)$ is optimal for unbounded fan-in and–or circuits of polynomial size for the n-input parity function. By contrast for all n-input symmetric functions there are threshold circuits of quadratic size and optimal $O(1)$ depth.

Our results show that for any $d>1$, every n-input symmetric function can be computed by a threshold circuit of size $O(n^{1+\varepsilon_d})$ and depth at most d, where ε_d goes to 0 exponentially in d. As a consequence we show that for all n-input symmetric functions there are threshold circuits of linear size and $O(\log\log n)$ depth.

The model

A threshold circuit is one built up from *threshold units* and *negations*. A threshold unit is defined by the number of inputs n and a threshold value k. The unit is denoted by $\leqslant_k^n(x)$, where x is a binary vector $(x_1, x_2, \ldots, x_n)$ of length n. It is defined by

$$\leqslant_k^n(x)=\begin{cases}1 & \text{if } \sum_{i=1}^n x_i \leqslant k,\\ 0 & \text{otherwise.}\end{cases}$$

Threshold circuits are built from threshold units in the usual way by allowing the output of a threshold unit or its negation to be the input to other gates. A threshold circuit can be represented by an acyclic directed labeled graph, where each threshold unit can be represented by a node labeled with its threshold value and each input is represented by a node labeled by an input variable. There is a directed edge from one node to another if the output of the former is an input to the latter. The edge is labeled with "$\neg$" if the input is negated. We are interested in two measures of the complexity of threshold circuits: *size*, which is the number of edges in the graph representing the circuit, and *depth*, which is the length of a longest path in the circuit. (For simplicity we have chosen not to count negations in depth or size. It is easy to see that including them in the count would at most increase the depth by 1 and the size by a factor of 2.)

The greater than threshold function, $\geqslant_k^n(x)$, the unbounded "and" function, and the unbounded "or" functions can all be constructed in size n and depth 1:

$$\geqslant_k^n(x) = \leqslant_{n-k}^n((\neg x_1, \ldots, \neg x_n)),$$

$$\bigwedge_{i=1}^{n} x_i = \geqslant_n^n((x_1, \ldots, x_n)),$$

$$\bigvee_{i=1}^{n} x_i = \geqslant_1^n((x_1, \ldots, x_n)).$$

Any symmetric function f can be computed in depth 2 and size $2k(n+1)$, where k is defined by choosing $a_1 \leqslant b_1 < a_2 \leqslant b_2 < \cdots < a_k \leqslant b_k$ with the property that $f(x)=1$ if and only if for some j, $a_j \leqslant \sum_{i=1}^n x_i \leqslant b_j$. Then,

$$f(x) = \geqslant_{k+1}^{2k}(\geqslant_{a_1}^n(x), \leqslant_{b_1}^n(x), \ldots, \geqslant_{a_k}^n(x), \leqslant_{b_k}^n(x)).$$

For the parity function of n inputs, $k = \lceil n/2 \rceil$. Thus, for n even, the size is bounded by $n(n+1)$. For n odd the size bound would appear to be $(n+1)^2$, but the threshold gate $\leqslant_n^n$ can be eliminated, thereby reducing the size to $n(n+1)$.

Results

The main result of this paper is that for all $d \geqslant 2$ and $n \geqslant 8^{2^d}$ there is a threshold circuit to compute any n-input symmetric function which has size

$$O\left(\sqrt{1+\frac{\log n}{2^d-1}} \cdot n^{1+1/(2^d-1)}\right)$$

and depth bounded by $6d+8$.[1] As a consequence, there is a threshold circuit for any n-input symmetric function which as size $O(n)$ and depth bounded by $O(\log \log n)$. The main result is presented in Section 4.

The proof of the main result relies heavily on the construction of threshold circuits for the *sum-reduction*^{n,m} problem, which is the problem of producing two $(m+\log n)$-

[1] In this paper we define $\log n$ to be the least integer k such that $2^k \geqslant n$.

bit integers, whose sum is equal to the sum of $2n$ m-bit integers. There are several important components to this construction. First, there is the construction of a *basic unit* to solve the problem which has size $O(mn^2)$ and depth 6. Second, there is the construction of a tree of basic units which has depth $6d$. The structure of the tree depends on the minimization of the size function for the resulting circuit. Third, since the minimization results in real values for the number of inputs to the basic units, we rely on a rounding technique to produce integer values. This rounding technique is of some independent interest and has the property that it keeps all rounded prefix products close to the original real prefix products. The construction of the threshold circuits for the *sum-reduction*^{n,m} problem is presented in Section 3.

It turns out that most of the ideas in the construction of threshold circuits for the *sum-reduction*^{n,m} function appear in a simpler form in the construction of threshold circuits for the parity function. Because the construction is simpler and the bounds are slightly better, we give a complete proof in Section 2 that for all $d \geqslant 1$ and $n \geqslant 2^{2^{d-1}}$ there is a threshold circuit for the n-input parity function which has size bounded by $(27/2\sqrt{2})n^{1+1/(2^d-1)}$ and depth bounded by $2d$.

2. Parity

In this section we show the following size–depth trade-off for the parity function.

Theorem 2.1. *For all $d \geqslant 1$ and $n \geqslant 2^{2^{d-1}}$ there is a threshold circuit for the n-input parity function which has size bounded by $(27/2\sqrt{2})n^{1+1/(2^d-1)}$ and depth bounded by $2d$.*

If we let $n \geqslant 4$ and $d = \log\log n$ then $n \geqslant 2^{2^{d-1}}$ and $n^{1/(2^d-1)} \leqslant 4$. Thus, we have the following result.

Corollary 2.2. *There is a threshold circuit for the n-input parity function which has size $O(n)$ and depth $O(\log\log n)$.*

Proof of Theorem 2.1. Ler *parity*^{n} be the parity function on n inputs. The construction for $d = 1$ is as described in Section 1. We call this circuit a *basic unit* with n inputs. Thus, the basic unit with n inputs has depth 2 and size bounded by $\frac{3}{2}n^2$.

For $d > 1$ and $n \geqslant 2^{2^{d-1}}$, we show how to build a threshold circuit of depth $2d$ for *parity*^{n} for every n. There are several steps in the construction. We will choose integers $v_1, v_2, \ldots, v_d$ such that $\prod_{i=1}^{d} v_i = n'$, where $n \leqslant n'$. We then build a tree of depth d with n' leaves, n of which are the original n inputs and $n' - n$ of which are dummy inputs set to zero. Each internal node of the tree is a basic unit. The root at level 0 is a basic unit with exactly v_1 inputs. Generally, for $1 \leqslant i \leqslant d$, there are $v_1 v_2 \cdots v_{i-1}$ basic units at level $i-1$, each with exactly v_i inputs. Figure 1 describes the circuit.

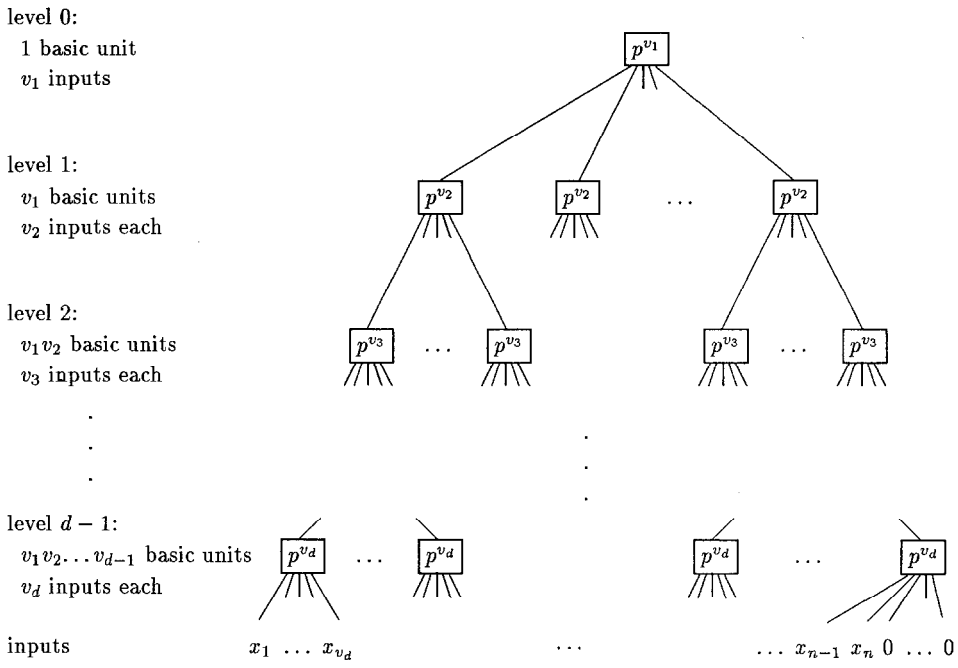

Fig. 1. Threshold circuit of depth $2d$.

Since the depth of each basic unit in the construction is 2, the total depth of the parity circuit is $2d$. The size of the circuit is bounded by

$$\tfrac{3}{2}\sum_{i=1}^{d}\left(\prod_{j=1}^{i-1} v_j\right)v_i^2 = \tfrac{3}{2}(v_1^2 + v_1v_2^2 + \cdots + v_1v_2\cdots v_{d-1}v_d^2).$$

There are two steps in choosing the sequence $v_1, v_2, \ldots, v_d$. First, we choose a sequence of positive real numbers $u_1, u_2, \ldots, u_d$ which minimizes $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2$ subject to $\prod_{i=1}^{d} u_i = n$. This enables us to show that $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2 \leqslant 2\sqrt{2}n^{1+1/(2^d-1)}$. Since the values of $u_1, u_2, \ldots, u_d$ are not necessarily integers, we employ a rounding technique which assigns, for each i, $v_i = \lceil u_i \rceil$ or $v_i = \lfloor u_i \rfloor$. Provided $u_1 \geqslant u_2 \geqslant \cdots \geqslant u_d \geqslant 2$, this assignment satisfies $n \leqslant \prod_{i=1}^{d} v_i \leqslant \frac{3}{2}n$ and $\frac{3}{2}\sum_{i=1}^{d}(\prod_{j=1}^{i-1} v_j)v_i^2 \leqslant \frac{27}{8}\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2$. Hence, the total circuit size is bounded by $(27/2\sqrt{2})n^{1+1/(2^d-1)}$.

Optimization Lemma 2.3. *Let $n \geqslant 1$ and $d > 1$ be fixed integers. If $u_1, u_2, \ldots, u_d$ are positive real numbers chosen to minimize $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2$ subject to $\prod_{i=1}^{d} u_i = n$, then*

(1) $u_i = u_{i+1}^2/2$ *for* $1 \leqslant i < d$,

(2) $u_d = 2^{(2^{d-1}-1)/(2^d-1)}n^{1/(2^d-1)}$,

(3) $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2 \leqslant 2\sqrt{2}n^{1+1/(2^d-1)}$.

Proof. Simple calculus can be used to demonstrate that if x^2+xy^2 is minimized subject to xy being held constant, then $x=y^2/2$. If $u_i \neq u_{i+1}^2/2$ for some $1 \leqslant i < d$, then all the terms in $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2$ can be held constant except the two terms $u_1u_2\cdots u_{i-1}u_i^2+u_1u_2\cdots u_iu_{i+1}^2$ by fixing $u_1, \ldots, u_{i-1}, u_{i+2}, \ldots, u_d$ and the product u_iu_{i+1}. This also fixes the entire product $u_1u_2\cdots u_d$. Keeping these constraints the sum $u_1u_2\cdots u_{i-1}u_i^2+u_1u_2\cdots u_iu_{i+1}^2$ is minimized when $u_i^2+u_iu_{i+1}^2$ is minimized. This is accomplished by letting $u_i=u_{i+1}^2/2$. This contradicts our assumption that $u_i \neq u_{i+1}^2/2$. Hence, we have shown (1).

(2) involves a calculation using (1) and the fact that $n=\prod_{i=1}^{d} u_i$. Using (1) it can be shown by induction on i that for $1 \leqslant i < d$,

$$u_{d-i}=u_d^{2^i}/2^{2^i-1}.$$

This implies that

$$n=\prod_{i=1}^{d} u_i=u_d^{2^d-1}/2^{2^d-1-1}.$$

Hence,

$$u_d=2^{(2^d-1-1)/(2^d-1)}n^{1/(2^d-1)}.$$

To prove (3) we note from (1) that for $1 \leqslant i < d$

$$u_1\cdots u_{i-1}u_i^2=\tfrac{1}{2}u_1\cdots u_iu_{i+1}^2.$$

That is, the ith term in the sum $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2$ is half as large as the $(i+1)$st. Hence,

$$\begin{aligned}\sum_{i=1}^{d}\left(\prod_{j=1}^{i-1} u_j\right)u_i^2 &= (2-1/2^{d-1})u_1u_2\cdots u_{d-1}u_d^2\\ &= (2-1/2^{d-1})nu_d\\ &= (2-1/2^{d-1})2^{(2^d-1-1)/(2^d-1)}n^{1+1/(2^d-1)}\\ &< 2\sqrt{2}n^{1+1/(2^d-1)}. \qquad \square\end{aligned}$$

Rounding Lemma 2.4. *Let $u_1, \ldots, u_d$ be real numbers with the property that $u_1 \geqslant u_2 \geqslant \cdots \geqslant u_d \geqslant 1$. Consider the following inductive definition of $v_1, \ldots, v_d$:*

$$v_i=\begin{cases}\lceil u_i \rceil & \text{if } v_1\cdots v_{i-1}\lfloor u_i \rfloor < u_1\cdots u_i,\\ \lfloor u_i \rfloor & \text{otherwise.}\end{cases}$$

Then, $u_1\cdots u_i \leqslant v_1\cdots v_i < u_1\cdots u_i(\lfloor u_i \rfloor+1)/\lfloor u_i \rfloor$ for $1 \leqslant i \leqslant d$.

Proof. The proof is by induction on i, $1 \leqslant i \leqslant d$. Note that v_1 must be equal to $\lceil u_1 \rceil$ because if $\lfloor u_1 \rfloor < u_1$, then it is by definition and, otherwise, u_1 is an integer and then $v_1=\lfloor u_1 \rfloor=\lceil u_1 \rceil$. Since $\lceil u_1 \rceil < u_1(\lfloor u_1 \rfloor+1)/\lfloor u_1 \rfloor$, $u_1 \leqslant v_1 < u_1(\lfloor u_1 \rfloor+1)/\lfloor u_1 \rfloor$.

Assume as an induction hypothesis that $u_1\cdots u_i \leqslant v_1\cdots v_i < u_1\cdots u_i(\lfloor u_i \rfloor+1)/\lfloor u_i \rfloor$. There are two cases to consider: either $v_{i+1}=\lceil u_{i+1} \rceil$ or $v_{i+1}=\lfloor u_{i+1} \rfloor$.

Case 1: $v_{i+1}=\lceil u_{i+1}\rceil$. Since $u_1\cdots u_i\leqslant v_1\cdots v_i$, then, clearly, $u_1\cdots u_iu_{i+1}\leqslant v_1\cdots v_i\lceil u_{i+1}\rceil=v_1\cdots v_iv_{i+1}$. The condition for choosing $v_{i+1}=\lceil u_{i+1}\rceil$ guarantees that $v_1\cdots v_i\lceil u_{i+1}\rceil<u_1\cdots u_{i+1}\lceil u_{i+1}\rceil/\lfloor u_{i+1}\rfloor\leqslant u_1\cdots u_{i+1}(\lfloor u_{i+1}\rfloor+1)/\lfloor u_{i+1}\rfloor$.

Case 2: $v_{i+1}=\lfloor u_{i+1}\rfloor$. The condition for choosing $v_{i+1}=\lfloor u_{i+1}\rfloor$ guarantees that $u_1\cdots u_{i+1}\leqslant v_1\cdots v_i\lfloor u_{i+1}\rfloor=v_1\cdots v_{i+1}$. By the induction hypothesis $v_1\cdots v_i<u_1\cdots u_i(\lfloor u_i\rfloor+1)/\lfloor u_i\rfloor$. Since $v_{i+1}\leqslant u_{i+1}$, then $v_1\cdots v_iv_{i+1}<u_1\cdots u_iu_{i+1}(\lfloor u_i\rfloor+1)/\lfloor u_i\rfloor$. Since $u_i\geqslant u_{i+1}\geqslant 1$, then $(\lfloor u_i\rfloor+1)/\lfloor u_i\rfloor\leqslant(\lfloor u_{i+1}\rfloor+1)/\lfloor u_{i+1}\rfloor$. Hence, $v_1\cdots v_iv_{i+1}<u_1\cdots u_iu_{i+1}(\lfloor u_{i+1}\rfloor+1)/\lfloor u_{i+1}\rfloor$. □

Proof of Theorem 2.1 (*conclusion*). We now conclude the proof of Theorem 2.1. Choose $u_1,\ldots,u_d$ so as to minimize $\sum_{i=1}^{d}(\prod_{j=1}^{i-1}u_j)u_i^2$ subject to $\prod_{i=1}^{d}u_i=n$. Since $n\geqslant 2^{2^{d-1}}$, then by Optimization Lemma 2.3(2)

$$\begin{aligned}u_d&=2^{(2^{d-1}-1)/(2^d-1)}n^{1/(2^d-1)}\\&\geqslant 2^{(2^{d-1}-1)/(2^d-1)}(2^{2^{d-1}})^{1/(2^d-1)}\\&=2.\end{aligned}$$

Since $u_d\geqslant 2$ and $u_i=u_{i+1}^2/2$ [Optimization Lemma 2.3(1)], then we must have $u_1\geqslant u_2\geqslant\cdots\geqslant u_d\geqslant 2$. Thus, the hypothesis of the Rounding Lemma 2.4 holds. Let $v_1,\ldots,v_d$ be defined as in the Rounding Lemma 2.4. Since $u_1\geqslant u_2\geqslant\cdots\geqslant u_d\geqslant 2$, $(\lfloor u_i\rfloor+1)/\lfloor u_i\rfloor\leqslant\frac{3}{2}$ and $v_i\leqslant\lceil u_i\rceil\leqslant\frac{3}{2}u_i$. Using these facts along with the conclusion of the Rounding Lemma 2.4 and Optimization Lemma 2.3(3), the total size of the circuit is bounded by

$$\begin{aligned}\tfrac{3}{2}\sum_{i=1}^{d}\left(\prod_{j=1}^{i-1}v_j\right)v_i^2&=\tfrac{3}{2}\sum_{i=1}^{d}\left(\prod_{j=1}^{i}v_j\right)v_i\\&\leqslant\tfrac{3}{2}\sum_{i=1}^{d}\left(\prod_{j=1}^{i}u_j\right)\frac{\lfloor u_i\rfloor+1}{\lfloor u_i\rfloor}\lceil u_i\rceil\\&\leqslant\tfrac{3}{2}\sum_{i=1}^{d}\left(\prod_{j=1}^{i}u_j\right)\tfrac{3}{2}\tfrac{3}{2}u_i\\&=\tfrac{27}{8}\sum_{i=1}^{d}\left(\prod_{j=1}^{i-1}u_j\right)u_i^2\\&\leqslant(27/2\sqrt{2})n^{1+1/(2^d-1)}.\end{aligned}$$

If we let $n'=v_1\cdots v_d$, then $n=u_1\cdots u_d\leqslant v_1\cdots v_d=n'$ and $n'=v_1\cdots v_d<u_1\cdots u_d(\lfloor u_d\rfloor+1)/\lfloor u_d\rfloor\leqslant\frac{3}{2}n$. □

3. Sum reduction

In order to construct the small size and depth threshold circuits for symmetric functions we need good threshold circuits for the *sum-reduction*^{n,m} problem. Given $2n$ m-bit integers, the *sum-reduction*^{n,m} problem is the problem of producing two $(m+\log n)$-bit integers, whose sum is equal to the sum of the input integers.

Theorem 3.1. *For all* $d \geqslant 1$ *and* $n \geqslant 8^{2^d-1}$ *there is a threshold circuit for the sum-reduction*^{n,m} *problem which has size bounded by*

$$O\left(\sqrt{1+\frac{\log n}{(2^d-1)m}} \cdot mn^{1+1/(2^d-1)}\right)$$

and depth bounded by $6d$.

If we let $n \geqslant 2^{16}$ and $d = \log\log n - 3$, then $n \geqslant 8^{2^d-1}$, and both $n^{1/(2^d-1)}$ and $\log n/[(2^d-1)m]$ are bounded by constants. By suppressing some large constants we have the following result.

Corollary 3.2. *There is a threshold circuit for sum-reduction*^{n,m} *which has size* $O(mn)$ *and depth bounded by* $O(\log\log n)$.

Proof of Theorem 3.1. For $d=1$ we construct a basic unit for *sum-reduction*^{n,m} of depth 6 and size $O(n^2m)$. The construction is essentially the same as the construction in the reduction of multiple-addition to binary-count described by Chandra et al. [4], except that the final stage of adding two binary numbers to obtain a single binary number is not done. The function *multiple-addition*^{n} is the problem of adding n n-bit numbers to form a single $(n+\log n)$-bit number. The function *binary-count*^{n} is the problem of producing a $(\log n)$-bit number which equals the number of 1's in an n-bit input. The function *parity*^{n} computes the low-order bit of *binary-count*^{n}. In general, the ith lowest bit of *binary-count*^{n} is a symmetric function which can be computed in depth 2 and size $O(n^2)$.

There are three stages in the computation of *sum-reduction*^{n,m} for $n \geqslant 8$ and $d=1$.

Stage 1: Use *binary-count*^{2n} to compute, for $1 \leqslant i \leqslant m$, the sum of the ith bit positions of all $2n$ numbers. These numbers can be "packed" into $\log 2n$ numbers, each of length $m+\log n$, whose sum is equal to the sum of the m input numbers. This stage can be done in depth 2 and size $O(mn^2)$.

Stage 2: Repeat Stage 1 with the $\log 2n$ numbers each of length $m+\log n$. In this case the $m+\log n$ results can be packed into $\log\log 2n$ numbers of the same length $m+\log n$. This stage can be done in depth 2 and size $O((m+\log n)\log^2 n)$.

Stage 3: Group the lowest order $\log\log\log 2n$ bit positions of the $\log\log 2n$ numbers together and continue forming groups of $\log\log\log 2n$ bit positions of the $\log\log 2n$ numbers together. There will be a total of $\lceil (m+\log n)/\log\log\log 2n \rceil$

groups of bits, each with $\log\log 2n$ numbers of length $\log\log\log 2n$. Use a brute-force sum of products to compute the sum of the numbers in each group. The result of the sum of each group is a single number of length $\leqslant 2\log\log\log 2n$. Because the bit positions of the groups start at multiples of $\log\log\log 2n$, the resulting sums of length $2\log\log\log 2n$ can be packed into two numbers of length $m+\log 2n$. This stage can be done in depth 2 and size $O((m+\log n)2^{(\log\log n)^2})$. The size follows because sum of product circuits for s-input, t-output functions have size at most $(s+t)2^s$.

It is clear that the depth of the basic unit is bounded by 6. The size of the first stage dominates the size of the other two stages. Hence, the size of the basic unit is $O(mn^2)$.

The construction of the threshold circuit for the iterated addition problem is analogous to that for the parity problem. Again the problem in question is broken up into a tree of height d of smaller basic units. The number of integers that fan-in to the root will be $2v_1$ and in general at level $i-1$ of the tree the fan-in will be $2v_i$. We will choose $v_1, v_2, \ldots, v_d$ so that $n \leqslant \prod_{i=1}^{d} v_i$. Although the input integers at the leaves of this tree have only m bits, the size grows for integers closer to the root. In general, for $1 \leqslant i \leqslant d$, there are $v_1 v_2 \cdots v_{i-1}$ basic units at level $i-1$, each of which has $2v_i$ input integers of up to $m+\log v_{i+1}+\cdots+\log v_d = \log(2^m v_{i+1} \cdots v_d)$ bits apiece.

Since the depth of each basic unit is 6, the total depth of the sum-reduction circuit is $6d$. The size of the circuit is bounded by

$$c\sum_{i=1}^{d}\left(\prod_{j=1}^{i-1} v_j\right)v_i^2 \log\left(2^m \prod_{j=i+1}^{d} v_j\right) = c(v_1^2 \log(2^m v_2 \cdots v_d) + v_1 v_2^2 \log(2^m v_3 \cdots v_d) + \cdots + v_1 v_2 \cdots v_{d-1} v_d^2 \log(2^m))$$

for some constant $c>0$. As was the case for parity, real values $u_1, u_2, \ldots, u_d$ are chosen which minimize

$$\sum_{i=1}^{d}\left(\prod_{j=1}^{i-1} u_j\right)u_i^2 \log\left(2^m \prod_{j=i+1}^{d} u_j\right)$$

subject to $u_1 u_2 \cdots u_d = n$, and then the Rounding Lemma 2.4 is used to convert these to integers which do not significantly change the value of the size formula.

Optimization Lemma 3.3. *Let $n \geqslant 1$ and $d>1$ be fixed integers. If $u_1, u_2, \ldots, u_d$ are real numbers >1 chosen to minimize $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2 \log(2^m \prod_{j=i+1}^{d} u_j)$ subject to $u_1 u_2 \cdots u_d = n$, then*

(1) *$u_{i-1} = u_i^2 a_{i+1}/(2\log u_i + 2a_{i+1} - 1)$, where $a_{i+1} = \log(2^m u_{i+1} \cdots u_d)$ for $1 < i \leqslant d$.*

(2) *$u_d < 4\sqrt{1+\log n/[m(2^d-1)]}\cdot n^{1/(2^d-1)}$ and $u_{i-1} > u_i^2/6$ for $i<d$.*

(3) *$\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2 \log(2^m \prod_{j=i+1}^{d} u_j) < 8\sqrt{1+\log n/[(2^d-1)m]}\cdot mn^{1+1/(2^d-1)}$.*

Proof. As was the case for parity we note that for any fixed values of $u_1, \ldots, u_{i-2}$ and $u_{i+1}, \ldots, u_d$ the constraint requires that the value of the product $u_{i-1}u_i$ is also fixed.

Furthermore, this fixes all terms in $\sum_{i=1}^{d}(\prod_{j=1}^{i-1} u_j)u_i^2 \log(2^m \prod_{j=i+1}^{d} u_j)$ except for

$$u_1 u_2 \cdots u_{i-2} u_{i-1}^2 \log(2^m u_i \cdots u_d) + u_1 u_2 \cdots u_{i-2} u_{i-1} u_i^2 \log(2^m u_{i+1} \cdots u_d).$$

These terms are minimized when

$$u_{i-1}^2(\log u_i + a_{i+1}) + u_{i-1} u_i^2 a_{i+1}$$

is minimized subject to $a_{i+1} = \log(2^m u_{i+1} \cdots u_d)$ being fixed. This expression is of the form $x^2(\log y + a) + xy^2 a$, where xy is fixed, which can be shown by basic calculus to be minimized when $x = ay^2/(2 \log y + 2a - 1)$. Substituting back the values u_{i-1}, u_i and a_{i+1} proves (1).

Repeated application of the formula for (1) will determine all the u_i in terms of u_d and then the constraint $u_1 \cdots u_d = n$ will yield their values in terms of n. More specifically,

$$u_{d-1} = u_d^2 \Big/ \left(\frac{2 \log u_d}{m} + 2 - \frac{1}{m}\right)$$

and $u_i < u_{i+1}^2$ for each $i < d$; so, $\log u_i < 2 \log u_{i+1}$. Thus, $2 \log u_i < 4 \log u_{i+1} \leqslant 4a_{i+1}$ for $i < d$. It then follows that $u_{i-1} > u_i^2/6$ for $i < d$. Using this one can show, by induction, that for $i \geqslant 1$,

$$u_{d-i} > 6u_d^{2^i} \Big/ \left[12\left(1 + \frac{\log u_d}{m}\right)\right]^{2^{i-1}}.$$

This implies that

$$n = \prod_{i=1}^{d} u_i > \frac{6^{d-1} u_d^{2^d - 1}}{[12(1 + \log u_d/m)]^{2^{d-1}-1}} \geqslant \frac{u_d^{2^d-1}}{[12(1+\log u_d/m)]^{2^{d-1}-1}}.$$

Hence,

$$u_d < \sqrt{12(1 + \log u_d/m)} \cdot n^{1/(2^d - 1)}.$$

Solving this yields

$$u_d < 4\sqrt{1 + \log n/[(2^d - 1)m]} \cdot n^{1/(2^d-1)}.$$

To prove (3) we note from (1) that for $1 \leqslant i < d$,

$$u_1 \cdots u_{i-1} u_i^2 \log(2^m u_{i+1} \cdots u_d) < \tfrac{1}{2} u_1 \cdots u_i u_{i+1}^2 \log(2^m u_{i+2} \cdots u_d).$$

This implies that each term is twice the preceding term and the entire sum is at most twice the last term. Hence,

$$\begin{aligned}\sum_{i=1}^{d}\left(\prod_{j=1}^{i-1} u_j\right)u_i^2 \log\left(2^m \prod_{j=i+1}^{d} u_j\right) &< 2u_1 \cdots u_{d-1} u_d^2 \log 2^m \\ &= 2nu_d m \\ &< 8\sqrt{1 + \log n/[(2^d-1)m]} \cdot m n^{1 + 1/(2^d-1)}. \qquad \square\end{aligned}$$

Proof of Theorem 3.1 (*conclusion*). We can now conclude the proof of Theorem 3.1. From Optimization Lemma 3.3(1) we observe that for all i, $u_i < u_{i+1}^2$. Therefore, $n = u_1 \cdots u_d < u_d^{2^d-1}$ and we get $u_d > n^{1/(2^d-1)} \geqslant 8$. Since $u_d \geqslant 8$, $u_d > (2 \log u_d/m) + 2 - 1/m$; so, $u_{d-1} = u_d^2/[(2 \log u_d/m) + 2 - 1/m] > u_d$. Furthermore, $u_{i-1} > u_i^2/6 > u_i$ for $i < d$. Thus, $u_1 \geqslant u_2 \geqslant \cdots \geqslant u_d \geqslant 8$. This allows the use of the Rounding Lemma 2.4 to obtain integers $v_1, v_2, \ldots, v_d$ such that for all i, $v_i \leqslant \lceil u_i \rceil \leqslant \frac{9}{8} u_i$ and

$$\prod_{j=1}^{i} u_j \leqslant \prod_{j=1}^{i} v_j < \prod_{j=1}^{i} u_i \frac{\lfloor u_i \rfloor + 1}{\lfloor u_i \rfloor} \leqslant \frac{9}{8} \prod_{j=1}^{i} u_i$$

since each $u_i \geqslant 8$. From this it can be seen that for any i and j these integers also satisfy $v_i v_{i+1} \cdots v_j \leqslant \frac{9}{8} u_i u_{i+1} \cdots u_j$. Using these facts, the size of the circuit is bounded by

$$\begin{aligned}
c \sum_{i=1}^{d} \left(\prod_{j=1}^{i-1} v_j \right) v_i^2 \log\left(2^m \prod_{j=i+1}^{d} v_j \right) &= c \sum_{i=1}^{d} \left(\prod_{j=1}^{i} v_j \right) v_i \log\left(2^m \prod_{j=i+1}^{d} v_j \right) \\
&\leqslant c \sum_{i=1}^{d} \frac{9}{8} \left(\prod_{j=1}^{i} u_j \right) \frac{9}{8} u_i \log\left(2^m \frac{9}{8} \prod_{j=i+1}^{d} u_j \right) \\
&\leqslant \frac{81c}{64} \sum_{i=1}^{d} \left(\prod_{j=1}^{i} u_j \right) u_i \log\left(2^m \frac{9}{8} \prod_{j=i+1}^{d} u_j \right) \\
&\leqslant \frac{11c}{8} \sum_{i=1}^{d} \left(\prod_{j=1}^{i-1} u_j \right) u_i^2 \log\left(2^m \prod_{j=i+1}^{d} u_j \right) \\
&< 11c \sqrt{1 + \frac{\log n}{(2^d-1)m}} \cdot m n^{1+1/(2^d-1)}.
\end{aligned}$$

If we let $n' = v_1 \cdots v_d$ then $n = u_1 \cdots u_d \leqslant n' < \frac{9}{8} n$ and the conditions of the theorem are satisfied. □

4. Symmetric functions

We now proceed to prove the main theorem.

Theorem 4.1. *For all $d \geqslant 2$ and $n \geqslant 8^{2^d}$, for any n-input symmetric function f there is a threshold circuit computing f of size*

$$O\left(\sqrt{1 + \frac{\log n}{2^d - 1}} \cdot n^{1+1/(2^d-1)} \right)$$

and depth bounded by $6d + 8$.

Proof. Let f be an n-input symmetric function, where $d \geqslant 2$ and $n \geqslant 8^{2^d}$. If n is odd, then we add one more input set to zero, so that the n inputs can be considered to be two $2\lceil n/2 \rceil$ numbers each of length one. We compute f in three stages.

Stage 1: Compute *sum-reduction*$^{\lceil n/2 \rceil, 1}$ to obtain two numbers of length $\log(\lceil n/2 \rceil)+1$ whose sum is the number of 1's in the original input. Since $n \geqslant 8^{2^d}$, $\lceil n/2 \rceil \geqslant 8^{2^{d-1}}$ and then by Theorem 3.1, this computation can be done in size

$$O\left(\sqrt{1+\frac{\log\lceil n/2 \rceil}{2^d-1}} \cdot \lceil n/2 \rceil^{1+1/(2^d-1)}\right)$$

and depth bounded by $6d$.

Stage 2: Use the well-known carry–look-ahead method for addition to compute the sum of the two numbers computed in Stage 1. The resulting number, which has exactly $\log n$ significant bits, is the number of 1's in the original input. This can be done in size $O(\log^3 n)$ and depth 4 [10].

Stage 3: Since the function f is symmetric, its value on a given input depends only on the result of Stage 2. Use the classical Lupanov construction [8] to compute f from the $\log n$ bit result of Stage 2. This can be done in size $O(n/\log n)$ and depth 4.

A straightforward calculation reveals that for $d \geqslant 2$ the size and depth bounds stated in the theorem hold. □

Theorem 4.2. *Any n-input symmetric function can be computed by a threshold circuit which has size* $O(n)$ *and depth* $O(\log \log n)$.

Proof. Simply replace Stage 1 in the proof of Theorem 4.1 by the threshold circuit given by Corollary 2.2. □

5. Open questions

Our results give linear-size threshold circuits for symmetric functions of $\Theta(\log \log n)$ depth. It is not clear whether or not this can be improved. Is it the case for example that all problems in NC^1, those computable by bounded fan-in logarithmic-depth circuits, can be computed in linear size and $O(\log \log n)$ depth?

For the parity function, Paturi and Saks [12] have improved our upper bound to $O(n^{1+\phi^{-d}})$, where ϕ is the golden ratio, and have shown a lower bound of $\Omega(n^2/\log^2 n)$ for the size of threshold circuits of depth 2. Lower bounds on the size of threshold circuits of depth >3 for the parity function are still unknown [12].

A more restrictive model than the one we use is that of threshold formulas, threshold circuits whose underlying graphs are trees. Although essentially our constructions are tree-like, the resulting circuits are not necessarily trees. Can similar size–depth tradeoffs to ours be shown for threshold formulas?

These results, more generally, leave open the question of whether nontrivial size–depth tradeoff lower bounds can be found for threshold circuits computing any specific Boolean functions in NP. Using the finer distinctions permitted by counting connections may make this a little easier than the corresponding question obtained by counting gates.

Acknowledgment

We thank the Editor, Michael Paterson, for pointing out a depth-2 threshold circuit for any symmetric function. We also thank the anonymous referee for several valuable suggestions.

References

[1] M. Ajtai, Σ_1^1 formulae on finite structures, *Ann. Pure Appl. Logic* **24** (1983) 1–48.
[2] E. Allender, A note on the power of threshold circuits, in: *Proc. 30th Ann. Symp. on Foundations of Computer Science* (1989) 580–585.
[3] D.S.M. Barrington, N. Immerman and H. Straubing, On uniformity within NC, in: *Proc. 3rd Ann. Conf. on Structure in Complexity Theory* (1988) 47–59.
[4] A.K. Chandra, L. Stockmeyer and U. Vishkin, Constant depth reducibility, *SIAM J. Comput.* **13** (1984) 423–439.
[5] M. Furst, J.B. Saxe and M. Sipser, Parity, circuits, and the polynomial-time hierarchy, *Math. Systems Theory* **17** (1984) 13–27.
[6] A. Hajnal, W. Mass, P. Pudlák, M. Szegedy and G. Turán, Threshold circuits of bounded depth, in: *Proc. 28th Ann. Symp. on Foundations of Computer Science* (1987) 99–110.
[7] J.T. Håstad, *Computational Limitations for Small-Depth Circuits* (MIT Press, Cambridge, MA, 1987).
[8] O.B. Lupanov, A method of circuit synthesis, *Izv. Vyssh. Uchebn. Zaved. Radiofiz* **1** (1958) 120–140.
[9] M. Minsky and S. Papert, *Perceptrons* (MIT Press, Cambridge, MA, 1969).
[10] D. Muller and F. Preparata, Bounds to complexities of networks for sorting and switching, *J. ACM* **22** (1975) 195–201.
[11] I. Parberry and G. Schnitger, Parallel computation with threshold functions, *J. Comput. System Sci.* **36** (1988) 278–302.
[12] R. Paturi and M.E. Saks, On threshold circuits for parity, in: *Proc. 31st Ann. Symp. on Foundations of Computer Science* (1990) 397–404.
[13] A.A. Razborov, Lower bounds for the size of circuits of bounded depth with basis $\{\&, \oplus\}$, *Mat. Zametki* **41** (4) (1987) 598–607; *Math. Notes* **41** (4) (1987) 333–338.
[14] J.H. Reif, On threshold circuits and polynomial computation, in: *Proc. 2nd Ann. Conf. on Structure in Complexity Theory* (1987) 118–123.
[15] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing* (MIT Press, Cambridge, MA, 1986).
[16] I. Wegener, *The Complexity of Boolean Functions*, Wiley–Teubner Series in Computer Science (Wiley, New York, 1987).