JOURNAL OF COMPUTER AND SYSTEM SCIENCES **54**, 199–212 (1997)
ARTICLE NO. SS971480

On the Computation of Boolean Functions by Analog Circuits of Bounded Fan-In*

György Turán†

Department of Mathematics, Statistics and Computer Science, University of Illinois at Chicago, Chicago, Illinois 60680; and Automata Theory Research Group of the Hungarian Academy of Sciences, Szeged, Hungary

and

Farrokh Vatan‡

Department of Mathematics, Statistics and Computer Science, University of Illinois, Chicago, Illinois 60680

Received June 9, 1995; revised September 14, 1995

We consider the complexity of computing Boolean functions by analog circuits of bounded fan-in, i.e., by circuits of gates computing real-valued functions, either exactly or as sign-representation. Sharp *upper bounds* are obtained for the complexity of the most difficult n-variable function over certain bases (sign-representation by arithmetic circuits and exact computation by piecewise linear circuits). Bounds are given for the computational power gained by adding *discontinuous* gate functions and *nondeterminism*. We also prove explicit *nonlinear lower bounds* for the *formula size* of analog circuits over bases containing addition, subtraction, multiplication, the sign function and all real constants.

1. INTRODUCTION

Boolean circuits form a basic model for computing Boolean functions. One can also build circuits using *analog gates*, i.e., gates that compute real-valued functions. Circuits of this kind will be called *analog circuits* in what follows. Important examples are provided by *arithmetic circuits* (i.e., circuits built of gates for arithmetic operations) and *analog neural networks*. Analog circuits can be viewed in several different ways. Let us first give a brief overview of some of the different possibilities.

Analog circuits can be considered as computing functions of *real variables*, either *exactly* or in some *approximate* sense. This is the approach of *algebraic complexity theory* for arithmetic circuits (Strassen [39]), of the theory of representing real functions as *superposition of simpler functions*, originating in Kolmogorov's theorem (Vitushkin [42]), respectively of the growing number of results on the approximating power of *neural networks* (e.g., Cybenko [10]).

Analog circuits can also be used to compute *Boolean functions*. Gashkov [14] studied the *exact* computation of Boolean functions by such circuits. One of the goals of *neural computation* is to *approximate* Boolean functions by analog neural networks (Rumelhart and McClelland [33], McClelland and Rumelhart [26]). Maass, Schnitger, and Sontag [24] and DasGupta and Schnitger [11] considered the relation of this model and threshold circuits. Siegelmann and Sontag [35, 36] studied the computational power of *recurrent* neural networks. Also, results on the *sign-representation* of Boolean functions by polynomials (see, e.g., the survey of Saks [34]) can be viewed in this context, using another notion of approximation.

A different approach to real-valued computation that received a lot of attention recently is the *real Turing machine* introduced by Blum, Shub, and Smale [2] and its variants (Koiran [21], Cucker, Karpinski, Koiran, Lickteig, and Werther [7]). A complexity theory based on these machines is developed [4–9, 20, 21]. The analog circuits corresponding to real Turing machines are *arithmetic threshold circuits*, i.e., circuits built using arithmetic and sign gates (Cucker and Torrecillas [9], Cucker and Grigoriev [6], Montaña and Pardo [28]; we note that in these papers these circuits are called *algebraic circuits*, respectively *arithmetic networks*). Thus the model of arithmetic threshold circuits can be viewed as the *nonuniform* version of real Turing machines. Another related model is that of *algebraic computation trees* (Ben-Or [1]).

It appears that analog circuits form a natural model of computation, and studying the complexity of Boolean functions in this framework may, among other things, contribute to the understanding of the relation between the discrete and continuous aspects of computation. As noted by Gashkov [14], an important task here is to explore the

* A preliminary version appeared in *Proc. 35th IEEE Symposium on the Foundations of Computer Science*, 1994, pp. 553–564.

† Partially supported by NSF Grant CCR-9208170. E-mail address: U11557@uicvm.uic.edu.

‡ E-mail address: U09042@uicvm.uic.edu. Partially supported by NSF Grant CCR-9208170. Current address: Electrical Engineering Department, UCLA, Los Angeles, CA 90095; e-mail: vatan@ee.ucla.edu.

0022-0000/97 $25.00

new phenomena that arise when compared to Boolean complexity. One of these new phenomena (also noted by Sontag [38] for neural networks) is that complexity depends very much on the *basis*, i.e., on the set of gate functions, that may be used. In certain bases, permitting the use of encoding Boolean functions as a single number, every Boolean function can be computed by a linear number of gates. Another related issue is the additional power gained by using large, or large precision, numbers. Concerning lower bounds, it is to be noted that if real constant inputs can be used then the number of circuits with a fixed structure is infinite. Therefore, even determining the *Shannon function* (the complexity of the most difficult function), that is considered to be a well-understood chapter of Boolean complexity theory, requires different techniques. This problem is also of interest from the point of view of determining the range of "pathological" bases mentioned above. A closely related problem, studied in the context of neural networks is that of determining the *Vapnik–Chervonenkis dimension* of classes of neural networks (see, e.g., Goldberg and Jerrum [15], Maass [22, 23], Macintyre and Sontag [25], and Karpinski and Macintyre [19]). There are several apparently different ways of introducing *nondeterminism* in this context. Another, perhaps not entirely hopeless, research direction is to extend *explicit* lower bounds for restricted classes of Boolean circuits to classes of analog circuits.

In this paper we consider some of these issues. It is to be noted that we only consider analog circuits of *bounded fan-in*. Before formulating our results, let us describe some of the relevant results of Gashkov [14]. He proved that if the basis consists of finitely many polynomials and all real numbers can be used as constant inputs then almost all n-variable Boolean functions require circuit size $\Omega(2^{n/2})$. The proof of this bound is based on *connected component counting* (for applications of this method in another context where the inputs are *real numbers* see, e.g., Ben-Or [1]). His argument remains valid if the circuits are only required to compute a *sign-representation*, i.e., to output a nonnegative (resp. negative) value if the function has value 1 (resp. 0). Gashkov also showed that if the basis contains finitely many *Lipschitz* functions and the constant inputs are restricted to some fixed interval, then almost all n-variable Boolean functions require $\Omega(2^{n/2})$ gates. If, in addition, the basis functions are Lipschitz functions with Lipschitz norm $\leqslant 1$, then the lower bound jumps up to $\Omega(2^n/n)$. These results show the limitations of analog computation even if one allows infinite precision operations with real numbers. Gashkov's argument for arithmetic circuits is presented in Section 3 for completeness, along with some of its extensions to more general classes. We discuss the important class of *arithmetic threshold circuits* referred to above and the *nondeterministic* version of these circuits. The lower bounds proved for these classes are $\Omega(2^{n/2})$ (resp. $\Omega(2^{n/4})$). We note that these lower bounds can also be derived from the results of Goldberg and Jerrum [15].

A question raised by these results is whether $2^{n/2}$ can occur as the order of magnitude of the Shannon function for some basis. It may be of some interest to note that in the context of Boolean complexity, orders of magnitude around $2^{n/2}$ typically occur as Shannon functions for bases with *unbounded fan-in*, while for complete bases with bounded fan-in one has always $\Theta(2^n/n)$.

In Section 4 we prove upper bounds that imply a positive answer to this question. It is shown that every n-variable Boolean function has a *sign-representation* of size $O(2^{n/2})$ by *arithmetic circuits*, i.e., circuits over the basis of addition, subtraction, and multiplication, using constant inputs from $[0, 1]$. Also, every n-variable Boolean function can be computed *exactly* by a *piecewise linear* (PL) circuit of size $O(2^{n/2})$, i.e., by a circuit over the basis of *addition*, *subtraction*, *halving*, and *absolute value*, using constant inputs from $[0, 1]$. In contrast, if addition and subtraction are replaced by $(x+y)/2$ and $(x-y)/2$, then, as all basis functions have Lipschitz norm $\leqslant 1$, the result of Gashkov implies that almost all n-variable Boolean functions need $\Omega(2^n/n)$ gates.

In Section 5 we show that "discontinuity can be traded for nondeterminism," by giving an efficient simulation of arithmetic threshold circuits by nondeterministic arithmetic circuits computing a sign-representation. The construction uses some techniques of Blum, Shub, and Smale [2]. We also show that nondeterministic arithmetic circuits with *integer* guesses are "too powerful" in the sense that every n-variable Boolean function can be sign-represented by such circuits of *linear* size. This can be contrasted with *digital nondeterminism* discussed, e.g., in Cucker, Karpinski, Koiran, Lickteig, and Werther [7], where the guesses are required to be Boolean. In this case the Shannon function is $\Omega(2^{n/2})$.

Section 6 contains *explicit nonlinear lower bounds*. It is shown that every *arithmetic expression* (i.e., formula over the basis of addition and multiplication, using arbitrary real constants) that is a *sign-representation* of the ELEMENT DISTINCTNESS function has size $\Omega(n^2/\log^2 n)$. The lower bound also applies to arbitrary finite bases of multilinear polynomials, and thus it generalizes the corresponding lower bound for Boolean formula size. This appears to be one of the first explicit lower bounds known for analog circuits. (Such a lower bound follows for a class of analog neural networks from the simulation results of Maass, Schnitger, and Sontag [24] and the lower bounds of Hajnal, Maass, Pudlák, Szegedy, and Turán [17] for the corresponding class of Boolean threshold circuits. Our lower bounds do not follow by simulation.) The proof combines connected component counting with Nechiporuk's method for Boolean formula complexity. Interpreted in the context of sign-representation of Boolean functions by polynomials referred to above (Saks [34]), this appears to be a new type of lower bound. Previous lower bounds apply to

the degree, the number of terms or decision tree complexity (see Nisan [29], Saks [34], Vatan [41]).

We extend this result by showing that every *arithmetic threshold formula* (i.e., a formula over the basis of addition, multiplication, the *sign* function, and all real constants) for ELEMENT DISTINCTNESS has size $\Omega(n^{3/2}/\log n)$.

As an application of the lower bound for arithmetic threshold formulas, we derive a lower bound for *Boolean threshold circuit* complexity. It follows that every *unbounded fan-in threshold circuit of depth d with unrestricted weights*, computing ELEMENT DISTINCTNESS, a function that belongs to AC^0, has size $\Omega(n^{1/2(d-1)}/(\log n)^{1/(d-1)})$ for every constant *d*. The previous similar lower bounds either apply to PARITY or INNER PRODUCT MOD 2, that are not in AC^0, or assume restricted weights (Dičiūnas [13], Gröger and Turán [16], Impagliazzo, Paturi, and Saks [18], Roychowdhury, Siu, and Orlitsky [32], Siu, Roychowdhury, and Kailath [37]).

2. PRELIMINARIES

The difference between the standard concepts of circuit complexity theory (see, e.g., Wegener [44]) and the definitions used here is that we consider bases $\mathscr{B}$ consisting of finitely many functions $f_1, ..., f_k$, where $f_i: \mathbb{R}^{n_i} \to \mathbb{R}$ is a *real* function of arity $n_i \geqslant 1$, and of a set of *constants* $\Delta \subseteq \mathbb{R}$, where either $\Delta = \mathbb{R}$ or $\Delta = [a, b]$ for some $a \leqslant b$. The input nodes of a circuit are labelled by variables or constants from Δ. The function computed by a circuit is defined in the usual manner. If $\mathbf{x} = (x_1, ..., x_n)$ are the inputs of C and $\mathbf{a} = (a_1, ..., a_m)$ are its constant inputs, then the function computed by C is denoted by $C(\mathbf{x}, \mathbf{a})$.

A circuit C can also be viewed as computing a Boolean function in the following way. C is a *sign-representation* of a Boolean function $f(\mathbf{x})$ if for every $\mathbf{x} \in \{0, 1\}^n$ it holds that $f(\mathbf{x}) = 1$ iff $C(\mathbf{x}, \mathbf{a}) \geqslant 0$.

Circuits over the basis $x + y$, $x - y$, $x \cdot y$ and $\mathbb{R}$ are called *arithmetic circuits*. Formulas, i.e., circuits of fan-out one, over this basis are called *arithmetic expressions*.

Circuits over the basis $x + y$, $x - y$, $x/2$, $|x|$ and $[0, 1]$ are called *PL* (*piecewise linear*) *circuits*.

A function $f(x_1, ..., x_m): \mathbb{R}^m \to \mathbb{R}$ is a *Lipschitz function* if for every $\mathbf{y} = (y_1, ..., y_m)$ and $\mathbf{z} = (z_1, ..., z_m)$ in $\mathbb{R}^m$ it holds that $|f(\mathbf{y}) - f(\mathbf{z})| \leqslant M \max_i |y_i - z_i|$. The *inf* of all M for which this property holds is called the *Lipschitz norm* of f. A basis is called a *Lipschitz basis* if all its functions are Lipschitz, and its constant inputs belong to some interval $[a, b]$. A basis is a *Lipschitz basis of norm* $\leqslant 1$ if, in addition, all its functions have Lipschitz norm $\leqslant 1$.

We also study the effect of allowing discontinuous basis functions and nondeterminism. For this purpose we consider different extensions of arithmetic circuits.

Circuits over the basis of $x + y$, $x - y$, $x \cdot y$, the function $\operatorname{sign}(x)$ defined by

$$\operatorname{sign}(x) = \begin{cases} 1 & \text{if } x > 0, \\ 0 & \text{if } x = 0, \\ -1 & \text{if } x < 0, \end{cases}$$

and $\mathbb{R}$ are called *arithmetic threshold circuits*. Formulas over this basis are called *arithmetic threshold expressions*.

A *nondeterministic arithmetic circuit* is an arithmetic circuit C with inputs $\mathbf{x} = (x_1, ..., x_n)$, constant inputs $\mathbf{a} = (a_1, ..., a_m)$ and *nondeterministic inputs* (called *nd-inputs* in what follows) $\mathbf{y} = (y_1, ..., y_k)$. As the results below refer to sign-representation, we formulate the definition of the Boolean function computed by a nondeterministic circuit only in that case. Thus, if we denote the function computed by C by $C(\mathbf{x}, \mathbf{y}, \mathbf{a})$, then C *sign-represents* the Boolean function f if for every $\mathbf{x} \in \{0, 1\}^n$ it holds that $f(\mathbf{x}) = 1$ iff for some $\mathbf{y} \in \mathbb{R}$, $C(\mathbf{x}, \mathbf{y}, \mathbf{a}) \geqslant 0$. For *nondeterministic arithmetic threshold circuits* it may be assumed w.l.o.g. that the output is always 0 or 1. The Boolean function computed by such a circuit is defined in the usual manner.

Nondeterministic arithmetic circuits can be generalized further by allowing *restricted* guesses. We formulate this for the case when the restricted guesses can be *integer* or *Boolean*. An $(\mathbb{R}, \mathbb{Z})$-nondeterministic arithmetic circuit C has inputs $\mathbf{x} = (x_1, ..., x_n)$, constant inputs $\mathbf{a} = (a_1, ..., a_m)$, and nondeterministic inputs $\mathbf{y} = (y_1, ..., y_k)$. Furthermore, each nd-input y_i has a *type* A_i, where A_i is $\mathbb{R}$ or $\mathbb{Z}$. If $C(\mathbf{x}, \mathbf{y}, \mathbf{a})$ denotes the function computed by C, then C *sign-represents* the Boolean function f if for every $\mathbf{x} \in \{0, 1\}^n$ it holds that $f(\mathbf{x}) = 1$ iff for some $y_1 \in A_1, ..., y_k \in A_k$ it holds that $C(\mathbf{x}, \mathbf{y}, \mathbf{a}) \geqslant 0$. The definition of $\{0, 1\}$-nondeterministic circuits are analogous (in this case *only* Boolean guesses are considered).

The main tool used in the lower bound arguments is usually referred to as *connected component counting*, based on the Milnor–Thom theorem [27]. The theorem of Warren [43] provides a convenient formulation of the bounds used in these arguments. We state this result and its corollary from Goldberg and Jerrum [15].

Let $p_1, ..., p_u$ be polynomials of degree at most d in v variables. A *consistent* $(+, -)$*-sign assignment* is a solvable system

$$p_1 \Delta_1 0, ..., p_u \Delta_u 0, \tag{1}$$

where $\Delta_i \in \{<, >\}$ for $i = 1, ..., u$. A *consistent* $(+, 0, -)$*-sign assignment* is a solvable system (1) where $\Delta_i \in \{<, =, >\}$ for $i = 1, ..., u$.

THEOREM 2.1 (Warren [43]). *There are at most* $(4edu/v)^v$ *consistent* $(+, -)$*-sign assignments of the form* (1).

COROLLARY 2.2 (Goldberg and Jerrum [15]). *There are at most* $(8edu/v)^v$ *consistent* $(+, 0, -)$*-sign assignments of the form* (1).

We also need the bound for quantifier-elimination to deal with the nondeterministic model. For this purpose we use Renegar's theorem [31] in the case of existential formulas.

THEOREM 2.3 (Renegar [31]). *Consider an existential formula*

$$(\exists \mathbf{y} \in \mathbb{R}^k)\ Q(\mathbf{x}, \mathbf{y}), \tag{2}$$

where $\mathbf{x} = (x_1, ..., x_m)$ *is a vector of free variables,* $\mathbf{y} = (y_1, ..., y_k)$ *is a vector of quantified variables, and* Q *is a Boolean combination of* ℓ *polynomial equalities or inequalities of degree bounded by* d. *Then* (2) *is equivalent to a quantifier-free formula of the form*

$$\bigvee_{i=1}^{I} \bigwedge_{j=1}^{J_i} (h_{ij} \Delta_{ij} 0), \quad \text{where } I \leqslant (\ell d)^{O(mk)},\ J_i \leqslant (\ell d)^{O(k)},$$

the degree of h_{ij} *is at most* $(\ell d)^{O(k)}$, *and* $\Delta_{ij} \in \{<, \leqslant, =, \neq, \geqslant, >\}$.

3. LOWER BOUNDS FROM COUNTING

In this section we discuss lower bounds for the complexity of almost all Boolean functions in different circuit models.

THEOREM 3.1 (Gashkov [14]). *Almost all* n*-variable Boolean functions require arithmetic circuits of size* $\Omega(2^{n/2})$ *to be sign-represented.*

Proof. The proof is a variant of the standard counting argument in Boolean complexity theory (see Wegener [44]). For fixed positive integer N, we count the number of different Boolean functions that can be sign-represented by arithmetic circuits of size $\leqslant N$. We assume that $n \leqslant N$. By adding an additional gate we may assume that the sign-representation is *strict*; i.e., the output is never 0. To construct such circuits, one can use Boolean variable inputs $x_1, ..., x_n$, real constant inputs $a_1, ..., a_N$ and arithmetic gates $G_1, ..., G_N$. (It can be assumed w.l.o.g. that the circuit has at most N constant inputs.) Every input of each gate G_k can be the output of any other gate or any of the inputs x_i or a_j. So there are at most $(4N^2)^N$ ways to wire up these gates. Each gate computes one of the three basic arithmetic operations, $+$, $-$, or $\times$; so there are at most $(4N^2)^N \cdot 3^N$ arithmetic circuits of size N, *not taking into consideration how the constant inputs are fixed.*

Now, fix one such circuit C, with constant inputs $a_1, ..., a_m$ (where $m \leqslant N$). We have to count the number of different Boolean functions sign-represented by C, when we fix the constant inputs $a_1, ..., a_m$ in *all* (*infinitely many*) *possible ways.* The output of C is a *polynomial* $C(\mathbf{x}, \mathbf{a})$, for $\mathbf{x} = (x_1, ..., x_n)$ and $\mathbf{a} = (a_1, ..., a_m)$. An induction shows that the degree of C is at most 2^N. Now consider the system

$$C(\mathbf{x}_i, \mathbf{a})\ \Delta_i 0, \qquad i = 1, ..., 2^n, \tag{3}$$

of 2^n polynomials in the variables $\mathbf{a}$. Here $\mathbf{x}_1, ..., \mathbf{x}_{2^n}$ is an enumeration of all vectors in $\{0, 1\}^n$, and $\Delta_i \in \{<, >\}$. If the system (3) is consistent, then the circuit C with any solution of (3) as its constant inputs, strictly sign-represents the same Boolean function. So the number of different Boolean functions strictly sign-represented by changing the constant inputs of C is actually the number of different consistent $(+, -)$-sign assignments of the system (3). By Warren's theorem (Theorem 2.1), this number is at most

$$\left(\frac{4e2^n \cdot 2^N}{m}\right)^m < 2^{3N^2},$$

assuming $n \geqslant 4$. Therefore the number of different n-variable Boolean functions that can be sign-represented by arithmetic circuits of size $\leqslant N$ is at most $(4N^2)^N \cdot 3^N \cdot 2^{3N^2} < 2^{4N^2}$, when N is large enough. Now, comparing this number with the number of n-variable Boolean functions, i.e., 2^{2^n}, concludes the proof. ∎

We note that the lower bound remains valid if division is added to the basis. In fact, in this case the output of the circuit C is a rational function of the form

$$C(\mathbf{x}, \mathbf{a}) = \frac{f(\mathbf{x}, \mathbf{a})}{g(\mathbf{x}, \mathbf{a})},$$

where f and g are polynomials of degree $\leqslant 2^N$. Then instead of the system (3), we consider the system

$$f(\mathbf{x}_i, \mathbf{a}) \cdot g(\mathbf{x}_i, \mathbf{a})\ \Delta_i 0, \qquad i = 1, ..., 2^n.$$

Thus the same argument as above implies the following.

COROLLARY 3.2. *Almost all* n*-variable Boolean functions require* $\Omega(2^{n/2})$ *gates to be sign-represented by a circuit over the basis* $\{+, -, \times, /\}$ *and* $\mathbb{R}$.

Now we consider the extension of these lower bounds to arithmetic threshold circuits.

THEOREM 3.3. *Almost all* n*-variable Boolean functions require* $\Omega(2^{n/2})$ *gates to be computed by an arithmetic threshold circuit. The same bound holds if division is also added to the basis.*

Proof. The proof is similar to the proof of Theorem 3.1. Our goal is to count the number of different Boolean functions that can be computed by arithmetic threshold circuits of size $\leqslant N$, for any fixed positive integer N.

There are at most $(4N^2)^N \cdot 4^N$ arithmetic threshold circuits of size $\leqslant N$ with Boolean inputs $\mathbf{x} = (x_1, ..., x_n)$ and constant inputs $\mathbf{a} = (a_1, ..., a_N)$, without fixing the constant inputs.

Let C be such a circuit with constant inputs $a_1, ..., a_m$, where $m \leqslant N$. We want to count the number of different Boolean functions computed by C, when we fix the constant inputs $a_1, ..., a_m$ in all (infinitely many) possible ways.

Let $v_1, ..., v_t$ be the sign gates of C, where $t \leqslant N$. W.l.o.g. we can assume that v_t is the output gate of C. The output of each v_i is either -1 or 0 or $+1$. So for each gate v_i, $1 \leqslant i \leqslant t$, there is a polynomial $P_i(\mathbf{x}, \mathbf{a}, \boldsymbol{\alpha})$ such that if $\boldsymbol{\alpha} \in \{-1, 0, 1\}^t$ is the vector of the outputs of $v_1, ..., v_t$, then the input of each v_i, $1 \leqslant i \leqslant t$, is $P_i(\mathbf{x}, \mathbf{a}, \boldsymbol{\alpha})$. We note that $\boldsymbol{\alpha}$ is a function of $\mathbf{x}$ and $\mathbf{a}$. The polynomial P_i has degree at most 2^N and depends on the jth component of the vector $\boldsymbol{\alpha}$ only if the input of v_i depends on the output of v_j. We claim that the number of different Boolean functions computed by C with different assignments to the constants $a_1, ..., a_m$, is *at most* the number of different consistent $(+, 0, -)$ sign-assignments to the polynomials

$$P_i(\mathbf{x}_j, \mathbf{a}, \boldsymbol{\alpha}), \quad 1 \leqslant i \leqslant t, \ 1 \leqslant j \leqslant 2^n, \ \boldsymbol{\alpha} \in \{-1, 0, 1\}^t.$$

Here $\mathbf{x}_1, ..., \mathbf{x}_{2^n}$ is a list of all vectors in $\{0, 1\}^n$. Let $\mathbf{a}, \mathbf{a}' \in \mathbb{R}^m$ be solutions of

$$P_i(\mathbf{x}_j, \mathbf{a}, \boldsymbol{\alpha}_\ell) \, \Delta_{ij\ell} \, 0, \quad 1 \leqslant i \leqslant t, \ 1 \leqslant j \leqslant 2^n, \ 1 \leqslant \ell \leqslant 3^t,$$

for some $\Delta_{ij\ell} \in \{<, =, >\}$, where $\boldsymbol{\alpha}_1, ..., \boldsymbol{\alpha}_{3^t}$ is a list of all vectors in $\{-1, 0, 1\}^t$. Then it holds that if the constant inputs are fixed to be $\mathbf{a}$ (resp. $\mathbf{a}'$), then the circuit produces the same output for every Boolean input vector $\mathbf{x}$. This follows by induction on the number of sign gates.

By Warren's theorem (Corollary 2.2), the number of consistent $(+, 0, -)$ sign-assignments is at most

$$\left(\frac{8e \cdot 2^N \cdot t \cdot 2^n \cdot 3^t}{m}\right)^m \leqslant 2^{3N^2},$$

assuming that n is sufficiently large. Therefore the number of different n-variable Boolean functions can be computed by arithmetic threshold circuits of size $\leqslant N$ is at most

$$(4N^2)^N \cdot 4^N \cdot 2^{3N^2} \leqslant 2^{4N^2}.$$

This gives the desired bound.

As it is noted after the Theorem 3.1, even if the division gate is added to the basis of arithmetic threshold circuits, the same bound can be obtained. ▮

As a corollary, one obtains the same lower bound for arithmetic circuits with sign(x), $|x|$ and arbitrary real constants, noting $|x| = x \cdot \text{sign}(x)$. This includes PL circuits with *arbitrary* real constants. The lower bound of Gashkov [14] for Lipschitz bases, applicable to PL circuits assumes that the constants are restricted to some bounded interval.

COROLLARY 3.4. *Almost all n-variable Boolean functions require $\Omega(2^{n/2})$ gates to be computed by PL circuits using arbitrary real constants.*

We close this section by discussing the case of nondeterministic circuits.

THEOREM 3.5. *Almost all n-variable Boolean functions require $\Omega(2^{n/4})$ gates to be computed by a nondeterministic arithmetic threshold circuit. The same bound holds if division is also added to the basis.*

Proof. The proof is similar to the proofs of Theorems 3.1 and 3.3; so we do not go through all the details.

Let C be a nondeterministic arithmetic circuit of size $\leqslant N$ with Boolean inputs $x_1, ..., x_n$, nd-inputs $y_1, ..., y_k$, and constant inputs $a_1, ..., a_m$ (where $k, m \leqslant N$). We want to count the number of different Boolean functions computed by C with all possible assignments to the constants $a_1, ..., a_m$.

Let $v_1, ..., v_t$ be the sign-gates of C ($t \leqslant N$), where v_t is the output gate of C. As in the proof of Theorem 3.3, we can associate a polynomial $P_i(\mathbf{x}, \mathbf{y}, \mathbf{a}, \boldsymbol{\alpha})$ to the sign-gate v_i, $1 \leqslant i \leqslant t$, such that if $\boldsymbol{\alpha} \in \{-1, 0, 1\}^t$ is the vector of the outputs of $v_1, ..., v_t$ then the input of v_i, $1 \leqslant i \leqslant t$, is $P_i(\mathbf{x}, \mathbf{y}, \mathbf{a}, \boldsymbol{\alpha})$. Here $\boldsymbol{\alpha}$ is a function of $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{a}$. Note that the degree of P_i is at most 2^N.

Let us consider a setting $\mathbf{a}$ of the constant inputs of C. Then a Boolean input $\mathbf{x}$ is accepted iff there is a setting $\mathbf{y}$ of the nondeterministic variables and a sequence $\boldsymbol{\alpha} \in \{-1, 0, 1\}^t$ of the outputs of the sign gates such that $\alpha_t = 1$ and for every $i = 1, ..., t$ it holds that $\text{sign}(P_i(\mathbf{x}, \mathbf{y}, \mathbf{a}, \boldsymbol{\alpha})) = \alpha_i$. Thus $\mathbf{x}$ is accepted iff

$$\exists \mathbf{y} \in \mathbb{R}^m \left(\bigvee_{\substack{\boldsymbol{\alpha} \in \{-1, 0, 1\}^t \\ \alpha_t = 1}} \bigwedge_{i=1}^{t} \text{sign}(P_i(\mathbf{x}, \mathbf{y}, \mathbf{a}, \boldsymbol{\alpha})) = \alpha_i \right).$$

Note that for a polynomial P and $\alpha \in \{-1, 0, 1\}$, the formula $\text{sign}(P) = \alpha$ is equivalent to $P < 0$ if $\alpha = -1$, to $P = 0$ if $\alpha = 0$ and to $P > 0$ if $\alpha = 1$. By the bound for quantifier elimination (Theorem 2.3), this formula is equivalent to a Boolean combination of

$$(3^t \cdot t \cdot 2^N)^{O(N^2)} = 2^{O(N^3)}$$

polynomial equations or inequalities of degree at most $2^{O(N^2)}$ in m variables. Thus the Boolean function represented by the nondeterministic circuit C with the setting $\mathbf{a}$ of the constant inputs depends on the number of consistent $(+, 0, -)$ sign-assignments to $2^n \cdot 2^{O(N^3)}$ polynomials of degree at most $2^{O(N^2)}$ in m variables.

Again, by Warren's theorem (Corollary 2.2) this is at most

$$\left(\frac{8e\cdot 2^N\cdot 2^n\cdot 2^{O(N^3)}}{m}\right)^m = 2^{O(N^4)}.$$

The rest of the proof is analogous to the previous arguments. ▮

4. UPPER BOUNDS

In this section we prove matching upper bounds for some of the lower bounds of the previous section.

4.1. *Arithmetic Circuits*

The first upper bound shows that the lower bound of Theorem 3.1 is tight.

THEOREM 4.1. *Every Boolean function* $f\colon \{0,1\}^n \to \{0,1\}$ *can be sign-represented by an arithmetic circuit of size* $O(2^{n/2})$.

Proof. We can assume w.l.o.g. that n is even. For every $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_{n/2}) \in \{0,1\}^{n/2}$, let $f_{\boldsymbol{\alpha}}$ be the subfunction of f on $\{x_{n/2+1}, ..., x_n\}$ defined by

$$f_{\boldsymbol{\alpha}}(x_{n/2+1}, ..., x_n) = f(\alpha_1, ..., \alpha_{n/2}, x_{n/2+1}, ..., x_n).$$

Each subfunction $f_{\boldsymbol{\alpha}}$ can be represented as a vector $[f_{\boldsymbol{\alpha}}] \in \{0,1\}^{2^{n/2}}$. The following lemma shows that we can encode vectors in $\{0,1\}^{2^{n/2}}$ as real numbers so that every vector can be recovered from its code by a small size circuit up to its sign. In the following we use another form of the sign function defined as

$$\mathrm{sg}(x) = \begin{cases} 1 & \text{if } x \geqslant 0, \\ 0 & \text{if } x < 0. \end{cases}$$

LEMMA 4.2. *For every* m, *there is a function* $\psi_m\colon \{0,1\}^m \to \mathbb{R}$ *and an arithmetic circuit* C_m *of size* $O(m)$ *having a single input and* m *outputs such that the following holds. For every* $\mathbf{x} = (x_1, ..., x_m) \in \{0,1\}^m$, *the circuit* C_m *over the input* $\psi_m(\mathbf{x})$ *gives the output* $\mathbf{y} = (y_1, ..., y_m)$ *such that* $y_i \neq 0$ *and* $x_i = \mathrm{sg}(y_i)$, *for every* $i = 1, ..., m$.

Proof. Consider the quadratic polynomial $g(z) = -z^2 + 4z$. Then $g([0,4]) = [0,4]$. For any $0 < a < 4$, the set $g^{-1}(a)$ contains two numbers; let us call them a^0 and a^1 where $a^0 < 2 < a^1$.

We define the encoding ψ_m inductively. For $m = 1$, let $\psi_1(0) = 1$ and $\psi_1(1) = 3$. Suppose ψ_m is defined. Consider $\varepsilon = (\varepsilon_1, ..., \varepsilon_m, \varepsilon_{m+1}) \in \{0,1\}^{m+1}$. Let $a = \psi_m(\varepsilon_1, ..., \varepsilon_m)$. Then define

$$\psi_{m+1}(\varepsilon) = \begin{cases} a^0 & \text{if } \varepsilon_{m+1} = 0; \\ a^1 & \text{if } \varepsilon_{m+1} = 1. \end{cases} \tag{4}$$

More explicitly, ψ_m is defined inductively as

$$\psi_1(0) = 1, \qquad \phi_1(1) = 3$$

$$\psi_{m+1}(\varepsilon_1, ..., \varepsilon_m, \varepsilon_{m+1}) = 2 - (-1)^{\varepsilon_{m+1}}\sqrt{4 - \psi_m(\varepsilon_1, ..., \varepsilon_m)}.$$

This completes the definition of the encoding ψ_m.

Now consider $\mathbf{x} = (x_1, ..., x_m) \in \{0,1\}^m$; and suppose $\psi_m(\mathbf{x}) = a$. From the definition of ψ_m it follows that $x_m = 1$ iff $a > 2$; $x_{m-1} = 1$ iff $g(a) > 2$, and so on. In general, it holds that

$$x_{m-i} = \mathrm{sg}(g^{(i)}(a) - 2)$$

(here $g^{(i)}$ is the result of i times iterating g). So an arithmetic circuit of size $O(m)$ computes all $y_{m-i} = g^{(i)}(a) - 2$, for $0 \leqslant i \leqslant m-1$. ▮

Now we describe the arithmetic circuit C_f which computes f. Let $\boldsymbol{\alpha}_0, ..., \boldsymbol{\alpha}_{t-1}$ (for $t = 2^{n/2}$) be the list of all vectors in $\{0,1\}^{n/2}$. Let $d_i = \psi_t([f_{\boldsymbol{\alpha}_i}])$ (for $i = 0, ..., t-1$), where ψ_t is the encoding defined by Lemma 4.2.

The desired circuit has the d_i's as its constant inputs. For the input $(x_1, ..., x_n)$ first it computes the sequences

$$(\sigma_0, ..., \sigma_{t-1}) \qquad \text{and} \qquad (\sigma'_0, ..., \sigma'_{t-1}) \tag{5}$$

such that $\sigma_j = 0$ for $j \neq i_0$ and $\sigma_{i_0} = 1$, where $i_0 = \sum_{i=1}^{n/2} x_i 2^{i-1}$, and similarly $\sigma'_j = 0$ for $j \neq i_1$ and $\sigma'_{i_1} = 1$ for $i_1 = \sum_{i=1}^{n/2} x_{i+n/2} 2^{i-1}$. A standard recursive construction gives a Boolean circuit of size $O(t)$ which computes these sequences. This circuit can be simulated by an arithmetic circuit of size $O(t)$. Then $\sum_{i=0}^{t-1} \sigma_i d_i = d_{i_0} = \psi_t([f_{\boldsymbol{\alpha}_{i_0}}])$, where $\boldsymbol{\alpha}_{i_0} = (x_1, ..., x_{n/2})$. Next d_{i_0} is fed in the circuit C_t of Lemma 4.2. Suppose $y_0, ..., y_{t-1}$ is the output of C_t; then the desired output is $\sum_{i=0}^{t-1} \sigma'_i y_i$. ▮

4.2. *Piecewise Linear Circuits*

Gashkov [14] showed that if $\mathscr{B}$ is any Lipschitz basis with constant inputs from some bounded interval Δ, then almost all n-variable Boolean functions require circuits of size $\Omega(2^{n/2})$ over $\mathscr{B}$. We show that for PL circuits this bound is tight. This also implies that the lower bound of Theorem 3.3 is tight.

The construction is a modification of the previous one. Before we go through the construction, let us note that the Boolean gates $\neg$, $\vee$, and $\wedge$ can be simulated by constant

size PL circuits. For example, an $\wedge$-gate is simulated by $|x+y-\frac{1}{2}|-\frac{1}{2}$.

First, we introduce an encoding of Boolean functions very similar to what is introduced in Lemma 4.2. Note that here, from the code of a vector we can recover each of its bits exactly, and not only up to their sign.

LEMMA 4.3. (i) *For every m there is a function $\varphi_m: \{0,1\}^m \to [0,1]$ and a PL circuit C_m of size $O(m)$ having a single input and m outputs such that the following holds. For every $\mathbf{x}=(x_1, ..., x_m) \in \{0,1\}^m$, the circuit C_m over the input $\varphi_m(\mathbf{x})$ gives the output $\mathbf{y}=(y_1, ..., y_m)$ so that $y_i \neq 0$ and $x_i = \mathrm{sg}(y_i)$, for every $i=1, ..., m$.*

(ii) *There is a PL circuit C'_m of size $O(m)$ having a single input and a single output which for every output y_j of the circuit C_m in* (i) *outputs* $\mathrm{sg}(y_j)$.

Proof. (i) Here we use the function h defined by

$$h(z)=\begin{cases}2z & \text{if} \quad z \leqslant \frac{1}{2},\\ -2z+2 & \text{if} \quad z > \frac{1}{2},\end{cases}$$

instead of the quadratic polynomial $g(z)$ in Lemma 4.2. Then $h([0,1])=[0,1]$, and as $h(z)=1-|2z-1|$, h can be computed by using four PL gates. Put $\varphi_1(0)=\frac{1}{3}$ and $\varphi_1(1)=\frac{2}{3}$. The inductive definition of φ_m is analogous to the definition of ψ_m by (4) in Lemma 4.2; and with same reasoning if $\mathbf{x}=(x_1, ..., x_m) \in \{0,1\}^m$ and $a=\varphi_m(\mathbf{x})$, then

$$x_{m-i} = \mathrm{sg}(h^{(i)}(a)-\tfrac{1}{2}).$$

Thus a PL circuit of size $O(m)$ can compute all $y_{m-i}=h^{(i)}(a)-\frac{1}{2}$.

(ii) Let A_m be the range of the function φ_m in (i); i.e., A_m is the set of encodings of the strings of length m. It follows by induction that

$$A_m = \left\{ \frac{i}{3\cdot 2^{m-1}} : 0 < i < 3\cdot 2^{m-1},\ 3 \nmid i \right\}.$$

This shows $A_m \subset A_{m+1}$ for every $m \geqslant 1$. For the set

$$A'_m = \{a-\tfrac{1}{2} : a \in A_m\}$$

it follows that

$$A'_m = \left\{ \frac{i}{3\cdot 2^{m-1}} :\ -3\cdot 2^{m-2} < i < 3\cdot 2^{m-2},\ 3 \nmid i \right\} \qquad (6)$$

if $m>1$ and $A'_1=\{-\frac{1}{6}, \frac{1}{6}\}$. Note $A'_m \subseteq [\frac{1}{2}, \frac{1}{2}]$.

Let y_j be an output of C_m for some input $a=\varphi_m(\mathbf{x})$. Then $y_j = h^{(m-j)}(a)-\frac{1}{2}$. As, by definition, $h(A_l)=A_{l-1}$, for every $l>1$, it follows that $y_j \in A'_j$.

Now consider the function

$$u(z)=\begin{cases}-2z-1 & \text{if} \quad z \leqslant -\frac{1}{4},\\ 2z & \text{if} \quad -\frac{1}{4} < z < \frac{1}{4},\\ -2z+1 & \text{if} \quad z \geqslant \frac{1}{4}.\end{cases}$$

Note that $u(z)=|2z+\frac{1}{2}|+|-2z+\frac{1}{2}|-2z$; so it can be computed by using nine PL gates.

The definition of u and (6) implies

$$u(A'_l) = A'_{l-1}, \qquad (7)$$

for every $l>1$. Since $u([-\frac{1}{2}, 0])=[-\frac{1}{2}, 0]$ and $u([0, \frac{1}{2}])=[0, \frac{1}{2}]$, we have

$$u(A_l) \cap [0, \tfrac{1}{2}] = A_{l-1} \cap [0, \tfrac{1}{2}]$$
$$u(A_l) \cap [-\tfrac{1}{2}, 0] = A_{l-1} \cap [-\tfrac{1}{2}, 0].$$

We claim that the circuit C'_m computing $\frac{1}{2}(3u^{(m+1)}(y)+1)$ satisfies the requirements. This follows from the following claim.

CLAIM. *For every $j \geqslant 1$, if $y_j > 0$ then $u^{(m+1)}(y_j)=\frac{1}{3}$ and if $y_j<0$ then $u^{(m+1)}(y_j)=-\frac{1}{3}$.*

Proof. Follows from (7) and the facts that $-\frac{1}{3}$ and $\frac{1}{3}$ are fixed-points of the mapping u and $u(A'_1)=\{-\frac{1}{3}, \frac{1}{3}\}$. ∎

This completes the proof of Lemma 4.3. ∎

THEOREM 4.4. *Every Boolean function $f: \{0,1\}^n \to \{0,1\}$ can be computed* exactly *by a PL circuit of size $O(2^{n/2})$.*

Proof. We describe the structure of the desired PL circuit C_f which computes f. As in the previous section, for $\boldsymbol{\alpha} \in \{0,1\}^{n/2}$, the vector $[f_{\boldsymbol{\alpha}}] \in \{0,1\}^{2^{n/2}}$ represents the subfunction $f_{\boldsymbol{\alpha}}$. Again, let $\boldsymbol{\alpha}_0, ..., \boldsymbol{\alpha}_{t-1}$ (for $t=2^{n/2}$) be the list of vectors in $\{0,1\}^{n/2}$. Let $e_i=\varphi_t([f_{\boldsymbol{\alpha}_i}])$ (for $i=0, ..., t-1$), where φ_t is the encoding defined by Lemma 4.3. Note that $0<e_i<1$. The circuit C_f has the e_i's as its constant inputs.

For the input $\mathbf{x}=(x_1, ..., x_n)$, C_f first finds $e_{i_0}=\varphi_t([f_{\boldsymbol{\alpha}_{i_0}}])$, where $\boldsymbol{\alpha}_{i_0}=(x_1, ..., x_{n/2})$. To do this, C_f computes the sequence $(\sigma_0, ..., \sigma_{t-1})$ as in (5), where $\sigma_i=1$ iff $(x_1, ..., x_{n/2})=\boldsymbol{\alpha}_i$. This needs $O(2^{n/2})$ Boolean gates and it can be simulated by $O(2^{n/2})$ PL gates. Note that exactly one of the σ_i's is 1. Next C_f adds each σ_i to e_i to obtain $h_i=\sigma_i+e_i$. So $h_{i_0}>1$ and $h_i<1$ for all $i \neq i_0$. Next C_f applies a circuit of size $O(2^{n/2})$ over the sequence $(h_0, ..., h_{t-1})$ to find h_{i_0}. For this C_f utilizes $\max(x, y)$ which can be computed by a constant size PL circuit, because $\max(x, y)=\frac{1}{2}(x+y+|x-y|)$. After finding h_{i_0}, e_{i_0} is obtained using $e_{i_0}=h_{i_0}-1$. Using the circuit C_t from Lemma 4.3(i) with input e_{i_0}, C_f obtains outputs $y_0, ..., y_{t-1}$.

Computing $(\sigma'_0, ..., \sigma'_{t-1})$ from $x_{n/2+1}, ..., x_n$ as in (5), C_f then finds y_{i_1} for which $\sigma'_{i_1} = 1$ using the same procedure as above. Finally, the circuit C'_t in Lemma 4.3(ii) can be used to find the desired output $\mathrm{sg}(y_{i_1})$. ∎

We would like to mention that the encoding and decoding procedures described in the proofs of Lemmas 4.2 and 4.3 can be viewed as utilizing the orbits of the *dynamical systems* (see [12]) defined by the logistic map $g(z)$ and the tent maps $h(z)$ and $u(z)$.

5. DISCONTINUITY AND NONDETERMINISM

In this section we show that there is a relationship between arithmetic threshold and nondeterministic arithmetic circuits, as nondeterministic arithmetic circuits can simulate arithmetic threshold circuits with only a constant factor increase in size.

THEOREM 5.1. *If the Boolean function f is computed by an arithmetic threshold circuit C, then it can be sign-represented by a nondeterministic arithmetic circuit of size $O(\mathrm{size}(C))$.*

Proof. Suppose C is an arithmetic threshold circuit which computes the Boolean function $f: \{0,1\}^n \to \{0,1\}$. Suppose C has t sign gates $G_1, ..., G_t$, listed in topological ordering, and let $z_i(\mathbf{x})$ be the input of G_i (note that z_i is not necessarily a polynomial function). We can assume that $z_i(\mathbf{x}) \neq 0$ for each i and each input $\mathbf{x} \in \{0,1\}^n$. Indeed, for each $i = 1, ..., t$, let $\alpha_i = \frac{1}{2} \min\{|z_i(\mathbf{x})| : \mathbf{x} \in \{0,1\}^n, z_i(\mathbf{x}) \neq 0\}$. Then substitute the sign gate G_i by a subcircuit of constant size computing $\frac{1}{2}(\mathrm{sign}(z_i + \alpha_i) + \mathrm{sign}(z_i - \alpha_i))$. The result is a new circuit of size $\leqslant 6\mathrm{size}(C)$ which computes the same Boolean function and the input of each sign gate in it is always nonzero.

We now construct a nondeterministic arithmetic circuit $\bar{C}$ which simulates C. The circuit $\bar{C}$ has t nd-inputs $u_1, ..., u_t$, which are used to produce the numbers $v_i = u_i^2$, $1 \leqslant i \leqslant t$. To obtain $\bar{C}$, we first construct a nondeterministic arithmetic circuit C' by the following process. Substitute each sign gate G_i of C by a multiplication gate $\overline{G_i}$ such that one of its inputs is the (only) input of G_i and its other input is v_i. Let $s_i(\mathbf{x}, \mathbf{u})$ denote the input of $\overline{G_i}$ other than v_i, where $\mathbf{x} = (x_1, ..., x_n)$ and $\mathbf{u} = (u_1, ..., u_t)$. Then s_i is a polynomial function of $\mathbf{x}$ and $\mathbf{u}$. Thus the output of $\overline{G_i}$ can be written as $\overline{G_i}(\mathbf{x}, \mathbf{u}) = s_i(\mathbf{x}, \mathbf{u}) \cdot v_i$. The output of C' is the polynomial $C'(\mathbf{x}, \mathbf{u})$.

Then the output of the desired circuit $\bar{C}$ is defined as

$$\bar{C}(\mathbf{x}, \mathbf{u}) := C'(\mathbf{x}, \mathbf{u}) - \gamma C''(\mathbf{x}, \mathbf{u})[S(\mathbf{x}, \mathbf{u}) - 1] - \tfrac{1}{2}, \qquad (8)$$

where $\gamma \geqslant 1$ is a constant input, C'' is an arithmetic circuit that will be defined later, and

$$S(\mathbf{x}, \mathbf{u}) = \prod_{i=1}^{t} ((s_i(\mathbf{x}, \mathbf{u})\, v_i - 1)^2 (s_i(\mathbf{x}, \mathbf{u})\, v_i + 1)^2 + 1).$$

We formulate some properties of $S(\mathbf{x}, \mathbf{u})$ that will be used later on.

LEMMA 5.2. (a) *For every $\mathbf{x} \in \{0,1\}^n$ and $\mathbf{u} \in \mathbb{R}^t$ it holds that $S(\mathbf{x}, \mathbf{u}) \geqslant 1$.*

(b) *For every $\mathbf{x} \in \{0,1\}^n$ and $\mathbf{u} \in \mathbb{R}^t$ it holds that $S(\mathbf{x}, \mathbf{u}) = 1$ iff for every $i = 1, ..., t$*

$$u_i = \pm \frac{1}{\sqrt{|z_i(\mathbf{x})|}}, \qquad v_i = \frac{1}{|z_i(\mathbf{x})|} \qquad (9)$$

and for every $i = 1, ..., t$

$$s_i(\mathbf{x}, \mathbf{u}) = z_i(\mathbf{x}), \qquad \overline{G_i}(\mathbf{x}, \mathbf{u}) = G_i(\mathbf{x}). \qquad (10)$$

(c) *If $S(\mathbf{x}, \mathbf{u}) = 1$ then $C'(\mathbf{x}, \mathbf{u}) = C(\mathbf{x})$.*

(d) *For every $\mathbf{x} \in \{0,1\}^n$ there is a rectangle $\Lambda_{\mathbf{x}} \subseteq \mathbb{R}^t$ such that $S(\mathbf{x}, \mathbf{u}) > 2$ for every $\mathbf{u} \notin \Lambda_{\mathbf{x}}$.*

Proof. (a) is obvious.

We assumed that the sign gates of C are ordered as $G_1, ..., G_t$ such that the input of G_i does not depend on the output of G_j for $j > i$. To prove (b), first assume $S(\mathbf{x}, \mathbf{u}) = 1$. Then an induction on i shows that (9) and (10) hold. Conversely, if (9) and (10) hold then

$$(s_i(\mathbf{x}, \mathbf{u})\, v_i - 1)(s_i(\mathbf{x}, \mathbf{u})\, v_i + 1) = 0$$

for every $i = 1, ..., t$. Therefore $S(\mathbf{x}, \mathbf{u}) = 1$.

(c) follows obviously from (b).

To show (d) we write

$$S(\mathbf{x}, \mathbf{u}) = ((s_1(\mathbf{x}, \mathbf{u})\, v_1 - 1)^2 (s_1(\mathbf{x}, \mathbf{u})\, v_1 + 1)^2 + 1) \times \prod_{i=2}^{t} ((s_i(\mathbf{x}, \mathbf{u})\, v_i - 1)^2 (s_i(\mathbf{x}, \mathbf{u})\, v_i + 1)^2 + 1).$$

As G_1 is the first sign-gate, s_1 only depends on $\mathbf{x}$ and $s_1(\mathbf{x}) = z_1(\mathbf{x}) \neq 0$. Thus solving

$$(s_1(\mathbf{x})\, u_1^2 - 1)(s_1(\mathbf{x})\, u_1^2 + 1) + 1 \leqslant 2$$

for u_1, we get a closed interval I_1 such that

$$\Lambda_{\mathbf{x}} := I_1 \times [0,1]^{t-1}$$

satisfies the requirement. ∎

The following lemma will be used to define the circuit C''.

LEMMA 5.3. *Suppose the polynomial $P(y_1, ..., y_m)$ is the output of an arithmetic circuit P. Then there is an arithmetic circuit Q computing a polynomial $Q(y_1, ..., y_m)$ such that $\mathrm{size}(Q) \leqslant 3\mathrm{size}(P)$, $Q(y_1, ..., y_m) \geqslant 1$ and $|P(y_1, ..., y_m)| < Q(y_1, ..., y_m)$ for all $y_1, ..., y_m \in \mathbb{R}$.*

Proof. The circuit Q is obtained from P by replacing subtractions by additions, substituting each input y_i By y_i^2+1 and each constant input a_j by the constant input a_j^2+1. So size$(Q) \leqslant 3$size(P). Now induction on size(P) shows that the circuit Q has the desired properties. ▮

The circuit C'' in (8) is obtained from C' as in the above lemma. So $C''(\mathbf{x}, \mathbf{u}) \geqslant 1$ and $|C'(\mathbf{x}, \mathbf{u})| < C''(\mathbf{x}, \mathbf{u})$, for all $\mathbf{x}$ and $\mathbf{u}$.

In view of Lemma 5.2, we define

$$\Gamma_{\mathbf{x}} := \prod_{i=1}^{t} \left\{ \pm \frac{1}{\sqrt{|z_i(\mathbf{x})|}} \right\} \subseteq \mathbb{R}^t$$

as the set of *correct guesses* for $\mathbf{x} \in \{0, 1\}^n$. Then Lemma 5.2 implies that for every $\mathbf{u} \in \Gamma_{\mathbf{x}}$ we have $C'(\mathbf{x}, \mathbf{u}) = C(\mathbf{x}) = f(\mathbf{x})$.

Lemma 5.2 implies that if $\mathbf{u} \in \Gamma_{\mathbf{x}}$ then

$$\bar{C}(\mathbf{x}, \mathbf{u}) = C'(\mathbf{x}, \mathbf{u}) - \tfrac{1}{2} = C(\mathbf{x}) - \tfrac{1}{2} = f(\mathbf{x}) - \tfrac{1}{2}.$$

Hence if $f(\mathbf{x}) = 1$ then $\bar{C}$ accepts $\mathbf{x}$ (with a correct guess $\mathbf{u}$ from $\Gamma_{\mathbf{x}}$). It remains to be shown that the constant γ in (8) can be chosen in such a way that if $f(\mathbf{x}) = 0$ then $\bar{C}(\mathbf{x}, \mathbf{u}) < 0$ for every $\mathbf{u} \in \mathbb{R}^t$.

Fix $\mathbf{x} \in \{0, 1\}^n$ such that $f(\mathbf{x}) = 0$. Let

$$\Omega_{\mathbf{x}} = \{\mathbf{u} \in \mathbb{R}^t : C'(\mathbf{x}, \mathbf{u}) \geqslant \tfrac{1}{2}\}.$$

Therefore $\Omega_{\mathbf{x}} \subseteq \mathbb{R}^t \backslash \Gamma_{\mathbf{x}}$.

Since $C'(\mathbf{x}, \mathbf{u})$ is a continuous function and $C'(\mathbf{x}, \mathbf{u}_0) = 0$ for every $\mathbf{u}_0 \in \Gamma_{\mathbf{x}}$, there is a $\delta_{\mathbf{x}} > 0$ so that for every $\mathbf{u}_0 \in \Gamma_{\mathbf{x}}$

$$\mathbf{u} \in \Omega_{\mathbf{x}} \Rightarrow d(\mathbf{u}, \mathbf{u}_0) \geqslant \delta_{\mathbf{x}}, \tag{11}$$

where d is the Euclidean metric for $\mathbb{R}^t$.

Now let

$$\gamma_{\mathbf{x}} = \min\left\{ S(\mathbf{x}, \mathbf{u}) - 1 : \mathbf{u} \in \Lambda_{\mathbf{x}} \Big\backslash \bigcup_{\mathbf{u} \in \Gamma_{\mathbf{x}}} B(\mathbf{u}; \delta_{\mathbf{x}}) \right\},$$

where $B(\mathbf{u}; r) = \{\mathbf{v} \in \mathbb{R}^t : d(\mathbf{u}, \mathbf{v}) < r\}$ and $\Lambda_{\mathbf{x}}$ is the rectangle in Lemma 5.2(d). Then $\gamma_{\mathbf{x}} > 0$ exists, because it realizes the minimum value of a nonzero continuous function over a compact set. Define

$$\gamma = \max_{f(\mathbf{x}) = 0} \{1/\gamma_{\mathbf{x}}, 1\}.$$

Let $f(\mathbf{x}) = 0$ and $\mathbf{u} \in \mathbb{R}^t$. If $\mathbf{u} \notin \Omega_{\mathbf{x}}$, i.e. $C'(\mathbf{x}, \mathbf{u}) < \frac{1}{2}$ then, because $\gamma \geqslant 1$, $C''(\mathbf{x}, \mathbf{u}) \geqslant 1$ (by Lemma 5.3) and $S(\mathbf{x}, \mathbf{u}) \geqslant 1$, Eq. (8) implies $\bar{C}(\mathbf{x}, \mathbf{u}) < 0$. Otherwise, we consider two cases:

If $\mathbf{u} \in \Omega_{\mathbf{x}} \backslash \Lambda_{\mathbf{x}}$ then $S(\mathbf{x}, \mathbf{u}) > 2$, and since $C''(\mathbf{x}, \mathbf{u}) > |C'(\mathbf{x}, \mathbf{u})|$ and $\gamma \geqslant 1$, it follows from (8) that $\bar{C}(\mathbf{x}, \mathbf{u}) < 0$.

Finally, if $\mathbf{u} \in \Omega_{\mathbf{x}} \cap \Lambda_{\mathbf{x}}$ then (by (11)) since

$$\mathbf{u} \notin \bigcup_{\mathbf{u}_0 \in \Gamma_{\mathbf{x}}} B(\mathbf{u}_0; \delta_{\mathbf{x}})$$

and by the very definition of γ, $\gamma[S(\mathbf{x}, \mathbf{u}) - 1] \geqslant 1$, so

$$\gamma C''(\mathbf{x}, \mathbf{u})[S(\mathbf{x}, \mathbf{u}) - 1] > C'(\mathbf{x}, \mathbf{u})$$

and, again, $\bar{C}(\mathbf{x}, \mathbf{u}) < 0$.

This completes the description and the proof of correctness of the circuit $\bar{C}$.

The bound on the size of $\bar{C}$ follows by noting that the size of C' and C'' is $O(\text{size}(C))$ by construction. As each s_i is computed at some gate of C', $S(\mathbf{x}, \mathbf{u})$ can also be computed with $O(\text{size}(C))$ additional gates. ▮

We also observe that the $(\mathbb{R}, \mathbb{Z})$-nondeterministic circuits are "too powerful," providing a different kind of example of the "pathologies" mentioned in the Introduction.

THEOREM 5.4. *Every Boolean function $f: \{0, 1\}^n \to \{0, 1\}$ can be sign-represented by some $(\mathbb{R}, \mathbb{Z})$-nondeterministic arithmetic circuit of size $O(n)$.*

Proof. Let $[f] \in \{0, 1\}^{2^n}$ represent f as a vector. This means that if $[f] = (\varepsilon_{2^n-1}, ..., \varepsilon_1, \varepsilon_0)$ then for $\mathbf{x} \in \{0, 1\}^n$, $f(\mathbf{x}) = \varepsilon_j$, where

$$j = \sum_{i=1}^{n} x_i 2^{i-1}. \tag{12}$$

We also consider $[f]$ as the nonnegative integer $\sum_{i=0}^{2^n-1} \varepsilon_i 2^i$.

The constant inputs of the circuit are the numbers 2^i and 2^{2^i} for $0 \leqslant i \leqslant n-1$, and $[f]$. It has four nd-inputs:

$$u_1 \in \mathbb{R}, \quad u_2 \in \mathbb{R}, \quad v \in \mathbb{Z}, \quad w \in \mathbb{R}.$$

On input $\mathbf{x} = (x_1, ..., x_n) \in \{0, 1\}^n$, the circuit first computes j by (12) and

$$m = 2^j = \prod_{i=1}^{n} ((2^{2^{i-1}} - 1)\, x_i + 1).$$

Then $f(\mathbf{x}) = 1$ iff for some integers v and w with $0 \leqslant w < m$ we have

$$[f] = 2vm + m + w. \tag{13}$$

This holds as (13) implies $v \geqslant 0$; and the jth bit of $2vm + w$ (for $v \geqslant 0$) is zero. Actually we can write a weaker condition, only requiring the integrality of v:

$$f(\mathbf{x}) = 1$$

iff

$$(\exists v \in \mathbb{Z})(\exists w \in \mathbb{R})[\, -1 < w < m \text{ and } [\, f\,] = 2vm + m + w\,].$$

To check the conditions inside the brackets note that if $A > 0$ then there exists a $u \in \mathbb{R}$ such that $Au^2 - 1 = 0$ and if $A \leqslant 0$ then for all $u \in \mathbb{R}$, $Au^2 - 1 < 0$. Therefore,

$$f(\mathbf{x}) = 1$$

iff

$$(\exists u_1 \in \mathbb{R})(\exists u_2 \in \mathbb{R})(\exists v \in \mathbb{Z})(\exists w \in \mathbb{R})[\, T(u_1, u_2, v, w) = 1\,],$$

where

$$T(u_1, u_2, v, w) = (((w+1)\, u_1^2 - 1)^2 + ((m-w)\, u_2^2 - 1)^2 + 1) \times (([\, f\,] - 2vm - m - w)^2 + 1).$$

Note that $T(u_1, u_2, v, w) \geqslant 1$ for every $(u_1, u_2, v, w) \in \mathbb{R}^4$; if $f(\mathbf{x}) = 1$ then the equation $T(u_1, u_2, v, w) = 1$ has a solution with $v \in \mathbb{Z}$, and if $f(\mathbf{x}) = 0$ then for all values of u_1, u_2, v, and w we have $T(u_1, u_2, v, w) > 1$. Consequently, $f(\mathbf{x}) = 1$ iff

$$(\exists u_1 \in \mathbb{R})(\exists u_2 \in \mathbb{R})(\exists v \in \mathbb{Z})(\exists w \in \mathbb{R})[\, 1 - T(u_1, u_2, v, w) \geqslant 0\,].$$

Since a circuit of size $O(n)$ can compute j, m, and T, the proof is complete. ∎

It may be of interest to compare Theorem 5.4 with the following bound for $\{0, 1\}$-nondeterministic circuits.

PROPOSITION 5.5. *Almost all n-variable Boolean functions require size $\Omega(2^{n/2})$ to be computed by a $\{0, 1\}$-nondeterministic arithmetic threshold circuit.*

Proof. The argument is a slight modification of the proof of Theorem 3.3. ∎

6. EXPLICIT LOWER BOUNDS

In this section we prove lower bounds that hold for *explicitly defined* Boolean functions.

6.1. *Arithmetic Expressions and Their Generalizations*

For the first lower bound, we consider bases consisting of finitely many multilinear polynomials and all constants from $\mathbb{R}$.

Let $f: \{0, 1\}^n \to \{0, 1\}$ be a Boolean function with variables $X = \{x_1, ..., x_n\}$ and $(S_1, ..., S_p)$ be a partition of X with $|S_i| = \ell_i$, $i = 1, ..., p$. Furthermore, let s_i be the number of subfunctions of f on S_i obtained by fixing the variables outside S_i in all possible ways, and let r_i be defined by

$$(2^{\ell_i + 4} r_i)^{r_i} = s_i, \tag{14}$$

for $i = 1, ..., p$.

THEOREM 6.1. *Let $\mathscr{B}$ be any basis of finitely many multilinear polynomials and all constants from $\mathbb{R}$. Then every formula sign-representing f over $\mathscr{B}$ has size*

$$\Omega\left(\sum_{1 \leqslant i \leqslant p} r_i\right).$$

Proof. We assume w.l.o.g. that the basis contains addition and the function

$$\psi(x, y, z) = xy + z.$$

(Adding these functions to the basis only makes the lower bounds stronger.)

Let F be a formula over $\mathscr{B}$ sign-representing f, and let t_i be the number of leaves labelled by variables from S_i. Adding a final gate that adds a small positive constant to the output of F, if necessary, it may be assumed that $F(\mathbf{x}) \neq 0$ for all $\mathbf{x} \in \{0, 1\}^n$; i.e., the sign-representation is *strict*.

Let Π_i be the set of all paths from a leaf labelled by some $x_j \in S_i$ to the output of F. Let Γ_i be the set of all gates of F where two paths from Π_i meet. Then it holds that $|\Gamma_i| \leqslant t_i$.

Consider an assignment σ of the variables outside S_i. Let F_σ be the resulting formula, and suppose that

$$P = (G_1, G_2, ..., G_m), \tag{15}$$

for $m > 2$, is a path in F_σ so that G_1 is a leaf of F_σ or a gate in Γ_i, G_m is a gate in Γ_i, or the root of F_σ, and $G_j \notin \Gamma_i$ for $1 < j < m$. Then, since $\mathscr{B}$ contains only multilinear polynomials, P is equivalent to the path $(G_1, \bar{G}, G_m)$, where the gate $\bar{G}$ computes $\psi(x, a, b)$ for constants a and b which depend on the constants of F, the assignment σ, and the gates G_j, $1 < j < m$. On each path like (15), substitute gates $G_2, ..., G_{m-1}$ by the single ψ-gate $\bar{G}$ and feed in the corresponding constant a and b as inputs y and z. Denote the resulting formula by $\overline{F_\sigma}$. So $\overline{F_\sigma}$ is a formula with size $t_i' = \Theta(t_i)$. Note that for different assignments σ and σ' of the variables outside S_i, the formula $\overline{F_\sigma}$ and $\overline{F_{\sigma'}}$ only differ on their constant inputs.

Let G be any formula of size t over a basis of multilinear polynomials, having inputs $Y = \{y_1, ..., y_k\}$. Consider the

family $\mathscr{G}$ of formulas by assigning constants to the constant inputs of G in all possible ways. The following lemma is a slight modification of a result of Gashkov [14].

LEMMA 6.2. *The number of Boolean functions strictly sign-represented by formulas in $\mathscr{G}$ is at most*

$$(2^{k+4}t)^t.$$

Proof. Let $\mathbf{a}=(a_1, ..., a_m)$ be the constants of G and let the polynomial computed by G be $Q(\mathbf{y}, \mathbf{a})$ (note that $m \leqslant t$). Consider the set of polynomials

$$\mathscr{P}=\{Q(\mathbf{y}, \mathbf{a}): \mathbf{y} \in \{0, 1\}^k\},$$

in the variables $a_1, ..., a_m$, as in the proof of Theorem 3.1. The difference between circuits and formulas is that in the case of formulas one can give a better bound for the degree of the polynomials involved. It follows by induction on the depth of G, using the multilinearity of the gates, that the degree of $Q(\mathbf{y}, \mathbf{a})$ is at most t. The number of Boolean functions strictly sign-represented by formulas in $\mathscr{G}$ is exactly the number of consistent $(+, -)$ sign-assignments to the polynomials in $\mathscr{P}$. Thus the lemma follows from Theorem 2.1. ∎

The lemma implies that the number of subfunctions on S_i that are strictly sign-represented by $\overline{F_\sigma}$, for some σ, is at most $(2^{\ell_i+4}t_i')^{t_i'}$. This number has to be at least s_i, completing the proof of Theorem 6.1. ∎

This lower bound can be applied for example to the ELEMENT DISTINCTNESS function (Beame and Cook; see Boppana and Sipser [3]). Assume that n is of the form $2\ell \log \ell$. Then every input $\mathbf{x}$ may be viewed as representing ℓ numbers of $2 \log \ell$ bits each. Let $\mathrm{ED}_n(\mathbf{x})=1$ iff these numbers are pairwise different. We note that the formula size of ED_n is $O(n^2/\log n)$ over the Boolean basis $\{\wedge, \vee, \neg\}$, and ED_n is an AC^0 function.

THEOREM 6.3. *Let $\mathscr{B}$ be any basis of finitely many multilinear polynomials and all constants from $\mathbb{R}$. Then any formula sign-representing ED_n over $\mathscr{B}$ has size $\Omega(n^2/\log^2 n)$.*

Proof. Consider the partition of the variables into ℓ sets of $2 \log \ell$ variables corresponding to the numbers represented. Then the number of subfunctions on each set is at least $\binom{\ell^2}{\ell-1}$. Hence for r_i in (14) it holds that

$$(2^{2 \log \ell+4}r_i)^{r_i} \geqslant \binom{\ell^2}{\ell-1} > \ell^{\ell-1},$$

so $r_i=\Omega(\ell)$. Thus Theorem 6.1 implies that any formula sign-representing ED_n has size $\Omega(\ell^2)=\Omega(n^2/\log^2 n)$. ∎

COROLLARY 6.4. *Every arithmetic expression sign-representing ED_n has size $\Omega(n^2/\log^2 n)$.*

Now we turn to the extension of the lower bound to bases that include the sign function as well. We may again assume w.l.o.g. that the basis contains addition, multiplication and the function ψ described in the proof of Theorem 6.1. In this case *sign-representation and exact computation are equivalent* up to three additional gates. Let f, S_i, ℓ_i, and s_i $(i=1, ..., p)$ be as in the beginning of the section.

THEOREM 6.5. *Let $\mathscr{B}$ be any basis of finitely many multilinear polynomials, the sign function, and all constants from $\mathbb{R}$. Then every formula computing f over $\mathscr{B}$ has size*

$$\Omega\left(\sum_{1 \leqslant i \leqslant p} \sqrt{\log s_i}\right).$$

Proof. Let F be a formula over $\mathscr{B}$ computing f, and let t_i be the number of leaves labelled by variables from S_i. Let Π_i, Γ_i, σ and $P=(G_1, G_2, ..., G_m)$ be as in the proof of Theorem 6.1. Now P is a chain of gates that compute either a multilinear polynomial or the sign function.

LEMMA 6.6. *P can be replaced by a path $(G_1, \overline{G_1}, ..., \overline{G_5}, G_m)$, where the $\overline{G_i}$, $i=1, ..., 5$, are either ψ or sign gates, and the y and z inputs of the ψ-gates are constants.*

Proof. A subchain of P that consists of gates computing multilinear polynomials can be replaced by a single ψ-gate $\psi(x, y, z)$, where y and z are constant inputs, as in the proof of Theorem 6.1. Thus we can assume that $G_2, ..., G_{m-1}$ is an alternating sequence of ψ and sign gates. It may also be assumed w.l.o.g. that G_2 is a ψ-gate (otherwise one can add a dummy ψ-gate) and that there is at least one sign gate (otherwise we are done). Then the function computed by the chain P can be written as $H_{y, z, \mathbf{u}}(x) := h(\mathrm{sign}(xy+z), \mathbf{u})$, where $\mathbf{u}$ contains the constant inputs of the gates $G_4, ..., G_{m-1}$ and h is some function.

For an arbitrary setting of the constant inputs y, z, and $\mathbf{u}$, there are the following possibilities for H:

(i) for some $a, b \in \mathbb{R}$, $b \neq 0$,

$$\begin{aligned} H(x)&=a && \text{if} \quad xy+z<0\\ H(x)&=a+b && \text{if} \quad xy+z=0\\ H(x)&=a+2b && \text{if} \quad xy+z>0; \end{aligned}$$

(ii) for some $a, b \in \mathbb{R}$, $a \neq b$,

$$\begin{aligned} H(x)&=a && \text{if} \quad xy+z \leqslant 0\\ H(x)&=b && \text{if} \quad xy+z>0; \end{aligned}$$

(iii) for some $a, b \in \mathbb{R}$, $a \neq b$,

$$H(x) = a \quad \text{if} \quad xy + z < 0$$
$$H(x) = b \quad \text{if} \quad xy + z \geqslant 0;$$

(iv) for $a \in \mathbb{R}$,

$$H(x) \equiv a.$$

Each possibility can be realized by a chain $\overline{G_1}, ..., \overline{G_5}$, where $\overline{G_1}$, $\overline{G_3}$, and $\overline{G_5}$ are ψ-gates and $\overline{G_2}$ and $\overline{G_4}$ are sign gates. ▮

Let $\overline{F_\sigma}$ denote the formula obtained by performing the above replacements for all paths. Again, it holds that for different assignments σ and σ' of the variables outside S_i, the formulas $\overline{F_\sigma}$ and $\overline{F_{\sigma'}}$ only differ on their constant inputs.

Now we formulate an analogue of Lemma 6.2. Let G be a formula of size t over a basis of multilinear polynomials and the sign function, having inputs $Y = \{y_1, ..., y_k\}$. Let $\mathscr{G}$ be the family of formulas obtained by assigning constants to the constant inputs of G in all possible ways.

LEMMA 6.7. *The number of Boolean functions computed by formulas in $\mathscr{G}$ is at most 2^{4t^2+5t}.*

Proof. We argue as in the proof of Theorem 3.3. Let $\mathbf{a} = (a_1, ..., a_m)$ be the constants of G, and $v_1, ..., v_r$ be the sign gates and v_r be the output gate of G, where each gate v_i computes a value α_i. We consider the polynomials

$$P_i(\mathbf{y}, \mathbf{a}, \boldsymbol{\alpha}).$$

Then it holds that the number of different Boolean functions computed by G with different assignments to the constants is at most the number of different consistent $(+, 0, -)$ sign-assignments to the polynomials $P_i(\mathbf{y}, \mathbf{a}, \boldsymbol{\alpha})$ for $\mathbf{y} \in \{0, 1\}^k$, $\boldsymbol{\alpha} \in \{-1, 0, 1\}^r$. Again, each polynomial has degree at most t. Thus Corollary 2.2 implies the upper bound

$$\left(\frac{8et\,2^k 3^t}{t}\right)^t < 2^{4t^2+5t}. \quad ▮$$

Lemma 6.7 implies $2^{4t_i^2+5t_i} \geqslant s_i$, completing the proof of Theorem 6.5. ▮

Applying Theorem 6.5 for the function ED_n we get the following lower bound.

THEOREM 6.8. *Let $\mathscr{B}$ be any basis of finitely many multilinear polynomials, the sign function, and all constants from $\mathbb{R}$. Then any formula computing ED_n over $\mathscr{B}$ has size $\Omega(n^{3/2}/\log n)$.*

Proof. Arguing as in Theorem 6.3, we have a lower bound of the form $\Omega(\ell \sqrt{\ell \log \ell}) = \Omega(n^{3/2}/\log n)$. ▮

As a special case, we obtain a lower bound for arithmetic threshold expressions.

COROLLARY 6.9. *Every arithmetic threshold expression for ED_n has size $\Omega(n^{3/2}/\log n)$.*

6.2. *An Application to Boolean Linear Threshold Circuits*

Finally we formulate an application of the lower bound of Corollary 6.9 for Boolean complexity, considering Boolean threshold circuits of unbounded fan-in with arbitrary weights. For definitions see, e.g., Hajnal *et al* [17].

THEOREM 6.10. *Every depth d threshold circuit computing ED_n has size $\Omega(n^{1/(2(d-1))}/(\log n)^{1/(d-1)})$.*

Proof. Consider a threshold circuit of depth d and size s computing ED_n, with s_i gates on level i, $i = 1, ..., d-1$. The last level contains a single gate. By repeating gates several times if necessary, the circuit can be transformed into a threshold formula with at most $(s_1 + 1) \cdots (s_{d-1} + 1) \cdot 2n$ leaves. Each threshold gate with fan-in u can be replaced by an arithmetic threshold expression of size $O(u)$ containing one occurrence of each input of the gate. Hence Corollary 6.9 implies

$$(s_1 + 1) \cdots (s_{d-1} + 1) \cdot 2n = \Omega\left(\frac{n^{3/2}}{\log n}\right),$$

and this with $(s_1 + 1) \cdots (s_{d-1} + 1) \leqslant (s+1)^{d-1}$ gives the desired bound. ▮

7. SOME FURTHER REMARKS AND OPEN PROBLEMS

It appears that "powerful" functions such as $\sin(x)$ lead to a linear Shannon function when added to the basis of arithmetic circuits (Gashkov [14], Sontag [38]), and at the same time their inclusion in the theory of real numbers leads to undecidability. On the other hand, if $\mathrm{sign}(x)$ and $|x|$ are added to the basis, the Shannon function remains exponential. As these functions are definable over the reals, adding them as new functions does not change the theory and thus it remains decidable. We also note that applications of important recent results in logic, relating decidability and the Vapnik–Chervonenkis dimension, are given in Macintyre and Sontag [25] and in Karpinski and Macintyre [19].

Theorem 4.1 implies that every n variable Boolean function can be computed by an arithmetic threshold circuit of size $O(2^{n/2})$. The circuits constructed in the proof have depth $\Omega(2^{n/2})$. On the other hand, we showed in [40] that if the depth of the arithmetic threshold circuits is restricted

to be *polynomial* in n, then almost all functions require size $\Omega(2^{n-O(\log n)})$. Thus for most Boolean functions there is a *size-depth trade-off* in this model. This is in contrast with Boolean circuits, where almost all functions can be computed by circuits that have asymptotically optimal size *and* depth.

We mention some recent results on the complexity of analog circuits. Wegener [45] considered the complexity of *encoding n* bits by a real number, i.e., the complexity of computing *any* injection from $\{0, 1\}^n$ to $\mathbb{R}$. He showed that the natural method of using the binary representation is *not* optimal and he determined the optimal bound for *formula size*. Pudlák [30] extended the exponential lower bound of Razborov from monotone Boolean circuits to circuits of *arbitrary real-valued gates* computing *monotone* functions. He used this result to prove an exponential lower bound for cutting plane proofs.

There are many open problems related to the results of this paper. The question whether arithmetic circuits or arithmetic threshold circuits are superpolynomially more powerful than Boolean circuits is open. This is also mentioned as an open problem in Koiran [21]. The corresponding problem for real Turing machines is noted as an open problem in Cucker and Grigoriev [6]. It is not known if all n-variable Boolean functions can be computed *exactly* by arithmetic circuits of size $O(2^{n/2})$. It would be also interesting to extend the lower bounds of Section 6 to formulas that may also use x^2 or $|x|$, and to prove explicit lower bounds for *planar* arithmetic circuits.

REFERENCES

1. M. Ben-Or, Lower bounds for algebraic computation trees, *in* "Proceedings, 15th ACM Symposium on Theory of Computing, 1983," pp. 80–86.
2. L. Blum, M. Shub, and S. Smale, On a theory of computation and complexity over the real numbers: NP-completeness, recursive functions and universal machines, *Bull. Amer. Math. Soc.* **21** (1989), 1–46.
3. R. B. Boppana and M. Sipser, The complexity of finite functions, *in* "Handbook of Theoretical Computer Science," Vol. A, "Algorithms and Complexity" (J. van Leeuwen, Ed.), pp. 757–804, Elsevier Science, New York, MIT Press, Cambridge, MA, 1990.
4. F. Cucker, $P_{\mathbb{R}} \neq NC_{\mathbb{R}}$, *J. Complexity* **8** (1992), 230–238.
5. F. Cucker, On the complexity of quantifier elimination: The structural approach, *Comput. J.* **36** (1993), 400–408.
6. F. Cucker and D. Grigoriev, On the power of real Turing machines over binary inputs, manuscript, 1994.
7. F. Cucker, M. Karpinski, P. Koiran, T. Lickteig, and K. Werther, On real Turing machines that toss coins, *in* "Proceedings, 27th ACM Symposium on Theory of Computing, 1995," pp. 335–342.
8. F. Cucker, M. Shub, and S. Smale, Separation of complexity classes in Koiran's weak model, *Theor. Comput. Sci.* **133** (1994), 3–14.
9. F. Cucker and A. Torrecillas, Two P-complete problems in the theory of reals, *J. Complexity* **8** (1992), 454–466.
10. G. Cybenko, Approximation by superposition of a sigmoidal function, *Math. Control Signals, Systems* **2** (1989), 303–314.
11. B. DasGupta and G. Schnitger, "Efficient Approximation with Neural Networks: A Comparison of Gate Functions," Technical Report TR CS-92-14, Penn. State Univ., Dept. C. S., June 1992.
12. R. L. Devaney, "An Introduction to Chaotic Dynamical Systems," Benjamin/Cummings, Redwood City, CA, 1986.
13. V. Dičiūnas, On the positive and the inversion complexity of Boolean functions, *Inform. Theor. Appl./Theor. Inform. Appl.* **27** (1993), 283–293.
14. S. B. Gashkov, The complexity of the realization of Boolean functions by networks of functional elements and by formulas in bases whose elements realize continuous functions, *Prob. Kibernet.* **37** (1980), 52–118. [Russian]
15. P. Goldberg and M. Jerrum, Bounding the Vapnik–Chervonenkis dimension of concept classes parameterized by real numbers, *Mach. Learning* **18** (1995), 131–148.
16. H. D. Gröger and Gy. Turán, A linear lower bound for the size of threshold circuits, *Bull. EATCS* **50** (1993), 220–222.
17. A. Hajnal, W. Maass, P. Pudlák, M. Szegedy, and Gy. Turán, Threshold circuits of bounded depth, *J. Comput. System Sci.* **46** (1993), 129–154.
18. R. Impagliazzo, R. Paturi, and M. E. Saks, "Size-Depth Trade-offs for Threshold Circuits," Technical Report TR CS-92-253, Univ. of California at San Diego, Comput. Sci. and Eng., July 1992.
19. M. Karpinski and A. Macintyre, Quadratic bounds for the VC dimension of sigmoidal neural networks, *in* "Proceedings, 27th ACM Symposium on Theory of Computing, 1995," pp. 200–208.
20. P. Koiran, Computing over the reals with addition and order, *Theor. Comput. Sci.* **133** (1994), 35–47.
21. P. Koiran, "A Weak Version of the Blum, Shub, & Smale Model," Technical Report NC-TR-94-5, NeuroCOLT Technical Reports Series, August 1994. A preliminary version appeared in "Proceedings, 34th IEEE Symposium on the Foundations of Computer Science, 1993," pp. 486–495.
22. W. Maass, Bounds for the computational power and learning complexity of analog neural nets, *in* "Proceedings, 25th ACM Symposium on Theory of Computing, 1993," pp. 335–344.
23. W. Maass, Neural nets with superlinear VC dimension, *Neural Comput.* **6** (1994), 877–884.
24. W. Maass, G. Schnitger, and E. D. Sontag, A comparison of the computational power of sigmoid and Boolean threshold circuits, *in* "Theoretical Advances in Neural Computation and Learning" (V. Roychowdhury, K.-Y. Siu, and A. Orlitsky, Eds.), pp. 127–151, Kluwer, Amsterdam, 1994; a preliminary version appeared in "Proceedings, 32nd IEEE Symposium on the Foundations of Computer Science, 1991," pp. 767–776.
25. A. Macintyre and E. D. Sontag, Finiteness results for sigmoidal "neural" networks, *in* "Proceedings, 25th ACM Symposium on Theory of Computing, 1993," pp. 325–334.
26. J. L. McClelland and D. E. Rumelhart, "Parallel Distributed Processing," Vol. 2, MIT Press, Cambridge, MA, 1986.
27. J. Milnor, On the Betti numbers of real varieties, *Proc. Amer. Math. Soc.* **15** (1964), 275–280.
28. J. L. Montaña and L. M. Pardo, Lower bounds for arithmetic networks, *Appl. Algebra Eng. Commun. Comput.* **4** (1993), 1–24.
29. N. Nisan, The communication complexity of threshold gates, *in* "Combinatorics, Paul Erdös is Eighty, Vol. 1," pp. 301–315, Bolyai Soc. Math. Studies, 1993.
30. P. Pudlák, Lower bounds for resolution and cutting plane proofs and monotone computations, preliminary draft, 1995.
31. J. Renegar, On the computational complexity and geometry of the first-order theory of the reals, Part I, *J. Symbolic Comput.* **13** (1992), 255–299.
32. V. P. Roychowdhury, K. Y. Siu, and A. Orlitsky, Lower bounds on threshold and related circuits via communication complexity, *IEEE Trans. Inf. Theory* **40** (1994), 467–474.

33. D. E. Rumelhart and J. L. McClelland, "Parallel Distributed Processing," Vol. 1, MIT Press, Cambridge, MA, 1986.
34. M. Saks, Slicing the hypercube, *in* "Surveys in Combinatorics," Cambridge Univ. Press, Cambridge, 1993.
35. H. T. Siegelmann and E. D. Sontag, Analog computation via neural networks, *Theor. Comput. Sci.* **131** (1994), 331–360.
36. H. T. Siegelmann and E. D. Sontag, On the computational power of neural nets, *J. Comput. System Sci.* **50** (1995), 132–150.
37. K. Y. Siu, V. P. Roychowdhury, and T. Kailath, Computing with almost optimal size threshold circuits, *in* "Proceedings, IEEE Int. Symp. Inf. Theory, 1991," pp. 370.
38. E. D. Sontag, Feedforward nets for interpolation and classification, *J. Comput. System Sci.* **45** (1992), 20–48.
39. V. Strassen, Algebraic complexity theory, *in* "Handbook of Theoretical Computer Science," Vol. A, "Algorithms and Complexity" (J. van Leeuwen, Ed.), pp. 633–672, Elsevier Science, Amsterdam/MIT Press, Cambridge, MA, 1990.
40. Gy. Turán and F. Vatan, A size-depth trade-off for the analog computation of Boolean functions, *Inform. Proc. Lett.* **59** (1996), 251–254.
41. F. Vatan, Some lower and upper bounds for algebraic decision trees and the separation problem, *in* "Proceedings, 7th Ann. IEEE Structure in Complexity Theory, 1992," pp. 295–304.
42. A. G. Vitushkin, On representation of functions by means of superpositions and related topics, *Enseign. Math.* **23** (1977), 255–320.
43. H. E. Warren, Lower bounds for approximation by non-linear manifolds, *Trans. Amer. Math. Soc.* **133** (1968), 167–178.
44. L. Wegener, "The Complexity of Boolean Functions," Teubner–Wiley, New York, 1987.
45. L. Wegener, On the complexity of encoding in analog circuits, *Inform. Proc. Lett.* **60** (1996), 49–52.

Printed in Belgium