Constant Depth Circuits, Fourier Transform, and Learnability

NATHAN LINIAL

Hebrew University, Jerusalem, Israel

YISHAY MANSOUR

Tel-Aviv University, Tel-Aviv, Israel

AND

NOAM NISAN

Hebrew University, Jerusalem, Israel

Abstract. In this paper, Boolean functions in AC^0 are studied using harmonic analysis on the cube. The main result is that an AC^0 Boolean function has almost all of its "power spectrum" on the low-order coefficients. An important ingredient of the proof is Hastad's switching lemma [8].

This result implies several new properties of functions in AC^0: Functions in AC^0 have low "average sensitivity;" they may be approximated well by a real polynomial of low degree and they cannot be pseudorandom function generators.

Perhaps the most interesting application is an $O(n^{\text{polylog}(n)})$-time algorithm for learning functions in AC^0. The algorithm observes the behavior of an AC^0 function on $O(n^{\text{polylog}(n)})$ randomly chosen inputs, and derives a good approximation for the Fourier transform of the function. This approximation allows the algorithm to predict, with high probability, the value of the function on other randomly chosen inputs.

A preliminary version of this paper was published in *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*. IEEE, New York, 1989, pp. 574–579.

The research of Y. Mansour was done while he was at the Laboratory for Computer Science, Massachusetts Institute of Technology, and was partially supported by National Science Foundation (NSF) grant CCR 86-5727, Army Research Office (ARO) program DALL 03-86-K-017, and ISEF fellowship.

The research of N. Linial was done while he was visiting IBM Almaden Research Center and Stanford University.

The research of N. Nisan was done while he was working at the Laboratory of Computer Science, Massachusetts Institute of Technology and was partially supported by NSF grant CCR 86-5727 and ARO program DALL 03-86-017.

Authors' addresses: N. Linial and N. Nisan, Department of Computer Science, Hebrew University, Jerusalem, Israel; Y. Mansour, Computer Science Department, Tel-Aviv University, Tel-Aviv Israel.

Categories and Subject Descriptors: F.1.2 [**Computation by Abstract Devices**]: Modes of Computation—*probabilistic computation*; F.2.1 [**Analysis of Algorithms and Problem Complexity**]: Numerical Algorithms and Problems—*computations of transforms, computations on polynomials*; G.3 [**Mathematics of Computing**]: Probability and Statistics—*probabilistic algorithms*; I.2.6 [**Artificial Intelligence**]: Learning—*concept learning*

General Terms: Algorithms, Theory, Verification

Additional Key Words and Phrases: AC^0 circuits, approximation, Boolean functions, circuits, complexity, Fourier transform, harmonic analysis learning.

1. *Introduction*

Harmonic analysis is widely used throughout classical mathematics (see [4]). Recently, Kahn et al. [9] suggested using harmonic analysis on the hypercube in the study of Boolean functions. They proved some inequalities which Fourier coefficients of Boolean functions must satisfy, and derived as a result bounds on the "influence" of variables on Boolean functions. Harmonic analysis was used in [3] to obtain lower bounds for the size of decision trees, DNF, and CNF.

In this paper, we study how the *computational complexity* of Boolean functions is related with their Fourier transform. Specifically, we study the Fourier transform of functions computable by constant depth circuits and derive an inequality that the transform of such functions satisfies. This inequality is then used to establish new results on complexity and learnability of constant depth circuits.

The best-known lower bound for constant depth circuits is that they require a very large size to compute the parity function [1, 5, 8, 20]. In fact, small constant depth circuits cannot even decently *approximate* the parity function (see [8]). This fact directly bears on the Fourier transform, because the Sth Fourier coefficient of f measures, by definition, the correlation between f the parity of the input bits in S. Consequently, each "high" Fourier coefficient of a function computable by a small constant depth circuit must be very small ("high" means coefficients corresponding to sets of large cardinality).

Our Main Lemma is an extension of this fact: Not only is each individual "high" Fourier coefficient small, but in fact the sum of squares (the "*power spectrum*") associated with all high Fourier coefficients is very small. Specifically:

MAIN LEMMA. *Let f be a Boolean function on n variables computable by a Boolean circuit of depth d and size M, and let t be any integer. Then*

$$\sum_{S \subset \{1 \cdots n\},\, |S| > t} \hat{f}(S)^2 \leq 2M2^{-t^{1/d}/20},$$

where $\hat{f}(S)$ denotes the Fourier Transform of f at S.

The first application of this lemma is an algorithm for learning AC^0 Boolean functions. To learn functions computed by a polynomial-size, constant-depth circuit, the algorithm proceeds as follows: It first observes the behavior of the circuit on $O(n^{\text{polylog}(n)})$ inputs chosen uniformly at random; this allows it to derive (with high probability) a very good approximation to all the "low" Fourier coefficients of the function computed by the circuit. Since the "high" coefficients are guaranteed by the Main Lemma to have very little "power," these approximations of the "low" Fourier coefficients are informa-

tive enough to predict the behavior of the circuit on inputs chosen uniformly at random. Since there are only a few "low" coefficients, the approximation can be done "efficiently."

There are three key ideas on which this learning algorithm relies. The first is that lower bounds (i.e., negative results) may be used to construct learning algorithms (i.e., positive results). A similar phenomenon appears in the study of pseudorandom generators, where lower bounds are used in order to deterministically simulate randomized algorithms [13, 19]. The second one says that learning can be achieved through estimating the Fourier coefficients, an observation that may be useful elsewhere as well. The third is the application of real arithmetic and real valued functions to approximate Boolean functions.

Our algorithm does not fall into the category of *distribution-free learning*, introduced by [18]. First and foremost, it runs in time $O(n^{\text{polylog}(n)})$ and not in polynomial time. Secondly, it learns circuits only under the uniform probability distribution on inputs, and not under an arbitrary distribution. On the positive side, the concept class being learned is substantially richer than previously achieved. Earlier positive results in learning involve classes whose combinatorial complexity is much more restricted, for example, k-DNF [18], k-decision lists [15], etc. Results involving richer classes, as NC^1, have been negative (see [10]).

Based on our Main Lemma, we derive a number of additional interesting properties of functions in AC^0.

(1) Every function in AC^0 can be approximated well by a real polynomial of low degree. This complements the results of [14] and [17], showing that such an approximation is possible over finite fields.
(2) Every function in AC^0 has low-average sensitivity to its input. Changing one bit of the input is very unlikely to change the value of the function, when the original input and the bit are chosen at random.
(3) Functions in AC^0 cannot be pseudorandom function generators in the sense of [6].
(4) Functions in AC^0 cannot distinguish between uniform distributions and polynomially bounded polylog-wise independent probability distributions. (A polynomially bounded distribution over Z_2^n is a distribution in which any input has probability less than $\text{poly}(n)/2^n$.)

The paper is organized as follows: Section 2 is devoted to notations and definitions. It includes all the necessary background on Fourier transform on the hypercube. The Main Lemma is proved in section 3. Section 4 is devoted to the learning algorithm and Section 5 contains further applications of the Main Lemma.

2. *Notation*

2.1. FOURIER TRANSFORM. Boolean functions on n variables will be considered as real valued functions $f: \{0,1\}^n \to \{-1,1\}$. The set of all real functions on the cube is a 2^n-dimensional real vector space with an inner product defined by

$$\langle g, f \rangle = 2^{-n} \sum_{x \in \{0,1\}^n} f(x)g(x) = E(gf)$$

(where E is expectation) and as usual the *norm* of a function is defined: $\|f\| = \sqrt{\langle f, f \rangle}$, which is the Euclidean norm.

Many of the elementary facts in harmonic analysis may be interpreted in the following way: Consider the linear space of real functions defined on a group, a clever choice of a basis for this linear space may be very helpful. This special basis is given by the *characters* of the group at hand. In the present case, the group is the cube $\mathbf{Z}_2^n$ and the basis is defined as follows: For each subset S of $\{1, \ldots, n\}$, define the function χ_S:

$$\chi_S(x_1, \ldots, x_n) = \begin{cases} +1 & \text{if } \sum_{i \in S} x_i \text{ is even,} \\ -1 & \text{if } \sum_{i \in S} x_i \text{ is odd.} \end{cases}$$

The following properties of these functions can all be easily verified:

—For every A, B: $\chi_A \chi_B = \chi_{A \Delta B}$, where $A \Delta B$ is the symmetric difference of A and B.

—The family $\{\chi_S\}$ for all $S \subset \{1 \cdots n\}$ forms an orthonormal basis, that is, if $A \neq B$, then $\langle \chi_A, \chi_B \rangle = 0$, and for every A, $\langle \chi_A, \chi_A \rangle = 1$.

Any real-valued function on the cube can be uniquely expressed as a linear combination of the χ_S's, namely, $\sum_S c_S \chi_S$, where c_S are real constants. These coefficients (c being viewed as a real function on the cube) constitute the function's Fourier transform. For a function f and $S \subset \{1, \ldots, n\}$, the Sth Fourier coefficient of S denoted by $\hat{f}(S)$ is what was previously called c_S, that is, $f = \sum_S \hat{f}(S) \chi_S$.

Since the χ_S's are an orthonormal basis, Fourier coefficients are found via:

$$\hat{f}(S) = \langle f, \chi_S \rangle.$$

For Boolean f, this specializes to:

$$\hat{f}(S) = \Pr\left[f(x) = \bigoplus_{i \in S} x_i\right] - \Pr\left[f(x) \neq \bigoplus_{i \in S} x_i\right],$$

where $x = (x_1, x_2, \ldots, x_n)$ is chosen uniformly at random in $\{0, 1\}^n$.

The orthonormality of the basis implies Parseval's identity:

$$\|f\|^2 = \sum_{S \subset \{1 \cdots n\}} \hat{f}(S)^2.$$

Note that if f is Boolean then $\|f\| = 1$.

Finally we define the *degree* of a Boolean function, $deg(f)$ to be the size of the largest set S such that $\hat{f}(S) \neq 0$. Note that this equals the degree of f as a real (multi-linear) polynomial.

2.2. AC^0 CIRCUITS. An AC^0 circuit consists of AND and OR gates, with inputs $x_1, \ldots, x_n$ and $\bar{x}_1, \ldots, \bar{x}_n$. Fanin to the gates is unbounded. The size of the circuit (i.e., the number of the gates) is bounded by a polynomial in n, and its depth is bounded by a constant. Without loss of generality, the circuit is leveled, where gates at level i have all their inputs from level $i - 1$, all gates at the same level have the same type, which is alternately AND and OR. (For a more detailed description, see [5], [8], and [20].) The set of functions computable by an AC^0 circuits of depth d is denoted by $AC^0[d]$.

2.3. RANDOM RESTRICTION. A restriction ρ is a mapping of the input variables to 0, 1, and $*$. The function obtained from $f(x_1, \ldots, x_n)$ by applying a restriction ρ, is denoted by f_ρ, its variables are those x_i for which $\rho(x_i) = *$, all other variables are set according to ρ.

For a set $S = \{x_{i_1}, \ldots, x_{i_{|S|}}\}$ and a vector $R = (r_1, \ldots, r_{|S|}) \in \{0,1\}^{|S|}$, let $S \leftarrow R$ denote the restriction ρ, such that $\rho(x_{i_j}) = r_j$, for $x_{i_j} \in S$, and $\rho(x) = *$, for $x \notin S$.

A random restriction with a parameter p is obtained by setting each x_i, independently. We choose a value from $\{*, 0, 1\}$, such that $\Pr[\rho(x_i) = *] = p$, and $\Pr[\rho(x_i) = 1] = \Pr[\rho(x_i) = 0] = (1 - p)/2$. In many cases, we abbreviate the notation and write $\Pr[*]$ rather than $\Pr[\rho(x_i) = *]$.

2.4. MISCELLANEOUS. The complement of a set $S \subset \{1 \cdots n\}$ is denoted by S^c. For a real number r, the value of $sign(r)$ is 1 if r is positive, -1 if it is negative and $sign(0) = 0$.

A *minterm* of a Boolean function is a minimal set of variables with the property that setting all of them to one forces the function to be one. (In the restriction language, it is a minimal set of variables S such that $f_{S \leftarrow 1}(x) \equiv 1$.) A *maxterm* is as minimal set of variables S that forces the function to be zero (i.e., $f_{S \leftarrow 0} \equiv 0$).

3. *Main Lemma*

Hastad's Switching Lemma [8] states that AC^0 functions tend to simplify in a significant way when subjected to random restrictions. This beautiful lemma is the main tool of the present article. We use a stronger statement than the one originally made by Hastad. It was observed by Hastad and Boppana (see [8, p. 65]) that the original proof yields this stronger version as well.

LEMMA 1 (HASTAD). *Let f be given by a CNF formula where each clause has size at most t, and choose a random restriction ρ with parameter p (i.e., $Pr[\rho(x_i) = *] = p$). With probability of at least $1 - (5pt)^s$, f_ρ can be expressed as a DNF formula each clause of which has size of at most s, and the clauses all accept disjoint sets of inputs.*

We require the following simple corollary.

COROLLARY 1. *If f is given by a CNF (or DNF) of bottom fanin at most t, and ρ is chosen at random with $Pr[*] = p$, then*

$$Pr\left[deg(f_\rho) > s\right] < (5pt)^s.$$

PROOF. Whenever f_ρ satisfies conditions (1) and (2) of Hastad's lemma the following also holds: For every set S, $|S| > s$, each clause of the DNF formula for f_ρ accepts exactly the same number of strings having even or odd parity on S. This happens because at least one of the variables in S does not appear in the clause (since the clause size is bounded by s). The corollary follows since the clauses all accept disjoint sets of inputs and thus f_ρ accepts an equal number of strings having even or odd parity on S. □

Repeated application of Hastad's lemma yields the following lemma.

LEMMA 2. *Let f be a Boolean function computed by a circuit of size M and depth d. Then*

$$Pr\left[deg(f_\rho) > s\right] \leq M2^{-s},$$

where ρ is a random restriction with

$$Pr[*] = \frac{1}{10^d s^{d-1}}.$$

PROOF. We view the restriction ρ as obtained by first having a random restriction with $\Pr[*] = 1/10$, and then $d-1$ consecutive restrictions each with $\Pr[*] = 1/(10s)$.

With high probability, after the first restriction, at the bottom level of the circuit all fanins are at most s. To see this, we consider two cases for each gate at the bottom level of the original circuit:

(1) The original fanin is at least $2s$. In this case, the probability that the gate was not eliminated by ρ, that is, that no input to this gate got assigned a 0 (assuming without loss of generality that the bottom level is an AND level) is at most $0.55^{2s} < 2^{-s}$;
(2) The original fanin is at most $2s$. In this case, the probability that at least s inputs got assigned a $*$ is at most $\binom{2s}{s}0.1^s < 2^{-s}$. Thus, the probability of failure at this stage is at most $m_1 2^{-s}$, where m_1 is the number of gates at the bottom level.

We now apply $d-2$ more restrictions with $\Pr[*] = 1/(10s)$. After each of these, we use Hastad's switching lemma to convert the lower two levels from CNF to DNF (or vice versa), and collapse the second and third levels (from the bottom) to one level, reducing the depth by one. For each gate of distance two from the inputs, the probability that it has a minterm (respectively, maxterm) of size larger than s, is bounded by 2^{-s}. The probability that *some* gate has a minterm (respectively, maxterm) larger than s is no more than $m_i 2^{-s}$, where m_i is the number of gates at level i.

After these $d-2$ stages we are left with a CNF (or DNF) formula of bottom fanin at most s. We now apply the last restriction with $\Pr[*] = 1/(10s)$ and by Corollary 1 get a function with degree at most s. The probability of failure at this stage is at most 2^{-s}.

To compute the total probability of failure, we observe that each gate of the original circuit contributed 2^{-s} probability of failure exactly once. □

At this point, we start analyzing how the Fourier transform of f relates to the probability of its restrictions having low degree. We start with a lemma relating the Fourier transform of a function f with the transforms of its restrictions.

LEMMA 3. *Let f be a Boolean function and S an arbitrary subset of the variables. Then, for any subset of the variables A:*

$$\hat{f}(A) = 2^{-|S^c|} \sum_{R \in \{0,1\}^{|S^c|}} \chi_{A \cap S^c}(R) \hat{f}_{S^c \leftarrow R}(A \cap S).$$

PROOF. Recall that $\hat{f}(A) = E_x[f\chi_A]$. In the right-hand side, we are simply first averaging over variables in S and then in variables in S^c. We can rewrite the right-hand side as:

$$2^{-|S^c|} \sum_{R_1 \in \{0,1\}^{|S^c|}} \chi_{A \cap S^c}(R_1) \left[2^{-|S|} \sum_{R_2 \in \{0,1\}^{|S|}} \chi_{A \cap S}(R_2) f_{S^c \leftarrow R_1}(R_2) \right].$$

First, we can rearrange the summation such that,

$$2^{-|S^c|} \sum_{R_1 \in \{0,1\}^{|S^c|}} 2^{-|S|} \sum_{R_2 \in \{0,1\}^{|S|}} \chi_{A \cap S^c}(R_1) \chi_{A \cap S}(R_2) f_{S^c \leftarrow R_1}(R_2).$$

Note that

$$\chi_{A \cap S^c}(R_1) \chi_{A \cap S}(R_2) = \chi_A(x),$$

where x, when restricted to the variables in S^c, is R_1 and when restricted to the variables in S is R_2. Similarly, $f_{S^c \leftarrow R_1}(R_2) = f(x)$. Furthermore, averaging over R_1 and R_2 is like averaging over $x \in \{0,1\}^n$. Finally, by definition $|S| + |S^c| = n$; therefore, we can rewrite the expression as,

$$2^{-n} \sum_{x \in \{0,1\}^n} \chi_A(x) f(x),$$

which is by definition $\hat{f}(A)$. □

LEMMA 4. *Let f be a Boolean function and S an arbitrary subset. For any $B \subset S$:*

$$\sum_{C \subset S^c} \hat{f}(B \cup C)^2 = 2^{-|S^c|} \sum_{R \in \{0,1\}^{|S^c|}} \hat{f}_{S^c \leftarrow R}(B)^2.$$

PROOF. By Lemma 3

$$\sum_{C \subset S^c} \hat{f}(B \cup C)^2 = \sum_{C \subset S^c} \left(2^{-|S^c|} \sum_{R \in \{0,1\}^{|S^c|}} \chi_{B \cup C}(R) \hat{f}_{S^c \leftarrow R}((B \cup C) \cap S) \right)^2.$$

Note that $\chi_{B \cup C}(R) = \chi_C(R)$ and $(B \cup C) \cap S = B$. Therefore, the above expression equals:

$$2^{-2|S^c|} \sum_{C \subset S^c} \left(\sum_{R \in \{0,1\}^{|S^c|}} \chi_C(R) \hat{f}_{S^c \leftarrow R}(B) \right)^2.$$

Now we can simply multiply and get

$$2^{-|S^c|} \sum_{R_1 \in \{0,1\}^{|S^c|}} \sum_{R_2 \in \{0,1\}^{|S^c|}} \hat{f}_{S^c \leftarrow R_1}(B) \hat{f}_{S^c \leftarrow R_2}(B) \left[2^{-|S^c|} \sum_{C \subset S^c} \chi_C(R_1 \oplus R_2) \right].$$

One can verify that the expression between the brackets is one if $R_1 = R_2$ and otherwise zero. Therefore, the expression can be simplified to

$$2^{-|S^c|} \sum_{R \in \{0,1\}^{|S^c|}} \hat{f}_{S^c \leftarrow R}(B)^2,$$

which completes the proof. □

LEMMA 5. *Let f be a Boolean function, S an arbitrary subset, and k an integer. Then*

$$\sum_{A,|A\cap S|>k} \hat{f}(A)^2 \le Pr[deg(f_{S^c\leftarrow R}) > k],$$

where R is a 0–1 assignment to the variables in S^c chosen at random.

PROOF. The main idea is the following: Consider an arbitrary assignment R. If $deg(f_{S^c\leftarrow R}) \le k$, then for any A, such that $|A \cap S| > k$, the corresponding Fourier coefficient is zero, that is, $\hat{f}_{S^c\leftarrow R}(A \cap S) = 0$. On the other hand, since $f_{S^c\leftarrow R}$ is a Boolean function, the value of $\sum_{|B|>k} \hat{f}_{S^c\leftarrow R}(B)^2$ is bounded by one. Therefore, it is sufficient to show that

$$\sum_{A,|A\cap S|>k} \hat{f}(A)^2 = E_R\left[\sum_{|B|>k} \hat{f}_{S^c\leftarrow R}(B)^2\right].$$

Rewrite the left-hand side as

$$\sum_{A,|A\cap S|>k} \hat{f}(A)^2 = \sum_{B\subset S,|B|>k} \sum_{D\subset S^c} \hat{f}(D\cup B)^2.$$

By Lemma 4, this equals

$$\sum_{B\subset S,|B|>k} 2^{-|S^c|} \sum_{R\in\{0,1\}^{|S^c|}} \hat{f}_{S^c\leftarrow R}(B)^2,$$

which can be rewritten as

$$2^{-|S^c|} \sum_{R\in\{0,1\}^{|S^c|}} \sum_{|B|>k} \hat{f}_{S^c\leftarrow R}(B)^2 = E_R\left[\sum_{|B|>k} \hat{f}_{S^c\leftarrow R}(B)^2\right],$$

which completes the proof. □

By averaging sums as those appearing in Lemma 5 over all subsets S, the sum of squares of high coefficients can be bounded.

LEMMA 6. *Let f be a Boolean function, t an integer, and $0 < p < 1$. Then*

$$\sum_{|A|>t} \hat{f}(A)^2 \le 2E_S\left(\sum_{|A\cap S|>pt/2} \hat{f}(A)^2\right),$$

where S is a subset chosen at random such that each variable appears in its independently with probability p, and $pt > 8$.

PROOF. Using Chernoff bounds, the probability that $|A \cap S| > pt/2$ is at least $1 - \exp(-tp/8)$ (see [7]). In our case, $tp > 8$; therefore, we can assume that the probability of $|A \cap S| > pt/2$ is at least $1/2$. Each set A contributes $\hat{f}(A)^2$ to at least half of the sets S, and the lemma follows. □

At this point, we have developed all the necessary machinery to prove the Main Lemma.

LEMMA 7 (MAIN LEMMA). *Let f be a Boolean function computed by a circuit of depth d and size M, and let t be any integer. Then*

$$\sum_{|A|>t} \hat{f}(A)^2 \leq 2M2^{-t^{1/d}/20}.$$

PROOF. Fix $p = 1/(10t^{(d-1)/d})$, and $s = pt/2 = t^{1/d}/20$. By Lemma 6,

$$\sum_{|A|>t} \hat{f}(A)^2 \leq 2E_S\left(\sum_{|A\cap S|>pt/2} \hat{f}(A)^2\right),$$

where S is chosen at random such that each variable appears in it with probability p. Using Lemma 5, this is bounded from above by

$$2E_S \Pr\left[\deg(f_{S^c \leftarrow R}) > \frac{pt}{2} = s\right].$$

Consider now the distribution of the restriction $S^c \leftarrow R$ induced by first choosing S at random such that each variable appears in it with probability p, and then choosing a random 0–1 assignment R to the bits in S^c. This is exactly the same distribution as choosing a restriction ρ at random with $\Pr[*] = p$. since by our choice of p and s, $p \leq 1/(10^d s^{d-1})$, Lemma 2 applies and the above quantity is bounded by

$$2M2^{-s} = 2M2^{-t^{1/d}/20} \qquad \square$$

4. *Learning Constant Depth Circuits*

Theoretical machine learning is mainly concerned with learning *concepts*, i.e., Boolean functions. The standard scenario is the following: A class of concepts is fixed and known to the learner who is trying to identify a specific member in the class that is unknown to him. To this end, the learner observes pairs of input/output of the concept. Based on these observations, the learner wishes to find some concept that is "close" to the unknown concept.

Various models of learning differ on a number of issues: First is the way for selecting input/output pairs for the learner to observe: They may be specified by the learning algorithm, randomly chosen from the uniform distribution, from some unknown distribution, or even by an adversary. The other issue is when are two concepts considered "close" to each other. This may be defined as their probability to agree on inputs drawn uniformly at random, from some unknown distribution, or from the distribution used to select inputs at the observation stage.

We consider a learning model with two phases: learning and prediction. In the learning phase the algorithm is presented randomly chosen inputs x, along with $f(x)$. During the prediction phase the algorithm is only presented random inputs x, and must output a prediction $\tilde{f}(x)$, for $f(x)$. For a given distribution D, an algorithm is called an (ϵ, δ, D) prediction algorithm if

$$\Pr_D\left[\tilde{f} \text{ disagrees with } f \text{ on more than an } \epsilon \text{ fraction of the inputs}\right] \leq \delta.$$

We present an (ϵ, δ, U) algorithm for learning circuits of depth d and size M, where U is the uniform distribution. For every fixed d, the algorithm runs in time quasi-polynomial in ϵ, δ, and M. In the learning phase the algorithm derives good approximations to the Fourier coefficients of f, and in the prediction stage these approximate coefficients are used to predict the value of f.

4.1. LEARNING PHASE. The algorithm observes f on m randomly chosen, sample points $x_1, \ldots, x_m$, where $m = 4(2n^k/\epsilon)\ln(2n^k/\delta)$ and $k = (20\log(2m/\epsilon))^d$. Its approximation for the Sth coefficient of f is

$$\alpha_S = \frac{\sum_{i=1}^{m} f(x_i)\chi_S(x_i)}{m}.$$

For all $|S| \leq k$, and $\alpha_S = 0$, for all $|S| > k$.

4.2. PREDICTION PHASE. The predicted value of f on input x is

$$\tilde{f}(x) = sign\left(\sum_{|S|\leq k} \alpha_S \chi_S(x)\right).$$

THEOREM 1. *The above algorithm is an (ϵ, δ, U) learning algorithm for circuits of depth d and size M, where U is the uniform distribution.*

The proof of Theorem 1 uses the following two lemmas: Lemma 8 shows with high probability all low-order coefficients are approximated well.

LEMMA 8

$$Pr\left[\text{For some } S, |S| \leq k, \left|\alpha_S - \hat{f}(S)\right| > \sqrt{\frac{\epsilon}{2n^k}}\right] \leq \delta$$

PROOF. For a subset S, consider the random variable $Y_S = f(x)\chi_S(x)$. The expected value of Y_S is, by definition, $\hat{f}(S)$. The algorithm estimates this expected value by averaging over m samples. The lemma follows through a standard application of Chernoff bounds (see [7]). □

LEMMA 9. *Let f be a Boolean function, and g an arbitrary function such that $\Sigma_S(\hat{f}(S) - \hat{g}(S))^2 \leq \epsilon$, then $Pr[f(x) \neq sign(g(x))] \leq \epsilon$.*

PROOF. Since f is a Boolean function, $f(x) \neq sign(g(x))$ implies that $|f(x) - g(x)| \geq 1$. Note that $\|f - g\|^2 = E_x(f(x) - g(x))^2$; thus, the probability that $|f(x) - g(x)| \geq 1$ does not exceed $\|f - g\|^2$. Finally, by Parseval's equality, $\|f - g\|^2 = \Sigma_S(\hat{f}(S) - \hat{g}(S))^2 \leq \epsilon$. □

Based on the above lemmas, we prove Theorem 1.

PROOF OF THEOREM 1. Consider the function $g = \Sigma_{|S|\leq k} \alpha_S \chi_S$. If $|S| > k$, then $\hat{g}(S) = 0$. But f has a circuit of depth d and size M, and an application of the Main Lemma yields:

$$\sum_{|S|>k} \left(\hat{f}(S) - \hat{g}(S)\right)^2 = \sum_{|S|>k} \hat{f}(S)^2 \leq \frac{\epsilon}{2}.$$

By Lemma 8, with probability of at least $1 - \delta$, there holds $(\hat{g}(S) - \hat{f}(S))^2 \leq \epsilon/2n^k$ for every set $|S| \leq k$. Whenever this is the case g satisfies the conditions of Lemma 9, so that $sign(g)$ disagrees with f on no more than an ϵ fraction of the inputs. □

5. *Further Corollaries*

In this section, the Main Lemma is used to derive some new properties of functions in AC^0.

5.1. Approximations by a Low Degree Polynomial.

As mentioned previously, Boolean functions can be thought of as taking real values. So it makes sense to approximate them with simple real functions such as low-degree polynomials. This complements the results of [14] and [17], showing that such approximation is possible over finite fields.

Lemma 10. *Let $f \in AC^0[d]$, then for every $\epsilon > 0$, there exists a polynomial p of degree at most $O(\log(n/\epsilon)^d)$ such that $\|f - p\| < \epsilon$.*

Proof. Approximate f by $p = \Sigma_{|S| \leq k} \hat{f}(S)\pi_S$, where $\pi_S = \Pi_{i \in S} x_i$, where the input bits x_i are taken to have values of 1 and -1 (respectively, false and true). The lemma becomes a restatement of the Main Lemma. □

It is interesting to note that the results of [14] and [17] for polynomials over finite fields yield more general, weighted approximations. These weights can correspond to any probability distribution. Our results apply only for the uniform distribution.

5.2. Low Average Sensitivity

Definition 1. Let f be a Boolean function, and $w \in \{0,1\}^n$. The *sensitivity* of f on w is the number of hamming neighbors w' of w such that $f(w) \neq f(w')$. The *average sensitivity* of f, $s(f)$, is the average over all $w \in \{0,1\}^n$ of the sensitivity of f on w.

This quantity measures how on average the value of f is sensitive to changes in the input. Equivalently, average sensitivity can be defined as the sum of influences of all variables on f (see [9]) in terms of the Fourier transform of f it becomes:

Lemma 11. *For any Boolean function f:*

$$s(f) = \sum_S |S| \hat{f}(S)^2.$$

Comment. In [9], this appears as $4\Sigma_S |S| \hat{f}(S)^2$, because the Boolean functions discussed there map to $\{0,1\}$, while here the range is $\{1, -1\}$.

An application of the Main Lemma implies:

Lemma 12. *For any function $f \in AC^0[d]$, we have $s(f) = O((\log n)^d)$.*

The bound given by this lemma is not far from optimal as the parity function on $(\log n)^{d-1}$ bits has sensitivity $(\log n)^{d-1}$, and can be computed in $AC^0[d]$.

This lemma gives a general, simple way to prove lower bounds for AC^0, and has recently found some new applications. In [16], it is used to obtain lower bounds on the number of negations required by AC^0 circuits. In [12], it is used to show that universal hashing cannot be done in AC^0.

5.3. No Pseudorandom Function Generators. A function f: $\{0,1\}^m \times \{0,1\}^n \to \{0,1\}$ is called a *pseudorandom function generator* if no oracle Turing machine M running in polynomial time can distinguish between a truly random oracle and the oracle $f(s, *)$, where s is chosen at random. For exact definitions as well as constructions of such generators, see [6]. (Here we are using a function generator that outputs one bit, in contrast to a string of bits.)

Lemma 13. *There does not exist a pseudorandom function generator in* AC^0.

Proof. The following algorithm exploits the low-average sensitivity of AC^0 functions, in order to distinguish them from a truly random one. The algorithm chooses a random x. Then, the algorithm flips a random bit in x, denote the result by x'. If $f(x) = f(x')$, then the algorithm guesses AC^0 *function*; otherwise, it guesses *random function*. □

5.4. Correlation with t-Wise Independent Probability Distributions. Consider probability distributions on $2^{\{x_1, \cdots x_n\}}$. Such a distribution μ is called *t-wise independent* if for every $x_{i_1} \cdots x_{i_t}$ and every $\epsilon_{i_1} \cdots \epsilon_{i_t}$ in $\{0,1\}$ there holds $\mu(x_{i_1} = \epsilon_{i_1} \cdots x_{i_t} = \epsilon_{i_t}) = 2^{-t}$. An easy but useful observation is that if μ is considered as a real function on the cube, then it is t-wise independent iff its Fourier transform vanishes on all S of cardinality between 1 and t.

Such distributions play an important role in the design of pseudorandom generators for AC^0 [2, 13]. Indeed, it is conjectured in [11] that any distribution that is polylog-wise independent is a pseudorandom generator for AC^0. Specifically, for a real function f on the cube and a probability distribution μ on it, let $E_\mu(f)$ (respectively, $E(f) = E_U(f)$) be the expectation of f when the input is chosen according to μ (respectively, uniformly). The conjecture is that for $f \in AC^0[d]$ and a $(\log^{d-1} n)$-wise independent μ the difference between $E(f)$ and $E_\mu(f)$ does not exceed 0.1, say. Here we show a result of a similar flavor that falls short, however, of proving the conjecture.

For the purpose of this section alone, Boolean functions map into $\{0,1\}$.

Lemma 14. *Let f be a Boolean function computable by a circuit of depth d and size M and let μ be a t-wise independent probability distribution, then*:

$$\left|E_U(f) - E_\mu(f)\right| \le 2^n \|\mu\| \cdot \sqrt{M}\, 2^{(-1/40)t^{1/d}}.$$

Proof. Notice that

$$E_\mu(f) = 2^n \langle f, \mu \rangle = 2^n \sum_{S \subset \{1 \cdots n\}} \hat{f}(S)\hat{\mu}(S),$$

where the last equality follows from the orthonormality of the basis of characters. Since f maps into $\{0,1\}$ its expectation equals $E_U(f) = \hat{f}(\varnothing)$, and

$\hat{\mu}(\varnothing) = 2^{-n}$. Also, μ is t-wise independent so $\hat{\mu}(S) = 0$, for $1 \leq |S| \leq t$. By Cauchy-Schwartz inequality

$$\left|E_U(f) - E_\mu(f)\right| = 2^n \left| \sum_{|S|>t} \hat{f}(S)\hat{\mu}(S) \right| \leq 2^n \sqrt{\sum_{|S|>t} \hat{f}(S)^2 \sum_{|S|>t} \hat{\mu}(S)^2}.$$

An application of the Main Lemma completes the proof. □

The quantity $\|\mu\|$ plays an important role in the above bound. Note that $\|\mu\| = \sqrt{\Sigma_x \mu^2(x)}$; therefore, $\|\mu\|^2$ is the collision probability of the distribution (the probability that two random values drawn independently according to μ have the same value). In order for our upper bound to be nontrivial (i.e., less than one), $\|\mu\|$ has to be exponentially small. For example, if μ is polynomially bounded, that is, for any x the probability $\mu(x) \leq poly(n)/2^n$, then we get a meaningful bound. Unfortunately, for our bound to be meaningful, the distribution μ has to be "fairly close" to the uniform distribution.

ACKNOWLEDGMENTS. We would like to thank Mauricio Karchmer, Mike Sipser, Robert Sloan, and Prasoon Tiwari for helpful discussions. We would like to thank the anonymous referees whose comments helped to both improve and simplify the presentation.

REFERENCES

1. AJTAI, M. Σ_1^1-formulae on finite structure. *Ann. Pure Appl. Logic 24* (1983), 1–48.
2. AJTAI, M., AND WIGDERSON, A. Deterministic simulation of probabilistic constant depth circuits. In *Advances in computing research*, Vol. 5. S. Micali, ed. JAI Press, Greenwich, Ct., 1989, pp. 199–222.
3. BRANDMAN, Y., HENNESSY, J., AND ORLITSKY, A. A spectral lower bound technique for the size of decision trees and two level circuits. *IEEE Trans. Comput. 39*, 2 (1990), 282–287.
4. DYM, H., AND MCKEAN, H. P. *Fourier Series and Integrals*. Academic Press, Orlando, Fla., 1972.
5. FURST, M., SAXE, J., AND SIPSER, M. Parity, circuits, and the polynomial-time hierarchy. *Math. Syst. Theory 17* (1984), 13–27.
6. GOLDREICH, O., GOLDWASSER, S., AND MICALI, S. How to construct random functions. *J. ACM 33*, 4 (Oct. 1986), 792–807.
7. HAGERUP, T., AND RUB, C. A guided tour to Chernoff bounds. *Inf. Proc. Lett. 33* (1989), 305–308.
8. HASTAD, J. AND BOPPANA, Computational limitations for small depth circuits. Ph.D. dissertation, MIT Press, Cambridge, Mass., 1986.
9. KAHN, J., KALAI, G., AND LINIAL, N. The influence of variables on Boolean functions. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science* (White Plains, N.Y., Oct.). IEEE, New York, 1988, pp. 68–80.
10. KEARNS, M., AND VALIANT, L. Cryptographic limitations on learning Boolean formulae and finite automata. In *Proceedings of the 21st Annual ACM Symposium on Theory of Computing* (Seattle, Wash., May). ACM, New York, 1989, pp. 433–444.
11. LINIAL, N., AND NISAN, N. Approximate inclusion-exclusion. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*. (Baltimore, Md., May 12–14). ACM, New York, 1990, pp. 260–270.
12. MANSOUR, Y., NISAN, N., AND TIWARI, P. The computational complexity of universal hashing. In *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*. (Baltimore, Md., May 12–14). ACM, New York, 1990, pp. 235–243.
13. NISAN, N., AND WIGDERSON, A. Hardness vs. randomness. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science* (White Plains, N.Y., Oct.) IEEE, New York, 1988, pp. 2–12.

14. RAZBOROV, A. A. Lower bounds for the size of circuits of bounded depth with basis AND, XOR. *Math. Zametski 41* (1987), 598–607 (in Russian). English translation in *Math Notes 41* (1987), 333–338.
15. RIVEST, R. L. Learning decision lists. *Machine Learning 2*, 3 (1987), 229–246.
16. SANTHA, M., AND WILSON, C. Polynomial size circuits with a limited number of negations. In *Proceedings of the 8th Annual Symposium on Aspects of Theoretical Computer Science*. IEEE, New York, 1991, pp. 228–237.
17. SMOLENSKY, R. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing* (New York City, N.Y., May 25–27). ACM, New York, 1987, pp. 77–82.
18. VALIANT, L. G. A theory of the learnable. *Commun ACM 27*, 11 (Nov. 1984), 1134–1142.
19. YAO, A. C. Theory and applications of trapdoor functions. In *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science*. IEEE, New York, 1982, pp. 80–91.
20. YAO, A. C. Separating the polynomial-time hierarchy by oracles. In *Proceedings of the 26th Annual Symposium on Foundations of Computer Science* (Portland, Ore. Oct.). IEEE, New York, 1985, pp. 1–10.

RECEIVED DECEMBER 1989; REVISED NOVEMBER 1991; ACCEPTED NOVEMBER 1991