INTERPOLATION AND APPROXIMATION OF SPARSE MULTIVARIATE POLYNOMIALS OVER GF(2)

Ron M. Roth[1] and Gyora M. Benedek[2]

Abstract. A function $f : \{0,1\}^n \to \{0,1\}$ is called t-sparse if the n-variable polynomial representation of f over $GF(2)$ contains at most t monomials. Such functions are uniquely determined by their values at the so-called critical set of all binary n-tuples of Hamming weight $\geq n - \lfloor \log_2 t \rfloor - 1$. An algorithm is presented for interpolating any t-sparse function f, given the values of f at the critical set. The time complexity of the proposed algorithm is proportional to n, t and the size of the critical set. Then, the more general problem of approximating t-sparse functions is considered, in which case the approximating function may differ from f at a fraction ϵ of the space $\{0,1\}^n$. It is shown that $O\Big((t/\epsilon) \cdot n\Big)$ evaluation points are sufficient for the (deterministic) ϵ-approximation of any t-sparse function, and that an order of $(t/\epsilon)^{\alpha(t,\epsilon)} \cdot \log n$ points are necessary for this purpose, where $\alpha(t,\epsilon) \geq 0.694$ for a large range of t and ϵ. Similar bounds hold for the t-term DNF case as well. Finally, a probabilistic polynomial-time algorithm is presented for the ϵ-approximation of any t-sparse function.

Key words. Interpolation of sparse polynomials; approximation of sparse polynomials; learning of Boolean functions; Reed-Muller codes.

AMS(MOS) subject classifications. 41A05, 41A10, 68Q20, 68R99, 94B35.

Abbreviated title. Interpolation and approximation of polynomials.

---

[1]IBM Research Division, Almaden Research Center, 650 Harry Road, San Jose, CA 95120.

[2]Rosh Intelligent Systems Ltd., P.O.Box 03552, Mevasseret Zion 90805, Israel.

This work was done in part while the authors were with the Computer Science Department, Technion – Israel Institute of Technology, Haifa 32000, Israel.

I. INTRODUCTION

Consider a Boolean function $f : \{0,1\}^n \to \{0,1\}$ which maps a vector $[x_{n-1}\, x_{n-2} \ldots x_0]$ into $f(x_{n-1}, x_{n-2}, \ldots, x_0)$. One common way to classify such functions is by the minimal number t for which there exists a *t-term disjunctive normal form* (in short, t-term DNF) expression equivalent to f; that is, an expression consisting of an inclusive OR of up to t products (AND) of variables, each variable possibly complemented (NOT). Another way of classifying Boolean functions is by the number of nonzero monomials in the (unique) n-variable polynomial representation of f over $GF(2)$,

$$f(x_{n-1}, x_{n-2}, \ldots, x_0) = \sum_{i=0}^{2^n-1} f_i \cdot x_{n-1}^{i_{n-1}} x_{n-2}^{i_{n-2}} \cdots x_0^{i_0}\,, \tag{1}$$

where $\mathbf{i} \triangleq [i_{n-1}\, i_{n-2} \ldots i_0]$ is the n-bit binary representation of i, and summation is carried out over $GF(2)$ (XOR). An n-variable Boolean function f is *t-sparse* if the number of nonzero monomials in the polynomial representation (1) of f is at most t.

In this work we first address the problem of *interpolating* t-sparse functions, that is: Given n, t and the values of a t-sparse function f at a subset P of $\{0,1\}^n$, can f be determined uniquely? if so, is there an efficient algorithm (i.e., in time complexity polynomial in n, t and $|P|$), by which f can be retrieved?

These questions arise in several applications, like in the study of function learnability and inductive inference [1][2][13][14][15]. In this model, a "student" tries to "learn" an underlying function f, given the values of f at some set of points $P \subseteq \{0,1\}^n$. Knowing the value of n (and, sometimes, t), the question is whether the student can retrieve f efficiently out of its values at P.

In Section II we show that, for the unique interpolation of f, the set P must contain a "critical set" consisting of all binary n-tuples of Hamming weight $\geq n - \lfloor \log_2 t \rfloor - 1$. This result applies to the *non-adaptive setting*, where the points of evaluation do not depend on values of f at previously-queried points. It turns out that adaptive schemes do not yield any significant reduction in the number of necessary queries. The existence of such a critical set has been proved (independently) also by Clausen et al. in [4]. Our result is somewhat stronger, showing that finding the *parity* of the truth table of f requires at least as many evaluation points as required for finding the truth table itself.

In Section III we present a deterministic non-adaptive algorithm which retrieves the underlying function f out of its values at this critical set in $O\left(t \cdot n \cdot \sum_{i=0}^{1+\lfloor \log_2 t \rfloor} \binom{n}{i}\right)$ bit operations. Establishing the correspondence between the interpolation problem and the decoding of certain error-correcting codes, our interpolation algorithm may also serve as a [syndrome-based] decoding algorithm for Reed-Muller codes [9, Ch. 13]. We conclude our interpolation discussion by showing that, for fixed $t > 1$, deciding whether there exists a t-sparse function passing via a given (arbitrary) set of evaluation points is NP-complete (Section IV).

Interpolation algorithms have been presented also by Ben-Or and Tiwari [3], Clausen et al. [4], and Grigoriev, Karpinski and Singer [6]; however, in their model, f is evaluated at n-tuples over an *extension field* $GF(2^m)$ (in which case t evaluation points can be shown to be sufficient), whereas in our case the evaluation points are confined to n-tuples over the ground field $GF(2)$. This extension field model has been motivated, in part, by the fact that the size of the critical set is non-polynomial in n and t.

Another way of overcoming the non-polynomial nature of Boolean interpolation is by considering the more general problem of *approximating* Boolean functions. In this scheme, we may end up with a function $\hat{f}$ whose truth table differs from that of f at less than $\epsilon \cdot 2^n$ entries for some (pre-specified) $0 < \epsilon \leq 1$. This scheme is widely used in the context of function learnability, with the functions usually being represented as DNF expressions.

Much work has been done on the (still unresolved) problem of finding an efficient algorithm for t-term DNF approximation [8][11][13][14][15]. The last two sections in this paper are devoted to the t-sparse polynomial approximation problem. In Section VI we present an approximation algorithm which, given n, t, ϵ and a (small) probability p of failure, finds an ϵ-approximation for any t-sparse function with probability $\geq 1 - p$, requiring $O\Big((t^2 n^2/\epsilon) \cdot \log(tn/p)\Big)$ bit operations. We believe that this result may shed light on the t-term DNF approximation problem as well, and it exhibits one of the advantages of the polynomial representation in studying the learnability of Boolean functions.

Preceding the presentation of the above algorithm, we obtain in Section V lower and upper bounds on the number of evaluation points required for the approximation of t-sparse functions. We show that $O\Big((t/\epsilon) \cdot n\Big)$ points are sufficient for the (deterministic) ϵ-approximation of any t-sparse function, and that an order of $(t/\epsilon)^{\alpha(t,\epsilon)} \cdot \log n$ points are necessary for this

purpose, where $\alpha(t,\epsilon) \geq 0.694$ for a large range of t and ϵ (Theorem 5.1 and Theorem 5.3). Similar bounds are derived for the t-term DNF case as well.

II. BACKGROUND AND BASIC RESULTS

Given a function f over $\{0,1\}^n$, let $\mathbf{f} \triangleq [f_0\, f_1\, \ldots\, f_{2^n-1}]'$ denote the column vector of coefficients of f as defined by (1) and let

$$\mathbf{F} = \begin{bmatrix} f(0,\ldots,0,0,0) \\ f(0,\ldots,0,0,1) \\ f(0,\ldots,0,1,0) \\ \vdots \\ f(1,\ldots,1,1,1) \end{bmatrix}$$

denote the truth table of f. Let A be the 2×2 matrix given by

$$A \triangleq \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix},$$

and define the $2^n \times 2^n$ matrix A_n as follows: $A_0 \triangleq [1]$ and, for $n \geq 1$, $A_n \triangleq A \otimes A_{n-1}$, where $\otimes$ stands for the direct (or Kronecker) product of matrices. For instance,

$$A_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Writing f as a polynomial in x_{n-1},

$$f(x_{n-1},\, x_{n-2},\, \ldots,\, x_0) \;=\; f_0(x_{n-2},\, x_{n-3},\, \ldots,\, x_0) \;+\; x_{n-1} \cdot f_1(x_{n-2},\, x_{n-3},\, \ldots,\, x_0)\,, \qquad (2)$$

it is easy to show by induction on n that for every function f over $\{0,1\}^n$, $\mathbf{F} = A_n\mathbf{f}$ [9, Ch. 13, §2]. Also, since $A_n = A_n^{-1}$, we have $\mathbf{f} = A_n\mathbf{F}$ and, in particular, the parity of the number

of 1's in $\mathbf{F}$ is equal to the coefficient of $x_{n-1}x_{n-2}\cdots x_0$ in the polynomial representation of f.

Remark 2.1. It is sometimes convenient to use the following equivalent definition of A_n. Given two binary n-vectors $\mathbf{i} = [i_{n-1}\, i_{n-2}\, \ldots\, i_0]$ and $\mathbf{j} = [j_{n-1}\, j_{n-2}\, \ldots\, j_0]$, we say that $\mathbf{j}$ is *dominated* by $\mathbf{i}$, denoted $\mathbf{j} \sqsubseteq \mathbf{i}$, if $j_k \leq i_k$ for all $0 \leq k \leq n-1$. It can be readily verified that $A_n[i,j] = 1$ if and only if $\mathbf{j} \sqsubseteq \mathbf{i}$, with $\mathbf{i}$ and $\mathbf{j}$ standing for the binary representations of i and j, $0 \leq i, j \leq 2^n - 1$ [9, Ch. 13, §2].

The t-sparse interpolation problem can now be formulated in the following coding theory terms. Assume that the values of f are given at some l points in $\{0,1\}^n$. These values can be written as a binary l-tuple $\mathbf{s} \triangleq H\mathbf{f}$, known as the *syndrome* of $\mathbf{f}$, where H is an $l \times 2^n$ sub-matrix of A_n. The interpolation process can now be viewed as the *decoding* of the vector $\mathbf{f}$ given the vector $\mathbf{s}$. In order to achieve unique interpolation, every $2t$ columns of H must be linearly independent, or else there would be two distinct t-sparse functions $\mathbf{f_1}$ and $\mathbf{f_2}$ such that $H\mathbf{f_1} = H\mathbf{f_2}$. On the other hand, if every $2t$ columns of H are linearly independent, then $\mathbf{s}$ determines $\mathbf{f}$ uniquely, provided the latter is t-sparse. Therefore, H must be a *parity-check matrix* of a binary linear code of length 2^n, dimension $\geq 2^n - l$ and minimum distance $\geq 2t+1$.

The above discussion leads us to the well-known relation between the interpolation problem and Reed-Muller codes [9, Ch. 13], which we briefly summarize below. For every $\mathbf{u} \in \{0,1\}^n$, denote by $\mathrm{w}(\mathbf{u})$ the Hamming weight of $\mathbf{u}$. Let $S(n,r)$ be the set of all vectors $\mathbf{u} \in \{0,1\}^n$ with $\mathrm{w}(\mathbf{u}) \geq n - r$ and let $V(n,r) \triangleq |S(n,r)| = \sum_{i=0}^{r} \binom{n}{i}$ (when $r > n$ or $r < 0$ we define $\binom{n}{r} \triangleq 0$). From the properties of the Pascal triangle it is easy to verify the identity $V(n,r) = V(n-1,r) + V(n-1,r-1)$.

Now, let $H_{n,r}$ be the binary $V(n,r) \times 2^n$ matrix consisting of the rows of A_n whose indices i are of binary representation $\mathbf{i} \in S(n,r)$, with the order of these rows maintained as in A_n. It is easy to verify that

$$H_{n,r} = \begin{bmatrix} H_{n-1,r-1} & 0 \\ H_{n-1,r} & H_{n-1,r} \end{bmatrix} . \tag{3}$$

The matrix $H_{n,r}$ is known as the parity-check matrix of the binary $(n-r-1)$-st order Reed-Muller code of length 2^n, the minimum distance of which is 2^{r+1}. The next lemma is a direct corollary of the known properties of Reed-Muller codes.

Lemma 2.1. *For $1 \leq t \leq 2^n$, the $V(n, 1 + \lfloor \log_2 t \rfloor)$ values of a t-sparse function $f : \{0,1\}^n \to \{0,1\}$ at $S(n, 1 + \lfloor \log_2 t \rfloor)$ are sufficient in order to determine uniquely any such function f.*

Proof. This follows from the fact that every $2t$ columns in $H_{n,1+\lfloor \log_2 t \rfloor}$ are linearly independent. □

The following lemma is the converse of Lemma 2.1. Moreover, we show that finding the coefficient of $x_{n-1}x_{n-2}\cdots x_0$ in f requires as many evaluation points as required for finding f itself.

Lemma 2.2. *In order to find the parity of the truth table of any t-sparse function $f : \{0,1\}^n \to \{0,1\}$, $1 \leq t \leq 2^n$, the values of f should be specified at all points of $S(n, 1 + \lfloor \log_2 t \rfloor)$.*

Proof. First, $[1\,1\,\ldots\,1]$ is the only evaluation point distinguishing the zero function from $x_{n-1}x_{n-2}\cdots x_0$. Now, let $0 < r \leq 1 + \lfloor \log_2 t \rfloor$ and assume that $\mathbf{z} \stackrel{\Delta}{=} [\underbrace{00\ldots0}_{r}\,11\ldots1]$ is not one of the evaluation points. Denote by $\overline{x}_j$ the complement of the variable x_j, and define the functions f and g by

$$f(x_{n-1}, x_{n-2}, \ldots, x_0) \stackrel{\Delta}{=} \overline{x}_{n-2}\overline{x}_{n-3}\cdots\overline{x}_{n-r} \cdot x_{n-r-1}x_{n-r-2}\cdots x_0 \tag{4}$$

and

$$g(x_{n-1}, x_{n-2}, \ldots, x_0) \stackrel{\Delta}{=} x_{n-1} \cdot \overline{x}_{n-2}\overline{x}_{n-3}\cdots\overline{x}_{n-r} \cdot x_{n-r-1}x_{n-r-2}\cdots x_0\ . \tag{5}$$

It is easy to see that the (nonzero) sum $f + g$ $(= \overline{x}_{n-1} \cdot f)$ vanishes at all points of $\{0,1\}^n$ except $\mathbf{z}$. Substituting $1 + x_j$ for $\overline{x}_j$ in the right-hand sides of (4) and (5), and expanding the expressions thus obtained, we conclude that f and g are both t-sparse functions taking the same value at every evaluation point. On the other hand we have $\mathrm{w}(\mathbf{F}) = 2$, whereas $\mathrm{w}(\mathbf{G}) = 1$. □

We can therefore summarize:

Theorem 2.1. *For $1 \leq t \leq 2^n$, the $V(n, 1 + \lfloor \log_2 t \rfloor)$ values of a t-sparse function $f : \{0,1\}^n \to \{0,1\}$ at $S(n, 1 + \lfloor \log_2 t \rfloor)$ are necessary and sufficient in order to determine uniquely any such function f.*

We now turn to the adaptive case, where the points of evaluation may depend on values

of the underlying function at previously-queried points. In such a scheme, any interpolation procedure can be described in a form of a tree: Each vertex corresponds to an interpolation query, whose result determines which one of the successive sub-trees we should go next. Each leaf in the tree is associated with at most one n-variable function, and every t-sparse function must be associated with at least one leaf.

Let $v_{\mathbf{0}}$ be a leaf corresponding to the zero function, let $l_{\mathbf{0}}$ be its distance from the root, and let $P_{\mathbf{0}}$ denote the $l_{\mathbf{0}}$ interpolation points queried from the root up to $v_{\mathbf{0}}$. For unique interpolation, none of the nonzero t-sparse functions should vanish at $P_{\mathbf{0}}$. Following similar arguments as given in the proof of Lemma 2.2, we obtain the lower bound $l_{\mathbf{0}} \geq V(n, \lfloor \log_2 t \rfloor)$. This leaves quite a marginal benefit, if any, in using adaptive interpolation algorithms, compared with the non-adaptive case.

III. INTERPOLATION ALGORITHM FOR t-SPARSE FUNCTIONS

There exists a well-known decoding algorithm for Reed-Muller codes, based on majority logic circuits [9, Ch. 13, §6,7]. However, this algorithm is not suitable for our purposes as its time complexity is proportional to 2^n. Instead, we describe a recursive procedure for solving deterministically the n-variable t-sparse interpolation problem: given the values of a t-sparse function $f : \{0,1\}^n \to \{0,1\}$ at the critical set specified in Theorem 2.1, the algorithm retrieves f in $O\Big(t \cdot n \cdot V(n, 1 + \lfloor \log_2 t \rfloor)\Big)$ bit operations. A similar algorithm has been discovered recently also by Hellerstein and Warmuth [7].

The procedure, named *INTERPOL* , is presented in Figure 1. Given an underlying n-variable t-sparse function f, the input to *INTERPOL* consists of the number of variables n; an integer r such that $t \leq 2^r - 1$; and the vector $\mathbf{s} = H_{n,r}\mathbf{f}$ of values of f at $S(n,r)$. The output of *INTERPOL* is the support F of the coefficient vector $\mathbf{f}$, i.e., the set of indices of the nonzero entries of $\mathbf{f}$.

The first steps of *INTERPOL* check whether either r or n is zero, in which case $\mathbf{s}$ is a scalar and, therefore, f can be determined in a straightforward manner. Note that when $r = 0$ and $\mathbf{s} \neq \mathbf{0}$ there is no solution for f, causing the procedure to raise the "*failure*" flag.

In case both n and r are nonzero, we enter the recursion stage. Let $\mathbf{f_0}$ and $\mathbf{f_1}$ denote the first and second halves of the coefficient vector $\mathbf{f}$, each $\mathbf{f_i}$ being a vector of length 2^{n-1}.

The basic idea is to find $\mathbf{f}$ by computing the two vectors $\mathbf{f_0}$ and $\mathbf{f_1}$ using recursive calls to *INTERPOL* . In order to find these vectors, we first issue the call *INTERPOL* $(n-1,\, r,\, \mathbf{s}_+)$, where $\mathbf{s}_+$ is a vector consisting of the last $V(n-1,r)$ entries of $\mathbf{s}$. This corresponds to interpolating the $(n-1)$-variable polynomial f_+, associated with the vector $\mathbf{f}_+ \stackrel{\Delta}{=} \mathbf{f_0} + \mathbf{f_1}$ (and represented by its support F_+). Interpolation failure at this stage indicates that t is greater than $2^r - 1$.

We now perform a second call to *INTERPOL* with the parameters *INTERPOL* $(n-1,\, r-1,\, \mathbf{s_0})$, where $\mathbf{s_0}$ is a vector consisting of the first $V(n-1,r-1)$ entries of $\mathbf{s}$. If no failure has occurred at this call, the computed set F_0 is the support of $\mathbf{f_0}$, allowing us to compute the support F_1 of $\mathbf{f_1} = \mathbf{f_0} + \mathbf{f}_+$ (or, in set notation, $F_1 = F_0 \oplus F_1$). The sum of the sizes of F_0 and F_1 now determines whether a third call to *INTERPOL* should be made. Such a call, when required, re-calculates the sets F_0 and F_1. To do this, we need to compute an intermediate vector $\mathbf{s_2}$ which consists of the entries of $\mathbf{s}$ of indices $i \geq 2^{n-1}$, $\mathbf{i} \in S(n, r-1)$ (note that $\mathbf{s_2}$ is of the same length $(V(n-1,r-1))$ as the previously-obtained vector $\mathbf{s_0}$). The set F_1 is re-computed by the call *INTERPOL* $(n-1,\, r-1,\, \mathbf{s_1})$, where $\mathbf{s_1} = \mathbf{s_0} + \mathbf{s_2}$, and then F_0 is updated accordingly. At this point we must have $\mathrm{w}(\mathbf{f}) = |F_0| + |F_1| \leq 2^r - 1$, unless t was not in the right range in the first place. Finally, the support F of $\mathbf{f}$ is obtained as the union of F_0 and a 2^{n-1}-offset of F_1.

To summarize, given n and t, the interpolation of t-sparse functions f over $\{0,1\}^n$ is carried out first by querying the values $\mathbf{s} = H_{n,1+\lfloor \log_2 t \rfloor}$ and then calling *INTERPOL* using the parameters *INTERPOL* $(n,\, 1 + \lfloor \log_2 t \rfloor\, ,\, \mathbf{s})$.

Lemma 3.1. *Let f be a t-sparse function over $\{0,1\}^n$ and let r be an integer such that $t \leq 2^r - 1$. Given the values $\mathbf{s} = H_{n,r}\mathbf{f}$ of f at $S(n,r)$, the output of INTERPOL $(n, r, \mathbf{s})$ equals the set of indices of the nonzero coefficients of f.*

Proof. Consider first the case when $r = 0$. Here t must be zero and, therefore, both $\mathbf{f}$ and $\mathbf{s}$ must be zero. Hence, if $\mathbf{s} \neq \mathbf{0}$, our assumption on the range of t is readily not satisfied, in which case *INTERPOL* returns "*failure*".

Assuming from now on that $r > 0$, we continue the proof by induction on n. When $n = 0$, we have either $f \equiv 0$ or $f \equiv 1$, according to the value of the scalar $\mathbf{s}$ (note that r might be greater than n).

```
procedure INTERPOL (n, r, s) output: F ;
/*
     Interpolation algorithm for n-variable t-sparse functions,  t ≤ 2^r − 1 .
     s = H_{n,r}f, where f is the t-sparse underlying (interpolated) function .
     F is the support of f, i.e., the set of indices of the nonzero entries
          of f .
     The procedure returns an error code "failure" in case there is no
          t-sparse function f satisfying s = H_{n,r}f .
 */
begin
     if r = 0 then
          if s = [0] then F ← ∅ else return "failure"
          /*  An empty set (F = ∅) corresponds to f ≡ 0 .   */
     else if n = 0 then
          if s = [1] then F ← {0} else F ← ∅
          /*  F = {0} corresponds to f ≡ 1 .   */
     else begin
          s_0 ← the V(n − 1, r − 1)-prefix of s ;
          s_+ ← the V(n − 1, r)-suffix of s ;
          F_+ ← INTERPOL (n − 1, r, s_+) ;
          if ("failure" while finding F_+) then return "failure"
          else begin
               F_0 ← INTERPOL (n − 1, r − 1, s_0) ;
               F_1 ← F_0 ⊕ F_+      /*  ≜ (F_0 ⋃ F_+) − (F_0 ⋂ F_+)   */ ;
               if |F_0| + |F_1| ≥ 2^r or ("failure" while finding F_0) then
               begin
                    s_2 ← [ the entries of s of indices i ≥ 2^{n−1}, i ∈ S(n, r − 1) ] ;
                    s_1 ← s_0 + s_2 ;
                    F_1 ← INTERPOL (n − 1, r − 1, s_1) ;
                    F_0 ← F_1 ⊕ F_+ ;
                    if |F_0| + |F_1| ≥ 2^r or ("failure" while finding F_1) then
                         return "failure"
               end;
               F ← { i  |  i ∈ F_0  or  i − 2^{n−1} ∈ F_1 }
          end
     end
end;
```

Figure 1: Procedure *INTERPOL* .

Now suppose that both n and r are nonzero. Recalling the definitions of $\mathbf{f_0}$, $\mathbf{f_1}$ and $\mathbf{f}_+$, by (3) we have $\mathbf{s_0} = H_{n-1,r-1}\mathbf{f_0}$ and $\mathbf{s}_+ = H_{n-1,r}\mathbf{f}_+ = H_{n-1,r}(\mathbf{f_0} + \mathbf{f_1})$. Note that our assumption on t implies $\mathrm{w}(\mathbf{f}_+) \leq t \leq 2^r - 1$ and, therefore, by the induction hypothesis, the execution of *INTERPOL* $(n-1,\, r,\, \mathbf{s}_+)$ will end up with the support F_+ of $\mathbf{f}_+$.

Second, we find either $\mathbf{f_0}$ or $\mathbf{f_1}$, and then solve for the other half. Note that at least one of these vectors must have weight $\leq t/2 \leq 2^{r-1} - 1$. Suppose first that $\mathrm{w}(\mathbf{f_0}) \leq \mathrm{w}(\mathbf{f_1})$. Noting that $\mathbf{s_0} = H_{n-1,r-1}\mathbf{f_0}$, the induction hypothesis implies that the execution of *INTERPOL* $(n-1,\, r-1,\, \mathbf{s_0})$ will result in the support F_0 of $\mathbf{f_0}$, allowing us to calculate the support F_1 of $\mathbf{f_1}$. Now, if, indeed, $\mathrm{w}(\mathbf{f_0}) \leq 2^{r-1} - 1$, all the above executions of *INTERPOL* must end successfully (i.e., without the "*failure*" flag raised) and, therefore, we must have

$$|F_0| + |F_1| = \mathrm{w}(\mathbf{f_0}) + \mathrm{w}(\mathbf{f_1}) \leq 2^r - 1 \ . \tag{6}$$

Furthermore, since there exists at most one solution $\mathbf{f}$ of weight $\leq 2^r - 1$ to $\mathbf{s} = H_{n,r}\mathbf{f}$, the existence of a solution $\mathbf{f_0}$ for $\mathbf{s_0} = H_{n,r}\mathbf{f_0}$, with a vector $\mathbf{f_1} = \mathbf{f}_+ + \mathbf{f_0}$ satisfying (6), is a *sufficient* criterion for a successful interpolation of $\mathbf{f}$.

Now, suppose that (6) does not hold, or that the execution of *INTERPOL* $(n-1,\, r-1,\, \mathbf{s_0})$ returns "*failure*" at one of its recursion levels. This implies that $\mathrm{w}(\mathbf{f_1}) \leq 2^{r-1} - 1$ and, therefore, $\mathbf{f_1}$ should be recovered successfully by *INTERPOL* . The corresponding vector $\mathbf{s_1} \triangleq H_{n-1,r-1}\mathbf{f_1}$ can be found by observing that $H_{n-1,r-1}$ is a sub-matrix of $H_{n-1,r}$; hence, $\mathbf{s_1}$ can be written as the sum of $\mathbf{s_0}$ and the vector $\mathbf{s_2}$ consisting of the entries of $\mathbf{s}$ of indices $i \geq 2^{n-1}$, $\mathbf{i} \in S(n, r-1)$. Now, failure to satisfy (6) this time implies that $\mathrm{w}(\mathbf{f}) \geq 2^r$, in which case *INTERPOL* returns "*failure*". □

Theorem 3.1. *The interpolation of n-variable t-sparse functions can be carried out in $O\Big(t \cdot n \cdot V(n,\, 1 + \lfloor \log_2 t \rfloor)\Big)$ bit operations.*

Proof. Let $\tau(n, r)$ denote the number of bit operations required while executing *INTERPOL* $(n, r$
We assume that set operations are bounded by the size of the sets times n, and that $\mathbf{s}$ is sorted according to ascending order of indices i, $\mathbf{i} \in S(n, r)$. Note that given such an index i, its successor can be evaluated by the rule $i \leftarrow i + \left\lceil 2^{n-r-\mathrm{w}(i+1)} \right\rceil$ (where $\lceil \cdot \rceil$ stands for the ceiling function), and it takes $O\Big(n \cdot V(n, r)\Big)$ bit operations to generate all such indices i. This rule can also be used to extract $\mathbf{s_2}$ out of $\mathbf{s}$ in $O\Big(n \cdot V(n-1, r-1)\Big)$ bit operations.

We thus have

$$\tau(n,r) = \tau(n-1,r) + 2\cdot\tau(n-1,r-1) + O\Big(n\cdot V(n-1,r-1)\Big) + O\Big(n\cdot 2^{\min(n,r)}\Big)\ ,$$

with the initial values $\tau(0,r) = O(1)$ and $\tau(n,0) = O(n)$. It follows by induction on n that there exists a constant β such that

$$\tau(n,r) \le \beta\cdot(2^{r+1}-1)\cdot(n+1)\cdot V(n,r)\ .$$

Hence, we conclude that the execution of *INTERPOL* $(n, 1+\lfloor\log_2 t\rfloor, \mathbf{s})$ involves $O\Big(t\cdot n\cdot V(n,\ 1+\lfloor\log_2 t\rfloor)\Big)$ bit operations. □

IV. THE INTERPOLATION DECISION PROBLEM IS INTRACTABLE

The time complexity of the procedure presented in Section III is polynomial in n when t is *fixed*. This observation can be put in contrast with the next theorem which establishes the intractability of the *t-Interpolation Decision Problem* (in short, t-ID), defined as follows: Given a fixed integer t, an instance of the problem consists of an integer n and a subset $R \triangleq \Big\{(\mathbf{v}_i\,;\,s_i)\Big\}_{i=1}^{m}$ of $\{0,1\}^n \times \{0,1\}$. The problem is to decide whether there exists an n-variable t-sparse function f such that $f(\mathbf{v}_i) = s_i$ for all $1 \le i \le m$.

Theorem 4.1. *For any fixed $t > 1$, the t-Interpolation Decision Problem is NP-complete.*

The proof of Theorem 4.1 is carried out, in part, by a reduction from the so-called *Hypergraph t-Colorability Problem* (in short, Hyper-t-Col) to the t-ID Problem. For fixed t, an instance of the Hyper-t-Col Problem consists of a finite set Q and a collection $\mathcal{C} = \{Q_1, Q_2, \ldots, Q_m\}$ of subsets (or *constraints*) $Q_i \subseteq Q$. The problem is to decide whether there exists a function (or *coloring*) $\chi : Q \to \{1, 2, \ldots, t\}$, such that each Q_i contains two elements y and z for which $\chi(y) \ne \chi(z)$. A similar reduction is used in [11] to show the intractability of the problem of deciding whether there exists a t-term DNF passing via a given set of points.

Lemma 4.1. [5, p. 221][11]. *For every fixed $t \ge 2$, the Hypergraph t-Colorability Problem is NP-complete.*

Lemma 4.1 holds even when each Q_i in $\mathcal{C}$ is of size ≤ 3. Without loss of generality we can also assume that every Q_i is of size ≥ 2 and that the Q_i are distinct. This means that $|\mathcal{C}| \le V(n,3) - (n+1)$, where $n = |Q|$.

Lemma 4.2. *For $t = 2$ and $t = 3$, the t-ID Problem is NP-complete.*

Proof. First, it is easy to verify that the t-ID Problem is in NP. Now, we use the following reduction from the Hyper-t-Col Problem to the t-ID Problem. Given an instance $\Big(Q = \{q_0, q_1, \ldots, q_{n-1}\}, \mathcal{C}\Big)$ of the Hyper-t-Col Problem, let $\mathbf{u}_i = [u_{i,0}\, u_{i,1}\, \ldots\, u_{i,n-1}] \in \{0,1\}^n$ be the characteristic vector of $Q - Q_i$, $1 \leq i \leq |\mathcal{C}|$. That is, $u_{i,j} = 0$ if and only if $q_j \in Q_i$. Also, let $\mathbf{e}_j$ denote the vector in $\{0,1\}^n$ of weight $n-1$ whose zero is at location j, $0 \leq j \leq n-1$. The corresponding instance of the t-ID Problem is now given by $(n, R_{(Q,\mathcal{C})})$, where

$$R_{(Q,\mathcal{C})} \triangleq \Big\{(\mathbf{e}_j\,;\,1)\Big\}_{j=0}^{n-1} \bigcup \Big\{(\mathbf{u}_i\,;\,0)\Big\}_{i=1}^{|\mathcal{C}|} .$$

Note that $\Big|R_{(Q,\mathcal{C})}\Big| < V(n,3)$.

The proof of the validity of the reduction is very similar to the proof in [11], and it is presented here for the sake of completeness. First, assume that $\Big(Q = \{q_0, q_1, \ldots, q_{n-1}\}, \mathcal{C}\Big)$ is t-colorable by a coloring $\chi : Q \to \{1, 2, \ldots, t\}$. We show the existence of a t-sparse function $f : \{0,1\}^n \to \{0,1\}$ which passes via the set $R_{(Q,\mathcal{C})}$. Let f be the n-variable polynomial defined by $f = \sum_{l=1}^{t} T_l$, where each T_l is a monomial given by

$$T_l = \prod_{j \text{ s.t. } \chi(q_j) \neq l} x_j \ ,$$

and $T_l \triangleq 1$ if all the q_j are colored by l (in which case $\mathcal{C}$ must be empty). Clearly, each variable x_j is missing from exactly one monomial ($T_{\chi(q_j)}$) and, therefore, $T_l(\mathbf{e}_j) = \delta(l, \chi(q_j))$, where $\delta(\cdot,\cdot)$ stands for the Kronecker delta function. We thus have $f(\mathbf{e}_j) = 1$ for all $0 \leq j \leq n-1$. Now, let $Q_i \in \mathcal{C}$, let $\mathbf{u}_i = [u_{i,0}\, u_{i,1}\, \ldots\, u_{i,n-1}]$ be the characteristic vector of $Q - Q_i$ and suppose, to the contrary, that $f(\mathbf{u}_i) = 1$. This implies $T_l(\mathbf{u}_i) = 1$ for at least one l and, therefore, we must have $u_{i,j} = 1$ for every j such that $\chi(q_j) \neq l$. Hence, whenever $u_{i,j} = 0$ we have $\chi(q_j) = l$, implying that all the elements of Q_i have the same color, a contradiction.

We now show that the existence of an n-variable t-sparse function f satisfying $f(\mathbf{v}) = s$ for every $(\mathbf{v}\,;\,s) \in R_{(Q,\mathcal{C})}$ implies the existence of a valid coloring of Q. Suppose that such a function f exists, and write $f = \sum_{l=1}^{k} T_l$, $k \leq t$ (≤ 3), where each T_l is a nonzero monomial in the variables x_j, $0 \leq j \leq n-1$. First, we show that if x_j appears in f at least once, then it must be missing from exactly one monomial. Indeed, assume that x_j appears in T_1, implying $T_1(\mathbf{e}_j) = 0$. Since $f(\mathbf{e}_j) = \sum_{l=1}^{k} T_l(\mathbf{e}_j) = 1$, we must have $T_l(\mathbf{e}_j) = 1$ for exactly

one monomial T_l, $2 \leq l \leq k$, the only monomial from which x_j is absent. Therefore, every variable x_j which appears in f can be assigned a well-defined index $l = l(x_j)$ of the monomial T_l from which it is missing.

Now, define a coloring $\chi : Q \to \{1, 2, \ldots, t\}$ as follows. If x_j appears in f, then $\chi(q_j) = l(x_j)$; otherwise, assign $\chi(q_j) = 1$. We now show that the above is indeed a valid coloring of Q. Suppose, to the contrary, that there exists a constraint $Q_i \in \mathcal{C}$ such that $\chi(q_j) = l_0$ for all $q_j \in Q_i$. Assume first that, for some $q_j \in Q_i$, the corresponding x_j appears in f (in which case $l_0 = l(x_j)$), and let $\mathbf{u}_i$ be the characteristic vector of $Q - Q_i$. Since x_j appears in each T_l, $l \neq l_0$, we have $f(\mathbf{u}_i) = T_{l_0}(\mathbf{u}_i) = 0$. This means that for at least one variable x_r appearing in T_{l_0}, the corresponding entry $u_{i,r}$ in $\mathbf{u}_i$ must be zero, implying $q_r \in Q_i$. On the other hand, $l(x_r) \neq l_0$ and, therefore, $\chi(q_r) \neq l_0$, contradicting the assumption that all elements of Q_i are colored by l_0.

It remains to consider the case where there exists a constraint $Q_i \in \mathcal{C}$ such that neither of the variables x_j, corresponding to $q_j \in Q_i$, appear in f. Now, since $f(\mathbf{u}_i) = 0$, f must have an even number of nonzero monomials. Hence, for every $q_j \in Q_i$, the corresponding vector $\mathbf{e}_j$ satisfies $f(\mathbf{e}_j) = 0$, resulting in a contradiction. □

Proof of Theorem 4.1. To complete the proof of the theorem we present a reduction from the t-ID Problem to the $(t+2)$-ID Problem. Let (n, R_t) be an instance of the t-ID Problem; the corresponding instance of the $(t+2)$-ID Problem is given by $(n+2, R_{t+2})$, where

$$\begin{aligned} R_{t+2} \;=\; & \Big\{ (11\,\mathbf{v}\,;\, s) \;\Big|\; (\mathbf{v}\,;\, s) \in R_t \Big\} \\ & \bigcup \Big\{ (0\,0\,\ldots\,0\,;\, 0) \Big\} \\ & \bigcup \Big\{ (0\,1\,0\,0\,\ldots\,0\,;\, 1), (1\,0\,0\,0\,\ldots\,0\,;\, 1) \Big\} \end{aligned}$$

(the size of R_{t+2} is, therefore, $|R_t| + 3$).

To prove the validity of the reduction, we must show that there exists an n-variable t-sparse function f_t satisfying R_t if and only if there exists an $(n+2)$-variable $(t+2)$-sparse function f_{t+2} satisfying R_{t+2}. Indeed, if f_t satisfies R_t, then

$$f_{t+2}(x_{n+1}, x_n, x_{n-1}, \ldots, x_0) \triangleq x_{n+1}x_n \cdot f_t(x_{n-1}, \ldots, x_0) + x_{n+1} + x_n$$

satisfies R_{t+2}.

On the other hand, let f_{t+2} be an $(n+2)$-variable $(t+2)$-sparse function satisfying R_{t+2}. Since $f_{t+2}(0,0,\ldots,0) = 0$ and $f_{t+2}(0,1,0,0,\ldots,0) = f_{t+2}(1,0,0,0,\ldots,0) = 1$, we can write $f_{t+2} = \phi + x_{n+1} + x_n$, where ϕ is an $(n+2)$-variable t-sparse function containing neither the linear terms x_{n+1} and x_n, nor the constant 1. We now define the t-sparse function $f_t : \{0,1\}^n \to \{0,1\}$ by $f_t(\mathbf{x}) \stackrel{\Delta}{=} \phi(11\,\mathbf{x})$. Since $f_{t+2}(11\,\mathbf{v}) = f_t(\mathbf{v})$ for every $\mathbf{v} \in \{0,1\}^n$, f_t must satisfy R_t. □

By the proof of Lemma 4.2 and Theorem 4.1 it follows that Theorem 4.1 still holds even if we restrict the size of $R = \left\{(\mathbf{v}_i\,;\,s_i)\right\}_i$ to be smaller than $V(n,3)$. When $S(n,\,1+\lfloor\log_2 t\rfloor)$ is contained in R, however, the t-Interpolation Decision Problem is easy to solve.

V. APPROXIMATION OF BOOLEAN FUNCTIONS

In the following sections we consider the problem of approximating t-sparse functions $f : \{0,1\}^n \to \{0,1\}$, given the values of f at various evaluation points in $\{0,1\}^n$. Let $\mathbf{F}$ and $\mathbf{G}$ be the truth tables of two functions $f,\, g : \{0,1\}^n \to \{0,1\}$, and let ϵ be a real number in the interval $(0,1]$. We say that f and g are ϵ-*close* if $\mathrm{w}(\mathbf{F}+\mathbf{G}) < \epsilon \cdot 2^n$, or, equivalently, if $\mathrm{Prob}[f(\mathbf{x}) \neq g(\mathbf{x})] < \epsilon$, where $\mathbf{x}$ is chosen uniformly from the space $\{0,1\}^n$. Given a function f, any function g which is ϵ-close to f will serve as an ϵ-*approximation* of f. Two functions which are not ϵ-close are said to be ϵ-*far*.

A set P of points in $\{0,1\}^n$ is called an ϵ-*approximation set* for t-sparse functions, if every two t-sparse functions $f,\, g : \{0,1\}^n \to \{0,1\}$, taking the same values at P, are necessarily ϵ-close. Having such a set P and the values of a t-sparse function f at P, we can ϵ-approximate f by taking any t-sparse function g whose truth table coincides with that of f at P. On the other hand, consider a set Q of points in $\{0,1\}^n$ such that knowing the values of *any* t-sparse function f at Q is sufficient for finding an $(\epsilon/2)$-approximating function $\hat{f}$ for f. In such a case, Q must be an ϵ-approximation set, since every two t-sparse functions f and g whose truth tables coincide at Q have the same $(\epsilon/2)$-approximating functions $\hat{f}$. By the triangle inequality f and g must therefore be ϵ-close.

We begin by obtaining bounds on the minimum size $L(n,t,\epsilon)$ of any ϵ-approximation set for t-sparse functions over $\{0,1\}^n$ (note that our discussion in the foregoing sections corresponds to the special case $\epsilon \leq 2^{-n}$). As in the interpolation case, we shall concentrate

on the non-adaptive model, pointing out that similar bounds can be obtained for the adaptive case as well.

Let $f : \{0,1\}^n \to \{0,1\}$ be a t-sparse function with each monomial being a product of at least k variables. Then, it is easy to see that the truth table of f is of Hamming weight $\leq t \cdot 2^{n-k}$. On the other hand, by the properties of Reed-Muller codes we have the following lemma.

Lemma 5.1. *Let the polynomial representation of a nonzero function $f : \{0,1\}^n \to \{0,1\}$ consist of a sum of monomials, each being a product of at most k variables. Then, the truth table $\mathbf{F}$ of f satisfies*

$$\mathrm{w}(\mathbf{F}) \geq 2^{n-k}.$$

Proof. The vector $\mathbf{F}$ is a nontrivial linear combination of columns of A_n (c.f. Section II) whose indices are of Hamming weight $\leq k$. Now, these column vectors are exactly the rows of $H_{n,k}$. Therefore, $\mathbf{F}$ is a nonzero codeword of the k-th order Reed-Muller code of length 2^n (which is the *dual* of the $(n-k-1)$-st order Reed-Muller code [9, p. 376]) and, as such, its weight is at least 2^{n-k}. □

Let $\Gamma(n,t,k)$ denote the set of all t-sparse functions $f : \{0,1\}^n \to \{0,1\}$ such that each monomial in the polynomial representation of f is a product of at most k variables. Note that if f and g are two distinct functions in $\Gamma(n,t,k)$, then $f - g \in \Gamma(n,2t,k) - \{0\}$ and, therefore, by Lemma 5.1, f and g are 2^{-k}-far. Hence, any two such functions should not take the same values at any ϵ-approximation set. Setting $k = \lfloor -\log_2 \epsilon \rfloor$, we obtain the following information bound

$$L(n,t,\epsilon) \;\geq\; L(n,t,2^{-k}) \;\geq\; \log_2 |\Gamma(n,t,k)| \;=\; \log_2 V\Big(V(n, \lfloor -\log_2 \epsilon \rfloor), t\Big)\,. \tag{7}$$

For a large range of values of n, t and ϵ, we can obtain a tighter lower bound on $L(n,t,\epsilon)$ which is presented in Theorem 5.1, following the next definitions.

Let $H : [0,1] \to [0,1]$ be the function given by

$$H(x) \;=\; \begin{cases} 0 & \text{if} \quad x = 0 \\ -x \cdot \log_2 x \;-\; (1-x) \cdot \log_2 (1-x) & \text{if} \quad 0 < x \leq \frac{1}{2} \\ 1 & \text{otherwise} \end{cases}\,,$$

and, for $0 \leq \rho \leq 1$, let $E(\rho)$ be the curve in the real plane defined by

$$E(\rho) \stackrel{\Delta}{=} \left\{ (\delta, \mu) \;\middle|\; H\Big((1-\mu)\delta\Big) = \rho \cdot \Big(1 + \delta H(\mu)\Big),\ 0 \leq \delta \leq 1,\ 0 \leq \mu \leq \tfrac{1}{2} \right\}.$$

Using the notations $\rho_1 \stackrel{\Delta}{=} \frac{5-\sqrt{5}}{10} \approx 0.276$ and $\rho_2 \stackrel{\Delta}{=} \frac{5+\sqrt{5}}{10} \cdot H\left(\frac{3-\sqrt{5}}{4}\right) \approx 0.509$, we now define the function $\gamma : [0,1] \to [0,1]$ by

$$\gamma(\rho) \;=\; \begin{cases} (1-\rho) \cdot H\left(\rho/(1-\rho)\right) & \text{if} \quad 0 \leq \rho \leq \rho_1 \\ \log_2\left(\frac{1+\sqrt{5}}{2}\right) \ (\approx 0.694) & \text{if} \quad \rho_1 < \rho \leq \rho_2 \\ \max_{(\delta,\mu) \,\in\, E(\rho)} \left\{ \frac{H(\delta)}{1+\delta H(\mu)} \right\} & \text{if} \quad \rho_2 < \rho \leq 1 \end{cases} . \tag{8}$$

Figure 2 depicts $\gamma(\rho)$ versus ρ. Some of the properties of $\gamma(\cdot)$ are summarized in Lemma 5.5 and Remarks 5.1 through 5.3 below.

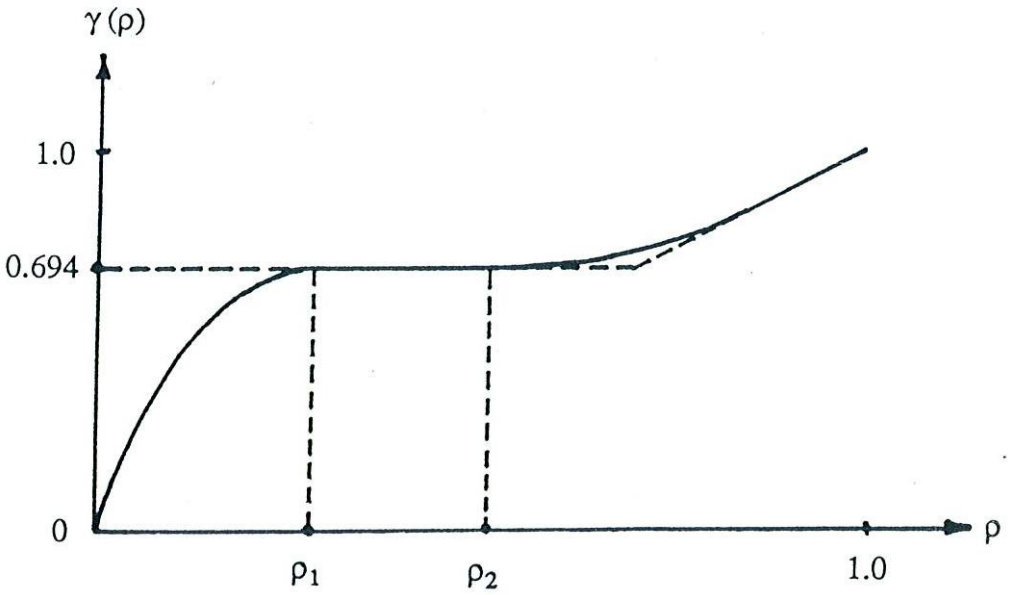

Figure 2: $\gamma(\rho)$ versus ρ.

Theorem 5.1. *For $t > 0$ and $\epsilon \in (0,1]$, let $\alpha(t,\epsilon) \stackrel{\Delta}{=} \gamma\left(\frac{\log t}{\log(t/\epsilon)}\right)$. Given any fixed $\theta > 0$, there exists an integer τ, depending only on θ, such that*

$$L(n,t,\epsilon) \;=\; \Omega\left((t/\epsilon)^{\alpha(t,\epsilon)-\theta} \cdot \log\left(\frac{1}{n + 2 - \log_2(t/\epsilon)} \;+\; \sqrt[\tau]{4\epsilon} \right)^{-1} \right) \tag{9}$$

whenever $\epsilon < \frac{1}{4}$ *and* $\tau \leq t/\epsilon \leq 2^n$.

Here $\Omega\Big(g(n)\Big)$ stands for an expression which is bounded from below by $a \cdot g(n)$ for some positive constant a. When $t/\epsilon \ll 2^n$ and $\epsilon \leq 1/n^c$ (for some fixed positive constant c), (9) becomes

$$L(n,t,\epsilon) \;=\; \Omega\,\Big((t/\epsilon)^{\alpha(t,\epsilon)-\theta} \cdot \log n\Big)\;.$$

On the other hand, when ϵ is constant (independent of n) the information bound (7) yields a better bound than (9). Note also that the range $t/\epsilon \geq 2^n$ has been excluded from Theorem 5.1 (see, however, Remark 5.4 below); in fact, in this range of parameters the values of the underlying function f can be specified at all points of $S(n,\, 1 + \lfloor \log_2 t \rfloor)$ (Theorem 2.1), still obtaining an algorithm whose time complexity is polynomial in t and $1/\epsilon$. As the proof of Theorem 5.1 is rather long, we postpone it to the end of this section.

The following is the analog of Theorem 5.1 for the t-term DNF case. Let $L_{DNF}(n,t,\epsilon)$ denote the minimum size of any DNF ϵ-approximation set, i.e., a set $P \subseteq \{0,1\}^n$ such that every two t-term DNF functions over $\{0,1\}^n$, taking the same values at P, are necessarily ϵ-close.

Theorem 5.2. *(i) For* $\epsilon < \frac{1}{4}$ *and* $t/\epsilon < 2^{n-1}$,

$$L_{DNF}(n,t,\epsilon) \;\geq\; \tfrac{1}{8} \cdot (t/\epsilon) \cdot \log_2 \left(\frac{1}{n+1-\log_2(t/\epsilon)} \;+\; 4\epsilon \right)^{-1} \quad .$$

(ii) For $\epsilon < \frac{1}{2}$ *and* $t/\epsilon \geq 2^{n-1}$,

$$L_{DNF}(n,t,\epsilon) \;=\; \Omega(2^n)\;.$$

The proof of Theorem 5.2 is presented after that of Theorem 5.1.[1]

The following theorem establishes a non-constructive upper bound on $L(n,t,\epsilon)$.

Theorem 5.3. *Given* n, t *and* $0 < \epsilon < 1$,

$$L(n,t,\epsilon) \;\leq\; \left\lceil \frac{\log\, V(2^n, 2t)}{-\log\,(1-\epsilon)} \right\rceil \tag{10}$$

[1]Due to integer roundings, the proofs of Theorem 5.1 and Theorem 5.2 yield slightly better lower bounds than the ones stated in the above theorems. However, for the sake of clarity, we chose to state these theorems in their present form.

and, therefore,

$$L(n,t,\epsilon) \;=\; O\Big(\,(t/\epsilon)\cdot n\,\Big)\;.$$

Proof. Let K denote the set of all $2t$-sparse functions f over $\{0,1\}^n$ with $\mathrm{w}(\mathbf{F}) \geq \epsilon \cdot 2^n$. It is sufficient to show that if L is an integer not smaller than the right-hand side of (10), then there exists an $L \times 2^n$ sub-matrix H of A_n (c.f. Section II) such that for any $f \in K$ we have $H\mathbf{f} \neq \mathbf{0}$.

For every $f \in K$ there exist less than $(1-\epsilon)\cdot 2^n$ rows $\mathbf{a}$ in A_n for which $\mathbf{a}\cdot\mathbf{f} = 0$. Therefore, for every integer L there exist less than $(1-\epsilon)^L \cdot 2^{nL} \cdot |K|$ distinct $L \times 2^n$ matrices H, with rows taken from A_n, such that $H\mathbf{f} = \mathbf{0}$ for at least one f in K (here H may contain the same row of A_n more than once). Now, $|K| \leq V(2^n, 2t)$ and, so, if

$$(1-\epsilon)^L \cdot 2^{nL} \cdot V(2^n, 2t) \;\leq\; 2^{nL}\;, \tag{11}$$

we can always find an $L \times 2^n$ matrix H for which $H\mathbf{f} \neq \mathbf{0}$ whenever $f \in K$. The theorem now follows by taking the logarithms of both sides of (11). □

The t-term DNF analog of Theorem 5.3 takes the form:

Theorem 5.4. *Given n, t and $0 < \epsilon < 1$,*

$$L_{DNF}(n,t,\epsilon) \;\leq\; \left\lceil \frac{2\cdot\log\left(\sum_{i=0}^{t} 2^i \cdot \binom{2^n}{i}\right)}{-\log(1-\epsilon)} \right\rceil \;=\; O\Big(\,(t/\epsilon)\cdot n\,\Big)\,.$$

Proof. The proof here is similar to that of Theorem 5.3. Let Γ_{DNF} denote the set of all t-term DNF functions over $\{0,1\}^n$. Given an integer L and two functions $f_1, f_2 \in \Gamma_{DNF}$ which are ϵ-far, there exist less than $(1-\epsilon)^L \cdot 2^{nL}$ ordered multisets P of L points in $\{0,1\}^n$ such that both f_1 and f_2 take the same values at P. Hence, if

$$(1-\epsilon)^L \cdot 2^{nL} \cdot |\Gamma_{DNF}|^2 \leq 2^{nL}\;,$$

we can always find a DNF ϵ-approximation set of size $\leq L$. The theorem now follows by the inequality $|\Gamma_{DNF}| \leq \sum_{i=0}^{t} 2^i \cdot \binom{2^n}{i}$. □

We now turn to the proof of Theorem 5.1, starting with a series of lemmas.

Lemma 5.2. *Given integers n, m, r and s, where $2 \leq m \leq n$ and $0 \leq r \leq m$, let B be an $L \times n$ binary matrix such that every $L \times m$ sub-matrix of B contains among its rows at*

least $V(m,r)-s$ distinct elements of $S(m,r)$. Then,

$$L \;\geq\; V(m-2,r-1)\cdot\left(\log_2\left(n-m+2\right)-\log_2\left(1+\frac{s(n-m+1)}{2V(m-2,r-1)}\right)\right)\;.$$

A special case of this combinatorial result, for $r=m$ and $s=0$, was proved in [12].

Proof. For every $\mathbf{u}\in S(m-2,r-1)$, let $L_{\mathbf{u}}$ denote the number of rows of B whose $(m-2)$-suffix is equal to $\mathbf{u}$, and let $C_{\mathbf{u}}$ denote the $L_{\mathbf{u}}\times(n-m+2)$ sub-matrix of B consisting of the $(n-m+2)$-prefixes of these rows (in case $L_{\mathbf{u}}=0$, $C_{\mathbf{u}}$ denotes an "empty" matrix). Assuming that $L_{\mathbf{u}}>0$, let $M_{\mathbf{u}}$ denote the number of pairs of identical columns in $C_{\mathbf{u}}$, and let N denote the number of distinct columns in $C_{\mathbf{u}}$, each such column appearing n_i times in $C_{\mathbf{u}}$, $i=1,2,\ldots,N$. We have

$$\sum_{i=1}^{N} n_i = n-m+2\;, \tag{12}$$

and

$$2\cdot M_{\mathbf{u}} \;=\; 2\cdot\sum_{i=1}^{N}\binom{n_i}{2} \;=\; \sum_{i=1}^{N} n_i^2 \;-\; \sum_{i=1}^{N} n_i\;.$$

Since $N\cdot\sum_{i=1}^{N} n_i^2 \geq \left(\sum_{i=1}^{N} n_i\right)^2$ we thus obtain

$$\begin{aligned} 2\cdot M_{\mathbf{u}} \quad &\geq \quad \frac{1}{N}\cdot\left(\sum_{i=1}^{N} n_i\right)^2 - \sum_{i=1}^{N} n_i \\ &\overset{(12)}{=} \quad \frac{(n-m+2)^2}{N} \;-\; (n-m+2)\;, \end{aligned}$$

or

$$N \;\geq\; \frac{(n-m+2)^2}{2M_{\mathbf{u}}+n-m+2}\;.$$

On the other hand, we must have $L_{\mathbf{u}} \geq \log_2 N$ to allow N distinct columns in $C_{\mathbf{u}}$. Therefore,

$$L_{\mathbf{u}} \;\geq\; \log_2\left(\frac{(n-m+2)^2}{2M_{\mathbf{u}}+n-m+2}\right)\;,$$

yielding

$$\begin{aligned} L \quad &\geq \quad \sum_{\mathbf{u}\,\in\, S(m-2,r-1)} L_{\mathbf{u}} \\ &\geq \quad \sum_{\mathbf{u}\,\in\, S(m-2,r-1)} \log_2\left(\frac{(n-m+2)^2}{2M_{\mathbf{u}}+n-m+2}\right) \end{aligned}$$

$$= \quad 2 \cdot V(m-2, r-1) \cdot \log_2 (n-m+2) \\ - \sum_{\mathbf{u} \in S(m-2,r-1)} \log_2 (2M_{\mathbf{u}} + n - m + 2) \ . \tag{13}$$

Let $\mathbf{c}_i$ and $\mathbf{c}_j$ be two identical columns (if any) in $C_{\mathbf{u}}$. These two columns define two row vectors, namely $[0\,1\,\mathbf{u}]$ and $[1\,0\,\mathbf{u}]$, both in $S(m,r)$, which are *missing* from the $L \times m$ sub-matrix of B consisting of the i-th and j-th columns, together with the last $m-2$ columns of B. Enumerating over all pairs of columns out of the first $n-m+2$ columns of B, we obtain

$$2 \ \cdot \sum_{\mathbf{u} \in S(m-2,r-1)} M_{\mathbf{u}} \ \leq \ s \cdot \binom{n-m+2}{2} \ . \tag{14}$$

Since the logarithmic function is convex, we can use Jensen's inequality [10, p. 277] to obtain

$$\begin{aligned} & \frac{\sum_{\mathbf{u} \in S(m-2,r-1)} \log_2 (2M_{\mathbf{u}} + n - m + 2)}{V(m-2, r-1)} \\ \leq \quad & \log_2 \left(\frac{\sum_{\mathbf{u} \in S(m-2,r-1)} (2M_{\mathbf{u}} + n - m + 2)}{V(m-2, r-1)} \right) \\ = \quad & \log_2 \left((n-m+2) + \frac{2 \sum_{\mathbf{u} \in S(m-2,r-1)} M_{\mathbf{u}}}{V(m-2, r-1)} \right) \\ \overset{(14)}{\leq} \quad & \log_2 \left((n-m+2) + \frac{s \cdot \binom{n-m+2}{2}}{V(m-2, r-1)} \right) \\ = \quad & \log_2 (n-m+2) \ + \ \log_2 \left(1 \ + \ \frac{s \cdot (n-m+1)}{2V(m-2, r-1)} \right) \ . \end{aligned} \tag{15}$$

Combining (13) and (15) we thus obtain

$$\begin{aligned} L \quad \geq \quad & 2 \cdot V(m-2, r-1) \cdot \log_2 (n-m+2) \\ & - \ V(m-2, r-1) \left(\log_2 (n-m+2) \ + \ \log_2 \left(1 + \frac{s \cdot (n-m+1)}{2V(m-2, r-1)} \right) \right) \\ = \quad & V(m-2, r-1) \left(\log_2 (n-m+2) \ - \ \log_2 \left(1 + \frac{s \cdot (n-m+1)}{2V(m-2, r-1)} \right) \right) \ . \end{aligned} \qquad \square$$

Lemma 5.3. *Given n, t and $2^{-n} \leq \epsilon \leq 1$, let $k \stackrel{\Delta}{=} \lfloor -\log_2 \epsilon \rfloor$ and let m and r be integers satisfying the following two conditions: (i) $\max(k, 2) \leq m \leq n$; and (ii) there exists an integer l, $0 \leq l \leq r$, such that $2^{m-k} \cdot V(r, l) + V(m, r-l-1) \ \leq \ 2t$. Then,*

$$L(n, t, \epsilon) \ \geq \ V(m-2, r-1) \cdot \left(\log_2 (n-m+2) - \log_2 \left(1 + \frac{(2^{m-k}-1)(n-m+1)}{2V(m-2, r-1)} \right) \right) \ .$$

Proof. Let $L \triangleq L(n,t,\epsilon)$ and let B be an $L \times n$ binary matrix whose rows form an ϵ-approximation set of size L. Let m and r be integers satisfying conditions (i) and (ii), and let C be an $L \times m$ sub-matrix of B consisting, say, of the last m columns of B. We now claim that C contains among its rows at least $V(m,r) - 2^{m-k} + 1$ distinct row vectors of $S(m,r)$.

Assume, to the contrary, that 2^{m-k} such rows are missing from C, say the rows $\mathbf{z}_i = [z_{i,m-1}\, z_{i,m-2} \ldots z_{i,0}]$, $1 \leq i \leq 2^{m-k}$. For each $\mathbf{z}_i$ we associate a term $\phi_i = \prod_{j=0}^{m-1} y_{i,j}$, where $y_{i,j} = x_j$ if $z_{i,j} = 1$, and $y_{i,j} = \overline{x}_j$ otherwise. It is easy to verify that for every $\mathbf{u} \in \{0,1\}^m$, $\phi_i(\mathbf{u}) = 1$ if and only if $\mathbf{u} = \mathbf{z}_i$. Substituting $1+x_j$ for $\overline{x}_j$ in ϕ_i, and expanding the expressions thus obtained, each ϕ_i becomes a sum of up to 2^r monomials.

For each i, $1 \leq i \leq 2^{m-k}$, let ξ_i denote the sum of those monomials in ϕ_i which are products of at most $m - r + l$ variables, where l is an integer guaranteed by condition (ii). Write $\phi \triangleq \sum_i \phi_i$ and $\xi \triangleq \sum_i \xi_i$ and let $\eta \triangleq \phi - \xi$. It can be verified that ξ is a sum of up to $2^{m-k} \cdot V(r,l)$ monomials and η is a sum of up to $V(m, r-l-1)$ monomials. Hence, by condition (ii), ϕ is $2t$-sparse. On the other hand, the truth table of ϕ, when regarded as an n-variable function, is of weight $2^{m-k} \cdot 2^{n-m} \geq \epsilon \cdot 2^n$ and, so, ϕ can be written as a sum of two functions which are both t-sparse and ϵ-far, contradicting the fact that they take the same values at an ϵ-approximation set. Since the above discussion applies to any $L \times m$ sub-matrix C of B, we can now apply Lemma 5.2 with $s = 2^{m-k} - 1$, thus concluding the proof of the lemma. □

When $k = n$ we can set $m = n$ and $r = l = 1 + \lfloor \log_2 t \rfloor$ in Lemma 5.3, yielding $L(n,t,2^{-n}) \geq V(n-2, \lfloor \log_2 t \rfloor)$, which, in view of Theorem 2.1, is quite close to the true value. Theorem 5.1 is virtually a restatement of Lemma 5.3, optimizing with respect to m, r and l and using the following well-known approximation of $V(n,r)$ (see, for instance, [9, p. 310]):

Lemma 5.4. *For every two integers n and $r = \mu \cdot n$, $0 \leq \mu \leq 1$,*

$$n \cdot H(\mu) - \tfrac{1}{2}\log_2(2n) \;\leq\; \log_2 V(n,r) \;\leq\; n \cdot H(\mu)\ .$$

Lemma 5.5. *Let $\psi, \omega_1, \omega_2 : [0,1] \times [0,1] \to [0,1]$ be given by*

$$\psi(\delta,\mu) \triangleq \frac{H(\delta)}{1 + \delta H(\mu)}\ ;$$

$$\omega_1(\delta,\mu) \stackrel{\Delta}{=} \frac{\delta H(\mu)}{1+\delta H(\mu)} \ ; \quad \textit{and} \quad \omega_2(\delta,\mu) \stackrel{\Delta}{=} \frac{H\Big((1-\mu)\delta\Big)}{1+\delta H(\mu)} \ .$$

For any $\rho \in [0,1]$, let $D_1(\rho)$ and $D_2(\rho)$ be the sets of pairs (δ,μ) in the unit square $[0,1]\times[0,1]$ defined by

$$D_i(\rho) = \Big\{ \, (\delta,\mu) \in [0,1] \times [0,1] \ \Big| \ \rho \ \geq \ \omega_i(\delta,\mu) \, \Big\} \ , \quad i = 1,2 \ , \tag{16}$$

and let $\gamma^* : [0,1] \to [0,1]$ be defined by

$$\gamma^*(\rho) \stackrel{\Delta}{=} \max_{(\delta,\mu) \, \in \, D_1(\rho) \bigcap D_2(\rho)} \{ \, \psi(\delta,\mu) \, \} \ . \tag{17}$$

Then,

$$\gamma^*(\rho) \geq \gamma(\rho) \ , \quad \rho \in [0,1] \ ,$$

where $\gamma(\cdot)$ is defined by (8).

Proof. Let $X_1(\rho)$ and $X_2(\rho)$ be the sets given by

$$X_1(\rho) \stackrel{\Delta}{=} \Big\{ \, (\delta,\mu) \in D_1(\rho) \bigcap D_2(\rho) \ \Big| \ \mu \geq \tfrac{1}{2} \, \Big\}$$

and

$$X_2(\rho) \stackrel{\Delta}{=} \Big\{ \, (\delta,\mu) \in D_1(\rho) \bigcap D_2(\rho) \ \Big| \ \mu \leq \tfrac{1}{2} \, \Big\} \ , \tag{18}$$

and let the functions $\gamma_1, \gamma_2 : [0,1] \to [0,1]$ be defined by

$$\gamma_i(\rho) \stackrel{\Delta}{=} \max_{(\delta,\mu) \, \in \, X_i(\rho)} \{ \, \psi(\delta,\mu) \, \} \ , \quad i = 1,2 \ . \tag{19}$$

Clearly, $\gamma^*(\rho) = \max\{\gamma_1(\rho), \gamma_2(\rho)\}$.

We start by analyzing the function $\gamma_1(\cdot)$. Given $\rho \in [0,1]$, $X_1(\rho)$ is equal to the set of pairs $(\delta,\mu) \in [0,1] \times [\frac{1}{2},1]$ satisfying both

$$\rho \geq \omega_1(\delta,\mu) = \omega_1(\delta,1) = \frac{\delta}{1+\delta} \tag{20}$$

and

$$\rho \geq \omega_2(\delta,\mu) \ . \tag{21}$$

Note that (20) is independent of μ, so is the expression $\psi(\delta,\mu) = \psi(\delta,1)$ which is to be maximized in (19) to obtain $\gamma_1(\rho)$. Also, (21) is satisfied for every ρ and δ if $\mu = 1$. Therefore, by (19) and (20) we can write

$$\gamma_1(\rho) = \max_{0 \leq \delta \leq \min\{1, \rho/(1-\rho)\}} \{\psi(\delta,1)\} \ . \tag{22}$$

The maximum value of $\psi(\cdot,1)$ in the interval $[0,1]$ is attained at $\delta_0 \stackrel{\Delta}{=} \frac{3-\sqrt{5}}{2}$, in which case $\psi(\delta_0,1) = \log_2\left(\frac{1+\sqrt{5}}{2}\right) \approx 0.694$. Hence, for $\rho \geq \rho_1 = \omega_1(\delta_0,1) = \frac{\delta_0}{1+\delta_0} = \frac{5-\sqrt{5}}{10} \approx 0.276$, we have $\gamma_1(\rho) = \psi(\delta_0,1) = \log_2\left(\frac{1+\sqrt{5}}{2}\right)$. Since $\psi(\delta,1)$ is monotonously increasing when $\delta < \delta_0$, the maximum in (22) for $\rho \leq \rho_1$ is attained when $\delta = \rho/(1-\rho)$. Hence,

$$\gamma_1(\rho) \;=\; \begin{cases} (1-\rho)\cdot H\Big(\rho/(1-\rho)\Big) & \text{if} \quad 0 \leq \rho \leq \rho_1 \\ \log_2\left(\frac{1+\sqrt{5}}{2}\right) & \text{if} \quad \rho_1 < \rho \leq 1 \end{cases} \;\;,$$

implying $\gamma_1(\rho) = \gamma(\rho)$ for $0 \leq \rho \leq \rho_2$.

We now turn to the function $\gamma_2(\cdot)$. For every $\delta \in [0,1]$ and $\mu \leq \frac{1}{2}$, we have $\delta H(\mu) \leq \delta \leq H(\delta/2) \leq H\Big((1-\mu)\delta\Big)$ and, therefore, (18) boils down to

$$X_2(\rho) = \Big\{\, (\delta,\mu) \in [0,1] \times [0,\tfrac{1}{2}] \;\;\Big|\;\; \rho \;\geq\; \omega_2(\delta,\mu) \,\Big\}\;,$$

implying $\gamma_2(\rho) \geq \gamma(\rho)$ for $\rho_2 < \rho \leq 1$. Therefore, $\gamma^*(\rho) \geq \gamma(\rho)$ for every $\rho \in [0,1]$. □

Remark 5.1. Referring to the notations of the last proof, we can verify that the functions γ_1 and γ_2, and therefore γ and γ^*, are all non-decreasing. Indeed, for any $\rho \leq \hat{\rho}$ we have $X_i(\rho) \subseteq X_i(\hat{\rho})$, $i = 1,2$.

Remark 5.2. For fixed δ, both $\psi(\delta,\mu)$ and $\omega_2(\delta,\mu)$ are monotonously non-increasing with respect to μ, whereas $\omega_1(\delta,\mu)$ is monotonously non-decreasing. Hence, if $\Big(\delta(\rho),\mu(\rho)\Big)$ is a pair attaining the maximum in (17) for a given ρ, we can assume that $\delta(\rho)$ and $\mu(\rho)$ are such that $\rho = \omega_2\Big(\delta(\rho),\mu(\rho)\Big)$. We thus have

$$\frac{\gamma^*(\rho)}{\rho} = \frac{\psi\Big(\delta(\rho),\mu(\rho)\Big)}{\omega_2\Big(\delta(\rho),\mu(\rho)\Big)} = \frac{H\Big(\delta(\rho)\Big)}{H\Big((1-\mu(\rho))\,\delta(\rho)\Big)} \geq 1\;; \tag{23}$$

that is, $\gamma^*(\rho)$ is always above (or on) the line $\rho \mapsto \rho$. Furthermore, it can be readily verified that both $\delta(\rho)$ and $\mu(\rho)$ are nonzero, unless $\rho \in \{0,1\}$, implying a strict inequality in (23) whenever $\rho \in (0,1)$.

Remark 5.3. Similarly,

$$\gamma_2(\rho) = \max_{\rho \geq\, \omega_2(\delta,\mu)} \psi(\delta,\mu) = \max_{\rho\,=\,\omega_2(\delta,\mu)} \psi(\delta,\mu)\;, \tag{24}$$

i.e, $\gamma_2(\rho) = \gamma(\rho)$ whenever $\rho_2 < \rho \leq 1$. Note also that

$$\rho_2 = \omega_2(\delta_0,\tfrac{1}{2}) = \tfrac{5+\sqrt{5}}{10}\cdot H\left(\tfrac{3-\sqrt{5}}{4}\right) \approx 0.509$$

and that

$$\gamma_2(\rho_2) \geq \frac{H(\delta_0)}{1+\delta_0 H(\frac{1}{2})} = \log_2\left(\tfrac{1+\sqrt{5}}{2}\right) . \tag{25}$$

In fact, by applying Lagrange multipliers on (24), it can be verified that (25) holds with equality; this implies that $\gamma(\cdot)$ is continuous (even differentiable) within the interval $[0,1]$ and that $\gamma^*(\rho)$ is actually *equal* to $\gamma(\rho)$.

Proof of Theorem 5.1. Given n, t and ϵ, let $h \stackrel{\Delta}{=} \lfloor \log_2 t \rfloor$, $k \stackrel{\Delta}{=} \lfloor -\log_2 \epsilon \rfloor$ (≥ 2) and $\rho \stackrel{\Delta}{=} h/(k+h)$. By the continuity of $\gamma(\rho)$, for every $\theta > 0$ there exists an integer $N_0(\theta)$ such that

$$\alpha(t,\epsilon) = \gamma\left(\tfrac{\log t}{\log(t/\epsilon)}\right) \leq \gamma\left(\rho + \frac{1}{k+h}\right) \leq \gamma(\rho) + \theta/2$$

whenever $k+h \geq N_0(\theta)$. Therefore, we choose τ to be at least $2^{N_0(\theta)+1}$, allowing us to replace the exponent $\alpha(t,\epsilon) - \theta$ in (9) by $\gamma(\rho) - \theta/2$, thus simplifying the analysis in the sequel.

Given n, k and h, let m, r and l be integers satisfying the following three conditions:

(a) $k \leq m \leq n$;

(b) $2^{m-k} \cdot V(r,l) \leq 2^h$; and —

(c) $V(m, r-l-1) \leq 2^h$.

Using the notation $\sigma(m,k,r) \stackrel{\Delta}{=} 2^{m-k-1}/V(m-2, r-1)$, by Lemma 5.3 we have

$$L(n,t,\epsilon) \geq V(m-2,r-1) \cdot \log_2\left(\frac{1}{n-m+2} + \sigma(m,k,r)\right)^{-1} \tag{26}$$

for any m, r and l satisfying (a)-(c). We now maximize $V(m-2,r-1)$ under the above three conditions.

Let $\mu \stackrel{\Delta}{=} l/r > 0$. By Lemma 5.4, (b) is implied by $2^{m-k} \cdot 2^{rH(\mu)} \leq 2^h$ and, so, we can set

$$r = \left\lfloor \frac{k+h-m}{H(\mu)} \right\rfloor . \tag{27}$$

Let m be in the range $\frac{1}{2}(k+h) \leq m \leq k+h$, and define $\lambda \stackrel{\Delta}{=} m/(k+h)$, $\frac{1}{2} \leq \lambda \leq 1$, and

$$\delta \stackrel{\Delta}{=} \frac{1-\lambda}{\lambda H(\mu)} ;$$

that is, $\lambda = \Big(1 + \delta H(\mu)\Big)^{-1}$ and, by (27), $r = \lfloor \delta \cdot m \rfloor$.

For any $\theta_1 > 0$ there exists an integer $N_1(\theta_1)$ such that

$$\begin{aligned} \log_2 V(m-2, r-1) &= \log_2 V\Big(\lambda(k+h) - 2, \lfloor \delta \cdot \lambda(k+h) \rfloor - 1\Big) \\ &\geq \lambda(k+h) \cdot \Big(H(\delta) - \theta_1\Big) \end{aligned}$$

whenever $\lambda(k+h) \geq N_1(\theta_1)$. Since $\frac{1}{2} \leq \lambda \leq 1$, we can set τ to be at least $2^{2N_1(\theta_1)+1}$, in which case

$$V(m-2, r-1) \geq 2^{(k+h)\cdot(\lambda H(\delta) - \theta_1)} \ . \tag{28}$$

Hence, whenever $t/\epsilon \geq \tau$ we have

$$V(m-2, r-1) \geq \tfrac{1}{4} \cdot (t/\epsilon)^{\beta} \ , \tag{29}$$

where β is any constant satisfying

$$\beta \leq \frac{H(\delta)}{1 + \delta H(\mu)} - \theta_1 = \psi(\delta, \mu) - \theta_1$$

(recall the notations of Lemma 5.5).

Plugging the values of λ, μ and δ into (c), we obtain the following condition

$$V\Big(\lambda(k+h), (1-\mu) \cdot \delta \cdot \lambda(k+h)\Big) \leq 2^h \tag{30}$$

which implies (c). By Lemma 5.4, (30) is satisfied when $\lambda(k+h) \cdot H\Big((1-\mu)\delta\Big) \leq h$, and, by the definition of ρ we thus obtain the condition

$$\rho \geq \lambda \cdot H\Big((1-\mu)\delta\Big) = \frac{H\Big((1-\mu)\delta\Big)}{1 + \delta H(\mu)} = \omega_2(\delta, \mu) \ , \tag{31}$$

i.e, $(\delta, \mu) \in D_2(\rho)$ (see Eq. (16)). Hence, at this point, (31), together with the definition of r in (27), guarantee conditions (b) and (c).

Refer now to condition (a). Clearly, $m \leq n$ since we require $m \leq k + h \leq n$. As for the lower bound on m, it can be easily verified that the inequality $k \leq m = \lambda(k+h)$ is satisfied if

$$\rho \geq \frac{\delta H(\mu)}{1 + \delta H(\mu)} = \omega_1(\delta, \mu) \ ,$$

i.e., $(\delta, \mu) \in D_1(\rho)$.

Now, let θ be fixed in the interval $(0,1]$ and let ρ_0 satisfy $\gamma(\rho_0) = \theta/2$. We distinguish between the following cases:

Case 1: $0 \le \rho < \rho_0$. In this case we have $\alpha(t,\epsilon) - \theta \le \gamma(\rho) - \theta/2 \le 0$ and, therefore, the theorem follows from the information bound $L(n,t,\epsilon) = \Omega(\log n)$ (Eq. (7)), which holds for any $t \ge 1$ and $\epsilon \le \frac{1}{2}$.

Case 2: $\rho_0 \le \rho \le 1 - \theta$. Let $\Big(\delta = \delta(\rho), \mu = \mu(\rho)\Big)$ be a pair which attains the maximum in (17). Note that, since $(\delta,\mu) \in D_1(\rho) \bigcap D_2(\rho)$, conditions (a), (b) and (c) are satisfied. By Remark 5.2 we also have

$$\rho \;=\; \omega_2(\delta,\mu) \;\le\; \psi(\delta,\mu) \;<\; \psi(\delta,\mu) + \omega_1(\delta,\mu)$$

for all $\rho_0 \le \rho \le 1-\theta$. Therefore, there exists a positive constant θ_2, depending only on θ, such that $\rho \le \psi(\delta,\mu) + \omega_1(\delta,\mu) - \theta_2$. This allows us to bound $\sigma(m,k,r)$ from above by

$$\begin{aligned}
\log_2 \sigma(m,k,r) \quad &= \quad m - k - 1 - \log_2 V(m-2, r-1) \\
&\overset{(28)}{\le} \quad \frac{k+h}{1+\delta H(\mu)} - k - 1 - (k+h)\Big(\psi(\delta,\mu) - \theta_1\Big) \\
&\le \quad (k+h)\Big(\rho - \psi(\delta,\mu) - \omega_1(\delta,\mu) + \theta_1\Big) - 1 \\
&\le \quad -(k+h)(\theta_2 - \theta_1) - 1 \;\le\; -k \cdot \frac{\theta_2 - \theta_1}{1-\rho_0} - 1
\end{aligned}$$

(assuming $\theta_2 \ge \theta_1$). Hence, we can set $\theta_1 = \frac{1}{2}\min(\theta, \theta_2)$, $\tau \ge 2(1-\rho_0)/\theta_2$, and, by Lemma 5.5, $\beta = \gamma(\rho) - \theta/2 \le \gamma^*(\rho) - \theta_1$ (in (29)), yielding

$$V(m-2, r-1) \;\ge\; \tfrac{1}{4} \cdot (t/\epsilon)^{\gamma(\rho) - \theta/2} \;\ge\; \tfrac{1}{4} \cdot (t/\epsilon)^{\alpha(t/\epsilon) - \theta}$$

and (assuming $\tau \ge 1$)

$$\sigma(m,k,r) \le \sqrt[4]{\epsilon}\ .$$

Case 3: $1 - \theta < \rho \le 1$. Set $r = m = h$ and $l = 0$; for these values we have $V(m-2, r-1) = \frac{1}{4} \cdot 2^h$, and it is easy to check that conditions (b) and (c) are satisfied. Condition (a) holds since, for this range of ρ, we have $k \le h < n$ (assuming $\theta \le \frac{1}{2}$). On the other hand,

$$(t/\epsilon)^{\alpha(t,\epsilon) - \theta} \;\le\; (t/\epsilon)^{1-\theta} \;\le\; 4 \cdot 2^{(k+h)(1-\theta)} \;=\; 4 \cdot 2^{(h/\rho)(1-\theta)} \;\le\; 4 \cdot 2^h$$

and

$$\sigma(m,k,r) \;=\; \frac{2^{m-k-1}}{2^{m-2}} \;=\; 2^{1-k} \;<\; 4\epsilon\ .$$

In cases 2 and 3 we thus have

$$V(m-2,r-1) \;=\; \Omega\left((t/\epsilon)^{\alpha(t,\epsilon)-\theta}\right)$$

and

$$\sigma(m,k,r) \le \sqrt[\tau]{4\epsilon}\;.$$

Recalling that $\log(n-m+2) \ge \log(n+2-k-h) \ge \log\Big(n+2-\log_2(t/\epsilon)\Big)$, the theorem is now implied by (26). □

Remark 5.4. We now consider briefly the case where $t/\epsilon > 2^n$. Referring to the notations of the last proof, our proof fails if the optimal value for m turns out to be greater than n. When $k+h \le n(1+\theta/2)$ we can still repeat the proof with $k' = \lfloor k/(1+\theta/2)\rfloor$ and $h' = \lfloor h/(1+\theta/2)\rfloor$, yielding

$$L(n,t,\epsilon) \;=\; \Omega\left((t/\epsilon)^{\alpha(t,\epsilon)-\theta}\right)\;.$$

Assume now that $k+h > n(1+\theta/2)$. Recalling that $m = (k+h)/(1+\delta H(\mu))$, we thus have to add the following condition

$$1+\delta H(\mu) \;\ge\; \frac{k+h}{n} \tag{32}$$

to conditions (a)-(c) while maximizing $V(m-2,r-1)$ in (26) (note that equality in (32) implies condition (a)). Instead of going through the steps of the proof of Theorem 5.1, we can obtain a simpler bound by assuming equality in both (31) and (32), resulting in the combined condition

$$\frac{1+\delta H(\mu)}{k+h} \;=\; \frac{1}{n} \;=\; \frac{H\Big((1-\mu)\delta\Big)}{h}\;. \tag{33}$$

For $h < n$ there exists a unique solution (δ,μ) to (33), satisfying

$$\delta \;=\; \frac{H^{-1}\left(\frac{h}{n}\right)}{1-\mu} \;=\; \frac{\frac{k}{n}+\frac{h}{n}-1}{H(\mu)}\;,$$

and when $h = n$ we can take $\delta = 1$. The lower bound is now obtained by plugging the solution for δ into the right-hand side of

$$L(n,t,\epsilon) \;\ge\; V(m,r) - 2^{m-k} + 1 \;=\; V\Big(n,\lfloor\delta\cdot n\rfloor\Big) - 2^{n-k} + 1\;,$$

the latter bound being a simplified version of Lemma 5.3. Finally, noting that $n-k < h - n\theta/2 = n\Big(H((1-\mu)\delta) - \theta/2\Big)$, we have

$$L(n,t,\epsilon) \;\ge\; 2^{n(H(\delta)-\theta/4)} - 2^{n(H((1-\mu)\delta)-\theta/2)} \;=\; \Omega\left(2^{n(H(\delta)-\theta)}\right)$$

for every fixed θ and for sufficiently large n (compare with part (ii) of Theorem 5.2).

Proof of Theorem 5.2. (i) The proof is similar to that of Lemma 5.3. Let $L \triangleq L_{DNF}(n,t,\epsilon)$ and let B be an $L \times n$ binary matrix whose rows are the elements of a DNF ϵ-approximation set of size L. Let $k \triangleq \lfloor -\log_2 \epsilon \rfloor$ (≥ 2), $h \triangleq \lfloor \log_2 t \rfloor$, and $m \triangleq k+h+1$, and let C be an $L \times m$ sub-matrix of B consisting (say) of the last m columns of B. We show that C contains among its rows at least $2^m - 2^{h+1} + 1$ distinct vectors of $\{0,1\}^m$.

Assume, to the contrary, that 2^{h+1} distinct row m-vectors are missing from C, and let $\phi_i = \prod_{j=0}^{m-1} y_{i,j}$, $1 \leq i \leq 2^{h+1}$, $y_{i,j} \in \{x_j, \overline{x}_j\}$, be as defined in the proof of Lemma 5.3. Let $\vee$ denote the inclusive OR operation and define the two functions

$$f \;=\; \vee_{i=1}^{2^h} \phi_i \quad \text{and} \quad g \;=\; \vee_{i=2^h+1}^{2^{h+1}} \phi_i \;,$$

both over $\{0,1\}^n$. Note that for $i \neq l$, ϕ_i and ϕ_l (regarded as functions over $\{0,1\}^n$) do not take the value 1 simultaneously at any point of $\{0,1\}^n$. Therefore, the truth table of $\phi = f + g$ is of weight $2^{h+1} \cdot 2^{n-m} = 2^{-k} \cdot 2^n \geq \epsilon \cdot 2^n$. On the other hand, our contrary assumption implies that ϕ takes the zero value at every evaluation point. It follows that the truth tables of f and g coincide at each evaluation point, in spite of the fact that they are ϵ-far.

Substituting $r = m = k+h+1 \leq \log_2(t/\epsilon) + 1 < n$ and $s = 2^{h+1} - 1$ in Lemma 5.2, we obtain,

$$\begin{aligned} L &\geq 2^{k+h-1} \cdot \left(\log_2(n+1-k-h) - \log_2 \left(1 + \frac{(2^{h+1}-1)(n-k-h)}{2^{k+h}} \right) \right) \\ &\geq 2^{k+h-1} \cdot \log_2 \left(\frac{1}{n+1-k-h} + \frac{1}{2^{k-1}} \right)^{-1} \\ &> \tfrac{1}{8} \cdot (t/\epsilon) \cdot \log_2 \left(\frac{1}{n+1-\log_2(t/\epsilon)} + 4\epsilon \right)^{-1} . \end{aligned} \tag{34}$$

(ii) Suppose that $\epsilon < \frac{1}{2}$ and that $t/\epsilon \geq 2^{n-1}$. The idea is to find t' and ϵ' such that $t' \leq t$, $\epsilon \leq \epsilon' < \frac{1}{2}$, and $2^{n-2} \leq t'/\epsilon' < 2^{n-1}$. Having done that, we substitute $k' \triangleq \lfloor -\log_2 \epsilon' \rfloor$ and $h' \triangleq \lfloor \log_2 t' \rfloor$ in (34), thus yielding

$$\begin{aligned} L_{DNF}(n,t,\epsilon) &\geq L_{DNF}(n,t',\epsilon') \\ &\geq 2^{k'+h'-1} \cdot \log_2 \left(\frac{1}{n+1-k'-h'} + \frac{1}{2^{k'-1}} \right)^{-1} \end{aligned}$$

$$\begin{aligned} &\geq 2^{n-4} \cdot \log_2 \Big(\frac{1}{n+1-k'-h'} + \frac{1}{2^{k'-1}} \Big)^{-1} \\ &\geq \Big(\tfrac{1}{16} \cdot \log_2 \tfrac{6}{5} \Big) \cdot 2^n \ . \end{aligned}$$

Indeed, assuming that $n \geq 4$, set $\epsilon' = \max(\epsilon, 2^{2-n})$ $(< \frac{1}{2})$ and $t' = \lceil \epsilon' \cdot 2^{n-2} \rceil$. We thus have $t'/\epsilon' \geq 2^{n-2}$, $\epsilon' \geq \epsilon$ and $t' = \max(1, \lceil \epsilon \cdot 2^{n-2} \rceil) \leq \max(1, \epsilon \cdot 2^{n-1}) \leq t$ (assuming $t \neq 0$). Furthermore,

$$\frac{t'}{\epsilon'} < \frac{\epsilon' \cdot 2^{n-2} + 1}{\epsilon'} = 2^{n-2} + \frac{1}{\epsilon'} \leq 2^{n-1} \ ,$$

as required. □

The discussion in this section can be extended easily to the adaptive scheme as well. In particular, the proof of Lemma 5.3 and, consequently, of Theorem 5.1 and Theorem 5.2, apply also to this case, except that the approximated function is now $2t$-sparse (or $2t$-term DNF).

VI. PROBABILISTIC POLY-TIME APPROXIMATION ALGORITHM

In this section we describe an algorithm for finding ϵ-approximations of t-sparse functions over $\{0,1\}^n$. The algorithm is adaptive and, given a (pre-specified) probability p of failure, its running time is $O\Big((t^2 n^2/\epsilon) \cdot \log(tn/p)\Big)$ bit operations. The approximation is carried out by actually finding monomials of the underlying function which are "short", i.e., each is a product consisting of few variables.

Given a function $f : \{0,1\}^n \to \{0,1\}$, for every $\mathbf{u} \in \{0,1\}^l$, $0 \leq l \leq n$, we associate a function $f_{\mathbf{u}}$ defined as follows: For $l = 0$ we have $f_{[\,]} \stackrel{\Delta}{=} f$, where $[\,]$ denotes the "empty vector". Now, given $f_{\mathbf{u}}$, $\mathbf{u} \in \{0,1\}^l$, $l < n$, we define $f_{[\mathbf{u}\,0]}$ and $f_{[\mathbf{u}\,1]}$ by the unique decomposition

$$f_{\mathbf{u}}(x_{n-l-1}, x_{n-l-2}, \ldots, x_0) = f_{[\mathbf{u}\,0]}(x_{n-l-2}, \ldots, x_0) + x_{n-l-1} \cdot f_{[\mathbf{u}\,1]}(x_{n-l-2}, \ldots, x_0) \quad (35)$$

(see Eq. (2) in Section II). In particular, when $l = n$, $f_{\mathbf{u}}$ is a coefficient of f. We now define a binary directed tree $T(f)$ whose vertices correspond to $f_{\mathbf{u}}$, $\mathbf{u} \in \bigcup_{l=0}^{n} \{0,1\}^l$, and for $l \neq n$ we have edges directed from $f_{\mathbf{u}}$ to the two vertices $f_{[\mathbf{u}\,0]}$ and $f_{[\mathbf{u}\,1]}$. Clearly, $f = f_{[\,]}$ is the root of $T(f)$ and $f_{\mathbf{u}}$, $\mathbf{u} \in \{0,1\}^n$, are its leaves. Now, let W be a binary sub-tree of $T(f)$ growing from the root $f_{[\,]}$ and let $\Lambda(W)$ denote the set of leaves of W. Using the notation

$\mathbf{x^u}$, $\mathbf{u} = [u_{l-1}\, u_{l-2}\, \ldots\, u_0] \in \{0,1\}^l$, for $x_{n-1}^{u_{l-1}} x_{n-2}^{u_{l-2}} \cdots x_{n-l}^{u_0}$, we can write, by (35), the identity

$$f \;=\; \sum_{\mathbf{u}\ \text{s.t.}\ f_{\mathbf{u}} \in \Lambda(W)} \mathbf{x^u} \cdot f_{\mathbf{u}} \tag{36}$$

for any binary sub-tree W of $T(f)$. Note that if f is t-sparse, $\Lambda(W)$ contains at most t leaves which are not identically zero.

The heart of our algorithm (procedure *APPROX* in Figure 3) is a partial *Depth First Search* (*DFS*) on the vertices of $T(f)$, starting at the root $f_{[]}$ and resulting in a binary sub-tree W_{DFS} of $T(f)$. Using randomization, we "weigh" the truth table of $\mathbf{x^u} \cdot f_{\mathbf{u}}$; if its *relative weight* (= the weight of a truth table divided by its size) turns out to be less than $\theta \stackrel{\Delta}{=} \epsilon/t$, we climb back to the father of $f_{\mathbf{u}}$ and, in this case, $f_{\mathbf{u}}$ becomes a so-called *negligible* leaf of W_{DFS}. Otherwise, we continue down into $T(f)$ unless $f_{\mathbf{u}}$ is a leaf of $T(f)$ (and, thus, of W_{DFS}); in this case $f_{\mathbf{u}} \equiv 1$ and, so, we have found a monomial $\mathbf{x^u}$ of f. Such a leaf $f_{\mathbf{u}}$, referred to as a *terminal* leaf, is now added to the approximating function of f. Exploring the sub-tree growing down from a non-negligible vertex $f_{\mathbf{u}}$ in $T(f)$, we then climb back to the father of $f_{\mathbf{u}}$.

The above procedure is implemented in *APPROX* as follows. The input parameters are n, t, ϵ and the allowed probability p of failure. We also assume that there exists a subroutine ("oracle") which, given $\mathbf{z} \in \{0,1\}^n$, returns the value of the underlying function f at $\mathbf{z}$. The output of *APPROX* is an ϵ-approximation $\hat{f}$ of f, represented by its nonzero monomials. The main module in *APPROX* consists of one call to the procedure *DFS* which traverses $T(f)$, inducing the sub-tree W_{DFS}.

The routine *DFS* is recursive, and at each recursion level we regard the vertex $f_{\mathbf{u}}$, $\mathbf{u} \in \{0,1\}^l$ (the input parameter to *DFS*) as a root of the sub-tree growing down from $f_{\mathbf{u}}$. The truth table of $f_{\mathbf{u}}$ is weighed by a procedure named *NONZERO* , which checks whether the relative weight of $\mathbf{x^u} \cdot f_{\mathbf{u}}$ is at least ϵ/t, using not-too-many samples of $f_{\mathbf{u}}$. The specifications of *NONZERO* are as follows:

(a) If the relative weight of $\mathbf{x^u} \cdot f_{\mathbf{u}}$ is at least $\theta = \epsilon/t$, *NONZERO* returns "true" with probability $\geq 1 - q$, where $q \stackrel{\Delta}{=} p/(tn+1)$.

(b) If $f_{\mathbf{u}}$ is identically zero, *NONZERO* always returns "false".

```
procedure APPROX (f; n, t, ε, p) output: f̂ ;
/*
    ε-approximation of a t-sparse Boolean function f over {0,1}^n.
    p is an upper bound on the probability of failure.
    f̂ is the approximating function of f.
    We assume the existence of a subroutine ("oracle") which, given
        z ∈ {0,1}^n, returns f(z).
 */
begin
    f̂ ← 0 ;   θ ← ε/t ;   q ← p/(tn + 1) ;
    /*  n, t, θ, q and f̂ are global for all subsequent routines.  */
    DFS ([ ], 0)
end;

procedure DFS (u, l) ;
/*  Perform DFS starting at the vertex f_u,   u ∈ {0,1}^l.  */
begin
    if NONZERO (u, l) then
        if l = n then     /*  a terminal leaf  */
            f̂ ← f̂ + x^u
        else begin  DFS ([u 0], l + 1) ;   DFS ([u 1], l + 1)  end
end;

procedure NONZERO (u, l) output: "true" / "false" ;
/*  Check whether the relative weight of x^u · f_u is at least θ.  */
begin
    if 2^{-w(u)} < θ then
        NONZERO ← "false"
    else begin
        i ← 0 ;
        N ← ⌈(log q)/(log (1 − θ · 2^{w(u)}))⌉ ;   /*  2^{-w(u)} = θ ⇒ N = 0.  */
        repeat
            i ← i + 1 ;
            choose at random z ∈ {0,1}^{n−l} ;
            b ← Σ_{v s.t. v ⊑ u} f([v z])
        until (i ≥ N) or (b = 1) ;
        NONZERO ← (b = 1)
    end
end;
```

Figure 3: ϵ-approximation of Boolean functions.

(Note that the output of *NONZERO* is unspecified if the relative weight of a nonzero $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ is less than θ). The value assigned to q guarantees an overall probability of failure which is not greater than p. In case *NONZERO* returns a "false" answer on weighing $f_{\mathbf{u}}$ (meaning that the relative weight of $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ is, most likely, smaller than ϵ/t), we return to the father of $f_{\mathbf{u}}$. The same holds also when *NONZERO* returns "true" and $f_{\mathbf{u}}$ is a terminal leaf, in which case the monomial $\mathbf{x}^{\mathbf{u}}$ is added to the approximating function $\hat{f}$. Otherwise, we continue down the tree $T(f)$.

We now consider the implementation of the routine *NONZERO* , in view of the above specifications (a) and (b). Given $\theta = \epsilon/t$, $q = p/(tn+1)$ and the parameter $\mathbf{u} \in \{0,1\}^l$, we pick at random a set P of $N = \left\lceil (\log q)/\log\left(1 - \theta \cdot 2^{\mathrm{w}(\mathbf{u})}\right) \right\rceil$ points in $\{0,1\}^{n-l}$ and then ask for the values of f at the set $Q_{\mathbf{u}} \subseteq \{0,1\}^n$ defined by

$$Q_{\mathbf{u}} = \left\{ [\mathbf{v}\,\mathbf{z}] \;\middle|\; \mathbf{v} \in \{0,1\}^l,\ \mathbf{v} \sqsubseteq \mathbf{u},\ \text{and } \mathbf{z} \in P \right\} .$$

By Remark 2.1, it is easy to verify that the values of $f_{\mathbf{u}}$ at P are given by

$$f_{\mathbf{u}}(\mathbf{z}) = \sum_{\mathbf{v} \text{ s.t. } \mathbf{v} \sqsubseteq \mathbf{u}} f([\mathbf{v}\,\mathbf{z}]), \quad \mathbf{z} \in P . \tag{37}$$

Sampling the truth table of $f_{\mathbf{u}}$ in this manner, *NONZERO* returns "false" if and only if $f_{\mathbf{u}}$ vanishes at P. The choice of N guarantees an error probability $\leq q$ for answering "false" instead of "true". Note that when $2^{-\mathrm{w}(\mathbf{u})} < \theta$, the relative weight of $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ is definitely smaller than θ and, therefore, no sampling is required. In the special case when $2^{-\mathrm{w}(\mathbf{u})} = \theta$ (in which case $N = 0$) we take one sampled value of $f_{\mathbf{u}}$. If this value is zero, the relative weight of $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ is proven to be less than θ and, therefore, $f_{\mathbf{u}}$ becomes a negligible leaf. Otherwise, *NONZERO* returns "true".

We now state the validity and the time complexity of *APPROX* .

Lemma 6.1. *The procedure NONZERO complies with the above specifications (a) and (b).*

Proof. First, note that *NONZERO* returns "true" only when it actually samples a nonzero value of $f_{\mathbf{u}}$. Therefore, when $f_{\mathbf{u}}$ is identically zero, *NONZERO* will always return "false", thus establishing requirement (b). As for requirement (a), assume that the relative weight of $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ is at least θ. This means that the relative weight of $f_{\mathbf{u}}$ is at least $\theta \cdot 2^{\mathrm{w}(\mathbf{u})}$ and, therefore, the probability of having N zero samples of $f_{\mathbf{u}}$ is not greater than $\left(1 - \theta \cdot 2^{\mathrm{w}(\mathbf{u})}\right)^N$,

which, due to the choice of N, is not greater than q (this applies also to the special case when $\theta = 2^{-\mathrm{w}(\mathbf{u})}$, where the truth table of $f_{\mathbf{u}}$ is all-one and, therefore, *NONZERO* will return "true" due to the one sample it makes). □

Lemma 6.2. *(i) DFS traverses at most $2t(n-1)+3$ vertices of $T(f)$; (ii) at most $tn+1$ of these vertices correspond to nonzero functions $f_{\mathbf{u}}$.*

Proof. Every nonzero vertex $f_{\mathbf{u}}$ in $T(f)$ is situated on a path from the root $f_{[]}$ to some nonzero leaf of $T(f)$. Since the number of such nonzero leaves is at most t, the number of nonzero vertices in $T(f)$ is at most $tn+1$, which is also an upper bound on the number of nonzero vertices in any sub-tree of $T(f)$. This proves part (ii) of the lemma.

As for part (i), let $f_{\mathbf{u}}$ be an inner vertex in W_{DFS}, i.e., a vertex which is not a leaf. Clearly, $f_{\mathbf{u}} \neq 0$, or else, by Lemma 6.1, *NONZERO* must return "false" on weighing $f_{\mathbf{u}}$, making $f_{\mathbf{u}}$ a negligible leaf, rather than an inner leaf, of W_{DFS}. Therefore, the inner vertices of W_{DFS} are all nonzero inner vertices of $T(f)$, the number of which is at most $r = t(n-1)+1$. Hence, the total number of vertices in W_{DFS} is at most $2r+1 = 2t(n-1)+3$. □

Lemma 6.3. *The procedure APPROX returns, with probability $\geq 1-p$, an ϵ-approximation $\hat{f}$ for any n-variable t-sparse function f.*

Proof. First, note that *NONZERO* might return a wrong answer ("false" instead of "true") only at a vertex $f_{\mathbf{u}}$ whose corresponding function $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ is of relative weight $\geq \theta = \epsilon/t$. By Lemma 6.2, the number of such vertices is at most $tn+1$ and, therefore, the probability of the answers of *NONZERO* being all correct is at least $(1-q)^{tn+1} \geq 1-(tn+1)q = 1-p$.

Consider now an execution of *APPROX* where all the answers of *NONZERO* are correct. Let $\widehat{\Lambda}$ denote the set of terminal leaves in W_{DFS}. The approximating function, computed by *APPROX* , is given by

$$\hat{f} \;=\; \sum_{\mathbf{u} \text{ s.t. } f_{\mathbf{u}} \in \widehat{\Lambda}} \mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}} \;.$$

Now, let $\widetilde{\Lambda} \triangleq \Lambda(W_{DFS}) - \widehat{\Lambda}$ denote the set of all negligible leaves of W_{DFS} and define the function $\tilde{f}$ by

$$\tilde{f} \;=\; \sum_{\mathbf{u} \text{ s.t. } f_{\mathbf{u}} \in \widetilde{\Lambda}} \mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}} \;. \tag{38}$$

By (36) we have $f = \hat{f} + \tilde{f}$. It suffices to show that the relative weight of $\tilde{f}$ if less than ϵ.

Let $f_{\mathbf{u}} \in \widetilde{\Lambda}$ be a negligible leaf encountered during any of the recursion levels of *DFS* . If $f_{\mathbf{u}}$ is identically zero, then the contribution of $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ to $\tilde{f}$ is zero. On the other hand, if $f_{\mathbf{u}} \neq 0$, then, assuming the answers of *NONZERO* being all correct, the relative weight of $\mathbf{x}^{\mathbf{u}} \cdot f_{\mathbf{u}}$ is less than ϵ/t. Now, the number of nonzero leaves in W_{DFS} is bounded from above by the number of nonzero leaves in $T(f)$ which, in turn, is upper-bounded by t. In particular, the number of nonzero negligible leaves in W_{DFS} is at most t and, therefore, the relative weight of $\tilde{f}$ is less than $t \cdot (\epsilon/t) = \epsilon$. □

Theorem 6.1. *The ϵ-approximation of any n-variable t-sparse function f can be performed, with probability $\leq p$ of failure, by querying $O\Big((t^2 n/\epsilon) \cdot \log(tn/p)\Big)$ values of f, involving $O(n)$ bit operations per query.*

Proof. The number of queries issued at each call to *NONZERO* is given by

$$|Q_{\mathbf{u}}| = 2^{\mathrm{w}(\mathbf{u})} \cdot N = 2^{\mathrm{w}(\mathbf{u})} \cdot \Big\lceil (\log\, q)/\log(1 - \theta \cdot 2^{\mathrm{w}(\mathbf{u})}) \Big\rceil = O\Big((1/\theta) \cdot \log(1/q)\Big) \ . \tag{39}$$

Now, by Lemma 6.2, the number of calls to *NONZERO* is at most $2t(n-1)+3$. Substituting $\theta = \epsilon/t$ and $q = p/(tn+1)$ in (39) yields a total number of $O\Big((t^2 n/\epsilon) \cdot \log(tn/p)\Big)$ queries. Finally, it is easy to verify that each query in *NONZERO* involves $O(n)$ bit operations. □

Finally, we show how *APPROX* can be used to approximate any t-sparse function, without knowing the value of t in advance. In such a scheme, we perform $M \leq \lceil \log_2 t \rceil$ iterations of *APPROX* ; the m-th iteration ($m = 1, 2, \ldots, M$) is executed with $t_m = 2^m$, and the global variables of *APPROX* are set to $\theta_m \leftarrow \epsilon/(t_m(n-1)+2)$ and $q_m \leftarrow p/(t_m n + 1)$. Let $W_{DFS}(m)$ denote the tree traversed by *DFS* during the m-th iteration, and let $W^*_{DFS}(m)$ denote the sub-tree of $W_{DFS}(m)$ obtained by deleting its negligible leaves. Note that each leaf of $W^*_{DFS}(m)$ corresponds to some nonzero function $f_{\mathbf{u}}$ and, therefore, the number of such leaves cannot exceed t. Now, the iteration process terminates when, for the first time, t_m becomes greater than or equal to the number of leaves of $W^*_{DFS}(m)$ (in which case $m = M \leq \lceil \log_2 t \rceil$). By arguments similar to those given in the proof of Lemma 6.2, the number of inner vertices in $W_{DFS}(M)$ is not greater than $t_M(n-1)+1$ and, therefore, the number of negligible leaves in $W_{DFS}(M)$ is at most $t_M(n-1)+2$. It thus follows that the relative weight of $\tilde{f}_M$, defined by (38) for the M-th iteration, is less than ϵ. Hence, the output $\hat{f}_M$ of the last iteration of *APPROX* is, with probability $\leq p$ of failure, an ϵ-approximation of f. It can be easily verified that the above process involves a total of $O\Big((t^2 n^2/\epsilon) \cdot \log(tn/p)\Big)$

queries.

Finding the t-term DNF counterpart of the above procedure remains still an open problem. The approximation problem in the t-term DNF case has been dealt with in the literature also in a wider context, namely, when the approximation factor ϵ is measured according to an arbitrary distribution induced on $\{0,1\}^n$ (and not necessarily according to the uniform distribution). For related work see, for instance, [8][11][13][14][15].

ACKNOWLEDGMENT

The authors wish to thank Noga Alon for the helpful discussions, and the anonymous referees for their valuable suggestions which helped improving the presentation of this paper. In particular, the approximation procedure for the case where t is unknown, discussed in Section VI, is due to one of the referees.

REFERENCES

[1] D. Angluin, *Learning k-term DNF formulas using queries and counterexamples,* Yale University, Dept. of Computer Science, RR-559 (1987).

[2] D. Angluin, C.H. Smith, *Inductive inference: theory and methods, Computing Surveys,* Vol. 15, 1983, pp. 237-269.

[3] M. Ben-Or, P. Tiwari, *A deterministic algorithm for sparse multivariate polynomial interpolation, 20-th Annual ACM Symp. on Theory of Computing,* 1988, pp. 301-309.

[4] M. Clausen, A. Dress, J. Grabmeier, M. Karpinski, *On zero-testing and interpolation of k-sparse multivariate polynomials over finite fields,* Institut für Informatik, Universität Bonn, Report No. 8522-CS, 1988.

[5] M. Garey, D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness,* W.H. Freeman, San Francisco, 1979.

[6] D.Y. Grigoriev, M. Karpinski, M.F. Singer, *Fast parallel algorithms for sparse multivariate polynomial interpolation over finite fields,* Institut für Informatik, Universität Bonn, Report No. 8523-CS, 1988.

[7] L. Hellerstein, M. Warmuth, Private communication.

[8] M. Kearns, M. Li, L. Pitt, L.G. Valiant, *On the learnability of Boolean formulae, 19-th Annual ACM Symp. on Theory of Computing,* 1987, pp. 285-295.

[9] F.J. MacWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes,* North-Holland, Amsterdam, 1977.

[10] R.J. McEliece, *The Theory of Information and Coding,* Addison-Wesley, Reading, Massachusetts, 1977.

[11] L. Pitt, L.G. Valiant, *Computational limitations on learning from examples,* Harvard University, Aiken Computation Laboratory, TR-05-86, 1986.

[12] G. Seroussi, N.H. Bshouty, *Vector sets for exhaustive testing of logic circuits, IEEE Trans. Inform. Theory,* Vol. IT-34, 1988, pp. 513-522.

[13] L.G. Valiant, *A theory of the learnable, Comm. ACM,* Vol. 27, 1984, pp. 1134-1142.

[14] L.G. Valiant, *Learning disjunctions of conjunctions, Proceedings of the 9-th IJCAI,* Vol. 1, 1985, pp. 560-566.

[15] L.G. Valiant, *Deductive learning,* Aiken Computational Laboratory, Harvard University, 1984.