Electronic Colloquium on Computational Complexity, Revision 1 of Report No. 127 (2005)

Hardness of Approximate Two-level Logic Minimization and PAC Learning with Membership Queries

Vitaly Feldman[*]
IBM Almaden Research Center
650 Harry rd.
San Jose, CA 95120
vitaly@post.harvard.edu

Abstract

Producing a small DNF expression consistent with given data is a classical problem in computer science that occurs in a number of forms and has numerous applications. We consider two standard variants of this problem. The first one is *two-level logic minimization* or finding a minimum DNF formula consistent with a given complete truth table (TT-MinDNF). This problem was formulated by Quine in 1952 and has been since one of the key problems in logic design. It was proved NP-complete by Masek in 1979. The best known polynomial approximation algorithm is based on a reduction to the SET-COVER problem and produces a DNF formula of size $O(d \cdot \mathrm{OPT})$, where d is the number of variables. We prove that TT-MinDNF is NP-hard to approximate within d^{γ} for some constant $\gamma > 0$, establishing the first inapproximability result for the problem. The other DNF minimization problem we consider is PAC learning of DNF expressions when the learning algorithm must output a DNF expression as its hypothesis (referred to as *proper* learning). We prove that DNF expressions are NP-hard to PAC learn properly even when the learner has access to membership queries, thereby answering a long-standing open question due to Valiant [40]. Finally, we provide a concrete connection between these variants of DNF minimization problem. Specifically, we prove that inapproximability of TT-MinDNF implies hardness results for restricted proper learning of DNF expressions with membership queries even when learning with respect to the uniform distribution only.

[*]Work done while the author was at Harvard University supported by grants from the National Science Foundation NSF-CCR-0310882, NSF-CCF-0432037, and NSF-CCF-0427129.

ISSN 1433-8092

1 Introduction

The problem of finding a minimal-size disjunctive normal form expression consistent with a given truth table (TT-MinDNF) is one of the oldest problems in computer science. It was formulated by the famous logician and philosopher Willard Van Quine in his work on mathematical logic [32, 33]. His algorithm for simplifying logical steps was also discovered in 1956 by Edward McCluskey in the context of circuit design [27]. Besides its important role in circuit design (in particular, two-level and multi-level logic synthesis for VLSI design of ASICs and Programmable Gate Arrays [11]) the problem has more recently appeared in reliability analysis [10], IP routing table compaction [24], and high-dimensional data representation [1]. This array of applications has led to an ongoing effort by many researchers to seek efficient heuristic and exact minimization procedures. We direct the interested reader to [11] for an overview of a large number of publications and some software tools. In the original Quine-McCluskey algorithm and in most of the later approaches, after a number of simplification steps the problem is reduced to an instance of the classical SET-COVER problem. Then, either an exact solution is found via the brute-force search, or an approximate solution is found using a certain heuristic. In the former case the size of the search space is not theoretically analyzed and in the latter no guarantees on the quality (i.e. size) of the output are given (both are usually measured empirically).

Far less work has been done on the theoretical side of this problem. Gimpel [17] and Paul [30] showed that Quine-McCluskey method can produce instances of SET-COVER that are NP-hard to solve. Then, in 1979, the full truth table version was proven NP-complete by Masek [26] (his manuscript was not published but the proof can be found in surveys by Czort [12] and Umans *et al.* [38]). Inapproximability results are only known for a generalization of TT-MinDNF that allows "don't care" values in the truth table (i.e., the truth table is partial). Allender *et al.* prove that this problem (we denote it by PTT-MinDNF) is NP-hard to approximate within any constant factor and cannot be approximated within $\log d$ factor unless $\mathsf{NP} \subseteq \mathsf{RTIME}(n^{polylog(n)})$, where d is the number of variables [4]. Using Gimpel's reduction from PTT-MinDNF to TT-MinDNF they also produced a simpler proof (than Masek's) for NP-hardness of TT-MinDNF.

On the approximation side the only known efficient approximating algorithm is the one resulting from using the greedy algorithm to solve the SET-COVER instance obtained in Quine-McCluskey algorithm [9]. It gives $\ln 2^d = O(d)$ approximation factor.

In this paper we present the first result on hardness of approximating TT-MinDNF. More specifically, we prove the following theorem.

Theorem 1.1 *There exists a constant $\gamma > 0$ such that it is NP-hard to approximate TT-MinDNF to within a factor d^γ, where d is the number of variables of the TT-MinDNF instance.*

This result implies that the approximation factor achieved by the greedy algorithm is at most polynomially larger than the best possible. In addition we prove the first hardness of approximation result for a natural restriction of SET-COVER problem, in which the ground set is $\{0,1\}^d$ and all subsets are subcubes (see Section 2 for a formal definition).

The unpublished results of Allender *et al.* were recently substantially strengthened by the same authors and Paul McCabe [3]. They prove NP-hardness of approximating TT-MinDNF within any constant factor and also show how to get the d^γ factor under the assumption that $\mathsf{NP} \not\subseteq \mathsf{DTIME}(n^{polylog(n)})$. Their independent work is based on a similar approach.

Learning is another context where finding a small DNF formula consistent (or almost) with the given data is a fundamental problem. The problem was formulated by Leslie Valiant in his seminal paper introducing the PAC model of learning [40] and has been the subject of numerous subsequent works. A number of questions related to PAC learning of DNF expressions were posed by Valiant [40, 41]. Specifically, he asked whether DNF expressions are learnable from random examples with or without the use of the membership query (MQ) oracle. Valiant's original definition required that the learning algorithm output a DNF expression but this restriction was later relaxed to any efficiently-evaluatable hypothesis with the stricter version being referred to as *proper* learning. All these variants of the DNF learning question remained open until a recent result by Alekhnovich *et al.* that established NP-hardness of the hardest variant: proper learning from random examples only [2]. Building on their proof, we resolve one more of Valiant's questions:

Theorem 1.2 (*informal)* *If* $\mathsf{NP} \neq \mathsf{RP}$ *then there is no polynomial-time PAC learning algorithm for DNF expressions that outputs a DNF expression even when the learning algorithm has access to the membership oracle.*

Besides, we observe that hardness of TT-MinDNF implies hardness of *strongly proper* learning of DNF expressions with MQs even with respect to the uniform distribution, where strongly proper means that the size (number of terms) of a hypothesis has to be upper-bounded by the DNF-size of the target function. Our inapproximability result then translates to hardness even when the size of a hypothesis is $O(\log^\gamma n)$ times larger than the size of the target. We note that, as proved by Jackson, unrestricted DNF expressions are learnable non-properly in this strong model [20], and hence our result highlights the importance of knowledge representation in this model.

Access to membership queries plays an instrumental role in numerous learning algorithms (many of which are proper), but hardness results for learning with MQs are still very scarce. Our results are the first to show that PAC learning (of any class) can be NP-hard even when MQs are available.

1.1 Relation to Other Work

Besides the results that we have already mentioned, one of the most significant results in DNF minimization is Umans' proof that finding a minimum DNF formula for a function given by a DNF formula (also called finding a minimum equivalent DNF and denoted MinEquDNF) is Σ_2^p-hard to approximate within N^γ for some constant $\gamma > 0$, where N is the size of the given DNF formula [37]. Despite the same goal in both problems the difference in input makes the nature of the problem (and, eventually, the proof techniques) very different. In particular, the gaps differ exponentially in terms of the size of hard instances. Hardness results for some other variants of DNF minimization can be found in a survey by Umans *et al.* [38].

Initial hardness results for properly learning DNF formulae due to Pitt and Valiant [31] show that unless $\mathsf{RP} = \mathsf{NP}$, k-term DNF formulae over n variables are not learnable by $2k$-term DNF. These results were strengthened by Nock *et al.* [29] who proved similar hardness even when learning by formulas of size $k^\alpha n^\beta$ (where $\alpha \leq 2$ and β is any constant). Finally, Alekhnovich *et al.* removed any bounds on the size of the hypothesis (other than those naturally imposed by the polynomial running time of the learning algorithm) [2]. Angluin and Kharitonov prove that if non-uniform one-way functions exist then MQs do not help predicting DNF formulae [5]. However, their reduction does not preserve the representation of a hypothesis and therefore cannot be combined with the result by Alekhnovich *et al.* to obtain hardness of proper learning with MQs.

Hardness results for learning of DNF expressions with MQs are only known for the *exact* model of learning (which is weaker than PAC learning) and only for strong proper learning (or slight relaxations similar to the one we prove for PAC learning with respect to the uniform distribution). The strongest results in this model are due to Hellerstein and Raghavan [19] and are based on information-theoretic hardness.

For proper PAC learning without MQs a number of hardness results are known for several other representations [8, 18, 22, 2].

Our hardness results for learning DNF expressions are contrasted by the fact that monotone DNF expressions are known to be strongly properly PAC learnable with MQs [40]. In addition to that, DNF expressions with k terms are known to be learnable by DNF expressions with 2^k terms when MQs are available [7]. It is also interesting to note that known non-trivial algorithms for learning unrestricted DNF formulae (running in time $2^{\tilde{O}(n^{\frac{1}{3}})}$ [23] and in time $2^{\tilde{O}(\sqrt{n})}$ with DNF hypotheses [2]) use only random examples and it is unknown whether they could be sped-up by using MQs.

When learning with respect to the uniform distribution DNF expressions are known to be PAC learnable (non-properly) with MQs [20] and PAC learnable properly in time $n^{O(\log(s/\epsilon))}$ [42].

1.2 Outline and Organization

The proof of the TT-MinDNF hardness result has two key components. The first one is a reduction from a more general problem of covering a subset of the Boolean hypercube with a given set of subcubes (we denote it by PHC-COVER) to TT-MinDNF. PHC-COVER can be seen as a geometric version of the general SET-COVER problem. The second component of the proof is a reduction from a multi-prover proof system with certain simple properties to PHC-COVER. This reduction follows the key ideas of the inapproximability result for SET-COVER by Lund and Yannakakis [25] and its generalization by Bellare *et al.* [6]. Finally, a low-error PCP by Raz and Safra [34] is used to obtain a multi-prover proof system with the desired properties, yielding the inapproximability result for TT-MinDNF. The low-error PCP of Raz and Safra was used in a similar way to obtain hardness of approximating within $\Omega(\log n)$ for SET-COVER under the assumption that $\mathsf{P} \neq \mathsf{NP}$ [34].

Besides the main reduction in Section 4.1, we show a reduction from hypergraph vertex cover problem to PHC-COVER. The reduction is based on families of sets in which none of the sets is covered by k others. This reduction together with a recent result by Dinur *et al.* [13] implies the same inapproximability result for TT-MinDNF under a stronger assumption $\mathsf{NP} \not\subseteq \mathsf{DTIME}(n^{\log(n)})$. This reduction is more direct and simple than both the reduction given in Section 4.1 and the reduction of Allender *et al.* [3] (which gives the same result).

The hardness of learning DNF expressions result is based on the proof by Alekhnovich *et al.* [2] that is, in turn, based on hardness of approximating the chromatic number of a graph by Feige and Kilian [16]. In essence, we augment the reduction from coloring to finding a small consistent DNF by providing a way to efficiently define the value of the target function on the whole hypercube without revealing any additional information about coloring and without changing the DNF-size of the target function substantially. This allows for simulation of the membership oracle in the DNF hardness reduction.

The rest of the paper is organized as follows. In Section 3 we show that to PHC-COVER can be reduced to TT-MinDNF in an approximation-preserving way. Then, in Section 4, we give a reduction from the low-error PCP by Raz and Safra [34] to PHC-COVER, establishing the desired hardness of approximation result. In Section 5 we prove the above-mentioned hardness results for proper learning with MQs.

2 Preliminaries

A Boolean partial function is a function $f : \{0,1\}^d \rightarrow \{0,1,*\}$. We say that a Boolean function g is consistent with a partial function f, if for every $a \in \{0,1\}^d$ such that $f(a) \neq *$, $g(a) = f(a)$. A subcube of a Boolean hypercube is a set $I_1 \times I_2 \times \cdots \times I_d$ where for each j, $I_j \subseteq \{0,1\}$. We identify each subcube with a term whose satisfying assignments are exactly the elements of the subcube.

The *size* of a DNF formula is the number of terms in it. The DNF-size of a function is the size of a minimum DNF formula equal to the function. Given the truth table of a function f the problem of finding the DNF-size of f is denoted TT-MinDNF. When f is a partial function the problem of finding the size of a minimum DNF consistent with f is referred to as PTT-MinDNF.

The problem of finding the size of a minimum cover of the d-dimensional Boolean hypercube with subcubes represented by the terms in $\mathcal{T} = \{T_i\}_{i=1}^m$ is referred to as HC-COVER. We also consider the following generalization of HC-COVER. Given a set of terms as above and a set of points $S \subseteq \{0,1\}^d$ find the size of a minimum cover of S by terms in $\mathcal{T}$. We refer to this generalized version as PHC-COVER. We say that PTT-MinDNF$(f) = C$ (HC-COVER$(\mathcal{T}) = C$, or PHC-COVER$(S, \mathcal{T}) = C$) if the size of a minimum DNF formula consistent with f (or, respectively, a cover for an instance $\mathcal{T}$ or $(S, \mathcal{T})$) equals C.

In all the above problems, we assume that the input is of size poly(2^d) (it cannot be larger as there are 3^d different terms). For PHC-COVER and HC-COVER the input can, in certain situations, be represented more concisely. However, for consistency with the definition of the usual set cover problem, we assume that all the 2^d points of the cube are given explicitly as part of the input. Hardness results for this setting imply hardness results for more concise representations of the same problem.

We use a dot '$\cdot$' to denote concatenation of bits and bit vectors. Unless defined otherwise we use a subscript to refer to an individual coordinate of a vector and use $[n]$ to denote the set $\{1,2,\ldots,n\}$. Let $\texttt{par}()$ denote the parity function defined for any bit vector. For any Boolean variable v and $b \in \{0,1\}$, let $\ell_v(b) = v$, if $b = 1$ and $\ell_v(b) = \bar{v}$, if $b = 0$. Similarly, for a vector of variables $w \in V^r$ and a vector $a \in \{0,1\}^r$, we define $\texttt{eq}(w,a) = \wedge_{i \leq r} \ell_{w_i}(a_i)$, or simply the term that checks if variables of w are set to a.

Besides the usual disjunctions and conjunctions we consider *threshold* or *halfspace* gates equal to $\texttt{sign}(\sum_i w_i x_i - \theta)$ for some real-valued $w_1, \ldots, w_n, \theta$. AND-of-thresholds (OR-of-thresholds) formula is a two-level formula with an AND (respectively OR) gate at the top (output) level and thresholds at the bottom level. The size of such a formula is the number of thresholds gates in it.

2.1 Learning Model

Our learning model is Valiant's well-known Probably Approximately Correct (PAC) learning model [40]. In this model, for a concept c and a distribution D over domain X an *example oracle* EX(c, D) is an oracle that upon request returns an example $\langle x, c(x) \rangle$, where x is chosen randomly with respect to D independently of any previous examples. The membership oracle MEM(c) is the oracle that given any point $x \in X$ returns the value $c(x)$. For $\epsilon \geq 0$ we say that function g ϵ-approximates function f with respect to distribution D if $\mathbf{Pr}_D[f(x) = g(x)] \geq 1 - \epsilon$. We say that an algorithm $\mathcal{A}$ efficiently learns concept class C if for every $\epsilon > 0$, $\delta > 0$, n, $c \in \mathcal{C}$, and distribution D over X, $\mathcal{A}(n, \epsilon, \delta)$, runs in time polynomial in n, $1/\delta$, $1/\epsilon$, $\texttt{size(c)}$ and, with probability at least $1 - \delta$, outputs an efficiently computable hypothesis h that ϵ-approximates c. Here $\texttt{size(c)}$ is the size of c in the representation associated with $\mathcal{C}$ (e.g. number of terms if the representation is DNF). In the basic PAC model $\mathcal{A}$ is allowed to use only random example oracle $\mathrm{EX}(c, D)$. We denote the model in which the learner also has access to $\mathrm{MEM}(c)$ by PAC+MQ.

If the hypothesis is output in the representation associated with C, the algorithm $\mathcal{A}$ is called *proper*. If, in addition, $\texttt{size(h)} \leq \texttt{size(c)}$, then the learning algorithm is called *strongly* proper. The distribution specific version of this model requires the learning algorithm to succeed only with respect to some specific distribution (in our case it will be the uniform distribution).

3 Hypercube Reductions

Below we show that the covering problems defined in the previous section have similar approximation complexity by describing efficient reductions from PHC-COVER to PTT-MinDNF, from PTT-MinDNF to TT-MinDNF, and from TT-MinDNF to HC-COVER. Our reductions preserve the approximation ratio and increase the number of variables by a small constant factor.

3.1 From PHC-COVER to PTT-MinDNF

It can be easily seen that PTT-MinDNF is an instance of PHC-COVER. For the other direction our reduction converts an instance of PHC-COVER given by a set $S \subseteq \{0,1\}^d$ and a set of terms $\mathcal{T}$, to an instance of PTT-MinDNF given by a function f where each element of $\mathcal{T}$ corresponds to a *prime implicant* of f. A prime implicant of a function f is a term T such that T is consistent with f (that is it does not accept points where f equals 0) and is not covered properly by another term consistent with f. Any DNF formula consistent with f can always be easily converted to a DNF formula of the same size that includes only prime implicants of f. Therefore, any DNF formula for f produced by our reduction corresponds to a cover of S by terms from $\mathcal{T}$. We now provide the details of this mapping.

Theorem 3.1 *There exists a polynomial-time algorithm that given an instance $(S, \mathcal{T})$ of PHC-COVER over d variables produces the truth table of partial function f over $2d$ variables such that $(S, \mathcal{T})$ has a cover of size C if and only if there exists a C-term DNF formula consistent with f.*

Proof: For a point $x \in \{0,1\}^d$, let $p[x]$ denote a point in $\{0,1\}^{2d}$ equal to $x \cdot \bar{x}$ (that is, x on first d coordinates and the bit complement of x on coordinates from $d+1$ to $2d$). For a term T over d variables, let $p[T]$ denote a term over $2d$ variables in which all the positive literals are the same as in T while each negative literal $\bar{x}_i$ is replaced by literal x_{d+i}. Let $g(y) = \bigvee_{T \in \mathcal{T}} p[T](y)$. Then we map $(S, \mathcal{T})$ to the instance of PTT-MinDNF given by the following function:

$$f(y) = \begin{cases} 0 & \text{if } g(y) = 0 \\ 1 & \text{if } y = p[x] \text{ and } x \in S \\ * & \text{otherwise} \end{cases}$$

Let $\mathcal{S} \subseteq \mathcal{T}$ be a set of C terms such that $S \subseteq \cup_{T \in \mathcal{S}} T$. We claim that $h(y) = \bigvee_{T \in \mathcal{S}} p[T](y)$ is consistent with f. Let y be a point in $\{0,1\}^{2d}$. If $f(y) = 0$, then $g(y) = 0$ and so $h(y) = 0$. If

$f(y) = 1$ then there exists x such that $y = p[x]$ and $x \in S$. Therefore, there exists $T \in \mathcal{S}$ such that $T(x) = 1$, which is equivalent to $p[T](p[x]) = 1$. In particular, $h(y) = 1$, which completes the proof of the claim.

For the other direction, let $h = \bigvee_{Z \in \mathcal{Z}} Z$ be a C-term DNF formula consistent with f. For a term $Z \in \mathcal{Z}$, let y be the point with the minimum number of 1's accepted by Z. By the consistency with f, we get that $f(y) \neq 0$ and hence $g(y) = 1$. Therefore let $m(Z)$ be a term of g that covers y and let $T_Z \in \mathcal{T}$ be some term for which $m(Z) = p[T_Z]$. We claim that $Z \subseteq m(Z)$. If for a point z, $Z(z) = 1$ then for every $i \leq 2d$, if $z_i = 0$ then $y_i = 0$. This is true since if $z_i = 0$ then Z does not include literal x_i and, therefore, by the minimality of y, $y_i = 0$. The term $m(Z) = p[T_Z]$ is monotone and, therefore, if it covers y then it covers z. This implies the claim that $Z \subseteq m(Z)$.

Define $\mathcal{T}' = \{T_Z \mid Z \in \mathcal{Z}\}$. If $x \in S$, then $f(p[x]) = 1$ and therefore, there exists $Z \in \mathcal{Z}$ such that $Z(p[x]) = 1$. This, in turn, implies that $T_Z(x) = 1$, that is, $\mathcal{T}'$ is a set of C subsets from $\mathcal{T}$ that covers S. □

3.2 From PTT-MinDNF to TT-MinDNF

The next step is an approximation preserving reduction from a partially-specified truth table to a fully-specified one. A part of this reduction is based on Gimpel's reduction from partially to fully specified truth-table [17].

Theorem 3.2 *There exists an algorithm that given the truth table of a partial function f on d variables and an integer $r \geq 1$ produces the truth table of partial function g over $d + r + 2$ variables such that there exists a C-term DNF consistent with f if and only if there exists $(2^{r-1}C + |f^{-1}(*)|)$-term DNF formula equal to g. The algorithm runs in time $2^{O(r+d)}$.*

Proof: The reduction has two components. The first component is Gimpel's reduction [17]. It converts f to a fully-specified function that has a distinct prime implicant for each point x where f equals $*$ thereby forcing any consistent DNF to include a term for every $*$ of the original function. The addition of new terms does not preserve approximation factors and therefore the second component replicates f 2^{r-1} times to ensure that the size of the cover is still dominated by the original problem (for large enough r). For a vector in $\{0,1\}^{d+r+2}$, we refer to its first d coordinates as $x_1, \ldots, x_d$; its next r variables as $y_1, \ldots, y_r$; and its last two variables as z_1, z_2. We define Boolean function g over $\{0,1\}^{d+r+2}$ as follows:

$$g(xyz) = \begin{cases} \mathtt{par}(y) & \text{if } f(x) = 1 \text{ and } z = 11 \\ 1 & \text{if } f(x) = * \text{ and} \\ & (z = \mathtt{par}(x) \cdot \neg\mathtt{par}(x) \text{ or } z = 11) \\ 0 & \text{otherwise} \end{cases}$$

Let $S = f^{-1}(*)$. We claim that there exists a C-term DNF consistent with f if and only if there exists a $(2^{r-1}C + |S|)$-term DNF equal to g. For the simpler direction, let $\mathcal{S}$ be a set of C terms such that $h(x) = \bigvee_{T \in \mathcal{S}} T(x)$ is consistent with f. For $b \in \{0,1\}$ we define $\mathtt{sw\text{-}z}(b) = z_{2-b}$ (that is, $\mathtt{sw\text{-}z}$ switches between z_1 and z_2 according to b). We claim that

$$g(xyz) \equiv (h(x) \wedge \mathtt{par}(y) \wedge z_1 \wedge z_2) \bigvee ((f(x) = *) \wedge \mathtt{sw\text{-}z}(\mathtt{par}(x)))\,.$$

By definition, the expression $R(xyz) \equiv (f(x) = *) \wedge \mathtt{sw\text{-}z}(\mathtt{par}(x))$ equals to $g(xyz)$ on all points with $z \neq 11$. In addition, for $z = 11$, $R(xy \cdot 11) \equiv (f(x) = *)$. The expression $L(xyz) =$

$h(x) \wedge \mathtt{par}(y) \wedge z_1 \wedge z_2$ only covers points for which $z = 11$ and $L(xy \cdot 11) = h(x) \wedge \mathtt{par}(y)$. Therefore, by the consistency of h with f, $L(xy \cdot 11)$ equals to $\mathtt{par}(y)$ when $f(x) = 1$ and does not cover any points for which $f(x) = 0$. Altogether, $g(xyz) \equiv L(xyz) \vee R(xyz)$.

Expressions $L(xyz)$ and $R(xyz)$ are not in DNF. To convert them to DNF we note that

$$\mathtt{par}(y) \equiv \bigvee_{a \in \{0,1\}^r, \mathtt{par}(a)=1} \mathtt{eq}(y, a)$$

and

$$(f(x) = *) \equiv \bigvee_{a \in S} \mathtt{eq}(x, a) \ .$$

Therefore

$$g(xyz) \equiv \left(\bigvee_{T \in \mathcal{S}, a \in \{0,1\}^r, \mathtt{par}(a)=1} T \wedge \mathtt{eq}(y, a) \wedge z_1 \wedge z_2 \right) \\ \bigvee \left(\bigvee_{a \in S} \mathtt{eq}(x, a) \wedge \mathtt{sw\text{-}z}(\mathtt{par}(a)) \right) ,$$

that is, g has a DNF expression with $C2^{r-1} + |S|$ terms.

For the other direction, let $\mathcal{T}$ be a set of $C2^{r-1} + |S|$ terms such that $g(xyz) = \bigvee_{T \in \mathcal{T}} T(xyz)$. For each $a \in S$, let $\tau_a \in \mathcal{T}$ be a term that accepts point

$$p(a) = a \cdot 0^r \cdot \mathtt{par}(a) \cdot \neg\mathtt{par}(a) \ .$$

We first prove that τ_a contains all the literals of $\mathtt{eq}(x, a)$. If τ_a does not contain literal $\ell_{x_i}(a_i)$, then let a^i be the point a with the i-th bit negated. Clearly τ_a will also accept the point $a^i \cdot 0^r \cdot \mathtt{par}(a) \cdot \neg\mathtt{par}(a)$. But this contradicts the consistency with g, since $\mathtt{par}(a) = \neg\mathtt{par}(a^i)$. It follows that for each $a \in S$, there is a *distinct* term in $\mathcal{S}$ that can only accept points with x part in S. We denote this set of terms by $\mathcal{T}^*$.

Now let $D = \{p \mid p \in \{0,1\}^r, \mathtt{par}(p) = 1\}$, p be any point in D and a be any point such that $f(a) = 1$. Then, by definition of g, there exists a term $\tau_{p,a}$ that accepts the point $a \cdot p \cdot 11$. We claim that $\tau_{p,a}$ contains all the literals of $\mathtt{eq}(y, p)$. If $\tau_{p,a}$ does not contain literal $\ell_{y_i}(p_i)$, then let p^i be the point p with the i-th bit negated. Clearly $\tau_{p,a}$ will also accept the point $a \cdot p^i \cdot 11$. But this contradicts the consistency with g, since $g(a \cdot p^i \cdot 11) = \mathtt{par}(p^i) = 0$. Now let $\mathcal{T}_p = \{\tau_{p,a} \mid f(a) = 1\}$ and let $h_p(x) = \bigvee_{T \in \mathcal{T}_p} T(x \cdot p \cdot 11)$. We claim that $h_p(x)$ is consistent with f. This is true since if $f(a) = 1$ then $\tau_{p,a} \in \mathcal{T}_p$ and $\tau_{p,a}(a \cdot p \cdot 11) = 1$. If $f(a) = 0$ then $g(a \cdot p \cdot 11) = 0$ and since $\mathcal{T}_p \subseteq \mathcal{T}$ then no term in $\mathcal{T}_p$ can accept point $a \cdot p \cdot 11$. As we have shown, all the $\mathcal{T}_p$'s for $p \in D$ are disjoint and they are clearly disjoint from $\mathcal{T}^*$ (since $0^r \notin D$). Therefore $|\mathcal{T}| \geq |S| + \sum_{p \in D} |\mathcal{T}_p|$. As $|D| = 2^{r-1}$ we get that there exists p such that $|\mathcal{T}_p| \leq C$ and hence h_p is a C-term DNF formula consistent with f. □

By a suitable choice of r in Theorem 3.1 one easily obtains the following corollary (the proof is omitted for brevity):

Corollary 3.3 *If TT-MinDNF can be approximated within $h(d)$ in time $t(d)$ then PTT-MinDNF can be approximated within $h(2d + \log d) + 1$ in time $t(2d + \log d) + 2^{O(d)}$.*

3.3 From TT-MinDNF to HC-COVER

We now give a simple reduction proving that finding a minimum cover of the whole hypercube by terms from a restricted set $\mathcal{T}$ is not substantially easier than finding a minimum cover of a subset of the hypercube. Note that this reduction is not required as a step in our main result and is provided to extend our hardness of approximation results to HC-COVER.

Theorem 3.4 *There exists an algorithm that, given the truth table of a function f on d variables and an integer $r \geq 1$, produces a set of terms $\mathcal{T}$ over $d+r$ variables such that there exists a C-term DNF expression equal to f, if and only if, $\{0,1\}^{d+r}$ can be covered by $2^r C + |f^{-1}(0)|$ terms from $\mathcal{T}$. The algorithm runs in time $2^{O(r+d)}$.*

Proof: The idea of the proof is to create $\mathcal{T}$ that contains all the terms consistent with f and terms that cover $\neg f$. As in the proof of Theorem 3.2, we will replicate f many times to preserve the approximation ratio .

Our instance of the HC-COVER problem is over $d+r$ variables where we refer to the first d variables as $x_1, \ldots x_d$ and to the next r variables as $y_1, \ldots, y_r$. Let $\mathcal{T}^f$ be the set of all the terms consistent with f (we can assume that it includes only the prime implicants of f but this is not essential). For $p \in \{0,1\}^r$ let $\mathcal{T}^f_p = \{T \wedge \mathtt{eq}(y,p) \mid T \in \mathcal{T}^f\}$, let $S = f^{-1}(0)$ and $\mathcal{T}^{\neg f} = \{\mathtt{eq}(x,a) \mid a \in S\}$. Then we define

$$\mathcal{T} = \mathcal{T}^{\neg f} \bigcup \left(\bigcup_{p \in \{0,1\}^r} \mathcal{T}^f_p \right).$$

We claim that there exists a C-term DNF equal to f, if and only if, there exists a set $\mathcal{S} \subseteq \mathcal{T}$ of size $C2^r + |S|$ that is a cover of $\{0,1\}^{r+d}$. Let $\mathcal{S}^f$ be a set of C terms such that $\bigvee_{T \in \mathcal{S}^f} T = f$. Then it is clear that

$$\mathcal{S} = \mathcal{T}^{\neg f} \bigcup \left\{ T \wedge \mathtt{eq}(y,p) \mid p \in \{0,1\}^r,\ T \in \mathcal{S}^f \right\}$$

is a cover of $\{0,1\}^{r+d}$, has size $C2^r + |S|$, and includes only terms from $\mathcal{T}$.

For the other direction, let $\mathcal{S} \subseteq \mathcal{T}$ be a cover of $\{0,1\}^{d+r}$. We observe that the only way to cover $S \times \{0,1\}^r$ is by including all the terms of $\mathcal{T}^{\neg f}$ in $\mathcal{S}$. For each $p \in \{0,1\}^r$, let $\mathcal{S}_p = \mathcal{S} \cap \mathcal{T}^f_p$. Only terms in $\mathcal{S}_p$ cover the subset $f^{-1}(1) \times \{p\}$ and therefore $h_p(x) = \bigvee_{T \in \mathcal{S}_p} T(x \cdot p)$ equals exactly $f(x)$. All the $\mathcal{S}_p$'s are mutually disjoint and are disjoint from $\mathcal{T}^{\neg f}$. Therefore if $|\mathcal{S}| \leq C2^r + |S|$, then $\mathtt{DNF\text{-}size}(f) \leq C$. □

As with Theorem 3.2, we obtain the following corollary.

Corollary 3.5 *If HC-COVER can be approximated within $h(d) = o(d)$ in time $t(d)$, then TT-MinDNF can be approximated within $h(2d + \log d) + 1$ in time $t(2d + \log d) + 2^{O(d)}$.*

We summarize the reductions in this section by the following equivalence theorem:

Theorem 3.6 *If there exists a constant $0 < \gamma \leq 1$ such that there is no polynomial-time algorithm approximating PHC-COVER, to within a factor d^γ then there is no polynomial-time algorithm approximating TT-MinDNF, PTT-MinDNF and HC-COVER to within a factor $\Omega(d^\gamma)$.*

4 Hardness of Approximation

Below we prove hardness of approximating TT-MinDNF by presenting two reductions to PHC-COVER both showing hardness of approximating within a factor of d^γ for a constant $\gamma > 0$. The first one is a reduction from the problem of finding a vertex cover of a k-uniform hypergraph. It is simple (relative to the other reduction and the reduction by Allender *et al.* [3]) but relies on a stronger assumption $\mathsf{NP} \not\subseteq \mathsf{DTIME}(n^{\log(n)})$. The second one is a general reduction from one-round multi-prover proof systems. When used with a low-error PCP it gives NP-hardness of approximating within a factor of d^γ. In addition, it makes an explicit connection between γ and standard parameters of a proof system that might be useful in obtaining inapproximability factors with specific or optimal exponent γ.

4.1 Reduction from Hypergraph Vertex Cover to PHC-COVER

A k-uniform hypergraph $H = (V, E)$ consists of a set of vertices V and a collection E of k-element subsets of V called hyperedges. A *vertex cover* of H is a subset $S \subseteq V$ such that every hyperedge in E intersects S. The Ek-Vertex-Cover problem is the problem of finding a minimum size vertex cover on a k-uniform hypergraph. The problem is alternatively called the minimum hitting set problem with sets of size k and is equivalent to the set cover problem where each element of the universe occurs in exactly k sets.

The first explicit hardness result shown for Ek-Vertex-Cover was due to Trevisan who showed an inapproximability factor of $k^{1/19}$ [36] (and a comparable result is implicit in Feige's proof of inapproximability of SET-COVER [15]). As we aim at obtaining a large inapproximability factor for covering the hypercube, we are interested in results that hold for large values of k. The first result stating the range of k explicitly is due to Dinur *et al.* [13] who give the following theorem.

Theorem 4.1 *There exists some $c > 0$ such that unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\log\log(n)})$, there is no polynomial-time algorithm for approximating Ek-Vertex-Cover for $k \leq (\log M)^{1/c}$ to within a factor of $\lfloor k/2 \rfloor - 0.01$, where M is the number of hyperedges in the k-uniform hypergraph.*

Remark 4.2 *It can be easily seen from the proof of this theorem, that the number of vertices N is smaller than M and therefore the result can be stated with the number of vertices N in place of M.*

4.1.1 Union-free Families

A family of sets $\mathcal{F}$ is called k-union-free if $A_0 \not\subseteq A_1 \cup A_2 \cup \cdots \cup A_k$ for all distinct $A_0, A_1, \ldots, \mathcal{A}_k \in \mathcal{F}$. They were introduced by Kautz and Singleton [21] (and then rediscovered by Erdös *et al.* [14]). A family of sets $\mathcal{F}$ is a (s, ℓ)-*combinatorial design* if each set in $\mathcal{F}$ has size s and the intersection of any two sets in $\mathcal{F}$ has size at most ℓ. If $\ell < s/k$ then (s, ℓ)-combinatorial design is k-union-free. The first efficient construction of combinatorial designs was given by Nisan and Widgerson [28]. For our purposes, k-union-free families can be obtained by derandomizing a straightforward randomized construction using the method of conditional probabilities (*cf.* [39, Lecture 21]).

Theorem 4.3 *There exists a k-union-free family of sets over $[d]$ of size m for $d = O(k^2 \log m)$. Moreover, such $\mathcal{F}$ can be constructed in time poly(m, d).*

4.1.2 Simple Reduction to PHC-COVER

Below we present a reduction from SET-COVER with each point occurring in k sets to PHC-COVER.

Theorem 4.4 *There exists a polynomial-time algorithm that, given a k-uniform hypergraph $H = (V, E)$ with $|V| = N$, produces an instance $(S, \mathcal{T})$ of PHC-COVER over $d = O(k^2 \log N)$ variables such that H has a vertex cover of size C, if and only if, $(S, \mathcal{T})$ has a cover of size C. The algorithm runs in time $O(N2^d)$.*

Proof: We first transform the k-uniform hypergraph vertex cover problem to its dual set cover problem with each point occurring in k sets. That is, for $v \in V$, let $S_v = \{e \mid e \in E, v \in e\}$ and $\mathcal{S} = \{S_v \mid v \in V\}$. Then $(E, \mathcal{S})$ is the equivalent instance of SET-COVER. Let $\mathcal{F} = \{P_v\}_{v \in V}$ be a k-union-free family (with N elements indexed by nodes in V). By Theorem 4.3, such $\mathcal{F}$ exists and can be efficiently constructed for $d = O(k^2 \log N)$. For any set $P \subseteq [d]$, let $\chi(P)$ be a characteristic vector of P, that is vector with $\chi(P)_i = 1$ when $i \in P$ and $\chi(P)_i = 0$, otherwise. For each $e \in E$, let $x^e = \chi(\cup_{v \in e} P_v)$. We define $\mathcal{T} = \{\mathtt{eq}(x, \chi(P_v)) \mid v \in V\}$, and define $S = \{x^e \mid e \in E\}$.

To prove the correctness of this reduction all we need to show is that for each $e \in E$ and $v \in V$, $\mathtt{eq}(x, \chi(P_v))$ covers x^e, if and only if, $e \in S_v$ or, in other words, $v \in e$. If $v \in e$ then $P_v \subseteq \cup_{u \in e} P_u$ and, therefore $x^e_i = 1$ for all $i \in P_v$. This implies that $\mathtt{eq}(x, \chi(P_v)) = \wedge_{i \in P_v} x_i$ accepts x^e. On the other hand, if $v \notin e$ then for each $u \in e$, by the properties of $\mathcal{F}$, $P_v \not\subseteq \cup_{u \in e} P_u$. This implies that $\mathtt{eq}(x, \chi(P_v))$ will not accept x^e. □

Corollary 4.5 *There exists a constant $\gamma > 0$ such that, unless $\mathsf{NP} \subseteq \mathsf{DTIME}(n^{\log(n)})$, there is no polynomial-time algorithm approximating PHC-COVER to within a factor d^γ, where d is the number of variables in the PHC-COVER instance.*

Proof: Given an instance of SAT on n variables, the reduction of Theorem 4.1 produces an instance of Ek-Vertex-Cover on $N = n^{O(\log \log n)}$ vertices for $k = (\log N)^{1/b}$, where $b = \max\{3, c\}$. The gap in vertex cover sizes between positive and negative instances is αk ($\alpha \approx 1/2$). Then reduction in Theorem 4.4 will produce an instance of PHC-COVER over $d = O(k^2 \log N) = O((\log N)^{1+2/b})$ with the same gap of αk, that in terms of d, is $\Omega(d^{1/(b+2)}) > d^\gamma$ for any constant $\gamma < \frac{1}{2+b}$ (and large enough d). The running time of the reduction and the produced instance are both bounded by

$$2^{O(d)} = 2^{O((\log N)^{1+2/b})} = n^{O((\log n)^{2/b} (\log \log n)^{1+2/b})} = O(n^{\log n})$$

(since $b \geq 3$). □

4.2 Reducing from Multi-prover Proof Systems

Below we prove hardness of approximating PHC-COVER by presenting a direct reduction from one-round multi-prover proof systems with certain properties to PHC-COVER. We then obtain the claimed result by coupling our reduction with the low-error PCP for NP due to Raz and Safra [34]. The reduction simulates the reduction from SET-COVER given by Bellare *et al.* [6] (which is a simple generalization of the reduction by Lund and Yannakakis [25]) on the Boolean hypercube. That is, the instance of PHC-COVER we create is the same as the instance of SET-COVER created in the reduction of Bellare *et al.* [6]. Therefore the analysis we give follows directly from the analysis of Lund and Yannakakis [25].

Following the definition by Bellare *et al.* [6] we distinguish five important parameters of one-round multi-prover proof systems and define the class $\mathrm{MIP}_1(\ell_r, p, \ell_a, \ell_q, \epsilon)$ as follows:

Definition 4.6 *$L \in MIP_1(\ell_r(n), p(n), \ell_a(n), \ell_q(n), \epsilon(n))$ if there exists a probabilistic polynomial-time verifier V, communicating with $p(n)$ provers such that for every $x \in \{0,1\}^n$ the verifier:*

- *tosses $\ell_r(n)$ random coins obtaining $r \in \{0,1\}^{\ell_r}$,*
- *computes $p(n)$ questions $q(r)_1, \ldots, q(r)_{p(n)}$ each of length at most $\ell_q(n)$,*
- *for each i, asks the i-th prover question $q(r)_i$ and gets $p(n)$ answers $a_1, \ldots, a_{p(n)}$ each of length at most $\ell_a(n)$,*
- *computes a predicate $V(x, r, a_1, \ldots, a_{p(n)})$ and accepts if and only if it is 1,*
- *has perfect completeness: if $x \in L$ then $\exists \bar{P} = P_1, \ldots, P_{p(n)}$ such that $\mathbf{Pr}_r[V \text{ accepts when interacting with } \bar{P}] = 1$;*
- *has soundness error at most $\epsilon(n)$: if $x \notin L$ then $\forall \bar{P} = P_1, \ldots, P_{p(n)}$, $\mathbf{Pr}_r[V \text{ accepts when interacting with } \bar{P}] \leq \epsilon(n)$.*

Our reduction will rely on three simple properties of V. The *functionality* property requires that for each $x \in \{0,1\}^n$ and each $a_1 \in \{0,1\}^{\ell_a}$ there is at most one vector $(a_2, a_3, \ldots, a_p)$ such that $V(x, r, a_1, a_2, \ldots, a_p) = 1$ for some $r \in \{0,1\}^{\ell_r}$. The second property, *uniformity*, requires that for each $i \in [p]$, queries of V to prover i are uniformly distributed over the set Q_i of all the possible queries to prover i. The last, *equality of question space sizes*, requires that $|Q_1| = |Q_2| = \cdots = |Q_p|$. Following Bellare *et al.* [6] we call V *canonical* if it has these three properties.

Similarly, we distinguish analogous parameters for a PCP system. We denote the class

$$\mathrm{PCP}(\ell_r(n), p(n), \ell_a(n), \ell_q(n), \epsilon(n))$$

to be the class of languages decidable by a PCP verifier V that uses $\ell_r(n)$ random bits, generates $p(n)$ questions of length $\ell_r(n)$, gets answers of length $\ell_a(n)$, has perfect completeness and soundness error $\epsilon(n)$.

4.2.1 Packing a Proof System into the Boolean Hypercube

The main tool for creating an approximation gap is a set system

$$\mathcal{B}_{m,l} = (B; C_1, C_2, \ldots, C_m)$$

where m, l are positive integers and for each $i \in [m]$, $C_i \subseteq B$. This set system has the property that if $I \subset [m]$ and $|I| \leq l$, then no union $\bigcup_{i \in I} D_i$ covers B, where D_i equals C_i or its complement.

Lemma 4.7 ([25]) *There exists $\mathcal{B}_{m,l} = (B; C_1, C_2, \ldots, C_m)$ for $|B| = O(2^{2l} m^2)$ and it can be constructed in time polynomial in $|B|$.*

The main construction of this section is given in the following lemma.

Lemma 4.8 *If $L \in MIP_1(\ell_r, p, \ell_a, \ell_q, \epsilon)$ with a canonical verifier V, then there exists an algorithm $\mathcal{A}$ that given x, produces an instance of PHC-COVER $(S_x, \mathcal{T}_x)$ over $d \leq \ell_r + p(\ell_q + 2^{\ell_a})$ variables such that*

- *if $x \in L$ then PHC-COVER$(S_x, \mathcal{T}_x) = p|Q_1|$, where Q_1 is the question space of the first prover.*
- *if $x \notin L$ then PHC-COVER$(S_x, \mathcal{T}_x) \geq \frac{1}{2}(2\epsilon)^{-1/p}|Q_1|$.*

Moreover, $\mathcal{A}$ runs in time polynomial in n and 2^d.

Proof: We start by describing a way to map an answer from a prover to a subset. In this mapping the first prover is treated differently from the rest because of the functionality property of V. As before, let $Q_i \subseteq \{0,1\}^{\ell_q}$ denote the set of questions that V asks prover i and let A_i be the answer space of prover i. Set $s_a = |A_2| + |A_3| + \ldots + |A_p|$ (note that $|A_1|$ is not included) and let $\mathcal{B}_{s_a,l} = (B; C_1, C_2, \ldots, C_{s_a})$ be a set system given by Lemma 4.7 for l to be specified later. We index the sets $C_1, C_2, \ldots, C_{s_a}$ by pairs (i, a_i) for $2 \leq i \leq p$ and $a_i \in A_i$. Let $A = \{(i, a_i) \mid i \in [p],\ a_i \in A_i\}$ be the set of all possible answers (answers from different provers correspond to different elements).

Now for each setting of a random string $r \in R = \{0,1\}^{\ell_r}$ and $(i, a_i) \in A$, we define a subset $C(r, i, a_i) \subseteq B$ as follows.

$$C(r,i,a_i) = \begin{cases} C_{i,a_i} & \text{if } i \geq 2 \\ B \setminus (C_{2,a_2} \cup \cdots \cup C_{p,a_p}) & \text{if } i = 1 \text{ and} \\ & \exists a_2, \ldots, a_p,\ V(x, r, a_1, a_2, \ldots, a_p) = 1 \\ \emptyset & \text{otherwise} \end{cases}$$

For $a_1 \in A_1$, $C(r, 1, a_1)$ is well-defined since V has the functionality property. The definition of $C(r, i, a_i)$ implies that if $V(x, r, a_1, a_2, \ldots, a_p) = 1$ then $\cup_{i \in [p]} C(r, i, a_i) = B$, that is, answers from provers that cause the verifier to accept correspond to "small" covers. Bellare *et al.* [6] define the following instance of SET-COVER. The ground set equals to $R \times B$ and for every $i \in [p]$, $q_i \in Q_i$, $a_i \in A_i$ the set system includes a subset

$$Z(i, q_i, a_i) = \{(r, b) \mid q_i = q(r)_i \text{ and } b \in C(r, i, a_i)\} ,$$

where $q_i = q(r)_i$ means that V generates query q_i to prover i on input x and random string r. In other words, the set system is $\mathcal{Z}_x = \{Z(i, q_i, a_i) \mid i \in [p],\ q_i \in Q_i,\ a_i \in A_i\}$.

We now show that exactly the same set system can be created on a hypercube of dimension $d = \ell_r + p\ell_q + |A|$. We refer to the first ℓ_r variables of the Boolean cube $\{0,1\}^d$ as $y_{r,1}, \ldots, y_{r,\ell_r}$, the next $p\ell_q$ variables as $z_{i,j}$ for $i \in [p]$ and $j \in [\ell_q]$, and the last $|A|$ variables as $z_{(i,a_i)}$ for $(i, a_i) \in A$.

For every $r \in R$ and $b \in B$, let $z(r, b) \in \{0,1\}^A$ be a Boolean vector of length $|A|$ such that $z(r,b)_{(i,a_i)} = 1$ whenever $b \in C(r, i, a_i)$. Furthermore, let $[r, b] = r \cdot q(r)_1 \cdots q(r)_p \cdot z(r, b)$. Let $S_x = \{[r, b] \mid r \in R,\ b \in B\}$. We now proceed to define the terms. For $i \in [p]$, $q_i \in Q_i$, and $a_i \in A_i$, let $T(i, q_i, a_i)$ be the term that checks that variables of i-th question equal to q_i and that the variable corresponding to answer a_i from prover i is set to 1, or formally

$$T(i, q_i, a_i) = \mathsf{eq}(z_{i,1} \cdots z_{i,\ell_q}, q_i) \wedge z_{(i,a_i)} .$$

Let $\mathcal{T}_x = \{T(i, q_i, a_i) \mid i \in [p],\ q_i \in Q_i,\ a_i \in A_i\}$. It is easy to verify that the term $T(i, q_i, a_i)$ covers a point $[r, b]$ if and only if $q_i = q(r)_i$ and $b \in C(r, i, a_i)$. Therefore the set system $(S_x, \mathcal{T}_x)$ corresponds exactly to the SET-COVER instance of Bellare *et al.* [6], where $[r, b]$ corresponds to (r, b) and $T(i, q_i, a_i)$ corresponds to $Z(i, q_i, a_i)$.

The analysis of Lund and Yannakakis [25] can now be used to prove that for $x \in L$, PHC-COVER$(S_x, \mathcal{T}_x) \leq \sum_{i \in P} |Q_i| = p|Q_1|$ and for $x \notin L$, PHC-COVER$(S_x, \mathcal{T}_x) \geq (1 - \epsilon l^p) l \cdot |Q_1|$.

Therefore by setting $l = (2\epsilon)^{-1/p}$ we will get the stated inapproximability gap of $(2\epsilon)^{-1/p}/(2p)$. For completeness we provide the details of this analysis using the notation of the above SET-COVER instance.

First, let $x \in L$ and let $\bar{P}$ be an honest deterministic prover. For $q_i \in Q_i$ denote by $P_i(q_i)$ the answer given by P_i to query q_i and set $\mathcal{S} = \{Z(i, q_i, P_i(q_i)) \mid i \in [p],\ q_i \in Q_i\}$. For every point (r, b) and $i \in [p]$, let $a'_i = P_i(q(r)_i)$. By perfect completeness of V, $V(x, r, \bar{a}') = 1$ and therefore $C(r, 1, a'_1) = B \setminus (C_{2,a'_2} \cup \cdots \cup C_{p,a'_p})$, i.e, $\cup_{i\in[p]} C(r, i, a'_i) = B$. This means that for some j, $b \in C(r, j, a'_j)$ and therefore $(r, b) \in Z(j, q(r)_j, a'_j)$. This means that $\mathcal{S} \subseteq \mathcal{Z}$ is a collection of size $\sum_{i\in[p]} |Q_i|$ that covers $R \times B$. By equality of question space sizes, $|\mathcal{S}| = \sum_{i\in[p]} |Q_i| = p|Q_1|$.

Let $x \notin L$ and $\mathcal{S} \subseteq \mathcal{Z}_x$ be a cover for $R \times B$. For a random string r and a set $C \subseteq B$, denote by (r, C) the set $\{(r, b) \mid b \in C\}$ and let $\mathcal{S}_r = \{Z(i, q(r)_i, a_i) \mid i \in [p],\ Z(i, q(r)_i, a_i) \in \mathcal{S}\}$, in other words $\mathcal{S}_r$ includes the terms from $\mathcal{S}$ that cover (r, B). We say that r is *good* if $|\mathcal{S}_r| \leq l$ and *bad* otherwise. Let δ be the fraction of good r's.

Claim 4.9 *There exists a prover $\bar{P}$ such that V will accept with probability δ/l^p.*

Proof: We define $\bar{P}$ with the following strategy: prover P_i on query q_i chooses a_i from the set $A^i_{q_i} = \{a \mid Z(i, q_i, a) \in \mathcal{S}\}$ randomly and uniformly (this set cannot be empty). Note that for every r, $\sum_i |A^i_{q(r)_i}| = |\mathcal{S}_r|$. If r is good, then $|\mathcal{S}_r| \leq l$ and hence there should exist $a'_1, \ldots, a'_p$ such that for every $i \leq p$, $a'_i \in A^i_{q(r)_i}$ and $V(x, r, a'_1, \ldots, a'_p) = 1$. To prove this, assume that for every $a_1 \in A^1_{q(r)_1}$, there exists $j(a_1)$ such that $V(x, r, a_1, \ldots, a_p) = 1$ but $a_{j(a_1)} \notin A^{j(a_1)}_{q(r)_{j(a_1)}}$. Then

$$Z(1, q(r)_1, a_1) \cap (r, B) = (r, C(1, \tau(1, a_1)) = (r, B \setminus \bigcup_{i\geq 2} C_{i,a_i}) \subseteq (r, \overline{C_{j(a_1),a_{j(a_1)}}}) \ .$$

This implies that

$$\left(\bigcup_{a_1 \in A^1_{q(r)_1}} \overline{C_{j(a_1),a_{j(a_1)}}} \right) \bigcup \left(\bigcup_{i\geq 2,\ a_i \in A^i_{q_i}} C_{i,a_i} \right) = B \ .$$

We obtained a union of at most l sets from $\mathcal{B}_{s_a,l}$ that does not include a set and its complement, and covers B. This contradicts the definition of $\mathcal{B}_{s_a,l}$, proving the existence of $a'_1, \ldots, a'_p$ as above. For good r's and each i, $|A^i_{q(r)_i}| \leq l$ and therefore, the probability that each P_i will answer with a'_i is at least l^{-p}. Hence this strategy has success probability at least δ/l^p. □ (Claim 4.9)

Claim 4.10 $|\mathcal{S}| \geq (1 - \delta)l|Q_1|$.

Proof: For each bad r, $|\mathcal{S}_r| \geq l$ and therefore $\sum_r |\mathcal{S}_r| = (1 - \delta)2^{\ell_r} l$. On the other hand, each subset $Z(i, q_i, a_i) \in \mathcal{S}$ appears once in all the sets for which $q_i = q(r)_i$. The uniformity property of V implies that each query q_i is asked for exactly $2^{\ell_r}/|Q_i| = 2^{\ell_r}/|Q_1|$ different r's (the last equality follows from the equality of question space sizes property). This implies that

$$|\mathcal{S}| = \frac{|Q_i|}{2^{\ell_r}} \sum_r |\mathcal{S}_r| \geq \frac{|Q_i|}{2^{\ell_r}} (1 - \delta)2^{\ell_r} l = (1 - \delta)l|Q_1| \ .$$

□ (Claim 4.10)

By Claim 4.9 and soundness of V, we get that $\delta \leq \epsilon \cdot l^p$. By Claim 4.10, this implies that $|\mathcal{S}| \geq (1 - \epsilon l^p)l|Q_1|$. □ (Lemma 4.8)

4.2.2 Obtaining Proof Systems with Canonical Verifiers

In this section, we show how to derive canonical multi-prover proofs systems from general PCPs for NP. The first step is obtaining a multi-prover system from a PCP. As shown by Bellare, Goldreich, and Safra, (their proof appears in Ta-Shma's paper [35]) the identity transformation of a PCP to an MIP (that is, just distributing p queries to p different provers) increases the soundness error of the proof system by a factor of at most p^p. That is,

Lemma 4.11 ([35])

$$PCP(\ell_r(n), p(n), \ell_a(n), \ell_q(n), \epsilon(n)) \subseteq MIP_1(\ell_r(n), p(n), \ell_a(n), \ell_q(n), p^p\epsilon(n)) \ .$$

The next step in our transformation is obtaining the functionality property.

Lemma 4.12 *If $L \in MIP_1(\ell_r, p, \ell_a, \ell_q, \epsilon)$ with a verifier V, then*

$$L \in MIP_1(\ell_r, p+1, p\ell_a, p\ell_q, \epsilon)$$

with a verifier V' that has the functionality property.

Proof: To get a verifier V' with the desired property, we add one more prover (which we place first in the enumeration). Given r, V' uses V to generate questions $q_1, \ldots, q_p$, asks all the "old" provers their respective questions, and asks the new prover question $(q_1, q_2, \ldots, q_p)$. Given answers $a_1, \ldots, a_p$ from the "old" provers and an answer $(a'_1, \ldots, a'_p)$ from the new prover, V' accepts if $a'_i = a_i$ for all $i \in [p]$, and $V(x, r, a_1, \ldots, a_p) = 1$. We first observe that, by definition, V' has the functionality property. Next, we observe that V' interacts with the original p provers exactly as V does and accepts only when V does. Therefore the soundness error of the new multi-prover system does not increase and, in particular, is at most ϵ. Perfect completeness is preserved since if the first prover answers his questions in the same way as the other p honest deterministic provers, then V' will accept whenever V accepts. Finally, the bounds on the length of queries and answers grow by a factor of at most p. □

Next we describe how to obtain the last two properties required to get a canonical verifier.

Lemma 4.13 *If $L \in MIP_1(\ell_r, p, \ell_a, \ell_q, \epsilon)$ with a verifier V, then*

$$L \in MIP_1((p+1)\ell_r, p, \ell_a, \ell_r + \ell_q, \epsilon)$$

with a verifier V' that has uniformity and "equality of answer space sizes" properties. Furthermore, if V has the functionality property then V' is canonical.

Proof: For each $q_i \in Q_i$, let R_{i,q_i} denote the set of random strings for which V generates question q_i for prover i. New verifier V' uses V to generate questions $q_1, q_2, \ldots, q_p$ and then asks questions $((q_1, j_1), (q_2, j_2), \ldots, (q_p, j_p))$ where j_i is an element of $[|R_{i,q_i}|]$ chosen randomly, uniformly, and independently of other choices. It is easy to see that after this modification the sets of possible questions are all of the same size 2^{ℓ_r} and the questions are distributed uniformly. These random bits can be disregarded by honest provers and therefore completeness is not changed. Clearly, randomly and independently chosen bits cannot help dishonest provers and therefore soundness error is still bounded by ϵ. Finally, the bound on questions size is at most $\ell_r + \ell_q$ and the number of random bits required is at most $(p+1)\ell_r$. The accepting predicate of V was not changed and thus functionality property is preserved in this transformation. □

We can now combine these transformations with the following theorem due to Raz and Safra [34],

Theorem 4.14 *For any $\beta \leq 1/4$, $\ell_q(n) \leq \log^{\beta} n$ there exist fixed positive constants $b_r, b_p, b_q, b_\epsilon$ such that* $\mathsf{SAT} \in PCP(b_r \log n, b_p, \ell_a(n), b_q \log n, 2^{-b_\epsilon \ell_a(n)})$.

obtaining the following result:

Lemma 4.15 *There exist fixed positive constants $c_r, c_p, c_q, c_\epsilon$ for which*

$$\mathsf{SAT} \in MIP_1(c_r \log n, c_p, \log\log n, c_q \log n, \log^{-c_\epsilon} n)$$

with a canonical verifier.

Proof: We start with the PCP from Theorem 4.14 and then apply Lemmas 4.11, 4.12, and 4.13 to get

$$\mathsf{SAT} \in \mathrm{MIP}_1((b_p+2)b_r \log n, b_p+1, b_p\ell_a(n), (b_pb_q+b_r)\log n, b_p^{b_p} 2^{-b_\epsilon \ell_q(n)}) \ .$$

We now choose $\ell_a(n) = (\log\log n)/b_p$ and obtain the desired result for $c_r = (b_p+2)b_r$, $c_p = b_p+1$, $c_q = (b_pb_q+b_r)$, and any $c_\epsilon > b_\epsilon/b_p$ ("strictly greater" is to offset the constant factor $b_p^{b_p}$). □

We can now use the results from Sections 3 and 4.2 to summarize our inapproximabilily results for covering problems on the Boolean hypercube.

Theorem 4.16 (subsumes Th. 1.1) *There exists a constant $\gamma > 0$ such that, unless* $\mathsf{P} = \mathsf{NP}$, *there is no polynomial-time algorithm approximating TT-MinDNF, PTT-MinDNF, HC-COVER, and PHC-COVER, to within a factor $d^\gamma = \Omega(\log^\gamma(N))$, where d is the number of variables and N is the size of an instance.*

By using our reduction from MIP to PHC-COVER(Lemma 4.8) with the canonical-verifier MIP obtained from the PCP of Raz and Safra [34] (Lemma 4.15), we get an inapproximability gap of $(2\log n)^{c_\epsilon/c_p}/(2c_p)$ for $d \leq (c_r + c_pc_q + c_p)\log n$. This implies the claim for PHC-COVER. We use Theorem 3.6 to extend the result to TT-MinDNF, PTT-MinDNF and HC-COVER.

5 Hardness of Proper PAC Learning with Membership Queries

In this section, we present our hardness results for proper PAC+MQ learning of DNF formulae. We first look at the learning model where the distribution over the input space is not restricted. In this setting our result is based on the hardness result for learning DNF expressions without MQs by Alekhnovich *et al.* [2]. As in their work, we prove a stronger result that shows hardness of learning DNF expressions by a more expressive[1] class of OR-of-thresholds.

Theorem 5.1 (subsumes Th. 1.2) *If there exists an algorithm $\mathcal{A}$ such that for every Boolean function c, distribution D and ϵ, $\mathcal{A}$, given access to $EX(c, D)$ and $MEM(c)$, runs in time $poly(n, \texttt{DNF-size}(c), 1/\epsilon)$ and with probability at least $3/4$ outputs an OR-of-thresholds formula h such that $\mathbf{Pr}_{x\in D}[h(x) = c(x)] \geq 1 - \epsilon$, then* $\mathsf{NP} = \mathsf{RP}$.

For consistency, we prove an equivalent formulation that CNF expressions are not learnable by AND-of-thresholds. The proof of Alekhnovich *et al.* is based on a reduction from approximating the chromatic number of a graph. Given a graph G, they produce a set of examples such that if

[1] More expressive in the sense that all size k DNF formulae can be expressed by size k OR-of-thresholds (but not vice versa).

the chromatic number of G is “small” then there exists a “small” CNF formula consistent with the examples. Otherwise, if the chromatic number of the underlying graph is “large”, then the size of the minimum AND-of-thresholds formula with “small” error on the induced distribution over the examples is “large”. Our contribution is to show that we can define (efficiently) values of the target function f on the rest of the hypercube so that in the case of the “small” chromatic number, f can still be represented by a relatively “small” CNF formula. This allows us to answer queries to the membership oracle without any knowledge of a “small” coloring.

5.1 From Coloring to Learning

Given a graph $G = (V, E)$, construct a target function f and a distribution D as follows. Fix some positive integer parameter r. The examples are from $\left(\{0,1\}^V\right)^r = \{0,1\}^{|V|\cdot r}$.

Definition 5.2 *Let $G(V, E)$ be a graph with n vertices and m edges. For a vertex $v \in V$, let $z(v)$ denote the vector with a 1 in the v-th position and 0 everywhere else. For an edge $e = (u, v)$ of G, let $z(e)$ be the vector with a 1 in positions u and v.*

Following the construction of Alekhnovich *et al.* [2], with each vector $(v_1, v_2, \ldots, v_r) \in V^r$, we associate a negative example $\langle z(v_1), \ldots, z(v_r), 0\rangle$. For each choice of k_1, k_2, such that $1 \le k_1 \le r$, $1 \le k_2 \le r$, $k_1 \ne k_2$, $e = (u, w) \in E$ and $v_i \in V$ for each $i = 1, 2, \ldots, r$, $i \ne k_1, k_2$, we associate a positive example

$$\langle z(v_1), \ldots, z(v_{k_1-1}), z(e), z(v_{k_1+1}), \ldots, z(v_{k_2-1}), \overline{0}, z(v_{k_2+1}), \ldots, z(v_r), 1\rangle \ .$$

Let S^+ denote the positive examples and S^- denote the negative examples. Set $S = S^+ \cup S^-$. The distribution D is uniform over the above set of examples S.

These examples define the values of f on points in S. In order to answer membership query we also need to define f on the rest of the hypercube. Let $x = (x^1, \ldots, x^r)$ be a point not in $S^+ \cup S^-$. If for all i, $x^i \in \{\bar{0}\} \cup \{z(v) \mid v \in V\}$ then $f(x) = 0$. We refer to this set of points as *0-vertex points*. If for some $i \in [r]$, there exists $(u, v) \notin E$ such that $x^i_u = x^i_v = 1$, then $f(x) = 0$. We call this set of points *non-edge* points. Otherwise, let $f(x) = 1$. We first note that the example oracle for f with respect to the distribution D and the membership query oracle for f can be simulated efficiently by a randomized algorithm with input G.

We now prove that if the chromatic number $\chi(G)$ is small, then there exists a small CNF formula equal to f.

Lemma 5.3 *If $\chi(G) \le n^\lambda$, then there is a CNF formula of size at most $n^{r\lambda} + r|E|$ equal to f.*

Proof: Suppose $V = \bigcup_{i=1}^{\chi} V_i$, where V_i are independent sets. Such sets must exist by the definition of χ. Define the CNF formula

$$g(x_1, x_2, \ldots, x_n) = \bigwedge_{i=1}^{\chi} \bigvee_{v \notin V_i} x_v \ .$$

This formula rejects all points in $\{\bar{0}\} \cup \{z(v) \mid v \in V\}$ and accepts all points in $\{z(e) \mid e \in E\}$.

We then define an expression F that rejects all the points in S^- and 0-vertex points

$$F(x^1, \ldots, x^r) = \bigvee_{k=1}^{r} g(x^k_1, \ldots, x^k_n) = \bigvee_{k=1}^{r} \bigwedge_{i=1}^{\chi} \bigvee_{v \notin V_i} x^k_v \ ,$$

and a CNF formula H on $r \cdot n$ variables that is negative on all the non-edge points and positive elsewhere

$$H(x^1, \ldots, x^r) = \bigwedge_{k \in [r];\ (u,v) \notin E} (\overline{x^k_u} \vee \overline{x^k_v}) \ .$$

We claim that in addition to S^- and 0-vertex points F rejects only non-edge points. By the definition of F, if F rejects a point $x^1, \ldots, x^r$ then for all $k \in [r]$, there exists $i \in [\chi]$ such that for all $v \notin V_i$, $x_v = 0$. Therefore if for some $k \in [r]$ and $u, v \in V$, $x^k_u = x^k_v = 1$ then $u, v \in V_i$ for some i. In particular, $(u, v) \notin E$ since V_i is an independent set. We therefore obtain that $F \wedge H$ rejects exactly points in S^-, 0-vertex points and the non-edge points, in other words, is identical to f. We remark that in order to answer membership queries to F we would need to know which non-edge points it accepts and which rejects. This would not be possible without knowing the "small" coloring. Therefore f is defined as $F \wedge H$ in order to hide all the coloring-dependent information in F by using H that rejects all the non-edge points.

Note that F above is not written as a CNF formula. It is, however, a disjunction of r CNF formulas, each having at most $\chi(G)$ clauses. Hence expanding the expression for F yields a CNF formula with at most $\chi(G)^r \leq n^{\lambda r}$ clauses. So $F \wedge H$ can be written as a CNF formula satisfying the conditions of the lemma. □

For the case when the chromatic number is large, we can use the original analysis of Aleknovich *et al.* [2]. This is possible since the distribution D and values of f on points in S are identical to those in their construction.

Lemma 5.4 (Th. 26 in [2]) *Let G be a graph such that $\chi(G) \geq n^{1-\lambda}$. Let $F = \wedge_{i=1}^{\ell} h_i$. If $\ell < \frac{1}{2\chi r}\left(\frac{\chi - 1}{\log n}\right)^r$ then F has error at least $\frac{1}{n^{2r\gamma+4}}$ with respect to D.*

Combining the two cases (Lemmas 5.3 and 5.4) gives us the following claim.

Lemma 5.5 *Suppose that there exists an algorithm $\mathcal{A}$ such that for every Boolean function c of CNF-size s, distribution D, and $\epsilon > 0$, $\mathcal{A}$, given access to EX(c, D) and MEM(c), runs in time $O(n^k \cdot s^k \cdot (\frac{1}{\epsilon})^k)$ and with probability at least $3/4$ outputs an AND-of-thresholds formula h such that $\mathbf{Pr}_{x \in D}[h(x) = c(x)] \geq 1 - \epsilon$. Then there exists a randomized algorithm that given a graph G on N vertices, can distinguish between the case when $\chi(G) \leq N^{\frac{1}{10k}}$ and the case when $N^{1-\frac{1}{10k}}$ in time $O(N^{9k+1})$. Moreover, the algorithm always succeeds when $\chi(G) \geq N^{1-\frac{1}{10k}}$.*

Proof: Fix $\epsilon = \frac{1}{N^6}$, and $r = 10k$. For a graph G let the target function f and the distribution D be defined as in Section 5.1. The dimension of the learning problem is $n = rN = 10kN$. Run the learning algorithm with respect to distribution D and with queries answered according to f. If it does not terminate after N^{9k+1} steps, output "$\chi \geq N^{1-\frac{1}{10k}}$". Otherwise, let h be the hypothesis the algorithm outputs. Calculate the error ϵ_h of h with respect to the distribution D. If $\epsilon_h < \frac{1}{N^6}$ output "$\chi \leq N^{\frac{1}{10k}}$", otherwise output "$\chi \geq N^{1-\frac{1}{10k}}$". We claim that this algorithm succeeds with probability at least 3/4 when $\chi \leq N^{\frac{1}{10k}}$ and always succeeds when $\chi \geq N^{1-\frac{1}{10k}}$.

If $\chi \leq N^{\frac{1}{10k}}$, then Lemma 5.3 implies that

$$s = \texttt{CNF-size}(f) \leq N^{\frac{1}{10k}10k} + 10k|E| \leq 10k \cdot N^2 \ .$$

Hence, the running time of $\mathcal{A}$ is at most

$$O(n^k \cdot s^k \cdot (\frac{1}{\epsilon})^k) = O((10k \cdot N)^k (10k \cdot N^2)^k (N^6)^k) = O(N^{9k}) < N^{9k+1}$$

for sufficiently large N and, with probability at least 3/4, the output hypothesis has error less than $\epsilon = \frac{1}{N^6}$. Hence the algorithm outputs "$\chi \leq N^{\frac{1}{10k}}$" with probability at least 3/4 in this case.

If $\chi \geq N^{1-\frac{1}{10k}}$, then by Lemma 5.4, an AND-of-thresholds with error less than $\epsilon = \frac{1}{N^6} = \frac{1}{N^{2\cdot 10k(\frac{1}{10k})+4}}$ must be of size at least $\frac{1}{2\chi r}\left(\frac{\chi-1}{\ln N}\right)^r$. Therefore in order to output a hypothesis with error at most ϵ, the running time of $\mathcal{A}$ must be at least

$$\frac{1}{2\chi r}\left(\frac{\chi-1}{\ln N}\right)^r \geq \frac{1}{20\chi \cdot k}\left(\frac{N^{1-\frac{1}{10k}}-1}{\ln N}\right)^{10k} \geq \frac{1}{20Nk}\left(\frac{N^{1-\frac{1}{10k}}}{2\ln N}\right)^{10k} \geq N^{10k-3}$$

for sufficiently large N. Hence if $\mathcal{A}$ terminates in $N^{9k+1} < N^{10k-3}$ steps, its error will be larger than ϵ, and the algorithm outputs "$\chi \geq N^{1-\frac{1}{10k}}$" with probability 1 in this case. □

Finally, we will require the following hardness result due to Feige and Kilian [16]:

Theorem 5.6 ([16]) *For any constant $\gamma > 0$, there exists a polynomial-time randomized reduction mapping instances f of SAT of length n to graphs G with $N = \mathsf{poly}(n)$ vertices with the property that if f is satisfiable then $\chi(G) \leq O(N^\gamma)$ and if f is unsatisfiable then $\chi(G) \geq \Omega(N^{1-\gamma})$. The reduction has zero-sided error.*

Combining Lemma 5.5 and Theorem 5.6 gives Theorem 5.1.

5.2 Hardness of Proper PAC+MQ Learning over the Uniform Distribution

We now show a simple application of the hardness of TT-MinDNF to the hardness of proper PAC learning of DNF expressions restricted to the uniform distribution over $\{0,1\}^n$. It is a very strong model in which, as proved by Jackson [20], DNF expressions are learnable non-properly.

It has been observed by Allender *et al.* that TT-MinDNF naturally reduces to exact learning of DNF with MQs [4]. We further this observation by reducing TT-MinDNF to PAC+MQ learning of DNF over the uniform distribution. We denote the uniform distribution over $\{0,1\}^n$ by U.

Theorem 5.7 *There exists a constant $\gamma > 0$ such that, if there exists an algorithm $\mathcal{A}$ that for every Boolean function c and $\epsilon > 0$, $\mathcal{A}$, given access to EX(c,U) and MEM(c), runs in time poly$(n, s = \texttt{DNF-size}(c), 1/\epsilon)$ and, with probability at least 3/4, outputs a DNF formula h of size at most $\log^\gamma(s/\epsilon)\cdot s$ that ϵ-approximates c with respect to U, then $\mathsf{NP} = \mathsf{RP}$.*

Proof: We reduce from TT-MinDNF and let γ be the constant from Theorem 1.1. Given the truth table of a function f over $d = \log n$ variables, we let the target concept be $c(x) = f(x_1 \cdots x_d)$. Clearly, $s = \texttt{DNF-size}(c) \leq 2^{\log n} = n$. The definition of $c(x)$ implies that EX(c,U) and MEM(c) can be efficiently simulated given the truth table of f. We then set $\epsilon = 1/(2n)$ and $\delta = 1/2$. A strongly proper algorithm on this input will (with probability at least 1/2) produce in time polynomial in $n = 2^d$ a DNF formula h of size $t \leq \log^\gamma(s/\epsilon)\cdot s$ that $\frac{1}{2n}$-approximates c. Now we choose a vector y of length $n-d$ randomly and uniformly and let h_y be the projection of h to first d variables with the last $n-d$ variables set to y. We claim that with probability at least 1/2, $f \equiv h_y$. To see this, note that

$$\frac{1}{2n} \geq \mathbf{Pr}_{x\in\{0,1\}^n}[c(x) \neq h(x)] = \mathbf{E}_{y\in\{0,1\}^{n-d}}\left[\mathbf{Pr}_{z\in\{0,1\}^d}[f(z) \neq h_y(z)]\right] ,$$

and thus for at least $\frac{1}{2}$ of y's, $\mathbf{Pr}_{z\in\{0,1\}^d}[f(z) \neq h_y(z)] = 0$, that is, $f(z) = h_y(z)$ for all z. For each y, the number of terms in h_y is at most t and therefore with probability at least 1/4, t approximates $\texttt{DNF-size}(f)$ within $\log^{\gamma}(s/\epsilon) = O(d^{\gamma})$. □

6 Conclusions and Open Problems

In this work we have conclusively answered the question of proper learning of DNF expressions in the PAC+MQ model. We also made some progress towards answering a similar question when the distribution is restricted to be uniform. It is easy to see that in the uniform case finding a DNF hypothesis $O(\log(s/\epsilon))$ times larger than the target can be done in time $n^{O(\log(s/\epsilon))}$ [42] and therefore, is unlikely to be NP-hard. This means that substantial improvements of the result will have to be based on different (probably stronger) assumptions. A more important direction would be to reduce restrictions on the output hypothesis in the hardness results (with the final goal being the circuit representation).

It would be also interesting to close the gap between $O(d)$ and d^{γ} for approximating TT-MinDNF.

Acknowledgments

I am grateful to Leslie Valiant for his advice and encouragement of this research. I would also like to thank Alex Healy, Shaili Jain, Emanuele Viola, and anonymous referees of STOC 2006 and JCSS for careful proofreading and valuable comments on the earlier versions of this manuscript.

References

[1] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data. *Data Mining and Knowledge Discovery*, 11(1):5–33, July 2005.

[2] M. Alekhnovich, M. Braverman, V. Feldman, A. Klivans, and T. Pitassi. The complexity of properly learning simple classes. *Journal of Computer and System Sciences*, 74(1):16–34, 2008.

[3] E. Allender, L. Hellerstein, P. McCabe, T. Pitassi, and M.Saks. Minimizing DNF formulas and AC^0 circuits given a truth table. In *Proceedings of IEEE Conference on Computational Complexity*, pages 237–251, 2006.

[4] E. Allender, L. Hellerstein, T. Pitassi, and M.Saks. On the complexity of finding minimal representations of boolean functions. 2004. Unpublished.

[5] D. Angluin and M. Kharitonov. When won't membership queries help? *Journal of Computer and System Sciences*, 50(2):336–355, 1995.

[6] M. Bellare, S. Goldwasser, C. Lund, and A. Russell. Efficient probabilistically checkable proofs and applications to approximations. In *Proceedings of STOC*, pages 294–304, 1993.

[7] A. Blum and S. Rudich. Fast learning of k-term DNF formulas with queries. *Journal of Computer and System Sciences*, 51(3):367–373, 1995.

[8] A. L. Blum and R. L. Rivest. Training a 3-node neural network is NP-complete. *Neural Networks*, 5(1):117–127, 1992.

[9] V. Chvàtal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4:233–235, 1979.

[10] O. Coudert and J. Madre. METAPRIME, an Interactive Fault-Tree Analyser. *IEEE Transactions on Reliability*, 43(1):121–127, 1994.

[11] O. Coudert and T. Sasao. Two-level logic minimization. In R. Brayton, S. Hassoun, and T. Sasao, editors, *Logic Synthesis and Verification*, pages 1–29. Kluwer, 2001.

[12] S. Czort. The complexity of minimizing disjunctive normal form formulas. Master's thesis, University of Aarhus, 1999.

[13] I. Dinur, V. Guruswami, S. Khot, and O. Regev. A new multilayered pcp and the hardness of hypergraph vertex cover. *SIAM Journal of Computing*, 34(5):1129–1146, 2005.

[14] P. Erdös, P. Frankl, and Z. Furedi. Families of finite sets in which no set is covered by the union of r others. *Israel Journal of Mathematics*, 51:79–89, 1985.

[15] U. Feige. A threshold of $\ln n$ for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[16] U. Feige and J. Kilian. Zero knowledge and the chromatic number. In *Proceedings of Conference on Computational Complexity (CCC-96)*, pages 278–289, May 24–27 1996.

[17] J.F. Gimpel. A method for producing a boolean function having an arbitrary prescribed prime implicant table. *IEEE Transactions on Computers*, 14:485–488, 1965.

[18] E. A. Gold. Complexity of automaton identification from given data. *Information and Control*, 37:302–320, 1978.

[19] L. Hellerstein and V. Raghavan. Exact learning of DNF formulas using DNF hypotheses. In *Proceedings of STOC*, pages 465–473, 2002.

[20] J. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55:414–440, 1997.

[21] W. Kautz and R. Singleton. Nonrandom binary superimposed codes. *IEEE Trans. Inform. Theory*, 10:363–377, 1964.

[22] M. Kearns, M. Li, L. Pitt, and L. Valiant. On the learnability of Boolean formulae. In *Proceedings of STOC*, pages 285–295, 1987.

[23] A. Klivans and R. Servedio. Learning DNF in time $2^{\tilde{\mathrm{O}}(\mathrm{n}^{1/3})}$. *Journal of Computer and System Sciences*, 68(2):303–318, 2004.

[24] H. Liu. Routing table compaction in ternary cam. *IEEE Micro*, 22(1):58–64, 2002.

[25] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM*, 41(5):960–981, 1994.

[26] W. Masek. Some NP-complete set covering problems. unpublished, 1979.

[27] E. L. Jr. McCluskey. Minimization of Boolean Functions. *Bell Sys. Tech. Jour.*, 35:1417–1444, 1956.

[28] N. Nisan and A. Wigderson. Hardness versus randomness. *Journal of Computer and System Sciences*, 49:149–167, 1994.

[29] R. Nock, P. Jappy, and J. Sallantin. Generalized graph colorability and compressibility of boolean formulae. In *Proceedings of the 9th International Symposium on Algorithms and Computation (ISAAC)*, 1998.

[30] W. J. Paul. Boolesche minimalpolynome und überdeckungsprobleme. *Acta Informatica*, 4:321–336, 1974.

[31] L. Pitt and L. Valiant. Computational limitations on learning from examples. *Journal of the ACM*, 35(4):965–984, 1988.

[32] W.V. Quine. The problem of simplifying truth functions. *Americal Mathematical Monthly*, 59:521–531, 1952.

[33] W.V. Quine. A way to simplify truth functions. *Americal Mathematical Monthly*, 62:627–631, 1956.

[34] R. Raz and S. Safra. A sub-constant error-probability low-degree test, and a sub-constant error-probability PCP characterization of NP. In *Proceedings of STOC*, pages 475–484, 1997.

[35] A. Ta-Shma. A Note on PCP vs. MIP. *Information Processing Letters*, 58(3):135–140, 1996.

[36] L. Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *Proceedings of STOC*, pages 453–461, 2001.

[37] C. Umans. Hardness of approximating Σ_2^p minimization problems. In *Proceedings of FOCS*, pages 465–474, 1999.

[38] C. Umans, T. Villa, and A. L. Sangiovanni-Vincentelli. Complexity of two-level logic minimization. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 25(7):1230–1246, 2006.

[39] S. Vadhan. Lecture notes on pseudorandomness. Available at `http://www.courses.fas.harvard.edu/~cs225/`, 2004.

[40] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[41] L. G. Valiant. Learning disjunctions of conjunctions. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, pages 560–566, 1985.

[42] K. Verbeurgt. Learning DNF under the uniform distribution in quasi-polynomial time. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 314–326, 1990.