Conjunctions of Unate DNF Formulas: Learning and Structure

Aaron Feigelson Lisa Hellerstein*

*Department of ECE, Northwestern University, Evanston, IL 60208-3118. Research partially funded by NSF Grants No. CCR-95-01660 and CCR-92-10957

Proposed running head: **Conjunctions of Unate DNF Formulas**

Corresponding author: Aaron Feigelson
Department of ECE
Northwestern University
2145 Sheridan Road
Evanston IL, 60208-3118

email: aaron@ece.nwu.edu

Abstract

A central topic in query learning is to determine which classes of Boolean formulas are efficiently learnable with membership and equivalence queries. We consider the class $\mathcal{R}^k$ consisting of conjunctions of k unate DNF formulas. This class generalizes the class of k-clause CNF formulas, and the class of unate DNF formulas, both of which are known to be learnable in polynomial time with membership and equivalence queries. We prove that $\mathcal{R}^2$ can be properly learned with a polynomial number of polynomial-size membership and equivalence queries, but can be properly learned in polynomial time with such queries if and only if P = NP. Thus the barrier to properly learning $\mathcal{R}^2$ with membership and equivalence queries is computational rather than informational. Few results of this type are known. In our proofs, we use recent results of Hellerstein, Pillaipakamnatt, Raghavan, and Wilkins, characterizing the classes that are polynomial-query learnable, together with work of Bshouty on the monotone dimension of Boolean functions. We extend some of our results to $\mathcal{R}^k$, and pose open questions on learning DNF formulas of small monotone dimension. We also prove structural results for $\mathcal{R}^k$. We construct, for any fixed $k \geq 2$, a class of functions f that cannot be represented by any formula in $\mathcal{R}^k$, but which cannot be "easily" shown to have this property. More precisely, for any function f on n variables in the class, the value of f on any polynomial-size set of points in its domain is not a witness that f cannot be represented by a formula in $\mathcal{R}^k$. Our construction is based on BCH codes.

Symbols used in the paper:

$\mathcal{R}^2$	script R squared
$\mathcal{M}_{b_1}(f)$	script M sub (b sub 1) of eff
$\geq$	greater-than-or-equal-to
$\oplus$	oplus
$\neg$	neg
$\in$	element-of
$\subseteq$	subseteq
$\equiv$	equiv
∞	infinity
$\bigwedge$	bigwedge
$\wedge$	wedge
$\neq$	not equal
$\vee$	vee
$\cup$	cup
δ_H	delta sub H

1 Introduction

1.1 Overview

A central topic in query learning is to determine which classes of Boolean formulas are efficiently learnable. A number of classes have been shown to be learnable in polynomial time using membership and equivalence queries (for an incomplete survey, see [4]). For classes that are difficult to learn, we would like to know whether the difficulty is purely informational (it takes more than a polynomial number of queries to learn the class) or whether it is computational (a polynomial number of queries may suffice, but it is difficult to generate and process them in polynomial time).

Aizenstein, Hegedűs, Hellerstein, and Pitt introduced the first technique for proving computational hardness results on proper learning in the membership and equivalence query model [1]. Related techniques were then presented by Pillaipakkamnatt and Raghavan [11]. They showed that certain classes of DNF formulas, including read-thrice DNF, are not properly learnable in polynomial time with membership and equivalence queries unless P = NP. Thus there is a computational barrier to properly learning these classes. It is not known whether there is also an informational barrier to learning these classes. That is, it is open whether these classes are *polynomial-query learnable*. A class is polynomial-query learnable if it can be properly learned using a polynomial number of (polynomial-size) membership and equivalence queries, given unlimited computational time.

More recently, Hellerstein, Pillaipakkamnatt, Raghavan, and Wilkins gave a characterization of the classes that are polynomial-query learnable [9]. (A related result was proved independently by Hegedűs [8].) Using this characterization, they showed that the class of CDNF formulas is polynomial-query learnable. However, it is not known whether there is a computational barrier to learning this class. That is, it is open whether this class can be properly learned in polynomial time with membership and equivalence queries.

We consider the class of conjunctions of two unate DNF formulas. We call this class $\mathcal{R}^2$. We show that this class is polynomial-query learnable, but can be properly learned in polynomial time with membership and equivalence queries if and only if P = NP. Therefore, the barrier to learning this class is computational, and not informational. It is easy to construct artificial classes that also have these properties, but it can be quite difficult to determine whether such properties hold for more natural classes of functions. [1]

[1] Aizenstein et al. did show that similar properties hold for the class of unions of k graphic halfspaces [1]. All functions in this class are expressible as monotone 2-DNF formulas and thus can be easily learned in terms of a 2-DNF representation using a polynomial number of membership queries (and no equivalence queries). However, Aizenstein et al. showed that unions of k graphic halfspaces cannot be properly learned in polynomial-time with membership and equivalence queries unless NP = co-NP. Using the techniques of Pillaikammnatt and Raghavan [11], and a result of Hellerstein et al. [9], this result can easily

The class of monotone DNF formulas is learnable in polynomial time with membership and equivalence queries [3]. The more general class of unate DNF formulas is also learnable in polynomial time in this model [5]. In a unate DNF formula, each variable either always appears in its negated form, or always appears without negation. The class $\mathcal{R}^2$ is a simple extension of the class of unate DNF formulas. It is also a simple extension of the class of 2-clause CNF formulas, because each clause of a 2-clause CNF formula is unate. The class of 2-clause CNF formulas is properly learnable in polynomial time [2].

The characterization of Hellerstein et al. [9] says that a class C can be properly learned using a polynomial number of polynomial-size membership and equivalence queries iff *C has polynomial-size certificates*. Roughly, C has polynomial-size certificates if, for all functions g not representable by a "small" formula in C, there is a polynomial-size set of points in the domain of g (the *certificate*), such that the value of g on those points proves that g is not representable by a "small" formula in C.

We show that $\mathcal{R}^2$ has polynomial-size certificates. It is therefore properly learnable with membership and equivalence queries by an algorithm using a polynomial number of polynomial-size queries. However, we show that $\mathcal{R}^2$ can be learned in polynomial time in this model if and only if P = NP. We also show that this NP-hardness result can be extended to $\mathcal{R}^k$, the class of conjunctions of k unate DNF formulas, for any constant $k \geq 2$. However, it is still open whether the certificate result can be extended to $k > 2$.

We also prove a structural result that has no immediate implications for learning. Using a construction based on BCH codes, we show that there are functions g on n variables that cannot be expressed by a formula in $\mathcal{R}^2$, but for which there is no certificate of size polynomial in n proving that fact. (This does not contradict our earlier claim that $\mathcal{R}^2$ has polynomial-size certificates. Recall that such certificates need only prove that g cannot be expressed as a "small" conjunction of two unate DNF formulas). We also extend this result to $\mathcal{R}^k$ for all constant $k \geq 2$. This is in sharp contrast to the situation for other classes of formulas such as monotone and unate DNF formulas, and Horn formulas. If a function cannot be expressed by a formula in one of these classes, a constant sized certificate exists proving that fact. It is easy to show that for the class of monotone DNF formulas, a certificate of size 2 exists, and for the class of unate DNF formulas, a certificate of size 4 exists. Ekin, Hammer, and Peled showed that for the class of Horn formulas, a certificate of size 3 exists [7].

It remains an intriguing open question whether the class $\mathcal{R}^k$ can be learned in polynomial time with membership and *improper* equivalence queries for any fixed $k > 1$.

be improved to state that unions of k graphic halfspaces can be properly learned in polynomial time with membership and equivalence queries if and only if P = NP.

1.2 Relation to work on monotone dimension

Our interest in conjunctions of unate DNF formulas is motivated in part by recent work of Bshouty [6]. Bshouty defined the *monotone dimension* of a Boolean function. The monotone dimension of a function f is the minimum number d such that f can be written as the conjunction of d unate DNF formulas. (In his paper, Bshouty actually gave a different definition. We show the equivalence of the two definitions in Section 2.3.)

The functions of monotone dimension 1 are precisely those representable by unate DNF formulas. The functions of monotone dimension k are precisely those representable by a formula in $\mathcal{R}^k$. For example, a simple function of monotone dimension 2 is $f(x_1, x_2) = x_1 \oplus x_2 = (x_1 \vee x_2) \wedge (\neg x_1 \vee \neg x_2)$. Since any clause in a CNF formula is unate, the monotone dimension of a function f is bounded above by CNF-size(f), the minimum number of clauses in any CNF formula representing f.

In a unate DNF formula, a variable has a *positive orientation* if it appears in the formula without negation, and a *negative orientation* if it appears with a negation. Any variable not appearing in the formula can be viewed as having either a negative or a positive orientation. An *orientation* of n variables is an assignment a to those n variables, where $a(x) = 0$ means x has a positive orientation, and $a(x) = 1$ means x has a negative orientation. Let $f : \{0,1\}^V \rightarrow \{0,1\}$. A set $\{a_1, \ldots, a_z\}$ of assignments to V is a *monotone basis* for f if f can be expressed as the conjunction of unate DNF formulas $f_1, \ldots, f_z$, such that $a_1, \ldots, a_z$ are orientations consistent with $f_1, \ldots, f_z$ respectively. Thus the monotone dimension of a function is the size of the smallest monotone basis for the function. We define the monotone dimension of a Boolean formula to be the monotone dimension of the function it represents.

Bshouty gave an algorithm for learning DNF formulas (with membership and improper equivalence queries) which builds a monotone basis for the target formula f as it learns. However, the basis built by the algorithm may not be of minimum size. Its size is bounded above by CNF-size(f). The running time of the algorithm is polynomial in n, the number of terms in f, and in CNF-size(f). Note that if f is a unate DNF formula, CNF-size(f) may be exponential in the number of terms of f. Thus in learning a unate DNF f, Bshouty's algorithm may take time exponential in n and the number of terms of f. In contrast, the algorithm of [5] for learning unate DNF formulas f finds a basis of size 1 for f and runs in time polynomial in n and the number of terms in f.

We are interested in the following general question: Is there a membership and improper equivalence query algorithm that learns DNF formulas in time polynomial in n, the size of f, and in the monotone dimension of f, rather than in n, the size of f, and CNF-size(f)? This general question is still open. A restricted version of this question is as follows: For fixed constant k, is there a membership and improper equivalence query algorithm that learns DNF formulas of monotone dimension k, in time polynomial in n, and the size of f? The NP hardness result of Section 4 suggests that if such an algorithm exists, it is unlikely to work by constructing a basis of minimum size. More specifically, unless P = NP there is no algorithm of this type that uses only equivalence queries from $\mathcal{R}^k$. The restricted question is still open for every fixed $k > 1$.

We show in Section 3 that the class $\mathcal{R}^2$ *can* be learned by a membership and equivalence query algorithm having polynomial query complexity (i.e., the number and size of the queries is polynomial in n and the size of the target formula), but the amount of time the algorithm takes is not polynomial.

2 Preliminaries

2.1 Basic Definitions

Let V be a finite set of Boolean variables. An *assignment* a to V is a mapping $a : V \to \{0,1\}$. Let $f : \{0,1\}^V \to \{0,1\}$ be a Boolean function. We sometimes treat a as a binary vector indexed by the elements of V. Let $f' : \{0,1\}^V \to \{0,1\}$. Then $f \subseteq f'$ if for all assignments a to V, $f(a) = 1$ implies $f'(a) = 1$. For the purposes of this paper, we restrict all assignments, functions, and formulas to be Boolean.

If $f(a) = 1$ then a is a *positive* assignment of f, otherwise a is a *negative* assignment of f. Let $x \in V$. If $i \in \{0,1\}$, then $a_{x \leftarrow i}$ is the assignment b such that $b(y) = a(y)$ for all $y \in V - \{x\}$, and $b(x) = i$. The assignment $a_{x \leftarrow \neg a(x)}$ is denoted by $a_{\neg x}$. If $x \notin V$, then $a_{x \leftarrow i}$ is the assignment to $V \cup \{x\}$ such that $b(y) = a(y)$ for all $y \in V$, and $b(x) = i$.

The *complement* of assignment a, denoted $\overline{a}$, is the assignment such that $\overline{a}(x) \neq a(x)$ for all $x \in V$. The *Hamming distance* between two assignments a and b, denoted by $\delta_H(a,b)$, is defined as follows: $\delta_H(a,b) = |\{x \in V \mid a(x) \neq b(x)\}|$.

For any assignment $b \in \{0,1\}^V$, and $V' \subseteq V$, we define the assignment $b|_{V'} : V' \to \{0,1\}$ such that for all $x \in V'$, $b|_{V'}(x) = b(x)$.

A partial assignment to V is a function $p : V \to \{0,1,\star\}$. If $p(x) = \star$, we say that x is not assigned by p. If a is an assignment to V, p/a denotes the *extension* of p obtained by setting the unassigned variables according to a, i.e., $(p/a)(x) = p(x)$ if $p(x) \neq \star$, and $(p/a)(x) = a(x)$ otherwise.

If p is a partial assignment to V, the *projection* f_p of f is the function $f_p : \{0,1\}^V \to \{0,1\}$ such that for all $a \in \{0,1\}^V$, $f_p(a) = f(p/a)$. There is also a natural definition of f_p on a restricted domain as follows: $f_p : \{0,1\}^{V_S} \to \{0,1\}$ where $V_S = \{x \in V | p(x) = \star\}$, and $f_p(a) = f_p(a|_{V_S})$ for all $a \in \{0,1\}^V$.

The function f is *monotone in* x if, for every assignment a to V, $f(a_{x\leftarrow 0}) \leq f(a_{x\leftarrow 1})$. Similarly, f is *anti-monotone in* x if for every assignment a to V, $f(a_{x\leftarrow 0}) \geq f(a_{x\leftarrow 1})$. The function f is a *monotone function* if it is monotone in every $x \in V$. It is an *anti-monotone function* if it is anti-monotone in every $x \in V$. It is a *unate function* function if, for every variable $x \in V$, it is monotone or anti-monotone in x.

Let S be a set of assignments to V. The set S is a *certificate* that f has property P if every function consistent with f on S has property P. That is, if $g : \{0,1\}^V \to \{0,1\}$ satisfies $f(a) = g(a)$ for all $a \in S$, then g has property P. For example, if a and b are assignments such that $f(a_{x\leftarrow 0}) = 1$, $f(a_{x\leftarrow 1}) = 0$, $f(b_{x\leftarrow 0}) = 0$ and $f(b_{x\leftarrow 1}) = 1$, then $S = \{a_{x\leftarrow 0}, a_{x\leftarrow 1}, b_{x\leftarrow 0}, b_{x\leftarrow 1}\}$ is a certificate that f is not unate.

2.2 Query Learning

In learning functions with queries, the goal of the learning algorithm is to identify a hidden *target* function f taken from a given class C of functions. The learning algorithm is expected to represent functions in C using an associated set of formulas $\mathcal{F}$. The algorithm is given as input the set of n variables V_n on which f is defined.

We consider here the most common combination of queries: membership and equivalence [3]. A membership query asks for the value of f on an assignment a. The answer to the query is $f(a)$. An equivalence query asks whether a hypothesis formula $h \in \mathcal{F}$ is equivalent to the function f, i.e., whether h represents f. The answer is "yes" if $h \equiv f$, otherwise the answer is a *counterexample* b such that $h(b) \neq f(b)$. *Improper* equivalence queries use hypotheses h that are not in $\mathcal{F}$. Throughout this paper, we use the term *equivalence query* to refer only to proper (i.e., not improper) equivalence queries; we explicitly use the longer term *improper equivalence query* where appropriate.

We restrict our attention to the problem of learning Boolean functions. Let $C = \bigcup_{n\geq 1} C_n$ be a class of Boolean functions, where C_n is a set of functions defined on a set V_n of n Boolean variables. Let $\mathcal{F} = \bigcup_{n\geq 1} \mathcal{F}_n$ be an associated class of Boolean formulas computing the functions in C. That is, $\mathcal{F}_n$ is a set of formulas on V_n computing only functions in C_n, and each function in C_n is computed by at least one formula in $\mathcal{F}_n$.

An algorithm *learns C with respect to representation class $\mathcal{F}$ with membership and equivalence queries* if, given oracles answering membership and equivalence queries for any function $f \in C$, the algorithm will eventually halt and output a formula $F \in \mathcal{F}$ such that $F \equiv f$ (the formula F computes the function f).

We often say simply that the algorithm *learns $\mathcal{F}$ with membership and equivalence queries*, and talk about target formulas rather than target functions. If an algorithm uses improper equivalence queries to learn $\mathcal{F}$, and those equivalence queries are all drawn from another class $\mathcal{H}$, we say the algorithm *learns $\mathcal{F}$ in terms of $\mathcal{H}$*.

With every class of formulas $\mathcal{F}$ we associate a *size* function mapping elements in $\mathcal{F}$ to the natural numbers. For $F \in \mathcal{F}$, the *size of formula F*, written $|F|_{\mathcal{F}}$, is the value of the size function on F. Where $\mathcal{F}$ is understood from context, we will write simply $|F|$. We assume that all size functions are such that $|F|$ is bounded above by a polynomial in the number of bits in a standard binary encoding of the formula F.

The *size* of function f with respect to formula class $\mathcal{F}$, written $|f|_{\mathcal{F}}$, is the size of the smallest formula in $\mathcal{F}$ computing f, as measured by the size function associated with $\mathcal{F}$. If f cannot be computed by a formula in $\mathcal{F}$, then $|f|_{\mathcal{F}} = \infty$. (Defining the size of functions not computable by formulas in $\mathcal{F}$ simplifies the statements of various results.)

A *polynomial-time* learning algorithm for $\mathcal{F}$ runs in time polynomial in n and $|f|_{\mathcal{F}}$, where f is the target function, and n is the number of variables on which f is defined.

The class $\mathcal{F}$ is *polynomially recognizable* if there is an algorithm that given (the encoding of) any formula F as input, decides whether $F \in \mathcal{F}$ in time polynomial in the length of the encoding of F.

A class $\mathcal{F}$ of formulas is *polynomial-query learnable* if there exists a membership and equivalence query algorithm for learning $\mathcal{F}$ with the following properties:
(1) All equivalence queries are formulas from $\mathcal{F}$, and they have size at most polynomial in $|f|$ and n, where f is the target function, and n is the number of variables on which f is defined.
(2) The total number of queries is polynomial in $|f|$ and n.

2.3 Monotone Dimension

The monotone dimension was introduced by Bshouty. We give the basic definitions. Additional details can be found in [6].

Let a and b be assignments to a set of n variables V_n. Let $a + b$ denote the sum of vectors a and b over GF[2]. The relation $a < b$ means that $a(x) \leq b(x)$ for all $x \in V_n$, and $a(y) < b(y)$ for some $y \in V_n$. Let c be an assignment to V_n. The relation $a <_c b$ means that $a + c < b + c$.

For a Boolean function $f : \{0,1\}^V \rightarrow \{0,1\}$, the *monotone closure of f with respect to c*, written $\mathcal{M}_c(f)$, is a Boolean function on V defined as follows:

$$\mathcal{M}_c(f)(b) = \begin{cases} 1 & (\exists a \leq_c b)\, f(a) = 1 \\ 0 & \text{otherwise.} \end{cases}$$

A set $\{b_1, \ldots, b_d\}$ satisfying

$$f \equiv \bigwedge_{i=1}^{d} \mathcal{M}_{b_i}(f).$$

is a *monotone basis* for f. The *monotone dimension* of a Boolean function f, written $Mdim(f)$, is the smallest number d such that there exists a monotone basis for f of size d.

The following lemma establishes the connection between functions of monotone dimension k, and conjunctions of k unate DNF formulas.

Lemma 1 *A Boolean function f has monotone dimension at most k iff it can be expressed as the conjunction of at most k unate DNF formulas.*

Proof: If $Mdim(f) \leq k$, then there exists a basis $\{b_1, \ldots, b_{k'}\}$, $k' \leq k$, such that $f = \mathcal{M}_{b_1}(f) \wedge \ldots \wedge \mathcal{M}_{b_k}(f)$. Since each $\mathcal{M}_{b_i}(f)$ is a unate function, each f_i can be written as a unate DNF formula.

Suppose $f_1, \ldots, f_k$ are unate DNF formulas such that $f = f_1 \wedge \ldots \wedge f_k$. For each f_i, since f_i is unate, there exists a consistent orientation b_i such that $\mathcal{M}_{b_i}(f_i) = f_i$. We will show that $f = \mathcal{M}_{b_1}(f) \wedge \ldots \wedge \mathcal{M}_{b_k}(f)$. Since $f \subseteq \mathcal{M}_b(f)$ for any assignment b, $f \subseteq \mathcal{M}_{b_1}(f) \wedge \ldots \wedge \mathcal{M}_{b_k}(f)$.

For all f_i, $\mathcal{M}_{b_i}(f) = \mathcal{M}_{b_i}(f_1 \wedge \ldots \wedge f_k) \subseteq \mathcal{M}_{b_i}(f_i)$. Therefore, $\mathcal{M}_{b_1}(f) \wedge \ldots \wedge \mathcal{M}_{b_k}(f) \subseteq \mathcal{M}_{b_1}(f_1) \wedge \ldots \wedge \mathcal{M}_{b_k}(f_k) = f_1 \wedge \ldots \wedge f_k = f$. ∎

The following is a useful lemma.

Lemma 2 *Let $f : \{0,1\}^V \rightarrow \{0,1\}$. Let $b_1, \ldots, b_k$ be assignments to V. Then $\{b_1, \ldots, b_k\}$ is not a monotone basis for f iff there exist k (not necessarily distinct) positive assignments $p_1, \ldots, p_k$ of f and a negative assignment q of f such that for all $i \in [1 \ldots k]$, $p_i <_{b_i} q$.*

Proof: Suppose $\{b_1, \ldots, b_k\}$ is not a basis for f. Then $f \neq \mathcal{M}_{b_1}(f) \wedge \ldots \wedge \mathcal{M}_{b_k}(f)$. Since $f \subseteq \mathcal{M}_{b_1}(f) \wedge \ldots \wedge \mathcal{M}_{b_k}(f)$, there exists an assignment q such that $f(q) = 0$ and $\mathcal{M}_{b_1}(f) \wedge \ldots \wedge M_{b_k}(f)(q) = 1$. Then for all $i \in [1 \ldots k]$, there exists a positive assignment p_i of f such that $p_i <_{b_i} q$.

Conversely, suppose there exist positive assignments $p_1, \ldots, p_k$ and a negative assignment q of f such that for all $i \in [1 \ldots k]$, $p_i <_{b_i} q$. Then $f(q) = 0$, but $\mathcal{M}_{b_1}(f)(q) \wedge \ldots \wedge \mathcal{M}_{b_i}(f)(q) = 1$. Therefore, $\{b_1, \ldots, b_k\}$ is not a monotone basis for f. ∎

The previous lemma immediately implies a characterization of functions with monotone dimension greater than k.

Lemma 3 *Let f be a Boolean function defined on variable set V. Then $Mdim(f) > k$ iff for all $k' \leq k$ assignments $b_1, \ldots, b_{k'}$ to V, there exist k' positive assignments $p_1, \ldots, p_{k'}$ of f, and one negative assignment q of f, such that for all $i \in [1 \ldots k']$, $p_i <_{b_i} q$.*

Lemma 2 also implies the following certificate characterization:

Lemma 4 *Let $f : \{0,1\}^V \to \{0,1\}$ and let $b_1, \ldots, b_k$ be assignments to V. Then S is a certificate that $\{b_1, \ldots, b_k\}$ is not a monotone basis for f iff S contains positive assignments $p_1, \ldots, p_k$ of f, and a negative assignment q of f, such that for all $i \in [1 \ldots k]$, $p_i <_{b_i} q$.*

Proof: Let S be a certificate that $\{b_1, \ldots, b_k\}$ is not a monotone basis for f. Suppose there are no assignments $p_1, \ldots, p_k, q \in S$ satisfying the properties stated in the lemma. Let $g : \{0,1\}^V \to \{0,1\}$ be such that for all assignments a to V, $g(a) = 1$ iff (1) $a \in S$ and $f(a) = 1$ or (2) there exist $p_1, \ldots, p_k \in S$ such that $f(p_1) = \ldots = f(p_k) = 1$, and for all $i \in [1 \ldots k]$, $p_i \leq_{b_i} a$. Then $g = \mathcal{M}_{b_1}(g) \wedge \ldots \wedge \mathcal{M}_{b_k}(g)$. Therefore, S is not a certificate that $\{b_1, \ldots, b_k\}$ is not a monotone basis for f. Contradiction.

The other direction of the lemma follows immediately from Lemma 2. ■

2.4 The formula classes $\mathcal{R}^k$ and $\mathcal{D}^k$

For $k \geq 1$, let $\mathcal{D}^k$ be the class of DNF formulas of monotone dimension at most k. Let $\mathcal{R}^k$ be the class of formulas $F = F_1 \wedge \ldots \wedge F_{k'}$, $k' \leq k$, where each F_i is a unate DNF formula. By Lemma 1, $\mathcal{D}^k$ and $\mathcal{R}^k$ represent the same class of functions, those of monotone dimension at most k. For $F \in \mathcal{D}^k$, we define $|F|_{\mathcal{D}^k}$ to be the number of terms in the DNF formula F. For $F \in \mathcal{R}^k$, we define the size function $|F|_{\mathcal{R}^k}$ to be the total number of terms in the (at most) k DNF formulas composing F.

Let f be a Boolean function such that $Mdim(f) \leq k$. By Lemma 1, it can be represented by a formula in $\mathcal{R}^k$. Then $|f|_{\mathcal{R}^k}$ is the minimum total number of terms in unate DNF formulas $F_1, \ldots, F_{k'}$, where $F_1 \wedge \ldots \wedge F_{k'}$ computes f and $k' \leq k$. Also, $|f|_{\mathcal{D}^k}=$ DNF-size(f), where DNF-size(f) is defined to be the minimum number of terms in any DNF formula representing f (all DNF formulas expressing f are in $\mathcal{D}^k$, because $Mdim(f) \leq k$).

Lemma 5 *Let f be a Boolean function. Let $s = |f|_{\mathcal{R}^k}$ and let $t = |f|_{\mathcal{D}^k}$. If $Mdim(f) \leq k$, then $s \leq tk$ and $t \leq s^k$. Otherwise, $|f|_{\mathcal{R}^k} = |f|_{\mathcal{D}^k} = \infty$.*

Proof: If $Mdim(f) > k$ then by Lemma 1 there is no formula in either $\mathcal{R}^k$ or $\mathcal{D}^k$ computing f, and $|f|$ is infinite with respect to both classes.

Suppose $Mdim(f) \leq k$. Let $F = F_1 \wedge \ldots \wedge F_{k'} \in \mathcal{R}^k$ be a formula expressing f whose total number of terms is s. The DNF obtained from F by "multiplying" the DNF formulas $F_1, \ldots, F_{k'}$ has at most $s^{k'} \leq s^k$ terms. Let $G = T_1 \vee \ldots \vee T_t$ in $\mathcal{D}^k$ be a DNF formula expressing f, where the T_i are terms. Let $\{b_1, \ldots, b_z\}$, $z \leq k$, be a monotone basis for f. Then $f = \mathcal{M}_{b_1}(f) \wedge \ldots \wedge \mathcal{M}_{b_z}(f)$. For $j \in [1 \ldots z]$, a unate DNF formula for $\mathcal{M}_{b_j}(f)$ can be obtained from G as follows: (1) Delete from each term T_i of G all negated literals $\neg x$ in T such that $b_j(x) = 0$, and all non-negated literals x such that $b_j(x) = 1$. (2) If any term becomes empty, replace the term with the constant 1. (The correctness of this procedure follows from the fact that $\mathcal{M}_b(f_1 \vee f_2) = \mathcal{M}_b(f_1) \vee \mathcal{M}_b(f_2)$.) Therefore, $\mathcal{M}_{b_j}(f)$ can be written as a unate DNF formula with at most t terms. The function f can be written as the conjunction of at most k unate DNF formulas, each of which has at most t terms, yielding a total of at most kt terms. ■

It follows from the above that for constant k, $|f|_{\mathcal{R}^k}$ and $|f|_{\mathcal{D}^k}$ are polynomially related. Thus a polynomial-time algorithm for learning $\mathcal{R}^k$ with membership and equivalence queries would also be a polynomial-time algorithm for learning $\mathcal{D}^k$ in terms of $\mathcal{R}^k$ with membership and improper equivalence queries.

Despite the similarities between the classes $\mathcal{R}^k$ and $\mathcal{D}^k$, there is an important difference. (Note that for the case $k = 1$, $\mathcal{R}^k$ is the class of unate DNF formulas, and $\mathcal{D}^k$ is the class of DNF formulas representing unate functions.) The class $\mathcal{R}^k$ is polynomially recognizable, because given a formula, it is easy to check whether it is of the form $g_1 \wedge \ldots \wedge g_k$, where $g_1, \ldots, g_k$ are unate DNF formulas. However, the following reduction from CNF-SAT shows that unless P = NP, the class $\mathcal{D}^k$ is not polynomially recognizable. Given a CNF formula g, construct a DNF formula $\neg g \vee h$ where h is a DNF formula computing parity of $x_1, \ldots, x_{k+1}$, and $x_1, \ldots, x_{k+1}$ are variables not appearing in g. It is easy to show that $Mdim(h) = 2^k$. If g is satisfiable, then $Mdim(\neg g \vee h) \geq 2^k$. If g is not satisfiable, $Mdim(\neg g \vee h) = 1$, the monotone dimension of the identically true function. Since $2^k > k$ for all $k \geq 1$, g is satisfiable iff $Mdim(\neg g \vee h) > k$.

For $k \geq 2$, our results on the NP-hardness of learning $\mathcal{R}^k$ with membership and equivalence queries (Sections 4 and 5) show the hardness of learning $\mathcal{R}^k$ in terms of $\mathcal{R}^k$. They also imply a hardness result for learning $\mathcal{D}^k$ in terms of $\mathcal{R}^k$ (because $|f|_{\mathcal{R}^k}$ and $|f|_{\mathcal{D}^k}$ are polynomially related). However, they do not imply the hardness of learning $\mathcal{D}^k$ in terms of $\mathcal{D}^k$. Moreover, the technique used to prove the hardness of learning $\mathcal{R}^k$ is not applicable to $\mathcal{D}^k$ because $\mathcal{D}^k$ is not a polynomially recognizable class. (However, we suspect that D^k is hard to learn in terms of D^k anyway.)

3 Conjunctions of two unate DNF formulas are polynomial-query learnable

In this section we show that $\mathcal{R}^2$ is polynomial-query learnable. In our proof, we actually show that $\mathcal{D}^2$ is polynomial-query learnable. Since $|f|_{\mathcal{R}^k}$ and $|f|_{\mathcal{D}^k}$ are polynomially related, and polynomial-query learnability does not take into account the time spent in computing representations, proving $\mathcal{D}^2$ is polynomial-query learnable implies that $\mathcal{R}^2$ is also polynomial-query learnable.

The proof that $\mathcal{D}^2$ is polynomial-query learnable relies on the connection between query complexity and the existence of polynomial-size certificates. We present a restricted version of a theorem of Hellerstein et al. (The original theorem applies to languages as well as functions.)

Theorem 1 [9] *A representation class $\mathcal{R}$ (for Boolean functions) is polynomial-query learnable iff there exist polynomials $p()$ and $q()$ such that for all $m, n > 1$ and all Boolean functions f on n variables, if $|f|_R > p(m,n)$, then there exists a certificate of size at most $q(m,n)$ that $|f|_R > m$.*

In fact, if the second part of the above theorem is satisfied with polynomials $p()$ and $q()$, then there exists a (computationally inefficient) membership and equivalence query algorithm for learning $\mathcal{R}$ that, in learning functions of size m, makes equivalence queries using hypotheses of size at most $p(m,n)$.

We prove that for $\mathcal{D}^2$, Theorem 1 is satisfied with $p = m^2$ and $q = O(n^2 + m^4)$. That is, we show there exist certificates of size $O(n^2 + m^4)$ separating functions f with DNF-size greater than m^2 or dimension greater than 2 (i.e., functions f such that $|f|_{D^2} > m^2$) from functions with DNF-size at most m and dimension at most 2.

One approach to proving that $\mathcal{D}^2$ has polynomial size certificates would be to show that (1) any function of monotone dimension greater than 2 has a certificate of size polynomial in n proving that fact and (2) any function that has monotone dimension at most 2 but DNF-size greater than m has a certificate of size polynomial in n and m proving its DNF-size is greater than m. This approach works for $\mathcal{D}^1$ (unate DNF): if a function is not unate, there is a certificate of size 4 attesting to that fact, and if it is unate but has DNF-size greater than m, there is a certificate of size polynomial in m and n that its DNF-size is greater than m. However, the results of Section 4 show that this approach will not work for $\mathcal{D}^2$, as functions with dimension greater than 2 do not necessarily have certificates of size polynomial in n that their dimension is greater than 2. We therefore use another approach. The next lemma specifies two types of functions for which we will find certificates.

Lemma 6 *Let $f : \{0,1\}^V \rightarrow \{0,1\}$ such that $Mdim(f) > 2$ or DNF-size(f) $> m^2$. Then there is a projection $g = f_p$ of f such that either*

Condition 1: *$Mdim(g) \leq 2$ and DNF-size(g) $> m^2$ or*

Condition 2: *$Mdim(g) > 2$ and DNF-size(g) $\leq 2m^2$.*

Proof: Let $f : \{0,1\}^V \rightarrow \{0,1\}$ such that $Mdim(f) > 2$ or DNF-size(f) $> m^2$. We will build a projection p on f such that f_p always maintains this initial condition, and such that eventually f_p satisfies either Condition 1 or 2. We start with the empty projection (i.e., with $f_p = f$).

Suppose $Mdim(f_p) \leq 2$. Then by the initial condition, DNF-size(f_p) $> m^2$. Therefore, f_p satisfies Condition 1.

Suppose $Mdim(f_p) > 2$. If DNF-size(f_p) $\leq 2m^2$, then f_p satisfies Condition 2. Assume DNF-size(f_p) $> 2m^2$. Let x be a variable of f_p, and consider the two projections of f_p setting x to 0 and 1 respectively. Since $f = xf_{p_{x \leftarrow 1}} \vee \neg x f_{p_{x \leftarrow 0}}$, either DNF-size($f_{p_{x\leftarrow 1}}$) $> \frac{1}{2}$ DNF-size(f_p) or DNF-size($f_{p_{x\leftarrow 0}}$) $> \frac{1}{2}$ DNF-size(f_p).

Extend p to either the 0 projection or the 1 projection on x, such that the resulting projection has DNF-size at least $\frac{1}{2}$ of the DNF-size of f_p. Since DNF-size(f_p) $> 2m^2$ before the extension of p, DNF-size(f_p) $> m^2$ after the extension of p. Thus f_p after the extension still obeys the initial condition.

Continue extending p in this way until f_p satisfies either Condition 1 or 2. It must eventually satisfy one of these conditions because the extension only continues while DNF-size(f_p) $> 2m^2$. ∎

In the above lemma, we showed that if $Mdim(f) > 2$ or DNF-size(f) $> m^2$, then f has a projection f_p that satisfies Condition 1 or 2. Moreover, $Mdim(f_p) > 2$ or DNF-size(f_p) $> m^2$. A certificate proving that $Mdim(f_p) > 2$ or DNF-size(f_p) $> m^2$ can easily be turned into a certificate that f has these properties. We therefore, without loss of generality, restrict our attention to functions f obeying Condition 1 or 2. We show that each such f has a polynomial-size certificate that $Mdim(f) > 2$ or DNF-size(f) $> m$. We first prove that if f obeys Condition 1, it has a polynomial-size certificate that its DNF-size is greater than m.

Lemma 7 *Let $C = \bigcup_{n\geq 1} C_n$ be the class of Boolean functions f such that $Mdim(f) \leq 2$ and DNF-size(f) $> m^2$. Then each $f \in C$ has a certificate of size $O(mn)$ that it has DNF-size greater than m.*

Proof: Let f be a function in C_n. Let $\{b_1, b_2\}$ be a monotone basis for f (b_1 and b_2 not necessarily distinct). Let f_1 be the (minimal) unate DNF formula for $\mathcal{M}_{b_1}(f)$, and let f_2 be the (minimal) unate DNF formula for $\mathcal{M}_{b_2}(f)$. Since $\{b_1, b_2\}$ is a basis for f, $f = f_1 \wedge f_2$. A DNF formula for f can be formed by "multiplying" the terms of f_1 by the terms of f_2. Since DNF-size(f) $> m^2$, either DNF-size(f_1) $> m$ or

DNF-size(f_2) $> m$. Suppose without loss of generality that DNF-size(f_1) $> m$. Let $T_1, \ldots, T_{m+1}$ be $m+1$ terms of f_1. Let $A = \{a_1, \ldots, a_{m+1}\}$ be the set of minimal assignments, with respect to the partial order $<_{b_1}$, satisfying $T_1, \ldots, T_{m+1}$. For each a_i, let $B_i = \{b | \; b = a_{i\,\neg x}$ for some variable x appearing in a literal of T_i $\}$.

Let $S = A \cup B_1 \cup \ldots \cup B_{m+1}$. Note that S contains $O(mn)$ assignments. We show that S is a certificate that DNF-size(f) $\geq m+1$. Assume that there exists a DNF g of size less than $m+1$ consistent with f on S. Then there must be two distinct assignments $a_i, a_j \in A$ satisfying the same term T of g. T_i contains a literal l not contained in T_j. Assignments a_i and a_j must differ on the value they give to the underlying variable x of the literal l. Therefore, neither x nor $\neg x$ appears in T. Thus assignment $a_{i\,\neg x} \in B_i$ satisfies T and $g(a_{i\,\neg x}) = 1$. However, $f(a_{i\,\neg x}) = 0$. Hence g is not consistent with f on S. ■

In the following lemmas, we show that any function $f : \{0,1\}^V \rightarrow \{0,1\}$ satisfying Condition 2 has a polynomial-size certificate that $Mdim(f) > 2$. Our proof is constructive.

Let b_1, b_2 be a pair of assignments to V. We define a *suitable triple* for $\{b_1, b_2\}$ (with respect to f) to be a triple (p_1, p_2, q) where p_1 and p_2 are positive assignments of f, q is a negative assignment, $p_1 \leq_{b_1} q$, and $p_2 \leq_{b_2} q$.

To construct the certificate that $Mdim(f) > 2$, we generate suitable triples for all pairs of assignments b_1, b_2. By Lemma 2, the elements in these triples form a certificate that $Mdim(f) > 2$. Our construction is based on the following technical lemma.

Lemma 8 *Let $f : \{0,1\}^V \rightarrow \{0,1\}$. Let b_1, b_2 be a pair of (not necessarily distinct) assignments to V such that for all $x \in V$, if $b_1(x) = b_2(x) = 0$ then f is monotone in x, and if $b_1(x) = b_2(x) = 1$ then f is anti-monotone in x.*

If $\{b_1, b_2\}$ is not a monotone basis for f, then f has a projection $f_z : \{0,1\}^W \rightarrow \{0,1\}$, $W \subseteq V$, such that either

1. *$|W| = 2$ and f_z has exactly three positive assignments, two of which are the complementary assignments $b_1|_W$ and $b_2|_W$, or*
2. *$|W| \geq 2$ and f_z has exactly two, complementary, positive assignments, which are $b_1|_W$ and $b_2|_W$.*

Proof: Let b_1, b_2, and f be as defined in the first paragraph of the lemma. Suppose $\{b_1, b_2\}$ is not a monotone basis for f.

By Lemma 2, there exists at least one suitable triple for b_1, b_2. Choose a suitable triple (p_1, p_2, q)

for b_1, b_2 such that the size of the set $W_{(p_1,p_2,q)} = \{x \in V | p_1(x), p_2(x)$, and $q(x)$ are not all equal $\}$ is minimized. Let $W = W_{(p_1,p_2,q)}$.

Claim 1: $p_1|_W = \overline{p_2|_W}$.

Suppose not. Then there exists $x \in W$ such that $p_1(x) = p_2(x)$. Without loss of generality, assume $p_1(x) = p_2(x) = 0$. Since $x \in W$, $q(x) = 1$. Because (p_1, p_2, q) is a suitable triple for b_1, b_2, $p_1 <_{b_1} q$ and $p_2 <_{b_2} q$, $f(p_1) = f(p_2) = 1$, and $f(q) = 0$. Therefore, $b_1(x) = b_2(x) = 0$. By the assumption in the statement of the lemma, f is monotone in x. Therefore, $f({p_1}_{x\leftarrow 1}) = 1$ and $f({p_2}_{x\leftarrow 1}) = 1$. Thus $({p_1}_{x\leftarrow 1}, {p_2}_{x\leftarrow 1}, q)$ is a suitable triple for b_1, b_2. But then $W_{({p_1}_{x\leftarrow 1},{p_2}_{x\leftarrow 1},q)} = W \setminus \{x\}$, which contradicts the minimality property of (p_1, p_2, q).

Claim 2: $b_1|_W = \overline{b_2|_W}$.

Suppose not. Then there exists $x \in W$ such that $b_1(x) = b_2(x)$. By Claim 1, $p_1(x) \neq p_2(x)$. Without loss of generality, assume $b_1(x) = b_2(x) = 0$, $p_1(x) = 0$, and $p_2(x) = 1$. Since $p_2(x) = 1$, $b_2(x) = 0$, and $p_2 <_{b_2} q$, it follows that $q(x) = 1$. By the assumption in the statement of the lemma, f is monotone in x. Therefore, since $f(p_1) = 1, f({p_1}_{x\leftarrow 1}) = 1$. But then $({p_1}_{x\leftarrow 1}, p_2, q)$ contradicts the minimality property of (p_1, p_2, q).

Claim 3: $b_1|_W = p_1|_W$ and $b_2|_W = p_2|_W$.

By Claim 2, $b_1|_W = \overline{b_2|_W}$. Since $p_2|_W <_{b_2|_W} q|_W$, it follows that $q|_W <_{b_1|_W} p_2|_W$. Thus $p_1|_W <_{b_1|_W} q|_W <_{b_1|_W} p_2|_W$. By Claim 1, $p_1|W = \overline{p_2|_W}$. It follows that $p_1|W = b_1|_W$, and hence $b_2|_W = p_2|_W$.

Let $z : V \to \{0, 1, *\}$ be the partial assignment such that for all $x \in W$, $z(x) = *$, and for all $x \notin W$, $z(x) = p_1(x)$ $(= p_2(x) = q(x))$. Consider the projection $f_z : \{0,1\}^W \to \{0,1\}$ induced by z. Since $f(p_1) = f(p_2) = 1$ and $f(q) = 0$, it follows that $f_z(p_1|_W) = f_z(p_2|_W) = 1, f_z(q|_W) = 0$. Thus $p_1|_W <_{b_1|_W} q|_W <_{b_1|_W} p_2|_W$ and f_z has at least 3 distinct assignments. Clearly, $|W| \geq 2$.

We will show that either f_z has exactly two positive assignments which are $p_1|_W$ and $p_2|_W$, or $|W| = 2$ and f_z has the unique negative assignment $q|_W$.

Case 1: $|W| = 2$

In this case, f_z has four different assignments. $p_1|_W$ and $p_2|_W$ are two positive assignments and $q|_W$ is a negative assignment. If the fourth assignment is negative, then f_z has exactly 2 positive assignments, $p_1|_W = b_1|_W$ and $p_2|_W = b_2|_W$. Otherwise, $q|_W$ is the unique negative assignment of f_p.

Case 2: $|W| > 2$

Suppose (for contradiction) that f_z has at least one positive assignment other than $p_1|_W$ and $p_2|_W$. Let a_1 be such a positive assignment, and let a_2 be a negative assignment of f_z such that $p_1|_W <_{b_1|_W} a_2 <_{b_1|_W} p_2|_W$ (a_2 exists because $q|_W$ satisfies this relation). Choose a_1 and a_2 such that $\delta_H(a_1, a_2)$ is minimized.

Suppose $a_1 <_{b_1|_W} a_2$. Then $(z/a_1, z/p_2, z/a_2)$ is a suitable triple for b_1, b_2. Since $a_1 \neq p_1|_W$, there exists $x \in W$ such that $a_1(x) = \overline{p_1}(x) = p_2(x)$. Since $a_1 <_{b_1|_W} a_2 <_{b_1|_W} p_2|_W$, $a_1(x) = a_2(x)$, and therefore $W \cup \{x\} \subseteq W_{(z/a_1, z/p_2, z/a_2)}$, contradicting the minimality property of (p_1, p_2, q). Therefore, $a_1 \not<_{b_1|_W} a_2$. Similarly, $a_2 \not<_{b_1|_W} a_1$.

Therefore, there exist $x, y \in W$ such that $a_1(x) = p_1(x)$, $a_2(x) = \overline{p_1}(x) = p_2(x)$, and $a_1(y) = p_2(y)$, $a_2(y) = \overline{p_2}(y) = p_1(y)$. If $a_{2\neg x} \neq p_1|_W$ (respectively, $a_{2\neg y} \neq p_2|_W$), then $a_1, a_{2\neg x}$, (respectively $a_1, a_{2\neg y}$) violate the minimal Hamming distance condition on a_1 and a_2. Therefore, $a_{2\neg x} = p_1|_W$ and $a_{2\neg y} = p_2|_W$. But then since $p_1|_W = \overline{p_2|_W}$, $|W| = 2$, which contradicts the premise of Case 2 that $|W| > 2$. Therefore, f_z has no positive assignments other than $p_1|_W$ and $p_2|_W$. ■

We are now ready to give the construction of the polynomial-size certificates.

Lemma 9 *Let Q be the class of Boolean functions f such that $Mdim(f) > 2$ and DNF-size$(g) \leq 2m^2$. Then each $f \in Q$ has a certificate of size $O(m^4 + n^2)$ that it has monotone dimension greater than 2.*

Proof: Let $f : \{0,1\}^V \to \{0,1\}$ be a function in Q. Let $T_1 \vee \ldots \vee T_k$ be a DNF formula for f where $k \leq 2m^2$. We will show that there exists a set S of size $O(m^4 + n^2)$ that contains suitable triples with respect to f for every potential basis pair $\{b_1, b_2\}$. By Lemma 4, S is a certificate that $Mdim(f) > 2$.

For each non-monotone variable x of f, choose a pair of assignments a, $a_{\neg x}$ such that $a(x) = 1$, $f(a) = 0$, and $f(a_{\neg x}) = 1$. Similarly, for each variable x of f such that f is not anti-monotone in x, choose a pair a, $a_{\neg x}$ such that $a(x) = 0$, $f(a) = 0$, and $f(a_{\neg x}) = 1$. Denote the resulting set of $O(n)$ assignments by A.

We now form another set B. For each pair of distinct variables $x, y \in V$, and for each of the four assignments $r : \{x, y\} \to \{0,1\}$ to x, y, do the following. If there exists a projection $f_p : \{0,1\}^{\{x,y\}} \to \{0,1\}$ of f such that f_p has the unique negative assignment r, choose one such p. In B, place the three assignments $p/r, p/r_{\neg x}, p/r_{\neg y}$. Since we place at most three assignments in B for each x, y, the size of B is $O(n^2)$.

We construct a third set C as follows. For each pair of terms T_i and T_j of the DNF for f, do the following. Let W be the set of variables which appear in opposite forms (i.e., negated in one, not negated in the other) in T_i and T_j. If $|W| \geq 2$, check whether there exists a projection $f_p : \{0,1\}^W \to \{0,1\}$ of f such that $f_p(a) = 1$ iff a either satisfies all literals in T_i formed from variables in W, or all literals in T_j formed from variables in W. If so, then pick one such p. Place in C the two distinct assignments p/b and p/c such that $b, c : \{0,1\}^W \to \{0,1\}$ and $f_p(b) = f_p(c) = 1$, and one assignment p/d such that $d : \{0,1\}^W \to \{0,1\}$ and $f_p(d) = 0$. Since DNF-size$(f) \leq 2m^2$, C contains $O(m^4)$ assignments.

Let $S = A \cup B \cup C$. We show that S has a suitable triple for every $\{b_1, b_2\}$.

First, let $\{b_1, b_2\}$ be such that $b_1(x) = b_2(x) = 0$ for some $x \in V$, and f is not monotone in x. Consider the pair of assignments a, $a_{\neg x}$ placed in A because f is not monotone in x. By the definition of a, $a(x) = 1$, $f(a) = 0$, and $f(a_{\neg x}) = 1$. Therefore, $a_{\neg x} <_{b_1} a$ and $a_{\neg x} <_{b_2} a$. So the triple (p_1, p_2, q) where $p_1 = p_2 = a_{\neg x}$, and $q = a$ is suitable for $\{b_1, b_2\}$.

Similarly, for any variable x of f such that f is not anti-monotone in x, the corresponding pair of assignments a and $a_{\neg x}$ in A will also form a suitable triple for any pair $\{b_1, b_2\}$ such that $b_1(x) = b_2(x) = 1$.

We now show that B and C contain suitable triples for all other possible pairs b_1, b_2. Let b_1, b_2 be such that for all variables x, if $b_1(x) = b_2(x) = 0$ then f is monotone in x, and if $b_1(x) = b_2(x) = 1$ then f is anti-monotone in x. By Lemma 8, there exists a projection $f_z : \{0,1\}^W \to \{0,1\}$ of f such that either

1. $|W| = 2$ and f_z has exactly three positive assignments, two of which are the complementary assignments $b_1|_W$ and $b_2|_W$, or
2. $|W| \geq 2$ and f_z has exactly two, complementary, positive assignments, which are $b_1|_W$ and $b_2|_W$.

If the first condition holds, let q be the unique negative assignment for f_z. Let $W = \{x, y\}$. Then by the definition of B, there exists a projection $f_{p'} : \{0,1\}^W \to \{0,1\}$ such that q is the unique negative assignment for $f_{p'}$, and B contains $p'/q, p'/q_{\neg x}$ and $p'/q_{\neg y}$. The assignments $q_{\neg x}$ and $q_{\neg y}$ are the unique complementary positive assignments for $f_{p'}$, and are therefore equal to $b_1|_W$ and $b_2|_W$. It is easy to verify that $(p'/q_{\neg x}, p'/q_{\neg y}, p'/q)$ is a suitable triple for $\{b_1, b_2\}$.

If the second condition holds, f_z has exactly two, complementary, positive assignments $b_1|_W$ and $b_2|_W$. Since $f(z/b_1) \neq f(z/{b_1}_{\neg x})$ for all $x \in W$, z/b_1 must satisfy a term T_i of the DNF for f such that T_i contains a literal for every variable in W. Similarly, z/b_2 must satisfy a term T_j such that T_j contains a literal for every variable in W. Since $b_1|_W = \overline{b_2|_W}$ have complementary assignments on W, $T_i \neq T_j$. Since z/b_1 and z/b_2 assign the same values to variables in $V - W$, for $x \in V$, T_i and T_j contain literals x and $\neg x$ respectively iff $x \in W$. Therefore, for some projection $f_{p'} : \{0,1\}^W \to \{0,1\}$, the set C must contain assignments p'/b, p'/c and p'/d where $b, c, d : W \to \{0,1\}$, such that b and c are the unique positive assignments of $f_{p'}$, d is a negative assignment of $f_{p'}$, b satisfies the literals in T_i formed from variables in W, and c satisfies the literals in T_j formed from variables in W. Without loss of generality, assume $b|_W = b_1|_W$ and $c|_W = b_2|_W$. Thus for all $x \in W$, $b(x) + b_1(x) \leq d(x) + b_1(x)$ and $c(x) + b_2(x) \leq d(x) + b_2(x)$. (addition is over GF[2].) This implies $p'/b(x) \leq_{b_1} p'/d(x)$ and $p'/c(x) \leq_{b_2} p'/d(x)$. Thus $(p'/b, p'/c, p'/d)$ is a suitable triple for $\{b_1, b_2\}$. ■

Theorem 2 *The classes $\mathcal{R}^2$ and $\mathcal{D}^2$ are polynomial-query learnable.*

Proof: Let $m, n > 1$, and let f be a Boolean function on n variables such that $|f|_{\mathcal{D}^2} > m^2$. We show that if $|f|_{\mathcal{D}^2} > m^2$, then f has a certificate of size $O(m^4 + n^2)$ that $|f|_{\mathcal{D}^2} > m$. The theorem then follows from Theorem 1, and previous arguments relating $\mathcal{R}^2$ and $\mathcal{D}^2$.

If $|f|_{\mathcal{D}^2} > m^2$, then either f has monotone dimension greater than 2 (and hence $|f|_{\mathcal{D}^2} = \infty$) or its DNF-size is greater then m^2. By Lemma 6, f either has a projection g satisfying Condition 1 ($Mdim(g) \leq 2$ and DNF-size$(g) > m^2$) or Condition 2 ($Mdim(g) > 2$ and DNF-size$(g) \leq 2m^2$). If f has a projection satisfying Condition 1, then by Lemma 7, f has a certificate of size $O(mn)$ that it has DNF-size greater than m. If f has a projection satisfying Condition 2, then by Lemma 9, f has a certificate of size $O(m^4 + n^2)$ that it has dimension greater than 2. Either certificate shows that $|f|_{\mathcal{D}^2} > m$. ■

4 Poly-time learning conjunctions of 2 unate DNF formulas is hard

We show that a polynomial-time membership and equivalence algorithm for learning $\mathcal{R}^2$ would imply a polynomial-time algorithm for the set-splitting problem, which is NP-hard. In the appendix, we extend this NP-hardness result to $\mathcal{R}^k$ for all constant $k > 2$. We will use a theorem of Pillaipakkamnatt and Raghavan [11]. First, we give the necessary definitions.

Let $\mathcal{F}$ and $\mathcal{C}$ be two classes of formulas. The class $\mathcal{F}$ is *testable with respect to* $\mathcal{C}$ if there exists an algorithm $\mathcal{A}_{\mathcal{F}}$ and a polynomial function $t()$ such that if $\mathcal{A}_{\mathcal{F}}$ is given encodings of formulas $c \in \mathcal{C}_n$ and $f \in \mathcal{F}_n$ as input, $\mathcal{A}_{\mathcal{F}}$ halts in time $t(n, |c|, |f|)$ (where $|c|$ and $|f|$ denote the size of the encodings of c and f) and outputs one of the following:

1. A *counterexample* a, which is an assignment such that $f(a) \neq c(a)$.
2. A formula $c' \in \mathcal{C}_n$ such that $f \equiv c'$.

The problem *REP*($\mathcal{C}$) *for* $\mathcal{F}$ takes as input a formula $f \in \mathcal{F}$ and determines whether there exists a formula in $\mathcal{C}$ that is equivalent to f.

Theorem 3 [11] *Let $\mathcal{C} = \bigcup_{n \geq 1} \mathcal{C}_n$ be a polynomially recognizable class of Boolean formulas and let $p()$ be a polynomial function such that for each $c \in \mathcal{C}_n$, the size of the encoding of c is at most $p(n)$. Let $\mathcal{F} = \bigcup_{n \geq 1} \mathcal{F}_n$ be a class of Boolean formulas that is testable with respect to $\mathcal{C}$. If $\mathcal{C}$ is learnable in polynomial time in the membership and equivalence query model, then $REP(\mathcal{C})$ for $\mathcal{F}$ is solvable in polynomial time.*

Proof: Let $\mathcal{H}^2$ be the class of DNF formulas in which each term has exactly one satisfying assignment (i.e., each term contains all n variables as literals). We will show that for $\mathcal{C} = \mathcal{R}^2$, $\mathcal{F} = \mathcal{H}^2$, the conditions of the above theorem hold.

It is clear that $\mathcal{R}^2$ is polynomial-time recognizable. We will show that $\mathcal{H}^2$ is testable with respect to $\mathcal{R}^2$ and that $REP(\mathcal{R}^2)$ for the class $\mathcal{H}^2$ is NP-hard. The theorem then implies that if P $\neq$ NP, $\mathcal{R}^2$ is not learnable in polynomial time with membership and equivalence queries.

Lemma 10 *$\mathcal{H}^2$ is testable with respect to $\mathcal{R}^2$.*

Suppose we are given formulas $g \in \mathcal{R}^2$ and $h \in \mathcal{H}^2$ both defined on variable set V. Let m_h be the number of terms of h, which is also the number of assignments satisfying h. We present an algorithm that, given $h \in \mathcal{H}^2$ and $g \in \mathcal{R}^2$, either produces a counterexample a such that $h(a) \neq g(a)$, or determines that $h \equiv g$.

g is the conjunction of two unate DNF formulas g_1 and g_2. Compute an equivalent DNF formula $\hat{g}$ by "multiplying" the terms of g_1 and g_2. Let $m_{\hat{g}}$ be the number of terms in $\hat{g}$.

First, we check if $h \subseteq \hat{g}$. For each term of h, check whether the unique satisfying assignment a of h also satisfies $\hat{g}$, (i.e., if $\hat{g}(a) = 1$). If not, output "No" with assignment a as the counterexample. There are at most m_h such tests.

Now we check if $\hat{g} \subseteq h$. For each term of $\hat{g}$, begin enumerating its satisfying assignments. For each satisfying assignment that is enumerated, check if $h(a) = 1$. If not, stop and output "No" with assignment a as the counterexample. Note that if some term of $\hat{g}$ yields more than m_h satisfying assignments, there must be some a such that $\hat{g}(a) = 1$ and $h(a) = 0$. Therefore, at most $m_h m_{\hat{g}}$ assignments are enumerated.

If $\hat{g} \subseteq h$ and $h \subseteq \hat{g}$, we output g. The total time is polynomial in n and the sizes of the encodings of g and h. ∎

To prove that REP($\mathcal{R}^2$) for $\mathcal{H}^2$ is NP-hard, we do a two-stage reduction from the known NP-complete problem *set-splitting*. First, we introduce the *assignment separator problem*.

Assignment Separator Problem

Instance: A set X of n Boolean variables $x_1 \dots x_n$. A set $A = \{a_1 \dots a_t\}$ of assignments to $x_1 \dots x_n$.

Question: Does there exist an assignment b to $x_1 \dots x_n$ such that for all distinct a_i, $a_j \in A$, there exist x_k, $x_l \in X$ such that

$$a_i(x_k) \neq a_j(x_k),\ a_i(x_k) = b(x_k)$$

$$a_i(x_l) \neq a_j(x_l),\ a_j(x_l) = b(x_l)$$

An assignment b satisfying the given properties is an *assignment separator* for (A, X).

Lemma 11 *The assignment separator problem is NP-complete.*

Proof: The problem is clearly in NP. We show that set-splitting is polynomial-time reducible to the assignment separator problem.

Set Splitting

Instance: Collection C of subsets of a finite set S.

Question: Is there a partition of S into two subsets S_1 and S_2 such that no subset in C is entirely contained in either S_1 or S_2?

Let $C = \{c_1, c_2, \ldots, c_m\}, S = \{s_1, s_2, \ldots, s_n\}$ be an instance of the set-splitting problem. If any set $c_i \in C$ has at most one element, the answer to the set-splitting problem is trivially "no". We assume therefore that each c_i has at least 2 elements.

For each subset c_i, create an $n + m$ dimensional Boolean vector a_i. The first n positions describe the subset: if $c_i \in C$ contains $s_j \in S$, then a_i has a 1 in position j, otherwise it has a 0. The final m positions pad the vector in a simple way: a_i has a 1 in position $n + i$ and has 0's elsewhere.

We also create m additional vectors, $z_1 \ldots z_m$, also of dimension $n + m$. Vector z_i has a 1 in position $n + i$ and 0's in the other $n + m - 1$ positions. Note that z_i is identical to a_i in the final m positions, and differs from a_i in at least two of the first n positions because set c_i contains more than one element.

Let $X = \{x_1, x_2, \ldots, x_{n+m}\}$. Let $A = \{a_1, a_2, \ldots, a_m\} \cup \{z_1, z_2, \ldots, z_m\}$. We claim that the answer to the assignment separator problem on this instance is "yes" if and only if the answer to the set-splitting instance is "yes".

Suppose b is an assignment separator for (A, X). Let $S_1 = \{s_i \in S | b(x_i) = 0\}$ and let $S_2 = S - S_1$. We show that S_1 and S_2 split the subsets in C. Consider a set $c_i \in C$ and its corresponding vectors a_i and z_i. There exist $x_k, x_l \in X$ such that

$$a_i(x_k) \neq z_i(x_k),\ a_i(x_k) = b(x_k)$$

$$a_i(x_l) \neq z_i(x_l),\ z_i(x_l) = b(x_l).$$

Because a_i and z_i are identical on the final m bits, we have $1 \leq k, l \leq n$. Moreover, z_i is 0 on the first n bits. Therefore, $a_i(x_k) = a_i(x_l) = 1$. Then $s_k, s_l \in c_i$. But we also have $b(x_k) = 1$, $b(x_l) = 0$. Then $s_k \in S_2$ and $s_l \in S_1$. Thus $c_i \not\subseteq S_1$ and $c_i \not\subseteq S_2$, as required.

Now suppose there is a solution (S_1, S_2) to the set-splitting instance S, C. Let b be an $n+m$ dimensional Boolean vector determined as follows:

$$b(x_i) = \begin{cases} 0 & 1 \le i \le n \text{ and } s_i \in S_1 \\ 1 & 1 \le i \le n \text{ and } s_i \in S_2 \\ 1 & n+1 \le i \le n+m. \end{cases}$$

We claim that b is an assignment separator for (A, X). Consider any two assignments in A. There are three cases:

Case I: The two assignments are a_i and a_j.

$$\begin{array}{rll} b(x_{n+i}) = & a_i(x_{n+i}) = 1 \neq 0 = a_j(x_{n+i}) & \\ & a_i(x_{n+j}) = 0 \neq 1 = a_j(x_{n+j}) & = b(x_{n+j}) \end{array}.$$

Case II: The two assignments are a_i and z_j. There exist $s_k, s_l \in c_i$ such that $s_k \in S_1$ and $s_l \in S_2$. Then $a_i(x_k) = a_i(x_l) = 1$. Also, $z_i(x_k) = z_i(x_l) = 0$ since $k, l \le n$. We have $b(x_k) = 0$, $b(x_l) = 1$.

Case III: The two assignments are z_i and z_j. Identical to Case I. ■

The following lemma will be useful in proving that $REP(\mathcal{R}^2)$ for $\mathcal{H}^2$ is NP-hard.

Lemma 12 *Let f be a function of monotone dimension 2 such that all variables of f are non-unate. If $\{a, b\}$ is a basis for f, then $a = \overline{b}$.*

Proof: Let $\{a, b\}$ be such that for some x, $a(x) = b(x)$. Without loss of generality, assume $a(x) = b(x) = 0$. Since f is non-unate in x, there exists an assignment c such that $c(x) = 1$, $f(c) = 0$, and $f(c_{x \leftarrow 0}) = 1$. Then $(c_{x \leftarrow 0}, c_{x \leftarrow 0}, c)$ is a suitable triple for a, b, and by Lemma 4, $\{a, b\}$ is not a basis for f. ■

Lemma 13 *The problem $REP(\mathcal{R}^2)$ for $\mathcal{H}^2$ is NP-hard.*

Proof: We show that the assignment separator problem is polynomial-time reducible to the problem $REP(\mathcal{R}^2)$ for $\mathcal{H}^2$. Let (A, X) be an instance of the assignment separator problem. Check whether $\delta_H(a_i, a_j) = 1$ for any $a_i, a_j \in A$. If so, the answer to the assignment separator problem is "no," and we can perform a trivial reduction.

Assume the Hamming distance between any two assignments in A is at least 2. Let $X' = \{x \in X |$ for some $a_i, a_j \in A$, $a_i(x) \neq a_j(x)\}$. For each $a_i \in A$, let $a'_i = a_i|_{X'}$. Let $A' = \{a'_i | a_i \in A\}$. Clearly (A, X) has an assignment separator iff (A', X') has an assignment separator.

For each $a'_i \in A'$, form a term T_i such that T_i is satisfied by a'_i, and T_i contains a literal x or $\neg x$ for every variable $x \in X'$. Let f be the DNF formula over the variables in X' which is the disjunction of all such T_i. Clearly $f \in \mathcal{H}^2$, since each term of f has exactly one satisfying assignment. Also, f's positive assignments are precisely the assignments in A', and f is non-unate in each variable in X'. We now show that (A', X') has an assignment separator iff f is representable as a formula in $\mathcal{R}^2$.

Suppose f is representable as a formula in $\mathcal{R}^2$. Then f has a monotone basis $\{b_1, b_2\}$. Since f is non-unate in all its variables, by Lemma 12, $b_1 = \overline{b_2}$. We show that b_1 is an assignment separator for (A', X').

Let a'_i, a'_j be distinct assignments in A'. Suppose $a'_i <_{b_1} a'_j$. Since the Hamming distance between a'_i and a'_j, and between all two assignments in A' is at least 2, there exists a negative assignment d of f such that $a'_i <_{b_1} d <_{b_1} a'_j$. Since $b_1 = \overline{b_2}$, $a'_j <_{b_2} d <_{b_2} a'_i$. But then $\mathcal{M}_{b_1}(f)(d) \wedge \mathcal{M}_{b_2}(f)(d) = 1$ which contradicts that $f = \mathcal{M}_{b_1}(f) \wedge \mathcal{M}_{b_2}(f)$ and $f(d) = 0$. By symmetry, we cannot have $a'_j <_{b_1} a'_i$. Hence a'_i and a'_j are not comparable with respect to $<_{b_1}$. It follows that there exist $x_k, x_l \in X'$ such that $b_1(x_k) = a'_j(x_k) \neq a'_i(x_k)$ and $b_1(x_l) = a'_i(x_l) \neq a'_j(x_l)$, so b_1 is an assignment separator for (A', X').

Conversely, suppose b is an assignment separator for (A', X'). We claim that in this case $\{b, \bar{b}\}$ is a monotone basis for f, and hence $Mdim(f) \leq 2$. Assume, for the purpose of contradiction, that $\{b, \bar{b}\}$ is not a monotone basis for f. Then there exists an assignment d such that $f(d) = 0$ but $\mathcal{M}_b(f)(d) = \mathcal{M}_{\bar{b}}(f)(d) = 1$. Thus there exist positive assignments $a'_i, a'_j \in A'$ of f such that $a'_i <_b d$ and $a'_j <_{\bar{b}} d$, and so $a'_i <_b d <_b a'_j$. But then for all x such that $a'_i(x) \neq a'_j(x)$, $a'_i(x) = b(x)$. This contradicts that b is an assignment separator for (A', X'). ■

The final theorem now follows:

Theorem 4 *There is a proper polynomial-time membership and equivalence query algorithm for learning $\mathcal{R}^2$ if and only if $P = NP$.*

Proof: The "only if" direction follows immediately from Lemmas 10 and 13 and from the theorem of Pillaipakkamnatt and Raghavan (Theorem 3).

Because $\mathcal{R}^2$ can be recognized and evaluated in polynomial time and $\mathcal{R}^2$ has polynomial-size certificates, results of Hellerstein et al. [9] imply that $\mathcal{R}^2$ can be learned in polynomial time using a Σ_4^p oracle. If P = NP, then $\Sigma_4^p = $ P, and we can learn $\mathcal{R}^2$ in polynomial time. ■

5 Certificates for monotone dimension greater than 2

In this section we construct a class of functions $\mathcal{F}$ which have dimension greater than 2, but no certificates of size polynomial in n attesting to that fact. An extension of this construction for dimension $k > 2$ is presented in the appendix.

We construct $\mathcal{F}$ so that for $f \in \mathcal{F}_n$, the Hamming distance between any two positive assignments of f is at least polylogarithmic in n. As shown below, if C is a certificate that $Mdim(f) > 2$, this Hamming distance property ensures that C has super-polynomial size. It is not enough, however, to construct arbitrary functions f obeying the above Hamming distance property; we must also ensure that, in fact, $Mdim(f) > 2$. This precludes the use of some simple classes of functions obeying the Hamming distance property. For instance, consider the Boolean functions f whose only positive assignments are the all 0's assignment, the all 1's assignment, and an assignment that is half 1's and half 0's. These functions certainly obey the Hamming distance property, but it can be shown that their monotone dimension is at most 2.

We construct the functions in $\mathcal{F}$ using a binary BCH code. We first present a well known result regarding binary BCH codes, and then explain its implications for our construction.

Lemma 14 (see, e.g., [10]) *For any positive integers m, t such that $t < 2^{m-1}$, there exists a binary BCH code with block length $n = 2^m - 1$, and $n - k \leq mt$ check digits, such that $\delta_H(s_1, s_2) > 2t + 1$ for any two codewords s_1, s_2.*

Let n be such that $n = 2^m - 1$ for some positive integer m for which $m^2 < 2^{m-1}$. Let $t = m^2 = \log^2(n+1)$. In addition, let n be large enough such that $2\log^3(n+1) < n$.

For these values of n and t, the above lemma guarantees the existence of a set S_n of binary n-vectors (codewords) having the following properties. The proof of these properties follow immediately from the above lemma and elementary properties of linear codes.

1. The vectors in S_n form a group with respect to component-wise addition over GF[2].
2. The vectors in S_n are a subspace of the space of n-vectors over GF[2].
3. There is an $(n-k) \times n$ matrix (called the parity check matrix) $A = [P|I]$, where P is an $(n-k) \times k$ matrix and I is the $(n-k) \times (n-k)$ identity matrix, such that S_n is the set of binary n-vectors satisfying the matrix equation $Ay = \vec{0}$ over GF[2].
4. The number of vectors in S_n is 2^k.

5. $\delta_H(s_i, s_j) \geq 2(\log^2(n+1)) + 1$ for any distinct $s_i, s_j \in S_n$.

6. $k \geq n - (\log^3(n+1))$.

We can also prove two additional properties of the submatrix P of the parity check matrix.

7. Each row of P is non-zero.

8. Each column of P is non-zero.

The proof of Property 7 is as follows. If a row of P contains all 0's, then that one check bit of the code computes the parity of an empty set of variables, and each codeword is zero in the position corresponding to that check bit. Since BCH codes are cyclic codes, any cyclic shift of a codeword is itself a codeword. Thus for any position, there is some codeword with a 1 in that position, which is a contradiction.

Property 8 follows immediately from the fact that no two codewords are Hamming distance 1 apart.

Definition of $\mathcal{F}$: For each n satisfying the conditions stated above, define the Boolean function f on n variables which is 1 on assignments corresponding to elements in S_n, and 0 on all other assignments. The class $\mathcal{F}$ consists of all such f.

Lemma 15 *For $f \in \mathcal{F}$, f is non-unate in each of its variables.*

Proof: Let $f : \{0,1\}^V \rightarrow \{0,1\}$ be a function in $\mathcal{F}_n$. Let $x_i \in V$. Since $A\vec{0} = \vec{0}$, $\vec{0} \in S_n$ and $f(\vec{0}) = 1$ (here we treating the vectors as assignments). Consider $\vec{0}_{x_i \leftarrow 1}$. Since no two vectors in S_n are Hamming distance 1 apart, $\vec{0}_{x_i \leftarrow 1} \notin S_n$ and thus $f(\vec{0}_{x_i \leftarrow 1}) = 0$. Hence f is not monotone in x_i.

We now show that f is not anti-monotone in x_i. Consider the column c of A corresponding to x_i. Recall that $A = [P|I]$.

Case 1: Column c is in P.
In this case, there exists a set C of columns of I such that the sum (over GF[2]) of the columns in $C \cup \{c\}$ is $\vec{0}$. Form an n-vector y' with 1's in the positions corresponding to columns in $C \cup \{c\}$, and 0's in all other columns. Thus $Ay' = \vec{0}$, $f(y') = 1$ and (because no two positive examples of f are Hamming distance 1 apart) $f(y'_{x_i \leftarrow 0}) = 0$. Therefore, f is not anti-monotone in x_i.

Case 2: Column c is in I.
Column c has exactly one 1, in some row r. Let c' be a column of P which has a 1 in row r (c' exists because no row of P is all 0's). Then c', c, and the other columns of I sharing 1's in the rows in which c'

has 1's, sum to $\vec{0}$ (over GF[2]). Let y' be the vector which is 1 precisely in the positions corresponding to these columns. Then $Ay' = \vec{0}$, $f(y') = 1$, and $f(y'_{x_i \leftarrow 0}) = 0$. Therefore f is not anti-monotone in x_i. ■

Lemma 16 *For all $f \in \mathcal{F}$, $Mdim(f) > 2$.*

Proof: Let $f \in \mathcal{F}_n$. By Lemma 15, f is non-unate in each of its variables, and therefore $Mdim(f) > 1$. Therefore, by Lemma 12, if f has dimension 2, then any monotone basis $\{b, d\}$ for f must consist of a complementary pair of assignments.

We will show that for any potential complementary basis pair $(b, \overline{b})$, there is a suitable triple (p_1, p_2, q). By Lemma 3, this proves f has monotone dimension greater than 2. Note that since b and $\overline{b}$ are complementary, (p_1, p_2, q) is a suitable triple for $b, \overline{b}$ iff $p_1 <_b q <_b p_2$.

Consider the group of binary n-vectors with respect to componentwise addition over GF[2]. S_n is a subgroup of that group. Consider the cosets $C_1, C_2, ..., C_q$ of S_n (since $|S_n| = 2^k$, $q = 2^{n-k}$). Let a, b be two assignments in the same coset of S_n. Suppose (p_1, p_2, q) is a suitable triple for $(a, \overline{a})$. Then $p_1, p_2 \in S_n$, $q \notin S_n$, and $p_1 + a < q + a < p_2 + a$. Therefore, $(p_1 + a + b) + b < (q + a + b) + b < (p_2 + a + b) + b$, and because a and b are in the same coset of S_n, $p_1 + a + b \in S_n$, $p_2 + a + b \in S_n$, and $q + a + b \notin S_n$. Thus $(p_1 + a + b, p_2 + a + b, q + a + b)$ is a suitable triple for $(b, \overline{b})$. To show that f has monotone dimension greater than 2, it therefore suffices to show for every coset C_i, that for one representative $b \in C_i$, there exists a suitable triple (p_1, p_2, q) for $b, \overline{b}$.

Let T be the set of binary n-vectors which are 0 in the first k positions, and take on all possible 2^{n-k} combinations of values in the last $n - k$ positions. We claim that these vectors are all in different cosets of S_n. Note that for any n-vector $y' \in T$ such that $y' = [0|r]$ (where 0 is the $1 \times k$ 0 vector, and r is a $1 \times (n-k)$ vector), $Ay' = r$. Thus for any two distinct vectors y' and y'' in T, $Ay' \neq Ay''$, which implies that y' and y'' are not in the same coset of S_n. Since there are 2^{n-k} cosets, T contains a representative of each coset of S_n.

Now, since $k \geq n - (\log^3(n+1))$ by Property 6, it follows (from our assumption that $n > 2\log^3(n+1)$) that $k > n - k$. Therefore, the columns of P are not all linearly independent and because no column of P is all 0's, there is a non-empty subset Q of at least two columns of P that sum to the 0 column. Let z be a binary n-vector with 1's in precisely the columns corresponding to elements in Q. Clearly $Az = 0$. Note that z has 0's in all positions corresponding to columns of I in $[P|I]$.

Let $b \in T$. Since $A(b + z) = Ab + Az = Ab$, it follows that b and $b + z$ are in the same coset of S. Note that since $b \in T$, by the definition of T, b contains 0's in all columns corresponding to P. Since z contains 1's only in columns corresponding to P, it follows that for all x_i, $b(x_i) \leq (b + z)(x_i)$. Thus $b < b + z$, and hence $\vec{0} <_b z$. Note that $\vec{0}$ and z are both positive assignments of f (since $A\vec{0} = Az = 0$). Since $\delta_H(\vec{0}, z) \neq 1$, there exists a q with $\delta_H(q, \vec{0}) = 1$ such that $\vec{0} <_b q <_b z$. But since $\delta_H(q, \vec{0}) = 1$, $q \notin S_n$ and hence $f(q) = 0$. It follows that $(\vec{0}, z, q)$ is a suitable triple for $b, \overline{b}$. Since b is a representative of an arbitrary coset of S_n, the lemma follows. ■

Let C be a certificate that a function $f \in \mathcal{F}_n$ has monotone dimension greater than 2. We will show that for every $p_1, p_2, q \in C$, (p_1, p_2, q) is a suitable triple for at most a polynomial number of pairs $b, \overline{b}$. Since there are an exponential number of pairs $b, \overline{b}$, it follows that C is of super-polynomial size. Note that if (p_1, p_2, q) is a suitable triple for $b, \overline{b}$, then $p_1 \leq_b q \leq_b p_2$ and hence $p_1 \leq_b p_2$.

Lemma 17 *Let p_1 and p_2 be binary n-vectors. Let $i = \delta_H(p_1, p_2)$. Then there are 2^{n-i} n-vectors b such that $p_1 \leq_b p_2$.*

Proof: Let Q be the set of variables (positions) on which p_1 and p_2 differ, and let R be the set of variables on which p_1 and p_2 agree. If an n-vector b agrees with p_1 on Q, then $b + p_1 \leq b + p_2$ (since $b + p_1$ is all 0 on Q, and $b + p_1$ and $b + p_2$ agree on R). There are 2^{n-i} such vectors b. However, if b disagrees with p_1 on some variable x_i in Q, then $(b + p_1)(x_i) = 1$, $(b + p_2)(x_i) = 0$, and so $p_1 \not\leq_b p_2$. ■

Lemma 18 *Let $f : \{0,1\}^V \rightarrow \{0,1\}$ be a function of n variables. Let C be a certificate that $Mdim(f) > 2$. If the Hamming distance between any two positive assignments of f is at least i, and C contains exactly m positive examples of f, then $m(m-1)(2^{n-i}) \geq 2^n$.*

Proof: By Lemma 3, for all assignments b to V, C must contain assignments p_1, p_2, q forming a suitable triple for $b, \overline{b}$.

Let (p_1, p_2) be a pair of distinct positive assignments in C. Since the distance between any two positive assignments of f is at least i, by Lemma 17 there are at most 2^{n-i} (ordered) complementary basis pairs $(b, \overline{b})$ such that $p_1 \leq_b p_2$.

Since C contains $m(m-1)$ pairs (p_1, p_2) of distinct positive assignments, and there are 2^n possible complementary ordered basis pairs $(b, \overline{b})$, it follows that $m(m-1)2^{n-i} \geq 2^n$. ■

We can now prove the main theorem of this section.

Theorem 5 *There exists a class of functions $\mathcal{F} = \bigcup_{n \geq 1} \mathcal{F}_n$ such that for all $f \in \mathcal{F}$, $Mdim(f) > 2$, and there is no polynomial $p(n)$ satisfying the following property: If $f \in \mathcal{F}_n$ then there is a certificate of size $p(n)$ that $Mdim(f) > 2$.*

Proof: Let $\mathcal{F}$ be as defined above.. By Lemmas 15 and 16, all functions in $\mathcal{F}$ have monotone dimension greater than 2. Recall that $f \in \mathcal{F}_n$ has the property that for any two positive assignments a, b of f, $\delta_H(a, b) \geq 2\log^2(n+1)+1$. Therefore, by Lemma 18, any certificate C that $Mdim(f) > 2$ must have m positive assignments where m satisfies $m(m-1)(2^{n-(2\log^2(n+1)+1)}) \geq 2^n$. That is, $m(m-1) > 2^{2\log^2(n+1)+1}$ and hence m is not polynomial in n. ■

Acknowledgments

We thank Lenny Pitt, Sally Goldman, and the anonymous referees for useful discussions and suggestions. We thank Nader Bshouty for pointing out the relationship given in Lemma 1.

Appendix

Poly-time learning conjunctions of k unate DNF formulas is hard

In this section we generalize the results of the Section 4 to show that conjunctions of k unate DNF formulas, for any constant $k \geq 2$, cannot be learned in polynomial time by a membership and (proper) equivalence query algorithm unless P = NP.

The class $\mathcal{H}^2$ was defined in Section 4 as the class of DNF formulas in which each term has exactly one satisfying assignment. We define the class of functions $\mathcal{H}^i$ recursively as follows for $i > 2$:

$h_i \in \mathcal{H}^i$ for $i > 2$ iff $h_i = \neg x h_{i-1} \vee x \neg T_{a_{i-1}}$ where

- $h_{i-1} \in \mathcal{H}^{i-1}$ is defined on variables W such that $x \notin W$,
- $a_{i-1} \in \{0,1\}^W$ is a positive assignment of h_{i-1}, and
- $T_{a_{i-1}}$ is the unique term containing all variables in W that is satisfied by a_{i-1}.

Note that $\neg T_{a_{i-1}}$ is satisfied by all assignments to W except for a_{i-1}.

Lemma 19 *For $i > 2$, let $h_i \in \mathcal{H}^i$ where $h_i = \neg x h_{i-1} \vee x \neg T_{a_{i-1}}$. Then $Mdim(h_i) = Mdim(h_{i-1}) + 1$.*

Proof:

$\boldsymbol{Mdim(h_i) \leq Mdim(h_{i-1}) + 1}$ Let $k = Mdim(h_{i-1})$. Then $h_{i-1} = f_1 \wedge f_2 \wedge \ldots \wedge f_k$ for some k unate DNF formulas $f_1, \ldots, f_k$. By the definition of h_i, $h_i = (\neg x f_1 \wedge \neg x f_2 \ldots \wedge \neg x f_k) \vee x \neg T_{a_{i-1}}$. Define a formula $h' = (x \vee f_1) \wedge (x \vee f_2) \wedge \ldots \wedge (x \vee f_k) \wedge \neg(x T_{a_{i-1}})$. Clearly $Mdim(h') \leq k+1$. It is easy to verify that $h_{i_{x \leftarrow 1}} \equiv h'_{x \leftarrow 1}$ and $h_{i_{x \leftarrow 0}} \equiv h'_{x \leftarrow 0}$. Therefore, $h_i \equiv h'$.

$\boldsymbol{Mdim(h_{i-1}) + 1 \leq Mdim(h_i)}$ By definition, $h_i = \neg x h_{i-1} \vee x \neg T_{a_{i-1}}$. Let $z = Mdim(h_i)$. Then $h_i = f_1 \wedge \ldots \wedge f_z$ for some unate DNF formulas $f_1, \ldots, f_z$. Assume that each of the unate DNF formulas $f_1, \ldots, f_z$ is minimal, meaning that removing a literal from any term would change the function computed by $f_1 \wedge \ldots \wedge f_z$. Consider the assignment $a_{i-1_{x \leftarrow 1}}$. It does not satisfy h_i so it must not satisfy some f_j, say f_z. Since each assignment at Hamming distance 1 from $a_{i-1_{x \leftarrow 1}}$ does satisfy h_i, we must have $f_z = \neg(x T_{a_{i-1}})$.

Consider f_j where $j < z$. Since the monotone dimension of f is z, $f_j \neq f_z$. Suppose f_j contains the literal $\neg x$. Then, since f_j is unate, there is some assignment c to $W \cup \{x\}$ such that $c(x) = 1$, $f_j(c) = 0$ and, consequently, $h_i(c) = 0$ (recall that W is the set of variables on which h_{i-1} is defined). By the definition of

h_i, the only such assignment c is ${a_{i-1}}_{x\leftarrow 1}$. Moreover, for any assignment c' at Hamming distance 1 from ${a_{i-1}}_{x\leftarrow 1}$, $h_i(c') = 1$ and hence $f_j(c') = 1$. Thus the literals appearing in the unate formula f_j are precisely those which are not satisfied by ${a_{i-1}}_{x\leftarrow 1}$, and it follows that $f_j = \neg(xT_{a_{i-1}}) = f_z$, which is a contradiction. Thus $\neg x$ does not appear in f_j.

For all j where $j < z$, let f'_j be the DNF consisting of those terms of f_j which do not contain x (there must be at least one such term because there exists at least one assignment c such that $c(x) = 0$, $h_i(c) = 1$, and hence $f_j(c) = 1$). We now show the following:

$$f_1 \wedge \ldots \wedge f_z \equiv (x \vee f'_1) \wedge \ldots \wedge (x \vee f'_{z-1}) \wedge f_z \tag{1}$$

Let p_0 and p_1 be the partial assignments to $W \cup \{x\}$ setting x to 0 and 1 respectively, and leaving the variables in W unassigned. We will show the equivalence of the two formulas in Equation 1 by showing that they are equivalent under the projections induced by p_0 and p_1. Clearly ${f_j}_{p_0} \equiv f'_j \equiv (x \vee f'_j)_{p_0}$. Therefore the two formulas are equivalent under the projection induced by p_0. Now consider p_1. For each $j < z$, $(x \vee f'_j)_{p_1} \equiv 1$ so the right side of Equation (1) becomes ${f_z}_{p_1}$. We must show that $(f_1 \wedge \ldots f_z)_{p_1} \equiv {f_z}_{p_1}$. It is trivial that $(f_1 \wedge \ldots \wedge f_z)_{p_1} \subseteq {f_z}_{p_1}$. Let $b : W \to \{0,1\}$ such that b does not satisfy $(f_1 \wedge \ldots \wedge f_z)_{p_1}$. Say b does not satisfy ${f_j}_{p_1}$. Then $b_{x\leftarrow 1}$ does not satisfy f_j, and therefore it does not satisfy h_i either. By the definition of h_i, $b_{x\leftarrow 1} = {a_i}_{x\leftarrow 1}$, which does not satisfy f_z or ${f_z}_{p_1}$. Thus we have ${f_z}_{p_1} \subseteq (f_1 \wedge \ldots \wedge f_z)_{p_1}$

We therefore have $h_i = \neg x h_{i-1} \vee x \neg T_{a_i} \equiv (x \vee f'_1) \wedge \ldots \wedge (x \vee f'_{z-1}) \wedge \neg(xT_{a_i})$. Then ${h_i}_{x\leftarrow 0} = h_{i-1} = f'_1 \wedge \ldots \wedge f'_{z-1}$. Since each f'_j is unate, we have that $Mdim(h_{i-1}) \leq z - 1$. ■

Lemma 20 *$REP(\mathcal{R}^k)$ for $\mathcal{H}^k$ is NP-hard for any constant k.*

Proof: We show that for any $k \geq 3$, $REP(\mathcal{R}^{k-1})$ for $\mathcal{H}^{k-1}$ is polynomial-time reducible to $REP(\mathcal{R}^k)$ for $\mathcal{H}^k$. Since by Lemma 13 $REP(\mathcal{R}^2)$ for $\mathcal{H}^2$ is NP-hard, this proves the lemma.

Let h_{k-1} be an instance of $REP(\mathcal{R}^{k-1})$ for $\mathcal{H}^{k-1}$. Construct a positive assignment a_{k-1} of h_{k-1} and use it to construct a formula $h_k = \neg x h_{k-1} \vee x \neg T_{a_{k-1}}$ in $\mathcal{H}^k$. By the previous lemma, $Mdim(h_k) = Mdim(h_{k-1}) + 1$. Therefore, h_{k-1} can represented by a formula in $\mathcal{R}^{k-1}$ if and only if h_k can represented by a formula in $\mathcal{R}^k$. ■

We now must show that the testability condition holds, as well.

Lemma 21 *$\mathcal{H}^k$ is testable with respect to $\mathcal{R}^k$ for any constant $k \geq 2$.*

Proof: We present an algorithm that, given $h_k \in \mathcal{H}^k$ and $g \in \mathcal{R}^k$, either produces a counterexample

a such that $h_k(a) \neq g(a)$, or determines that $h_k \equiv g$.

Expanding h_k, we have

$$\begin{aligned} h_k &= \neg x_{k-1} h_{k-1} \vee x_{k-1} \neg T_{a_{k-1}} \\ &= \neg x_{k-1}(\neg x_{k-2} h_{k-2} \vee x_{k-2} \neg T_{a_{k-2}}) \vee x_{k-1} \neg T_{a_{k-1}} \\ &= \neg x_{k-1} \neg x_{k-2} h_{k-2} \vee \neg x_{k-1} x_{k-2} \neg T_{a_{k-2}} \vee x_{k-1} \neg T_{a_{k-1}} \\ &= \neg x_{k-1} \neg x_{k-2} \ldots \neg x_2 h_2 \vee (\textstyle\bigvee_{j=2}^{k-2} \neg x_{k-1} \ldots \neg x_{j+1} x_j \neg T_{a_j}) \vee x_{k-1} \neg T_{a_{k-1}} \end{aligned}$$

Let $h = h_k$ and let n be the number of variables on which h is defined. Consider the 2^{k-2} projections of h obtained by fixing $x_2, \ldots, x_{k-1}$. For each projection h_p, take the maximum $i \in \{2, \ldots, k-1\}$ such that x_i is fixed to 1 by the projection. If no x_i is fixed to 1, then $h_p = h_2$. Otherwise, $h_p = (\neg T_{a_i})_p$, which is either a CNF clause of $n-k+2$ literals (if none of $x_{i-1} \ldots x_2$ are fixed by p to satisfy the clause) or is identically true (if some $x \in \{x_{i-1}, \ldots, x_2\}$ is fixed by p to satisfy the clause).

Thus, each projection h_p is either: a) identically 1; b) a clause of $n-k+2$ literals; or c) h_2.

We test the equivalence of h_p and g_p for each partial assignment p. Note that any counterexample to $h_p \equiv g_p$ can be extended to be a counterexample to $h_k \equiv g$ by setting $x_2, \ldots, x_{k-1}$ according to p.

a) $h_p \equiv 1$. Let $g = g_1 \wedge \ldots \wedge g_k$ where $g_1, \ldots, g_k$ are unate DNF formulas. Then $g_p = g_{1p} \wedge \ldots \wedge g_{kp}$. If any of $g_{1p}, \ldots, g_{kp}$ contain some literal $y \notin \{x_2, \ldots, x_{k-1}\}$ then any assignment a which does not satisfy y does not satisfy g_p. Output a as a counterexample. Otherwise, $g_p \equiv 0$ or $g_p \equiv 1$ and we either output an arbitrary assignment as a counterexample or move on to the next projection.

b) h_p is a clause of $n-k+2$ literals. For each literal y in the clause, test the equivalence of the projections $h_{p,y\leftarrow 1}$ and $g_{p,y\leftarrow 1}$. Since $h_{p,y\leftarrow 1} \equiv 1$, we apply the previous case. In addition, test whether $g_p(a) = 0$ for the unique assignment a setting the literals in h_p to 0.

c) $h_p = h_2$. This is the base of the recursion and we apply the algorithm given in the proof of Lemma 10.

If $h_p \equiv g_p$ for each p, then $h_k \equiv g$. If not, the algorithm produces a counterexample via one of the above cases. ■

Again applying the theorem of Pillaipakkamnatt and Raghavan (Theorem 3), we have the following theorem.

Theorem 6 *If $P \neq NP$, then there is no proper polynomial-time membership and equivalence query algorithm for learning $\mathcal{R}^k$, for any constant $k \geq 2$.*

Certificates for functions of monotone dimension greater than k

In this section we generalize the results of Section 5 to construct, for any constant $k > 2$, a class of functions $\mathcal{F}^k$ which have monotone dimension greater than k, but no certificates of size polynomial in n proving this property.

We use the recursive construction from the previous section, but change the base case. The functions in $\mathcal{F}^k$ are defined recursively as follows: f_k defined on variable set V is in $\mathcal{F}^k$ iff $f_k = \neg x(f_{k-1}) \vee x(\neg T_{a_{k-1}})$ for some $f_{k-1} \in \mathcal{F}^{k-1}$ defined on $V - \{x\}$, a_{k-1} is a satisfying assignment for f_{k-1}, and $T_{a_{k-1}}$ is the term corresponding to that assignment. The base case $\mathcal{F}^2$ is the class $\mathcal{F}$ from Section 5.

We need only show that the existence of polynomially sized certificates for $\mathcal{F}^k$ implies the existence of polynomially sized certificates for $\mathcal{F}^{k-1}$.

Let $f_k = \neg x(f_{k-1}) \vee x(\neg T_{a_{k-1}}) \in F_k$, with $f_{k-1} \in F_{k-1}$ and $f_{k-1}(a_{k-1}) = 1$, where f_k is defined on variables V (with $|V| = n$) and f_{k-1} is defined on $V - \{x\}$. The proof of Lemma 19 can be used directly to show that $Mdim(f_k) = Mdim(f_{k-1}) + 1$.

Suppose C_k is a certificate that $Mdim(f_k) > k$. We show that restrictions of the assignments in C form a certificate that $Mdim(f_{k-1}) > k - 1$.

Lemma 22 *Let $f \in \mathcal{F}^k$ where $k \geq 3$. If C_k is a certificate that $Mdim(f_k) > k$, then there is a certificate C_{k-1} that $Mdim(f_{k-1}) > k - 1$ such that $|C_k| \geq |C_{k-1}|$.*

Proof: Define $C_{k-1} = \{c|_{V-\{x\}} \mid c(x) = 0, c \in C_k\}$. We show that C_{k-1} is a certificate that $Mdim(f_{k-1}) > k - 1$.

Let $b_1, \ldots, b_{k-1}$ be assignments to $V - \{x\}$. For $1 \leq i \leq k-1$, let $b'_i = b_{i\,x\leftarrow 0}$. Let $b'_k = a_{k-1\,x\leftarrow 1}$. It follows from Lemma 4 that C_k must contain positive assignments $p'_1, \ldots, p'_k$ and a negative assignment q' of f such that $p'_i <_{b'_i} q'$ for $1 \leq i \leq k$. We will show that $p'_i(x) = q'(x) = 0$ for $1 \leq i \leq k - 1$.

To see that $q'(x) = 0$, note that $q' \neq b'_i$ for any i because $p'_i <_{b'_i} q'$. In particular, $q' \neq b'_k$. Since the only negative assignment of f setting x to 1 is b'_k, we have $q'(x) = 0$. For $i < k$, since both b'_i and q' set x to 0, we must have $p'_i(x) = 0$ because $p'_i <_{b'_i} q'$.

For $1 \leq i < k$, let $p_i = p'_i|_{V-\{x\}}$ and let $q = q'|_{V-\{x\}}$. Then $p_1, \ldots, p_{k-1}, q \in C_{k-1}$. Since $b'_i(x) = p'_i(x) = q'(x) = 0$, $p'_i <_{b'_i} q'$ implies $p_i <_{b_i} q$ for $1 \leq i \leq k-1$.

Then by Lemma 3, C_{k-1} is a certificate that $Mdim(f_{k-1}) > k - 1$. ∎

By the recursive definition of $\mathcal{F}^k$ and the fact that $\mathcal{F}^2$ does not have certificates of size polynomial in

n, we have the following theorem:

Theorem 7 *There exists a class of functions $\mathcal{F}^k$ such that every function in $\mathcal{F}^k$ has monotone dimension greater than k, but there is no polynomial $p(n)$ satisfying the following property: If $f_k \in \mathcal{F}^k$ is defined on n variables, then there is a certificate of size $p(n)$ that f_k has monotone dimension greater than k.*

Proof: Let $\mathcal{F}^k$ be recursively defined as above. Since each function in $\mathcal{F}^2$ has monotone dimension > 2, each function in $\mathcal{F}^k$ has monotone dimension $> k$ (by repeated application of the fact $Mdim(f_k) = Mdim(f_{k-1}) + 1$).

Suppose there exists a polynomial $p(n)$ such that each $f_k \in \mathcal{F}^k$ has a certificate of size at most $p(n)$ that $Mdim(f_k) > k$, where n is the number of variables on which f_k is defined. Let $f_k \in \mathcal{F}^k$ and let C_k be a certificate of size at most $p(n)$ that $Mdim(f_k) > k$. By Lemma 22, C_{k-1} is a certificate of size less than $p(n)$ that $Mdim(f_{k-1}) > k-1$. Recursively, there is a certificate of size less than $p(n)$ that $Mdim(f_2) > 2$. Since f_2 is defined on $n - k + 2$ variables, we define polynomial q such that $q(n - k + 2) = p(n)$. Then functions in $\mathcal{F}^2$ have polynomially sized certificates that they have monotone dimension > 2, contradicting the proof of Theorem 5. ■

References

[1] Aizenstein, H., Hegedűs, T., Hellerstein, L., and Pitt, L. (1996), Complexity theoretic hardness results for query learning, *To appear in Computational Complexity.*

[2] Angluin, D. (1987), Learning k-term dnf formulas using queries and counterexamples, Technical Report YALE/DCS/RR-559, Yale University.

[3] Angluin, D. (1988), Queries and concept learning, *Machine Learning* **2,** 319–342.

[4] Angluin, D. (1992), Computational learning theory: Survey and selected bibliography, *in* "Proceedings of the 24th Annual ACM Symposium on Theory of Computing," pp. 351–369.

[5] Angluin, D., Hellerstein, L., and Karpinski, M. (1993), Learning read-once formulas with queries, *J. ACM* **40,** 185–210.

[6] Bshouty, N. (1995), Exact learning boolean functions via the monotone theory, *Information and Computation* **123,** 146–153.

[7] Ekin, O., Hammer, P., and Peled, U. (1995), Horn functions and submodular boolean functions, Technical Report RRR1-95, Rutgers Center for Operations Research.

[8] Hegedűs, T., (1995) Generalized Teaching Dimensions and the Query Complexity of Learning, *in* "Proceedings of the Eighth Annual ACM Conference on Computational Learning Theory," pp. 108-117.

[9] Hellerstein, L., Pillaipakkamnatt, K., Raghavan, V., and Wilkins, D. (1996), How many queries are needed to learn? *Journal of the ACM* **43** (5), 840-862.

[10] Lin, S. (1971), *An Introduction to Error-Correcting Codes*, Prentice-Hall.

[11] Pillaipakkamnatt, K. and Raghavan, V. (1994), On the limits of proper learnability of subclasses of dnf formulas, *in* "Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory," pp. 118–129.