JACM 43(5):840-862, 1996. Preliminary version appeared in Proceedings of STOC '95

How Many Queries are Needed to Learn?

Lisa Hellerstein, † EECS Dept., Northwestern University
Krishnan Pillaipakkamnatt, ‡ CS Dept., Hofstra University
Vijay Raghavan, ‡§ CS Dept., Vanderbilt University
Dawn Wilkins, ‡CIS Dept., University of Mississippi

Abstract

We investigate the query complexity of exact learning in the membership and (proper) equivalence query model. We give a complete characterization of concept classes that are learnable with a polynomial number of polynomial sized queries in this model. We give applications of this characterization, including results on learning a natural subclass of DNF formulas, and on learning with membership queries alone. Query complexity has previously been used to prove lower bounds on the time complexity of exact learning. We show a new relationship between query complexity and time complexity in exact learning: If any "honest" class is exactly and properly learnable with polynomial query complexity, but not learnable in polynomial time, then P $\neq$ NP. In particular, we show that an honest class is exactly polynomial-query learnable if and only if it is learnable using an oracle for Σ_4^p.

1 Introduction

Today concept learning is studied under two rigorous frameworks which model the notions of effective and efficient learning. The first framework was introduced by Valiant [31] and

†This work was partially supported by NSF grants CCR-9210957 and CCR-9501660.
‡This work was partially supported by NSF grant CCR-9212011.
§Corresponding Author: raghavan@vuse.vanderbilt.edu, Computer Science Department, Box 1679-B, Vanderbilt University, Nashville, TN 37235. Also supported by NSF grant CCR-9510392.

has come to be known as the Probably Approximately Correct (PAC) model of learning. An important breakthrough in the PAC model was the proof [10] that a combinatorial parameter of concept classes plays a central role in efficient learning. This parameter, called the Vapnik-Chervonenkis dimension, provides both necessary and sufficient conditions for learnability with a reasonably small (i.e., polynomial) sample size. In this paper, we give a similar characterization for efficient learning in the second framework.

The second framework, developed primarily by Angluin [3], concerns itself with exact learning with queries. Of the many kinds of queries considered by Angluin, membership and equivalence queries have established themselves as the standard combination to be used for exact learning. There are many reasons for this. First, as shown by Angluin, a concept class that is efficiently exactly learnable with equivalence queries is also efficiently PAC-learnable. Second, research has revealed that there are many interesting concept classes that are not efficiently learnable with either membership or equivalence queries alone, but that are efficiently learnable with both kinds of queries [2, 5, 6, 18, 28]. Equivalence queries may be classified as being *proper* or *improper*. In a proper equivalence query, the hypothesis proposed must be from the same class as the target concept. In this paper, we restrict our attention to proper equivalence queries, and use the term "equivalence query" to refer only to proper equivalence queries.

We give necessary and sufficient conditions for exact learning with a polynomial number of equivalence and membership queries. There are many results in learning theory that are related to our work. For example, Angluin [4] showed that the presence of the approximate fingerprint property in concept classes is a sufficient condition for non-learnability with equivalence queries alone. Recently, Gavaldà [16] proved the converse—the absence of the approximate fingerprint property is a sufficient condition for learnability with a polynomial number of equivalence queries. Goldman and Kearns [17] showed that if a concept class is learnable with a polynomial number of membership queries alone, then it has polynomial teaching dimension. Kannan [21] gave a different kind of sufficient condition for learnability with a polynomial number of improper equivalence and membership queries—that the hypothesis class contain the target class and be closed under union and intersection. This result was refined in [14] to show that membership queries are not needed in this process—if the hypothesis class satisfies the required closure conditions, then the target class is learnable with equivalence queries alone.

Our main results are as follows. We show that a target class is polynomial-query learnable with equivalence and membership queries if and only if it has *polynomial certificates* (defined later). This result immediately implies that a target class is not polynomial-*time* learnable with equivalence and membership queries if it does not have polynomial certificates. A similar characterization has been obtained independently by Hegedüs [19], who builds on Moshkov's earlier work [24] on "conditional tests". An interesting question is whether there are concept classes that are polynomial-query learnable but not polynomial-time learnable. We do not know the answer to this question; however, we prove that if an "honest" concept class is polynomial-query learnable but not polynomial-time learnable with equivalence and membership queries, then $\mathrm{P} \neq \mathrm{NP}$. Equivalently, any honest concept class that has polynomial certificates is learnable within the polynomial-time hierarchy. We give a few applications of our characterization, including a proof that the concept class of functions whose DNF and CNF sizes are polynomially related is polynomial-query learnable with equivalence and membership queries. Bshouty [12] has already shown that this concept class is learnable in polynomial time if one is allowed to use depth-three formulas as hypotheses in equivalence queries; our result shows that DNF (or CNF) hypotheses suffice for polynomial-query learning.

Another application emerges when we consider projection-closed concept classes. We show that if a projection-closed Boolean concept class $\mathcal{C}$ has polynomial teaching dimension, then $\mathcal{C}$ has polynomial certificates. By the characterization mentioned above, $\mathcal{C}$ is polynomial-query learnable with equivalence and membership queries. However, there is more: we show that equivalence queries are not needed at all here! In other words, for projection-closed Boolean concept classes, Goldman and Kearns' [17] characterization of polynomial teaching dimension as a necessary condition for polynomial-query learnability with membership queries is also a sufficient condition. Since many natural concept classes are projection-closed we expect that this result will have many applications. We give one new application: the concept class of Boolean formulas representable as both $\log n$-DNF and $\log n$-CNF formulas has polynomial teaching dimension. Bshouty et al. [14] have shown that this class is learnable as depth-three formulas using membership queries and an NP-oracle; our result shows that the class is learnable in polynomial time as a $(n, 2\log n)$ universal set representation using membership queries. This representation is not known to be evaluatable in polynomial time, but can be evaluated with an oracle for NP $\cap$ co-NP.

The rest of the paper is organized as follows. Section 2 contains definitions relevant to the rest of the paper. Section 3 has our characterization of exact learnability for Boolean concept classes. Section 4 is a generalization of Section 3 for all concept classes. Section 5 shows that "honest" classes that are polynomial-query learnable are also learnable at least in the polynomial-time hierarchy. Section 6 gives some applications of our characterization and finally, Section 7 deals with exact learning using only membership queries.

2 Definitions

We closely follow Angluin's [4] terminology. Let Σ be an alphabet. Then Σ^* denotes the set of all finite-length strings over Σ, $\Sigma^{\leq n}$ denotes the set of all strings over Σ of length at most n, and Σ^n denotes the set of all strings over Σ of length exactly n. For a set S, $|S|$ denotes the cardinality of S, and for a string w, $|w|$ is the length of w.

A *representation class* $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ is a 4-tuple where Σ and Δ are finite alphabets $R \subseteq \Delta^*$, and μ is a many-to-one mapping from R to subsets of Σ^*. A *concept* is any subset of Σ^*. For any concept f, χ_f denotes the characteristic function of f. That is, for any string x, $\chi_f(x) = 1$ if x is in f and $\chi_f(x) = 0$ otherwise. An *example* is a pair (x, b), where $x \in \Sigma^*$ and $b \in \{0, 1\}$. Δ is the alphabet of representations. The strings in R are the set of valid representations in the representation class $\mathcal{R}$. The function μ is a map from representations to concepts. The *size* of a representation $r \in R$ is $|r|$. The *size* of a concept

c is $\min\{|r| : \mu(r) = c\}$. If a concept c does not have a representation in R (i.e., there exists no $r \in R$ such that $\mu(r) = c$), then its size is infinite. (Defining the size of such concepts to be infinite simplifies the statement of various results.) The *concept class* $\mathcal{C}$ defined by the representation class $\mathcal{R}$ is the set of concepts that have representations in R. That is, $\mathcal{C} = \{\mu(r) : r \in R\}$. For any positive integer m, we also define $\mathcal{C}_m = \{\mu(r) : r \in R, |r| \leq m\}$. Thus, $\mathcal{C} = \bigcup_{m \geq 1} \mathcal{C}_m$.

Throughout the paper, we assume that for any representation class $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$, the following problems are computable:

1. For $r \in \Delta^*$, decide if $r \in R$.

2. For $x \in \Sigma^*$, $r \in R$, decide if $x \in \mu(r)$.

In the literature, the size of a concept is not always the string length of its (minimum) representation. Our theorems also hold for more general definitions of concept size. We discuss this issue in Section 6.1.

A Boolean concept is a subset of $\{0,1\}^n$ for some positive integer n. A Boolean concept class is one in which all concepts are Boolean. The strings in $\{0,1\}^n$ are considered to be assignments to the set $V_n = \{v_1, v_2, \ldots, v_n\}$ of n Boolean variables. Thus if $f \subseteq \{0,1\}^n$ is a Boolean concept, the characteristic function χ_f is a Boolean function defined on the variables in V_n. Where it causes no confusion, we will use f itself to denote χ_f, and thus treat f as a function defined on V_n. We will also talk of learning Boolean functions rather than Boolean concepts. If $x \in \{0,1\}^n$ and $f(x) = 1$, then x is a *positive assignment* of f. If $f(x) = 0$ then x is a *negative assignment* of f.

If $\mathcal{R}$ is a representation class defining a class of Boolean concepts $\mathcal{C}$, then $\mathcal{C}_{m,n} = \{\mu(r) : r \in R,\ |r| \leq m, \text{ and } \mu(r) \subseteq \{0,1\}^n\}$. That is, $\mathcal{C}_{m,n}$ contains all concepts (functions) in $\mathcal{C}$ defined on n variables that have size at most m. We define $\mathcal{C}_{(n)} = \bigcup_{m \geq 1} \mathcal{C}_{m,n}$.

Two concepts $f, g \subseteq \Sigma^*$ are *consistent with each other over a set* $X \subseteq \Sigma^*$ if for all $x \in X$ $\chi_f(x) = \chi_g(x)$. A concept $f \subseteq \Sigma^*$ is *consistent with a set* Y of examples if for all $(x, b) \in Y$ $\chi_f(x) = b$.

Angluin models the learning process using a *minimally adequate teacher* in the following way: The teacher selects some concept $c \in \mathcal{C}$. The goal of the learner (or learning algorithm) $\mathcal{A}$ is to output a hypothesis string $h \in R$ such that $\mu(h) = c$. That is, the concept c is the unknown *target concept* that $\mathcal{A}$ attempts to learn. If $\mathcal{A}$ is a learning algorithm for $\mathcal{R}$, it will eventually output such a string, regardless of which concept c is chosen from $\mathcal{C}$. The algorithm $\mathcal{A}$ may acquire information about the target concept c by asking the teacher two kinds of queries—*equivalence queries* and *membership queries*.

An equivalence query, EQ(h), where $h \in R$, effectively asks, "Is $\mu(h) = c$?" If $\mu(h) = c$, the teacher answers "Yes"; otherwise, the teacher answers "No" and gives a *counterexample* string $\beta \in \Sigma^*$ such that $\chi_c(\beta) \neq \chi_{\mu(h)}(\beta)$. A membership query, MQ(β), effectively asks if the string β is a member of the unknown concept c; the teacher answers with $\chi_c(\beta)$.

We assume that if $\mathcal{C}$ is a class of Boolean concepts, the teacher gives the learning algorithm the value n as input, where n is the number of variables of the target function. (If equivalence queries are allowed, this assumption can be removed by having the algorithm ask an initial equivalence query using the smallest possible hypothesis. If a counterexample is received, its length will be n.) We also assume that such algorithms ask membership queries only on strings in $\{0,1\}^n$ and that they ask equivalence queries only with hypotheses h that represent functions on n variables. Thus all counterexamples given to these equivalence queries are strings in $\{0,1\}^n$.

The *query complexity* of a learning algorithm $\mathcal{A}$ at any stage in a run is the sum (at that stage) of the lengths of the (i) inputs to equivalence queries (ii) inputs to membership queries and (iii) counterexamples received in response to equivalence queries. If there exists a learning algorithm for $\mathcal{R}$ that can use an unlimited amount of computation time but has polynomial query complexity, then $\mathcal{R}$ is said to be *polynomial-query learnable.* Formally, we say that the class $\mathcal{R}$ is *polynomial-query learnable (with equivalence and membership queries)* if there exists an algorithm $\mathcal{A}$ and a two-variable polynomial p such that, for any positive integer m, and for any unknown target concept $c \in \mathcal{C}_m$:

1. $\mathcal{A}$ uses membership queries, and equivalence queries of the form $\mathrm{EQ}(h)$, where $h \in R$,
2. $\mathcal{A}$ eventually halts and outputs a string $r \in R$ such that $\mu(r) = c$, and
3. At any stage, if n is the length of the longest counterexample received so far in response to an equivalence query, the query complexity of $\mathcal{A}$ at that stage does not exceed $p(m,n)$.

If, at every stage, $\mathcal{A}$ also has time complexity polynomial in n and m, then $\mathcal{R}$ is said to be *polynomial-time learnable.* This is the standard definition of polynomial-time learnability in the membership and equivalence query model.

For the Boolean concept classes, a simpler definition of polynomial-query learnability is obtained by replacing Condition 3 above with the following:

3a. The total query complexity of $\mathcal{A}$ is at most $p(m,n)$, where n is the number of input variables to the target function.

Clearly, if a representation class $\mathcal{R}$ is not polynomial-query learnable, then it is not polynomial-time learnable either.

3 Learning Boolean Concept Classes

In this section, we give results on polynomial query learnability for Boolean concept classes. In the next section, we extend the results to the general case. Although the general case includes the Boolean case, the Boolean case is much simpler, and we present it first in order to highlight the ideas behind the results.

Let $\mathcal{C}$ be a class of Boolean concepts defined by a representation class $\mathcal{R}$. Consider the following standard Majority learning algorithm. (This algorithm is described, for example, in [3]. It is also called the *halving algorithm.*) Assuming that the learner is allowed arbitrary hypotheses as inputs to equivalence queries, the algorithm works as follows: Because $\mathcal{C}$ is a Boolean concept class, it is given as input n, the number of variables on which the target f is defined. Initially, let $K = \mathcal{C}_{(n)}$. At any stage, K holds all concepts in $\mathcal{C}$ that are consistent with the information known about the target concept. For any set T of concepts, let $MAJ(T)$ denote the *majority concept.* That is, for any string x, $MAJ(T)$ contains x if and only if at least half of the concepts in T contain x. Repeat the following two steps until the equivalence query returns "Yes": (i) Make an equivalence query with $MAJ(K)$; (ii) if x is the counterexample received, eliminate from K all concepts that disagree with the target concept on the evaluation of x. Since each counterexample eliminates at least half of the concepts in K, the algorithm identifies the target concept in at most $\lceil \log_2 |\mathcal{C}_{(n)}| \rceil$ iterations.

The above Majority algorithm does not imply polynomial-query learnability of $\mathcal{C}$ for two simple reasons. First, the upper bound on the number of queries, $\lceil \log_2 |\mathcal{C}_{(n)}| \rceil$, may not be bounded by a polynomial in m, the size of the target concept, and n. This problem can be easily solved using a standard trick. If we knew the size m of the target concept, we could start the above algorithm with $K = \mathcal{C}_{m,n}$ rather than $K = \mathcal{C}_{(n)}$. In this way, the number of queries would be at most $\lceil \log_2 |\mathcal{C}_{m,n}| \rceil$, rather than $\lceil \log_2 |\mathcal{C}_{(n)}| \rceil$. Since every concept in $\mathcal{C}_{m,n}$ can be represented as a string of length at most m in Δ^*, $|\mathcal{C}_{m,n}| \leq (1 + |\Delta|)^{m+1}$, and hence $\lceil \log_2 |\mathcal{C}_{m,n}| \rceil$ is at most $(m+1)\log_2(1 + |\Delta|)$. Lacking knowledge of the size of the target concept, we start by guessing $m = 1$, and then running the algorithm with $K = \mathcal{C}_{1,n}$. If the algorithm ends up with K empty (i.e. if all concepts in $\mathcal{C}_{1,n}$ are inconsistent with the target), we double our guess for m and repeat. The number of queries performed is then at most $\lceil \log_2 |\mathcal{C}_{2m,n}| \rceil (\log_2 m)$, which is still polynomial in m.

The second, more serious, problem with the above Majority algorithm is that it makes equivalence queries with $MAJ(K)$ as the hypothesis; however, such a hypothesis may not be polynomially representable in $\mathcal{R}$ (i.e., the smallest string that represents $MAJ(K)$ in R may be superpolynomially larger than the smallest string used to represent a concept in K.) We will show that this problem can be circumvented if $\mathcal{R}$ *has polynomial certificates.*

Polynomial certificates (for Boolean concept classes): Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class defining a Boolean concept class $\mathcal{C}$, i.e., $\Sigma = \{0,1\}$. The class $\mathcal{R}$ *has polynomial certificates* if there exist two-variable polynomials p and q, such that for all $m, n > 0$, and for all $f \subseteq \{0,1\}^n$ such that the size of the concept f is greater than $p(m,n)$, there exists a set $Q \subseteq \{0,1\}^n$ such that $|Q| \leq q(m,n)$ and no concept in $\mathcal{C}_{m,n}$ is consistent with f over Q.

The intuition behind this definition is the following: If a Boolean concept class has polynomial certificates, then for every Boolean concept (not necessarily the class) that has a "large" representation, there is a proof (in the form of the set Q) that the concept does not have a small representation.

Theorem 1 *Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class defining a Boolean concept class $\mathcal{C} = \bigcup_{m \geq 1} \mathcal{C}_m$. If $\mathcal{R}$ has polynomial certificates then $\mathcal{R}$ is polynomial-query learnable with* *equivalence and membership queries.*

Proof: Let p and q be the (not necessarily unique) two-variable polynomials whose existence is guaranteed by the fact that $\mathcal{R}$ has polynomial certificates. We give an algorithm for learning $\mathcal{R}$ that has polynomial query complexity. It is a variation of the Majority learning algorithm.

Since $\mathcal{R}$ represents a class of Boolean concepts, we may assume that the algorithm is given as input n, the number of variables of the target concept f. We will also assume that the algorithm is given as input m, the size of the target concept f. This assumption can be removed by using the doubling trick mentioned above.

Begin the algorithm by setting $K = \mathcal{C}_{m,n}$. Then repeat the following until an equivalence query returns "Yes":

1. If the size of $MAJ(K)$ is at most $p(m,n)$, then ask an equivalence query with a hypothesis $h \in R$ representing $MAJ(K)$, such that $|h| \leq p(m,n)$. Let x be the counterexample received. Eliminate from K all concepts that disagree with f on the evaluation of x.

2. If the size of $MAJ(K)$ is greater than $p(m,n)$, then find a set Q such that no concept in $\mathcal{C}_{m,n}$ is consistent with $MAJ(K)$ over Q and such that $|Q| \leq q(m,n)$. The existence of Q is guaranteed by the fact that $\mathcal{R}$ has polynomial certificates. For all $x \in Q$, ask a membership query to discover the value of $f(x)$. Eliminate from K all concepts that disagree with f on some $x \in Q$.

Step 1 above is the basic step of the Majority learning algorithm, and it eliminates at least half of the concepts in K. Now consider Step 2. From the construction of Q and the fact that the size of f is m, there must exist some $x \in Q$ such that $MAJ(K)$ and f have different values on input x. Therefore, eliminating all concepts that disagree with f on x eliminates at least half the concepts in K. It follows that steps 1 and 2 are executed at most $\lceil \log_2 |\mathcal{C}_{m,n}| \rceil \leq (m+1)\log_2(1+|\Delta|)$ times (where Δ is the alphabet of R), and that the total query complexity of the algorithm is $O(m(p(m,n)+q(m,n)+n))$. ■

We now show the converse of the above theorem.

Theorem 2 *Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class defining a Boolean concept class $\mathcal{C} = \bigcup_{m \geq 1} \mathcal{C}_m$. If $\mathcal{R}$ is polynomial-query learnable then $\mathcal{R}$ has polynomial certificates.*

Proof: Let $\mathcal{A}$ be a membership and equivalence query algorithm for learning $\mathcal{R}$ that has polynomial query complexity. We may assume without loss of generality that immediately before $\mathcal{A}$ halts and outputs a representation r of the target function, $\mathcal{A}$ asks an equivalence query with hypothesis r that is answered "Yes."

Suppose that the query complexity of $\mathcal{A}$ is $p(m',n')$, where m' is the size of the target function, and n' is the number of variables on which the target function is defined.

Let f be any Boolean concept over n variables whose size is greater than $p(m,n)$, for some m. Consider simulating the algorithm $\mathcal{A}$ as follows. Give $\mathcal{A}$ the input n as the number of variables and run $\mathcal{A}$ until its query complexity exceeds $p(m,n)$ or $\mathcal{A}$ halts. Each time $\mathcal{A}$ asks a membership query on a string $x \in \{0,1\}^n$, give the answer $f(x)$ to $\mathcal{A}$ and continue the simulation of $\mathcal{A}$. Each time $\mathcal{A}$ asks an equivalence query with a hypothesis string h, do the following. If $|h| > p(m,n)$, abort the simulation. If $|h| \leq p(m,n)$, then $h \neq f$. Therefore, there exists an $x \in \{0,1\}^n$ such that $h(x) \neq f(x)$. Answer "No" to the equivalence query, return x as the counterexample, and continue the simulation.

There is a technical point we have omitted in discussing the above simulation. The answers given to all queries during the simulation are consistent with f being the target concept. However, if f has infinite size, then there is no guarantee as to what the algorithm will do (the definition of polynomial query complexity gives conditions for the behavior of the algorithm only on target concepts of finite size). Therefore, in the above simulation, the algorithm may terminate abnormally, ask a membership query on a string not in $\{0,1\}^n$, or ask an equivalence query with some hypothesis $h \notin R$. During the simulation, if any of these occur, abort the simulation.

Let Q be the set of strings $x \in \{0,1\}^n$ for which answers to membership queries were returned or which were returned as counterexamples during the simulation. Clearly $|Q| \leq p(m,n)$. We now show that no concept in $\mathcal{C}_{m,n}$ is consistent with f over Q.

Let $g \in \mathcal{C}_{m,n}$ and suppose, to the contrary, that g is consistent with f on the set Q. Then in the above simulation, all answers to membership queries and all counterexamples given to equivalence queries agree with g being the target. By the definition of $\mathcal{A}$, $\mathcal{A}$ halts within query complexity $p(m,n)$ when g is indeed the target. More particularly, within query complexity $p(m,n)$, it asks an equivalence query with a hypothesis r that represents g. Since the simulation continues until the query complexity of $\mathcal{A}$ exceeds $p(m,n)$, the simulation must ask an equivalence query with a hypothesis r representing g. Such an equivalence query will be answered with a counterexample x of length n such that $g(x) \neq f(x)$. The counterexample x will be added to the set Q. However, this contradicts the assumption that g is consistent with f over Q. ■

The contrapositive of the above theorem gives a non-learnability result: If $\mathcal{R}$ does not have polynomial certificates, then $\mathcal{R}$ is not polynomial-query learnable. In fact, the above proof combines elements from two different kinds of proofs of non-learnability—the information-theoretical proof [4] that classes that possess the approximate fingerprint property are not learnable with equivalence queries alone, and the complexity-theoretical proofs [1, 27] that classes for which the representation problem is hard are not learnable in polynomial time with equivalence and membership queries.

The above two theorems immediately imply the following corollary.

Corollary 1 *A representation class $\mathcal{R}$ defining a class of Boolean concepts is polynomial-query learnable if and only if it has polynomial certificates.*

4 Learning General Concept Classes

In this section, we extend the definitions and theorems of the previous section to general concept classes. That is, we consider classes made up of concepts that are arbitrary subsets of Σ^*. The main difficulty in extending our results is that in learning a concept that is not Boolean, a learning algorithm may receive counterexamples of variable length. Moreover, the definition of polynomial query complexity depends on the length of the counterexamples.

We begin by giving a new, more general, definition of polynomial certificates.

Polynomial certificates: Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class defining a concept class $\mathcal{C} = \bigcup_{m \geq 1} \mathcal{C}_m$. The class $\mathcal{R}$ *has polynomial certificates* if there exist two-variable polynomials p and q, such that for all $m, n > 0$, and for all $f \in \Sigma^*$ such that no concept in $\mathcal{C}_{p(m,n)}$ is consistent with f over $\Sigma^{\leq n}$, there exists a subset $Q \subseteq \Sigma^{\leq n}$ such that $\sum_{x \in Q} |x| \leq q(m,n)$, and no concept in $\mathcal{C}_m$ is consistent with f over Q.

Note that the last part of this definition states that the total length of the strings in Q is at most $q(m,n)$, rather than that the *number* of strings in Q is at most $q(m,n)$.

We give general versions of the two theorems from the previous section. For any set S of examples, let $Cons(S,m)$ denote the set of concepts in $\mathcal{C}_m$ that are consistent with S.

Theorem 3 *Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class defining a concept class $\mathcal{C} = \bigcup_{m \geq 1} \mathcal{C}_m$. If $\mathcal{R}$ has polynomial certificates then $\mathcal{R}$ is polynomial-query learnable with equivalence and membership queries.*

Proof: Let p and q be the (not necessarily unique) two-variable polynomials whose existence is guaranteed by the fact that $\mathcal{R}$ has polynomial certificates. In Figure 1, we give an algorithm for learning $\mathcal{R}$. The algorithm does not assume that it is given as input the size of the target concept.

The algorithm is a variant of the algorithm given in the proof of Theorem 1. Some of the differences are minor. Instead of explicitly keeping a set K of concepts consistent with all the examples seen so far, it just keeps the examples in a set S (from which the set K can be derived). Also, since the algorithm does not receive as input the size of the target concept, it often updates its estimate m for the size of the target concept. (As written, the algorithm may increment m by only one at each update; however, it can easily be modified to double m.)

A more important difference is that the algorithm of Figure 1 makes an equivalence query with hypothesis w in each iteration of the outer **While** loop. In what follows, we show that with or without the inclusion of this equivalence query, the algorithm maintains the required property that its query complexity is polynomial at every stage. The inclusion of the equivalence query with hypothesis w ensures that the algorithm eventually terminates by guaranteeing the elimination of at least one candidate for the target concept in $Cons(S,m)$.

We now prove that the algorithm uses only a polynomial number of polynomial sized equivalence and membership queries, eventually terminates, and is correct. Let m^* be the

Algorithm **Learn from a Minimally Adequate Teacher**

$n \leftarrow 0$
$S \leftarrow \emptyset$
$m \leftarrow$ Least i such that $\mathcal{C}_i \neq \emptyset$
$h \leftarrow$ A representation of size at most m of any concept in $\mathcal{C}_m$
While $EQ(h) \neq$"Yes" **do**
    Let x be the counterexample to h
    $w \leftarrow$ A representation of size at most m of any concept in $Cons(S, m)$
    If $EQ(w) =$"Yes" **then**
        Output w and halt
    Endif
    Let y be the counterexample to w
    $S \leftarrow S \cup \{(x, MQ(x)), (y, MQ(y))\}$
    $m \leftarrow$ Least i such that $Cons(S, i) \neq \emptyset$
    $n \leftarrow \max(n, |x|, |y|)$
    While no concept in $\mathcal{C}_{p(m,n)}$ is consistent with $MAJ(Cons(S, m))$ over $\Sigma^{\leq n}$ **do**
        Let X be a set of strings such that $\Sigma_{x \in X}|x| \leq q(m, n)$ and
        no concept in $\mathcal{C}_m$ is consistent with $MAJ(Cons(S, m))$ over X
        $S' \leftarrow \bigcup_{x \in X}\{(x, MQ(x))\}$
        $S \leftarrow S \cup S'$
        $m \leftarrow$ Least i such that $Cons(S, i) \neq \emptyset$
    Endwhile
    $h \leftarrow$ A representation of size at most $p(m, n)$ of any concept that is consistent
        with $MAJ(Cons(S, m))$ over $\Sigma^{\leq n}$
Endwhile

Figure 1: An Algorithm for Learning $\mathcal{R}$

size of the target concept. As a first step we show that for any value of $m < m^*$ the number of iterations of the inner and outer **While** loops is no greater than $\log_2 |\mathcal{C}_m| + n + 1$, where n is the length of the longest counterexample received thus far. Note that the value of m is always increased when $|Cons(S, m)| = 0$. Hence we only need to count the maximum number of iterations before $Cons(S, m)$ becomes empty.

For each counterexample x received in response to an equivalence query with hypothesis h at the beginning of an iteration of the outer loop, one of the following two cases holds: (i) the length of counterexample x is at most l, the length of the longest counterexample at that point, or (ii) the length of counterexample x is greater than l. In all iterations (except the first) the equivalence query was made with the majority concept. If case (i) holds when $(x, MQ(x))$ is added to S, then the number of concepts in $Cons(S, m)$ consistent with $(x, MQ(x))$ is at most $\frac{1}{2}|Cons(S, m)|$. If case (ii) holds then no such guarantee can be offered. However, the number of iterations in which case (ii) can be true is at most n.

Now consider the set S' of examples added to set S in any iteration of the inner loop. Here again one of two cases is true: (i) there exists a string $x \in X$ such that the target concept and $MAJ(Cons(S, m))$ disagree on x or (ii) the target concept and $MAJ(Cons(S, m))$ are consistent with each other over X. If case (i) holds, then the addition of example $(x, MQ(x))$ to S reduces the number of concepts in $Cons(S, m)$ by at least a factor of $\frac{1}{2}$. If case (ii) holds, then since each concept in $\mathcal{C}_m$ disagrees with $MAJ(Cons(S, m))$ on at least one string in X, none of the concepts in $\mathcal{C}_m$ is consistent with the target concept over X. That is, $Cons(S, m)$ becomes empty after S' is added to S. Since in all but at most $n + 1$ iterations of the inner and outer loops the number of concepts in $Cons(S, m)$ reduces by a factor of at least $\frac{1}{2}$, the number of iterations before $Cons(S, m)$ becomes empty is $\log_2 |\mathcal{C}_m| + n + 1 \leq (m + 1) \log_2(1 + |\Delta|) + n + 1$.

The initial value of m is no greater than m^*. Every example added to S is consistent with the target concept and therefore $Cons(S, m^*)$ is never empty. Further, each equivalence query with hypothesis w in the outer **While** loop is guaranteed to eliminate at least one concept in $Cons(S, m)$, for all $m < m^*$. Hence the value of m must at some point increase to m^*. Reasoning as above, we conclude that after at most $(m^* + 1) \log_2(1 + |\Delta|) + n + 1$ additional iterations the only remaining concept in $Cons(S, m^*)$ is the target concept. At this stage, the algorithm terminates since the response to an equivalence query must be "Yes." Finally, it is easy to verify that all queries used are of polynomial size. ■

Theorem 4 *Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class defining a concept class $\mathcal{C} = \bigcup_{m \geq 1} \mathcal{C}_m$. If $\mathcal{R}$ is polynomial-query learnable then $\mathcal{R}$ has polynomial certificates.*

Proof: The proof is similar to the proof of Theorem 2.

Let $\mathcal{A}$ be a membership and equivalence query algorithm for learning $\mathcal{R}$ with polynomial query complexity. Let the query complexity of $\mathcal{A}$ be bounded above by $p(m, n)$, where p is a polynomial, m is the size of the target concept, and n is the length of the longest counterexample received. We may assume without loss of generality that in learning any concept in $\mathcal{C}$, $\mathcal{A}$ asks an equivalence query immediately before terminating that is answered "Yes."

Consider any pair of fixed integers $m, n \geq 1$. Let $f \subseteq \Sigma^*$ be a concept such that no concept in $\mathcal{C}_{p(m,n)}$ is consistent with f over $\Sigma^{\leq n}$. We show that there exists a set $Q \subseteq \Sigma^{\leq n}$ such that $\sum_{x \in Q} |x| \leq p(m,n)$ and no concept in $\mathcal{C}_m$ is consistent with f over Q.

Consider simulating the algorithm $\mathcal{A}$ as follows, until the query complexity of $\mathcal{A}$ exceeds $p(m,n)$ or $\mathcal{A}$ terminates. Each time $\mathcal{A}$ asks a membership query on a string x, give the answer $\chi_f(x)$ to $\mathcal{A}$. If $\mathcal{A}$ makes an equivalence query $EQ(h)$ where $h \in R$ and $|h| \leq p(m,n)$, give $\mathcal{A}$ the answer "No" together with a counterexample $x \in \Sigma^{\leq n}$ such that $\chi_f(x) \neq \chi_{\mu(h)}(x)$. The existence of such a string x is guaranteed by the fact that no concept in $\mathcal{C}_{p(m,n)}$ is consistent with f over $\Sigma^{\leq n}$. If during the simulation, $\mathcal{A}$ terminates abnormally, or asks an equivalence query with some hypothesis $h \notin R$, abort the simulation.

Let Q be the set of strings $x \in \Sigma^*$ for which answers to membership queries were returned or which were returned as counterexamples, during the above simulation. Since the simulation is aborted once the query complexity of $\mathcal{A}$ exceeds $p(m,n)$, $\sum_{x \in Q} |x| \leq p(m,n)$. By the same arguments as used in the proof of Theorem 2, it can be shown that no concept in $\mathcal{C}_m$ is consistent with f over Q. ∎

Corollary 2 *A representation class $\mathcal{R}$ is polynomial-query learnable if and only if it has polynomial certificates.*

5 Learning Honest Classes in the Polynomial Hierarchy

In the proof of Theorem 4, we gave an algorithm (Figure 1) for learning a representation class $\mathcal{R}$ with polynomial query complexity, given that $\mathcal{R}$ has polynomial certificates. In this section, we show that a variant of the algorithm can be implemented in the polynomial-time hierarchy. The new algorithm (Figure 2) is conceptually the same as the algorithm in Figure 1 except that (i) $Cons$ has been replaced with $RCons$, and (ii) the majority function MAJ has been replaced by a "pseudo-majority" function $PMAJ$, where $RCons$ and $PMAJ$ are defined as follows. For any set S of examples, $RCons(S, m)$ denotes the set of representations of size at most m that represent concepts in $\mathcal{C}$ which are consistent with S. That is, $RCons(S, m) = \{r \in R : |r| \leq m,\ \mu(r) \text{ is consistent with } S\}$. Let $T \subseteq R$ be a set of representations. For any $k > 1$, $\frac{1}{k}$-$PMAJ(T)$ is the set of $\frac{1}{k}$ majority concepts over T. That is, $\frac{1}{k}$-$PMAJ(T) = \{c \subseteq \Sigma^* : \text{for all } x \in \Sigma^*, |\{r \in T : \chi_{\mu(r)}(x) = \chi_c(x)\}| \geq \frac{1}{k}|T|\}$.

The sole reason for introducing these changes is to ensure that all necessary computations of the learning algorithm can be carried out in the polynomial-time hierarchy. First, the preference of a pseudo-majority function over a true majority function comes about from the crucial observation (as in Gavaldà [16]) that a good *approximation* of the majority concept suffices to ensure that the learning algorithm uses a polynomial number of queries. While a true majority function may be hard to compute, a pseudo-majority function defined as above can be computed in the polynomial-time hierarchy using Stockmeyer's result [30] on approximating functions in #P. Second, the pseudo-majority function in the algorithm in

Figure 2 is carried out over *representations* rather than *concepts* as in the algorithm in Figure 1. The point here is that while a pseudo-majority of representations can be computed in the polynomial-time hierarchy, a pseudo-majority of concepts requires being able to check if two concepts are equal, a problem which is undecidable even for natural concept classes (a good example is the class of Context-Free Grammars).

We need the following mild technical requirement for Theorem 5. A representation class $\mathcal{R} = (\Sigma, \Delta, R, \mu)$ is *honest* if it satisfies the following two conditions:

1. *$\mathcal{R}$ is recognizable in polynomial time:* Given a string $h \in \Delta^*$, it can be decided in time polynomial in $|h|$ if h is in R,

2. *$\mathcal{R}$ is evaluatable in polynomial time:* Given a string $w \in \Sigma^*$ and a representation $h \in R$ of a concept, it can be checked in time polynomial in $|w|$ and $|h|$ if w is a string in $\mu(h)$.

Theorem 5 *If an honest representation class $\mathcal{R}$ is polynomial-query learnable but not polynomial-time learnable then $P \neq NP$.*

Proof: If P is distinct from the polynomial hierarchy, then P $\neq$ NP. Therefore, to prove the theorem, it suffices to show that if there exists an honest representation class that is polynomial-query learnable but not polynomial-time learnable, then P is distinct from the polynomial hierarchy. In other words, we only need to show that every honest representation class $\mathcal{R}$ that is polynomial-query learnable is also learnable in polynomial time using oracles in the polynomial hierarchy.

It suffices to show that the following "hard" steps in the algorithm of Figure 2 can be solved using calls to oracles in the polynomial hierarchy. (Since (i) the number of representations of size at most m is bounded from above by $(1 + |\Delta|)^{m+1}$ and (ii) the use of a $\frac{1}{512}$ pseudo-majority concept instead of the $\frac{1}{2}$ true-majority concept in Theorem 3 affects query complexity by only a constant factor, the correctness and query complexity of the algorithm in Figure 2 follows as in Theorem 3.)

- $m \leftarrow$ *Least i such that $\mathcal{C}_i \neq \emptyset$*

This can be solved by iteratively answering the following question for each value of i: Does there exist a $u \in R$ such that $|u| \leq i$? Since $\mathcal{R}$ is honest a "Yes" answer to this question with a representation of size i can be verified. Hence, this can be solved using calls to a Σ_1^p oracle.

- $w \leftarrow$ *Any representation in $RCons(S, m)$*

- $m \leftarrow$ *Least i such that $RCons(S, i) \neq \emptyset$*

Both these steps can be solved easily by using a Σ_1^p oracle in essentially the same way. For example, the second step above can be solved by iteratively answering the following question for each value of i: Does there exist a $u \in R$ such that $|u| \leq i$ and $\mu(u)$ is consistent with S? Again, since $\mathcal{R}$ is honest, a "Yes" answer can be verified in time polynomial in $|S|$ and i.

- *Is no concept in $\mathcal{C}_{p(m,n)}$ consistent with any concept in $\frac{1}{32}$-$PMAJ(RCons(S,m))$ over $\Sigma^{\leq n}$?*

- *$h \leftarrow$ A representation of size at most $p(m,n)$ of any concept in $\mathcal{C}$ consistent with any concept in $\frac{1}{512}$-$PMAJ(RCons(S,m))$ over $\Sigma^{\leq n}$*

We show that there exists a test such that if the test succeeds then we are guaranteed that there is no concept in $\mathcal{C}_{p(m,n)}$ consistent with any concept in $\frac{1}{32}$-$PMAJ(RCons(S,m))$ over $\Sigma^{\leq n}$; if the test fails, we can compute a representation of size $p(m,n)$ of a concept in $\mathcal{C}$ that is consistent with some concept in $\frac{1}{512}$-$PMAJ(RCons(S,m))$ over $\Sigma^{\leq n}$. The decision problem translates into: Is it true that for all $u \in R$ such that $|u| \leq p(m,n)$ there exists $x \in \Sigma^{\leq n}$ such that

$$\frac{|\{u' \in RCons(S,m) : \chi_{\mu(u')}(x) = \chi_{\mu(u)}(x)\}|}{|RCons(S,m)|} < \frac{1}{32}.$$

Now, the decision problem "Does there exist a representation $u \in R$ such that $|u| \leq m$ and $\mu(u)$ is consistent with S?" is in NP (the representation class is honest), and the number of choices of u that are "Yes"-certificates for the decision problem is precisely $|RCons(S,m)|$. Therefore the function that computes the value of $|RCons(S,m)|$ is in #P. Using Stockmeyer's [30] results we can find an approximation r of $|RCons(S,m)|$ such that $\frac{r}{2} \leq |RCons(S,m)| \leq 2r$ using an oracle in Δ_3^p. Similarly, for a given $u \in R$, $|u| \leq p(m,n)$, and a given $x \in \Sigma^{\leq n}$, the decision problem "Does there exist a representation d, $|d| \leq m$, such that $\mu(d)$ is consistent with S and $\chi_{\mu(d)}(x) = \chi_{\mu(u)}(x)$?" is in NP and the corresponding #P-function is precisely $|\{u' \in RCons(S,m) : \chi_{\mu(u')}(x) = \chi_{\mu(u)}(x)\}|$. Hence, for a given u and x, we can use Stockmeyer's technique again to find an approximation s such that $\frac{s}{2} \leq |\{u' \in RCons(S,m) : \chi_{\mu(u')}(x) = \chi_{\mu(u)}(x)\}| \leq 2s$.

For any given u and x, if $\frac{s}{r} < \frac{1}{128}$, then we are assured that $\frac{|\{u' \in RCons(S,m) : \chi_{\mu(u')}(x) = \chi_{\mu(u)}(x)\}|}{|RCons(S,m)|} < \frac{1}{32}$. Hence, we check if for all $u \in R$ such that $|u| \leq p(m,n)$ there exists $x \in \Sigma^{\leq n}$ such that $\frac{s}{r} < \frac{1}{128}$. If there exists a $u \in R$ such that $|u| \leq p(m,n)$ and for all $x \in \Sigma^{\leq n}$ we have $\frac{s}{r} \geq \frac{1}{128}$, then we are assured that there exists a concept in $\mathcal{C}_{p(m,n)}$ that is consistent with $\frac{1}{512}$-$PMAJ(RCons(S,m))$ over $\Sigma^{\leq n}$. Thus the test "*Is no concept in $\mathcal{C}_{p(m,n)}$ consistent with any concept in $\frac{1}{32}$-$PMAJ(RCons(S,m))$ over $\Sigma^{\leq n}$?*" can be approximated using a Π_2^p process relative to a Δ_3^p-oracle (or equivalently, a Π_4^p-oracle) in such a way that a "Yes" answer is always correct. Moreover, if the answer is "No" then the problem "*$h \leftarrow$ A representation of size at most $p(m,n)$ of any concept in $\mathcal{C}$ consistent with any concept in $\frac{1}{512}$-$PMAJ(RCons(S,m))$ over $\Sigma^{\leq n}$*" can be solved using a $\Sigma_2^p(\Delta_3^p) = \Sigma_4^p$-oracle.

- *Let X be a set of strings such that $\Sigma_{x \in X}|x| \leq q(m,n) + n$ and no concept in $\mathcal{C}_m$ is consistent with some concept in $\frac{1}{2}$-$PMAJ(RCons(S,m))$ over X*

Since $\mathcal{R}$ has polynomial certificates, we know that there is some X_0 such that $\Sigma_{x \in X_0}|x| \leq q(m,n)$ and for every $c \in \mathcal{C}_m$

$$\frac{|\{d : d \in RCons(S,m), \mu(d) \text{ is consistent with } c \text{ over } X_0\}|}{|RCons(S,m)|} \leq \frac{1}{32}.$$

As before, given S, m, X and a representation of size m of c we can find approximations r and t such that

$$\frac{r}{2} \leq |RCons(S,m)| \leq 2r$$

and

$$\frac{t}{2} \leq |\{d : d \in RCons(S,m), \mu(d) \text{ is consistent with } c \text{ over } X\}| \leq 2t.$$

Now we can implement the above line by searching for a set X of assignments such that $t/r \leq 1/8$. Since X_0 exists, and since t/r is at most a factor of 4 away from the value it approximates, this test must succeed—at least for X_0. Similarly, if it succeeds for any set X, then we are assured that

$$\frac{|\{d : d \in RCons(S,m), \mu(d) \text{ is consistent with } c \text{ over } X\}|}{|RCons(S,m)|} \leq \frac{1}{2}$$

thus solving the problem. The search for such an X requires using a Σ_2^p process relative to a Δ_3^p-oracle, or equivalently, a Σ_4^p-oracle.

■

Corollary 3 *If an honest representation class $\mathcal{R}$ has polynomial certificates and is not polynomial-time learnable then $P \neq NP$.*

6 Applications of the Characterization

6.1 Alternative Definitions of Size

Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class defining a concept class $\mathcal{C}$. Thus far, we have defined the size of a representation $r \in R$ to be $|r|$, the length of the string r. However, other definitions of size are possible. For example, if the strings in R are encodings of DNF formulas, then we might define the size of $r \in R$ to be the number of terms of the DNF encoded by r.

The theorems of the previous sections hold for virtually all of the alternative definitions of size used in the literature. More precisely, let $\sigma : R \rightarrow \mathcal{N}$ be any function, where $\mathcal{N}$ is the set of natural numbers. Define the size of $r \in R$ (with respect to σ) to be $\sigma(r)$ and the size of a concept c (with respect to σ) to be $\min\{\sigma(r) : \mu(r) = c\}$ if c can be represented by a string in r, and infinite otherwise. This new definition of size may then be used to specify $\mathcal{C}_m$ and polynomial certificates.

It is easy to show that Theorems 3 and 4 hold for all size functions σ such that the length of representation r is bounded from above by a polynomial in $\sigma(r)$. If, in addition, $\sigma(r)$ can be computed in time polynomial in the length of r, then Theorem 5 holds as well.

For Boolean concept classes, Theorems 1 and 2 hold for all size functions σ satisfying the following property: there exists a two variable polynomial p such that for all representations r, if r represents a function on n variables, $|r| \leq p(\sigma(r), n)$. Again, if $\sigma(r)$ can be computed in time polynomial in the length of r, then Theorem 5 holds as well.

6.2 Applications

We now show that the class of Deterministic Finite State Automata (DFA) is polynomial-query learnable. Since this result follows from the fact that the class is polynomial-time learnable [2, 28], the proof below is intended only to illustrate that a simple proof of polynomial-query learnability can be obtained by showing that a concept class has certificates. See [20] for a definition of DFA, regular languages, etc.

Assume some standard DFA representation in which the length of a representation of an automaton is at most a fixed polynomial in its number of states. Define the size of a representation to be the number of states in the automaton represented. By the comments in the previous subsection, we may apply Theorem 3.

Theorem 6 *The class of DFA is polynomial-query learnable.*

Proof: By Theorem 3, it suffices to prove that DFA have polynomial certificates. Equivalently, we prove that there exist two-variable polynomials p and q such that for all non-negative integers m and n, if $L \subseteq \Sigma^*$ is a language not consistent with any DFA of at most $p(m,n)$ states over $\Sigma^{\leq n}$, then there is a set X of at most $q(m,n)$ words in $\Sigma^{\leq n}$ such that no DFA of at most m states is consistent with L over X.

Fix $p(m,n) = m$ and $q(m,n) = m^2 + m$. For any DFA M, let L_M denote the language accepted by M. Let $L \subseteq \Sigma^*$ be a (not necessarily regular) language. Define the symmetric relation of *n-inequivalence*, $\not\sim_n$, as follows: for all $x, y \in \Sigma^{\leq n}$, $x \not\sim_n y$ if and only if there exists a $z \in \Sigma^{\leq n}$ such that $|xz| \leq n$, $|yz| \leq n$, and exactly one of xz and yz is in L. Let $A_L(n)$ denote the maximum number of pairwise n-inequivalent strings in $\Sigma^{\leq n}$.[§] Kaneps and Freivalds [22] proved that there is a DFA M of $A_L(n)$ states such that $L_M \bigcap \Sigma^{\leq n} = L \bigcap \Sigma^{\leq n}$ (this proof is closely related to the proof of the correctness of Angluin's algorithm [2]).

From the above, it follows that if a language L is not consistent with any DFA of $p(m,n) = m$ states over $\Sigma^{\leq n}$, then there exists a set Q of $m+1$ pairwise n-inequivalent strings in $\Sigma^{\leq n}$. For each pair of distinct strings x_j and x_k in Q, let x_{jk} be a string witnessing their n-inequivalence, i.e. $|x_j x_{jk}| \leq n$, $|x_k x_{jk}| \leq n$, and exactly one of $x_j x_{jk}$ and $x_k x_{jk}$ is in L. Let $X = \cup_{0 \leq j < k \leq m+1} \{x_j x_{jk}, x_k x_{jk}\}$. Note that $|X| \leq m(m+1) = q(m,n)$. To finish the proof, observe that if any DFA M' of m or fewer states is consistent with L over X then some distinct x_j and x_k in Q must reach the same state of M'. But then M' either accepts or rejects both of $x_j x_{jk}$ and $x_k x_{jk}$, a contradiction. ∎

As another application of our characterization, we now show that Boolean functions with bounded DNF and CNF sizes are polynomial-query learnable as DNF formulas (or as CNF formulas). The basic idea is to use the probabilistic method [15] to prove the existence of polynomial certificates for this class. The result then follows from Corollary 1. A rigorous definition of the representation class can be easily worked out along the lines of [4].

A Boolean variable v or its complement $\overline{v}$ is called a *literal*. A conjunction of literals is a *term*. A disjunction of terms is called a *Disjunctive Normal Form* (DNF) formula. A

[§]The measure $A_L(n)$ was introduced independently by a number of people, including Karp [23] and Breitbart [11].

disjunction of literals is a *clause*. A conjunction of clauses is called a *Conjunctive Normal Form* (CNF) formula. Let $V_n = \{v_1, v_2, \ldots, v_n\}$ be a set of n Boolean variables and L_n the set of all literals over V_n. An *assignment* α is a mapping α: $L_n \rightarrow \{0,1\}$ such that for all $1 \leq i \leq n$, $\{\alpha(v_i), \alpha(\overline{v_i})\} = \{0,1\}$. For any literal v, the assignment $\alpha_{v\leftarrow 1}$ is defined by $\alpha_{v\leftarrow 1}(x) = \alpha(x)$ if $x \notin \{\overline{v}, v\}$, $\alpha_{v\leftarrow 1}(v) = 1$ and $\alpha_{v\leftarrow 1}(\overline{v}) = 0$. The assignment $\alpha_{v\leftarrow 0}$ is similarly defined. For any Boolean formula f, any variable v and any $b \in \{0,1\}$, $f_{v\leftarrow b}$ is defined as $f_{v\leftarrow b}(\alpha) = f(\alpha_{v\leftarrow b})$. The DNF size of a Boolean function h, denoted $\tau(h)$, is the least number of terms required to express h in DNF. The CNF size of h is $\tau(\overline{h})$. The CDNF size of a Boolean function h is the maximum of $\tau(h)$ and $\tau(\overline{h})$. The class CDNF_m consists of all Boolean functions h which have CDNF size at most m. Finally, we define the class of CDNF functions as $\bigcup_{m\geq 0} \mathrm{CDNF}_m$.

Theorem 7 *The class of CDNF functions is polynomial-query learnable.*

Proof: By Corollary 1 it suffices to prove that the class of CDNF functions has polynomial-sized certificates (where the size of a function is defined to be its CDNF size). More precisely, we show that for every Boolean function f of CDNF size at least $p(m,n) = mn + 1$, there exists a polynomial certificate W containing at most $q(m,n) = 2(\log_2 3)m^2n^2 + 3(\log_2 3)mn$ assignments such that no formula in CDNF_m is consistent with f over W.

First note that if $\tau(f) = r$ and $\tau(\overline{f}) = s$ then for any variable v,

- $r/2 \leq \tau(f_{v\leftarrow 0}) \leq r$ or $r/2 \leq \tau(f_{v\leftarrow 1}) \leq r$, and
- $s/2 \leq \tau(\overline{f}_{v\leftarrow 0}) \leq s$ or $s/2 \leq \tau(\overline{f}_{v\leftarrow 1}) \leq s$.

(The lower bound follows from the fact that $f \equiv v f_{v\leftarrow 1} + \overline{v} f_{v\leftarrow 0}$.) More generally, if $\tau(f) \geq p(m,n)$ and $\tau(\overline{f}) \geq 2p(m,n)$ then we can find a set $\{v_1, v_2, ..., v_k\}$ of variables and values $b_1, b_2, ..., b_k \in \{0,1\}$ such that one of $f_{v_1\leftarrow b_1,...,v_k\leftarrow b_k}$ and $\overline{f}_{v_1\leftarrow b_1,...,v_k\leftarrow b_k}$ has size at least $p(m,n)$ and the other has size at most $2p(m,n)$. Since a certificate for such a restriction of f or $\overline{f}$ to k fewer variables can be extended to a certificate for f, there is no loss of generality in supposing (as we do in what follows) that f satisfies $\tau(f) \geq p(m,n)$ and $\tau(\overline{f}) \leq 2p(m,n)$.

We show that there exists a set X of positive assignments of f, such that for every disjunction h of (at most) m implicants of f, more than half of the assignments in X do not satisfy h. Suppose no such set exists. Let J be the set of all positive assignments of f. Since f has CDNF size at least $mn + 1$, it must have at least one negative assignment, and hence $|J| \leq 2^n - 1$. Then we can construct a DNF formula g satisfying f as follows. Initialize g to the empty DNF formula ($False$). Repeat the following until all assignments in J satisfy g: Append to the formula g the disjunction h of (at most) m implicants of f such that h is satisfied by at least half of all assignments in J which do not already satisfy g. Then g contains k terms, for some k satisfying $\frac{1}{2}^k(2^n - 1) < 1$. This implies that $\tau(f) \leq mn < p(m,n)$, a contradiction.

We now claim that there exists a subset $A \subseteq X$ of at most $mn(\log_2 3)$ assignments such that for every disjunction h of (at most) m implicants of f, there is some assignment in A that does not satisfy h. To see this, pick $mn(\log_2 3)$ assignments from X at random (with replacement) to form A. Let $H(f)$ denote the set of disjunctions of at most m implicants of f. Note that $|H(f)| \leq 3^{nm}$ since the total number of possible implicants is 3^n. Therefore,

$$\begin{aligned}
&\text{Prob}[\exists h \in H(f)\text{: } h \text{ is satisfied by all assignments in } A] \\
&\qquad \leq \Sigma_{h\in H(f)} \text{ Prob}[h \text{ is satisfied by all assignments in } A] \\
&\qquad < 3^{mn}(1/2)^{mn(\log_2 3)} \quad \leq 1
\end{aligned}$$

which proves the claim. Fix any such set A.

If any formula $h \in \text{CDNF}_m$ is satisfied by all $mn(\log_2 3)$ positive assignments in A, then h must contain a term t that is not an implicant of f. We now show that any non-implicant of f that is satisfied by an assignment in A must also be satisfied by at least one assignment in a set D of negative assignments of f. To construct the set D, we use the following terminology. If v is a variable that occurs in a term t, define $t(v) = 1$ if the literal v occurs in t and $t(v) = 0$ if the literal $\overline{v}$ occurs in t. For each $x \in A$ define the set D_x of assignments for x as follows: $D_x = \{z_c : c \text{ is a term in } \overline{f}, \text{ for all } v \in c\ z_c(v) = c(v) \text{ and for all } v \notin c\ z_c(v) = x(v)\}$. Note that every assignment in D_x is a negative example of f. We claim that if a non-implicant of f, say t, is satisfied by some $x \in A$, then t is also satisfied by some assignment in D_x. To see this, let y be any negative assignment of f that satisfies t. (There has to be such an assignment since t is not an implicant of f.) Clearly any literal l in t must be satisfied by both x and y. Since $f(y) = 0$ there must be some term c in $\overline{f}$ such that y satisfies all literals in c. Consider the assignment z_c in the definition of D_x. Each literal l in t that corresponds to a variable in c is satisfied by z_c. Each literal l in t that does not correspond to a variable in c is satisfied by x and hence by z_c.

Let $D = \cup_{x\in A} D_x$. From the argument above, no formula in CDNF_m is consistent with f over $A \cup D$. Moreover, $|A \cup D| \leq |A| + |A| \cdot \tau(\overline{f}) \leq mn(\log_2 3)(1 + 2p(m, n)) = q(m, n)$. ■

7 Learning with Membership Queries

The main result of this section is that a projection-closed Boolean concept class is polynomial-query learnable with membership queries alone if and only if it has polynomial teaching dimension. Since the "only if" part of the proof follows from Goldman and Kearns [17], we prove only the "if" part.

Let $\mathcal{C} = \bigcup_{m\geq 1, n\geq 1} \mathcal{C}_{m,n}$ be a Boolean concept class defined by a representation class $\mathcal{R}$, where $\mathcal{C}_{m,n}$ is the set of Boolean concepts on n variables that have a representation in $\mathcal{R}$ of size no greater than m. Note that $\mathcal{C}$ contains only concepts with finite representations.

For any assignment x to V_n, $v \in V_n$, and any $b \in \{0, 1\}$, we define $x_{v\leftarrow b}$ to be the assignment to V_n setting all $v' \in V_n - \{v\}$ as in x, and setting v to b. Let c be a concept in $\mathcal{C}_{m,n}$, v a variable in V_n, and $b \in \{0, 1\}$. The concept c' obtained by *projecting v to b in c*, denoted $c_{v\leftarrow b}$, is on n variables, and is defined by $\chi_{c'}(x) = \chi_c(x_{v\leftarrow b})$ for all assignments x to V_n. We say that $\mathcal{C}$ is *projection-closed* if for all non-negative integers m and n, all $c \in \mathcal{C}_{m,n}$, all $v \in V_n$, and all $b \in \{0, 1\}$, the size of the concept $c_{v\leftarrow b}$ is at most m.

A *specification set* for $c \in \mathcal{C}_{(n)}$ is a set X of assignments such that no other concept in $\mathcal{C}_{(n)}$ is consistent with c over X. This corresponds to the notion of a *specifying sample* in

[8] and *teaching sequence* in [17], and is also related to work of Shinohara and Miyano [29]. We say that the class $\mathcal{C}$ has *polynomial teaching dimension* if there exists a two-variable polynomial function p such that for all $n \geq 1$, and all $c \in \mathcal{C}_{(n)}$, there exists a specification set for c of size at most $p(m,n)$, where m is the size of c. We say that the representation class $\mathcal{R}$ of a Boolean concept class $\mathcal{C}$ is *polynomial membership query learnable* if there exists an algorithm $\mathcal{A}$ and a two-variable polynomial function p such that for all $m, n \geq 1$, and for all unknown target concepts $c \in \mathcal{C}_{m,n}$, given n as input, $\mathcal{A}$ uses at most $p(m,n)$ membership queries (and no equivalence queries) and outputs a representation of c in R.

Theorem 8 *Let $\mathcal{C} = \bigcup_{m \geq 1, n \geq 1} \mathcal{C}_{m,n}$ be a projection-closed Boolean concept class defined by a representation class $\mathcal{R}$. If $\mathcal{C}$ has polynomial teaching dimension then $\mathcal{R}$ is polynomial membership query learnable.*

Proof: Suppose that $\mathcal{C}$ has teaching dimension p, where p is a two-variable polynomial. We show that $\mathcal{C}$ has polynomial certificates. Let $c \subseteq \{0,1\}^n$ be any concept of size greater than m. If $|\mathcal{C}_{m,n}| = 1$, then any assignment on which c differs from the one concept in $\mathcal{C}_{m,n}$ will suffice to form a certificate that that c has size greater than m. If $|\mathcal{C}_{m,n}| > 1$, then since $\mathcal{C}$ is projection-closed, $\mathcal{C}_{m,n}$ must contain the concepts corresponding to the constant functions $True$ and $False$, and it is easy to prove the following: There exists an integer i, $0 \leq i < n$, and an assignment of $b_j \in \{0,1\}$ to each variable v_j, $1 \leq j \leq i$ such that $c_{v_1 \leftarrow b_1, v_2 \leftarrow b_2, \ldots, v_i \leftarrow b_i}$ is not in $\mathcal{C}_{m,n}$ but both $c_1 = c_{v_1 \leftarrow b_1, v_2 \leftarrow b_2, \ldots, v_i \leftarrow b_i, v_{i+1} \leftarrow 0}$ and $c_2 = c_{v_1 \leftarrow b_1, v_2 \leftarrow b_2, \ldots, v_i \leftarrow b_i, v_{i+1} \leftarrow 1}$ are in $\mathcal{C}_{m,n}$. Now, if X_1 and X_2 are specification sets of size at most $p(m,n)$ for c_1 and c_2 respectively, let $X_1' = \{x_{v_1 \leftarrow b_1, \ldots, v_i \leftarrow b_i, v_{i+1} \leftarrow 0} \mid x \in X_1\}$, and let $X_2' = \{x_{v_1 \leftarrow b_1, \ldots, v_i \leftarrow b_i, v_{i+1} \leftarrow 1} \mid x \in X_2\}$. Finally, note that no concept in $\mathcal{C}_{m,n}$ is consistent with c over $X = X_1' \cup X_2'$ (otherwise, there would be a concept in $\mathcal{C}_{m,n}$ equivalent to $c_{v_1 \leftarrow b_1, \ldots, v_i \leftarrow b_i}$) and that $|X| \leq 2p(m,n)$.

We have just shown that $\mathcal{R}$ has polynomial certificates. Therefore, by Corollary 2, any concept in $\mathcal{C}_{m,n}$ can be learned by a membership and equivalence query algorithm that has query complexity $p'(m,n)$, for some polynomial p'. We convert this algorithm into a membership query algorithm (that takes n as input) by simulating each equivalence query $EQ(h)$ with polynomially many membership queries as follows. Since n is given, we may assume that h represents a concept in $\mathcal{C}_{(n)}$. Since the algorithm has query complexity $p'(m,n)$, the size of h is at most $p'(m,n)$, and thus h has a specification set X of size $p(p'(m,n),n)$. Construct the specification set X for h and make membership queries on all assignments in X. If h is consistent with the target concept over X, then return "Yes" for the equivalence query; otherwise return "No" and some assignment in X on which h disagrees with the target concept. The theorem follows. ∎

An easy application of Theorem 8 is that a monotone Boolean function f over n variables is learnable using a number of membership queries at most a polynomial in $\max\{\tau(f), \tau(\overline{f})\}$— one can construct a specification set for f containing at most $n+1$ assignments for each prime implicant of f and $\overline{f}$. (This query bound was previously shown using other methods in [14] and [9]).

A common example of a class that is not polynomial membership query learnable is the class of *monomials*. A monomial is a single term DNF formula. The identically false function is represented by an *inconsistent* monomial containing both a literal and its negation. (If the identically true function is also considered to be represented by a monomial, then this class is projection-closed.) This class includes representations of all Boolean functions with exactly one positive assignment, and therefore the specification set for the identically false function on n variables must include all 2^n assignments on n variables. Thus the class does not have polynomial teaching dimension.

Goldman and Kearns proved that a class which they also call "monomials" does have polynomial teaching dimension. However, under their definition, the constant function $False$ is not representable as a monomial, and hence their class is not projection-closed. Similarly, under their definition, the class of k-term DNF formulas has polynomial teaching dimension, but is not projection-closed. Under a more standard definition, the class of k-term DNF formulas includes all formulas with at most k terms (including the 0-term DNF formula representing the function $True$), and is projection-closed. Since this class includes the monomials (as defined in the previous paragraph), it does not have polynomial teaching dimension, and is therefore not polynomial membership query learnable.

Let $V_n = \{v_1, v_2, \ldots, v_n\}$ be a set of n Boolean variables. A *monotone decision list* over V_n is a list $L = \langle (y_1, b_1), \ldots, (y_l, b_l) \rangle$ such that $l \geq 0$, each $y_i \in V_n$, and each $b_i \in \{0, 1\}$. The function f represented by L is defined as follows: for each assignment α to V_n, $f(\alpha) = b_j$ where $1 \leq j \leq l$ is the least value such that $\alpha(y_j) = 1$; $f(\alpha) = 0$ if there is no such j. Goldman and Kearns proved that this class has teaching dimension at most $2n - 1$. This class is not projection-closed, but becomes so if a representation of $True$ is added to the class. Specification sets for the new class can be obtained by (i) adding the assignment setting all variables to 0 to the Goldman and Kearns' specification set of a monotone decision list, and (ii) creating a specification set for $True$, consisting of the single assignment setting all variables to 0. Thus the new class has teaching dimension at most $2n$, and consequently monotone decision lists are polynomial membership query learnable. In fact, a polynomial-time membership query algorithm can easily be derived from the teaching dimension proof in [17].

We now show that the class of $\log n$-DNF $\cap$ $\log n$-CNF formulas is polynomial membership query learnable. A k-DNF formula is a DNF formula in which every term has at most k literals. A k-CNF formula is a CNF formula in which every clause has at most k literals. The class r-DNF $\cap$ s-CNF contains all Boolean formulas that can be represented as r-DNF formulas and as s-CNF formulas. This class is projection-closed. (We assume that $True$ and $False$ are special: they are both 0-DNF and 0-CNF formulas.)

A (n, k)–universal set is a set $X = \{b_1, b_2, ..., b_t\} \subseteq \{0, 1\}^n$ such that every term of at most k literals is satisfied by at least one assignment in X. A straightforward application of the probabilistic method [15] shows that there is a (n, k)–universal set of size $t = O(k2^k \log n)$. Naor and Naor [26] show that a (n, k)–universal set of size $t = O(k2^{3k} \log n)$ can be constructed in time polynomial in n and 2^k.

Theorem 9 *The class $\mathcal{Q} = \log n$-DNF $\cap \log n$-CNF is learnable with a polynomial number of membership queries.*

Proof: By Theorem 8, it suffices to show that $\mathcal{Q}$ has polynomial teaching dimension. We do this by showing that for any concept $c \in \mathcal{Q}$, any $(n, 2\log n)$-universal set X is a specification set for c. If $c' \in \mathcal{Q}$ is a concept not equivalent to c, then one can construct a partial assignment of 0's and 1's to a subset of the n variables that either satisfies a term of c and a term of $\overline{c'}$ or a term of $\overline{c}$ and a term of c'. Any extension of such a partial assignment to a full assignment will be evaluated differently by c and c'. Since every term of c, $\overline{c}$, c', and $\overline{c'}$ has at most $\log n$ literals, and X is a $(n, 2\log n)$-universal set, all partial assignments of $2\log n$ variables are represented in X. Hence X must contain an assignment on which c and c' differ. Finally, note that the size of X is bounded by a polynomial in n. ■

The set X used in the proof above provides a "universal" specification set for all the concepts in $\mathcal{Q}$. Therefore, a membership query based algorithm for learning $\mathcal{Q}$ only needs to construct a $(n, 2\log n)$-universal set of assignments and make membership queries on those assignments—the answers to the queries specify the concept uniquely. Clearly, the proof generalizes to the class of $O(\log n)$-DNF $\cap$ $O(\log n)$-CNF formulas and also shows that the number of concepts in the class is at most $2^{poly(n)}$. The class $\mathcal{Q}$ contains the class of $\log n$-depth decision trees, shown by Kushilevitz and Mansour [25] to be exactly learnable in polynomial time with membership queries using a representation based on Fourier coefficients. More generally, $\mathcal{Q}$ contains the class of $\log n$-depth branching programs, which is also polynomial-query learnable as a consequence of Theorem 9. Bshouty et al. [14] used the Monotone Theory to show that the class $\mathcal{Q}$ itself is learnable with membership queries and an NP-oracle, in a representation of depth-three Boolean formulas. Our result (combined with Naor and Naor's efficient construction of universal sets) shows that one can learn $\mathcal{Q}$ in polynomial time with a polynomial number of membership queries in a $(n, 2\log n)$-universal set representation of concepts in $\mathcal{Q}$. The only caveat here is that we do not know if this representation is "honest"—that is, given any arbitrary assignment x and a $(n, 2\log n)$-universal set representation of a concept c in $\mathcal{Q}$, we do not have a polynomial time algorithm to decide if x satisfies c. However, it is easy to prove that this decision problem is in NP $\cap$ co-NP.

8 Conclusions

We have presented a characterization of exact learning with a minimally adequate teacher. The characterization suggests many new questions. For example, Angluin and Kharitonov [7] proved that many concept classes—NFA, CFG, Boolean Formulas, finite unions and intersections of DFA—are not even predictable with membership queries, given cryptographic assumptions. Do any of these classes have polynomial certificates? If they do not, we have an assumption-free proof that they are not exactly learnable with queries. If they do, we have a proof that they are exactly learnable with queries at least in some low level of the polynomial hierarchy. Another interesting open question is whether our proof of the polynomial-query learnability of CDNF formulas can be extended to the class of all DNF formulas.

An intriguing possible use of the characterization is in the development of efficient (polynomial time) learning algorithms for a concept class. As a first step, one could show the existence of polynomial certificates for the class and thereby establish at least polynomial-query learnability. The proof of certificates could then direct the search for an efficient polynomial time learning algorithm. We need more applications of the characterization to decide if this really is a viable approach.

Bshouty [13] and Hegedüs [19] have observed that a concept class is learnable with a polynomial number of membership queries if and only if it has polynomial teaching dimension and has polynomial certificates. (The proof follows along the same lines as the proof of Theorem 8.) Moreover, Hegedüs has developed this further to give upper and lower bounds on the number of membership queries needed to learn a concept class.

Bshouty [13] has also noted the following interesting relationship of polynomial query learnability to the decision problem of whether a given representation of a concept is of smallest size. For many representation classes (for example, Boolean circuits, Boolean formulas, DNF formulas) this decision problem is easily shown to be in Π_2^p but it is not known if the problem is Π_2^p-hard. By way of Corollary 3, if an honest representation class $\mathcal{R}$ admits a polynomial-query learning algorithm with membership and equivalence queries of size no greater than the target concept, then the problem of deciding if a given representation r in $\mathcal{R}$ is of smallest size is in Σ_2^p. Consequently, the existence of such a learning algorithm for (say) Boolean circuits would show that the corresponding decision problem is not Π_2^p-hard (assuming $\Sigma_2^p \neq \Pi_2^p$).

Finally, we would like to know if the $(n, 2\log n)$-universal set representation of class $\mathcal{Q}$ in the last section is "honest." An efficient algorithm for evaluating assignments given such a representation would immediately imply an exact membership query learning algorithm for class $\mathcal{Q}$.

References

[1] H. Aizenstein, L. Hellerstein, and L. Pitt. Read-thrice DNF is Hard to Learn with Membership and Equivalence Queries. *Proceedings of the 33rd IEEE Conference on the Foundations of Computer Science*, 1992.

[2] D. Angluin. Learning Regular Sets from Queries and Counterexamples. *Information and Computation*, 75:87–106, 1987.

[3] D. Angluin. Queries and Concept Learning. *Machine Learning*, 2:319–342, 1988.

[4] D. Angluin. Negative Results for Equivalence Queries. *Machine Learning*, 5:121–150, 1990.

[5] D. Angluin, M. Frazier, and L. Pitt. Learning Conjunctions of Horn Clauses. *Proceedings of the 31st Annual IEEE Symposium on the Foundations of Computer Science*, pages 186–192, 1990.

[6] D. Angluin, L. Hellerstein, and M. Karpinski. Learning Read-Once Formulas with Queries. *Journal of the ACM*, pages 185–210, 1993.

[7] D. Angluin and M. Kharitonov. When Won't Membership Queries Help? *Proceedings of the ACM Symposium on Theory of Computing*, pages 444–454, 1991.

[8] M. Anthony, G. Brightwell, and J. Shawe-Taylor. On Specifying Boolean Functions by Labelled Examples. Technical report, London School of Economics, London, U.K., 1992.

[9] J. C. Bioch and T. Ibaraki. Complexity of Identification and Dualization of Positive Boolean Functions. Technical report RRR 25-93, Rutgers Center for Operations Research, 1993.

[10] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. Warmuth. Learnability and the Vapnik-Chervonenkis Dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[11] Y. Breitbart. Complexity of the Calculation of Predicates by Finite Automata. PhD Thesis, Technion, Haifa, Israel, June 1973.

[12] N. Bshouty. Exact Learning Boolean Functions via the Monotone Theory. *Information and Computation*, 123(1):146–153, Nov 1995.

[13] N. Bshouty. Personal communication, November 1994.

[14] N. Bshouty, R. Cleve, S. Kannan, and C. Tamon. Oracles and Queries that are Sufficient for Exact Learning. *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*, pages 130–139, 1994.

[15] P. Erdös and J. Spencer. *Probabilistic Methods in Combinatorics.* Probability and Mathematical Statistics A Series of Monographs and Textbooks. Academic Press, 1974.

[16] R. Gavaldà. On the Power of Equivalence Queries. *Proceedings of EUROCOLT '93*, pages 193–203, 1993.

[17] S. Goldman and M. Kearns. On the Complexity of Teaching. *Proceedings of the Fourth Annual Workshop on Computational Learning Theory*, pages 303–314, 1991.

[18] T. Hancock. Identifying μ-Formula Decision Trees with Queries. *Proceedings of the Third Annual Workshop on Computational Learning Theory*, pages 23–37, 1990.

[19] T. Hegedüs. Generalized Teaching Dimensions and the Query Complexity of Learning. *Proceedings of the Eighth Annual Conference on Computational Learning Theory*, pages 108–117, 1995.

[20] John Hopcroft and Jeffrey Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979.

[21] S. Kannan. On the Query Complexity of Learning. *Proceedings of the Sixth Annual ACM Conference on Computational Learning Theory*, pages 58–66, 1993.

[22] J. Kaneps and R. Freivalds. Running Time to Recognize Nonregular Languages by 2-way Probabilistic Finite Automata. In J. Leach Albert, B. Monien, and M. Rodriǵuez Artalejo, editors. *ICALP '91 (18th International Colloquium on Automata, Languages, and Programming)*, Vol. 510 of *Lecture Notes in Computer Science*, pages 355–361. Springer-Verlag, 1991.

[23] R. M. Karp. Some Bounds on the Storage Requirements of Sequential Machines and Turing Machines *J. Assoc. Comput. Mach.*, 14, pages 478–489, 1967.

[24] M. Y. Moshkov. Conditional Tests. *Problemy Kibernitiki* (in Russian), 40:131–170, 1983.

[25] E. Kushilevitz and Y. Mansour. Learning Decision Trees Using the Fourier Spectrum. *SIAM Journal of Computing*, 22(6):1331–1348, 1993.

[26] J. Naor and M. Naor. Small-bias Probability Spaces: Efficient Constructions and Applications. *Proceedings of the 22nd Annual ACM Symposium on Theory of Computing*, pages 213–223, 1990.

[27] K. Pillaipakkamnatt and V. Raghavan. On the Limits of Proper Learnability of Subclasses of DNF Formulas. *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory*, pages 118–129, 1994.

[28] R. Rivest and R. Schapire. Inference of Finite Automata using Homing Sequences. *Proceedings of the 21st Annual ACM Symposium on Theory of Computing*, pages 411–420, 1989.

[29] A. Shinohara and S. Miyano. Teachability in Computational Learning. *New Generation Computing*, 8:337–347, 1991.

[30] L. Stockmeyer. On Approximation Algorithms for #P. *SIAM Journal on Computing*, 14:849–861, 1985.

[31] L. Valiant. A Theory of the Learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Algorithm **Learn from a Minimally Adequate Teacher (Revisited)**

$n \leftarrow 0$
$S \leftarrow \emptyset$
$m \leftarrow$ Least i such that $\mathcal{C}_i \neq \emptyset$
$h \leftarrow$ A representation of size at most m of any concept in $\mathcal{C}_m$
While $EQ(h) \neq$"Yes" **do**
  Let x be the counterexample to h
  $w \leftarrow$ Any representation in $RCons(S, m)$
  If $EQ(w) =$"Yes" **then**
    Output w and halt
  Endif
  Let y be the counterexample to w
  $S \leftarrow S \cup \{(x, MQ(x)), (y, MQ(y))\}$
  $m \leftarrow$ Least i such that $RCons(S, i) \neq \emptyset$
  $n \leftarrow \max(n, |x|, |y|)$
  While no concept in $\mathcal{C}_{p(m,n)}$ is consistent with any concept in $\frac{1}{32}$-$PMAJ(RCons(S, m))$
    over $\Sigma^{\leq n}$ **do**
    Let X be a set of strings such that $\Sigma_{x \in X} |x| \leq q(m, n)$ and no concept
    in $\mathcal{C}_m$ is consistent with some concept in $\frac{1}{2}$-$PMAJ(RCons(S, m))$ over X
    $S' \leftarrow \bigcup_{x \in X} \{(x, MQ(x))\}$
    $S \leftarrow S \cup S'$
    $m \leftarrow$ Least i such that $RCons(S, i) \neq \emptyset$
  Endwhile
  $h \leftarrow$ A representation of size at most $p(m, n)$ of any concept in $\mathcal{C}$ consistent
    with a concept in $\frac{1}{512}$-$PMAJ(RCons(S, m))$ over $\Sigma^{\leq n}$
Endwhile

Figure 2: An Algorithm for Learning $\mathcal{R}$ in the Polynomial Hierarchy