International Journal of Foundations of Computer Science

ORACLES IN Σ_2^p ARE SUFFICIENT FOR EXACT LEARNING*

Johannes Köbler and Wolfgang Lindner†
Institut für Informatik, Humboldt-Universität zu Berlin, Unter den Linden 6
D-10099 Berlin, Germany

Received (received date)
Revised (revised date)
Communicated by Editor's name

ABSTRACT

We study the learnability of representation classes in Angluin's exact learning model. In particular, we consider the following three query types: equivalence queries, equivalence and membership queries, and membership queries only. We show in all three cases that polynomial query complexity implies already polynomial-time learnability, provided that the learner additionally has access to an oracle in Σ_2^p. It follows that boolean circuits are polynomial-time learnable with equivalence queries and the help of an oracle in Σ_2^p. [a]

Keywords: Query Learning, Complexity, Polynomial Hierachy, Universal Hashing

1. Introduction

When considering efficient learning notions two questions arise immediately. First, one may ask for the combinatorial, or information-theoretic complexity of the concept classes learnable in the model when no resource bounds on the learner are involved. Second, one may consider the computational complexity of the learning task in the model.

In the PAC-learning paradigm [17], both questions are well understood. The Vapnik-Chervonenkis dimension of a concept class gives a necessary and sufficient condition for PAC-learnability when the learner has unlimited computational power [5]. Moreover, it follows from [4] that every honest class which can be learned from a polynomial number of examples, can already be learned by a polynomial-time algorithm with additional access to an oracle in $\mathcal{NP}$ [14].

In the context of exact learning via queries [1], combinatorial properties characterizing the polynomial query complexity have been given for each of the three possible combinations of equivalence and membership queries. As shown in [2, 6], a representation class $\mathcal{R}$ is polynomial-query learnable with equivalence queries if

*Some of these results were presented at the 8th International Workshop on Algorithmic Learning Theory [12].

†Supported by DFG grant Ko1053/1-1.

and only if it does not have approximate fingerprints. Further, it has been shown in [10] that $\mathcal{R}$ is polynomial-query learnable with equivalence and membership queries if and only if $\mathcal{R}$ has polynomial certificates. Finally, in [9, 10] it has been observed that $\mathcal{R}$ is polynomial-query learnable with membership queries alone if and only if $\mathcal{R}$ has polynomial-size specifying sets.

Concerning the computational complexity, it has been shown that the learning tasks for honest concept classes can always be accomplished in polynomial time with additional access to an oracle in certain levels of the polynomial-time hierarchy. An oracle in Σ_5^p has been used for both the cases of equivalence queries and membership queries alone [6]. In the case of equivalence and membership queries an oracle in Σ_4^p is sufficient [10]. In the particular case of learning boolean circuits with equivalence queries an oracle in Σ_3^p has been shown to be sufficient in [6]. Moreover, an oracle in $\mathcal{NP}$ *is* sufficient for learning honest classes with membership queries and learning boolean circuits with equivalence queries, though at the expense of a *randomized* computation with expected polynomial time [6]. Thus the precise location of the oracle complexity that is sufficient for deterministic polynomial-time learning with the help of an additional oracle in Angluin's exact learning model is still an interesting open question.

In this paper we further investigate the computational complexity of the learning task for honest concept classes with polynomial query-complexity. We show that in all three cases, polynomial query complexity implies already polynomial-time learnability where the learner additionally has access to an oracle in Σ_2^p. We conclude that boolean circuits are polynomial-time learnable with equivalence queries and the help of an oracle in Σ_2^p. Under the widely accepted hypothesis that the polynomial-time hierarchy does not collapse this improves on the above mentioned results concerning query learning in deterministic polynomial time with the help of an oracle.

2. Preliminaries

We adopt the terminology from [2, 6, 10]. Let Σ be an alphabet. Then $\Sigma^* = \bigcup_{n\geq 0} \Sigma^n$ is the set of all strings over Σ where Σ^n consists of all strings of length exactly n. The length of a string $x \in \Sigma^*$ is denoted by $|x|$. For a subset $X \subseteq \Sigma^n$, X^l denotes the set $\{x_1 \cdots x_l \in \Sigma^{nl} : x_i \in X \text{ for } i = 1, \ldots, l\}$. For a finite set A, we denote by $|A|$ the cardinality of A and by log we denote the function $\log n = \max\{1, \lceil \log_2 n \rceil\}$.

We assume that the reader is familiar with fundamental complexity theoretic concepts such as (oracle) Turing machines and the polynomial-time hierarchy (see, for example, [3]).

A *representation class* $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ is a 4-tuple where Σ and Δ are finite alphabets, $R \subseteq \Delta^*$ is the set of *representations*, and μ is a mapping from R to subsets of Σ^*, called *concepts*. The *concept class* defined by $\mathcal{R}$ is $\mathcal{C} = \{\mu(r) : r \in R\}$. The *size* of a representation $r \in R$ is $|r|$. The *size* of a concept $c \in \mathcal{C}$ is $|c| = \min_{\mu(r)=c} |r|$, i.e., the size of the smallest representation of c. Concepts $c \notin \mathcal{C}$ are said to have infinite size.

A representation class $\mathcal{R}$ is said to be *honest*, if

- for a given $r \in \Delta^*$ it can be checked in time polynomial in $|r|$ whether r belongs to R, and
- for a given string $x \in \Sigma^*$ and a representation $r \in R$ it can be decided in time polynomial in $|x|$ and $|r|$ whether x belongs to $\mu(r)$.

Let $\mathcal{R} = \langle \Sigma, \Delta, R, \mu \rangle$ be a representation class for a concept class $\mathcal{C}$. Then R_m denotes the set of all representations $r \in R$ of size at most m,

$$R_m = \{r \in R : |r| \leq m\}.$$

The corresponding class of concepts is

$$\mathcal{C}_m = \{\mu(r) : r \in R_m\}.$$

By abuse of notation we denote the *characteristic function* of a concept c also by c, i.e., for all $x \in \Sigma^*$,

$$c(x) = \begin{cases} 0, & x \notin c, \\ 1, & x \in c. \end{cases}$$

If r is a representation, then we simply write $r(x)$ instead of $\mu(r)(x)$.

In this paper we study the exact learning model where the "learner" asks the "teacher" questions of a certain type about the *target concept* c. The goal of the exact learning algorithm is to determine a representation of the target concept, regardless of which concept $c \in \mathcal{C}$ is actually chosen as the target concept. In particular, we study the following two types of queries.

- In a *membership query*, the learning algorithm asks whether a string x belongs to the target concept c and the teacher answers with $c(x)$.
- In an *equivalence query*, the learning algorithm asks whether a representation $h \in R$ represents c. (In this context, $\mu(h)$ is also called the hypothesis of the query.) The teacher answers with "Yes" in case $\mu(h) = c$, and by giving some *counterexample* string y with $h(y) \neq c(y)$, otherwise.

A learning algorithm $\mathring{A}$ *exactly identifies* a representation class $\mathcal{R}$ if for all target concepts $c \in \mathcal{C}$, $\mathring{A}$ eventually halts and outputs a string $r \in R$ with $\mu(r) = c$. A learning algorithm $\mathring{A}$ has *polynomial query complexity* if there is a two-variable polynomial p such that for any unknown target concept c of size m, in each stage in the run of $\mathring{A}$, the sum of the lengths of all inputs to membership and equivalence queries of $\mathring{A}$ as well as the length of the counterexamples given by the teacher is bounded by $p(l, m)$, where l refers to the length of the longest counterexample received up to this stage, or $l = 0$ if no equivalence queries have been used. The learning algorithm $\mathring{A}$ has *polynomial-time complexity* if in each stage in the run of $\mathring{A}$ also the time used by $\mathring{A}$ is bounded by $p(l, m)$. A representation class $\mathcal{R}$ is said to be *polynomial-query learnable* (with equivalence and/or membership queries) if there is an algorithm $\mathring{A}$ with polynomial query complexity which exactly identifies $\mathcal{R}$. If, additionally, $\mathring{A}$ has polynomial time complexity, then $\mathcal{R}$ is called *polynomial-time query learnable*.

3. Generating Good Hypotheses

Let us briefly recall the general strategy of the so-called “majority vote” or “halving” learning algorithm. Here we successively eliminate a certain fraction from a current hypothesis space $\mathcal{K}$. Elimination is done according to the information obtained from the teacher in response to equivalence and/or membership queries. This means that all concepts are eliminated from $\mathcal{K}$ that are not consistent with the answers provided by the teacher. Since we have no initial knowledge of the size of the target concept, we start with $\mathcal{K}$ set to $\mathcal{C}_m$ for the smallest size-bound m such that $\mathcal{C}_m$ is not empty. Then the elimination step is repeated until we identify the target concept, or $\mathcal{K}$ becomes empty. In the latter case we proceed with the next size-bound m for which there is some concept consistent with the answers provided by the teacher so far. Since the target concept is always consistent with the answers provided by the teacher, it will never be eliminated. Thus m will eventually reach the size of the target concept, and then we identify the target concept.

Obviously, the number of iterations we need to identify the target concept depends on the size of the fraction we eliminate from $\mathcal{K}$ in every iteration. In all of the three models we are considering, that is, equivalence queries, equivalence and membership queries, and only membership queries, the combinatorial properties characterizing the respective polynomial query complexities always ensure the existence of queries such that any answer from the teacher eliminates at least an inverse polynomial fraction of concepts from $\mathcal{K}$. Since the initial size of $\mathcal{K}$ is at most exponential in m, this guarantees the termination of the learning algorithm within a polynomial number of steps.

Concerning the *computational* complexity of the learning algorithm first note that with the help of a sample set $S \subseteq \{0,1\}^n \times \{0,1\}$ that always contains all the information received from the teacher we can represent the current hypothesis space $\mathcal{K}$ by its representations consistent with S,

$$R_m(S) = \{r \in R_m : [\forall (x,b) \in S : r(x) = b]\}.$$

Then for a honest representation class, the set $R_m(S)$ is polynomial-time decidable since S will always contain only polynomially many pairs (x,b). Furthermore, we can check whether there is still more than one concept left in the current $R_m(S)$ in polynomial time with the help of an oracle in $\mathcal{NP}$. So the crucial question is how to *find* queries such that any answer from the teacher guarantees the elimination of a sufficiently large fraction from the current set $R_m(S)$ of hypotheses. In all three cases it is central to find a hypothesis that agrees on every example with a significant fraction of $R_m(S)$.

Definition 1 *A concept $c \subseteq \Sigma^*$ is* δ-good on $X \subseteq \Sigma^*$ for a finite set $\mathcal{T}$ *of concepts if, for all $x \in X$, c represents at least a δ-fraction of $\mathcal{T}$,*

$$(\forall x \in X) : \frac{|\{d \in \mathcal{T} : d(x) = c(x)\}|}{|\mathcal{T}|} \geq \delta.$$

Similarly, c is called δ-good on X *for a finite set T of representations if the above condition holds with T in place of $\mathcal{T}$.*

Let us emphasize that the fraction corresponding to the *concepts* in $\mathcal{T}$ might be very different from the fraction corresponding to the *representations* in T, even if T represents precisely $\mathcal{T}$.

In [6] it has been shown that a good hypothesis for a set of representations can be computed in polynomial time with an oracle in Σ_3^p. The proof is based on universal hashing [7], a well-known technique which can be used to estimate the size of a finite set X of strings [15]. Here we show that generating a good hypothesis can be actually accomplished in polynomial time with the help of an oracle in Σ_2^p. Our improvement makes use of a refinement of the hashing-technique (the refinement has been developed in [11] to show that boolean circuits are polynomial-time learnable with equivalence queries where the learner additionally has access to an oracle in Σ_2^p), so let us procceed to recall the definitions and basic properties of hashing that we need.

A *linear hash function* h from $\{0,1\}^m$ to $\{0,1\}^k$ is given by a boolean (k,m)-matrix (a_{ij}) and maps any string $x = x_1 \dots x_m$ to a string $y = y_1 \dots y_k$, where y_i is the inner product $a_i \cdot x = \sum_{j=1}^{m} a_{ij}x_j \pmod 2$ of the i-th row a_i and x. Let $X \subseteq \{0,1\}^m$ and let h be a linear hash function from $\{0,1\}^m$ to $\{0,1\}^k$. Then we say that h *hashes* X if for all pairs of different strings $x, y \in X$, $h(x) \neq h(y)$. More generally, if H is a sequence $(h_1, \dots, h_s)$ of linear hash functions from $\{0,1\}^m$ to $\{0,1\}^k$, then we say that H *hashes* X if for every $x \in X$ there is some i, $1 \leq i \leq s$, such that $h_i(x) \neq h_i(y)$, for all $y \in X - \{x\}$. Note that the predicate "H hashes X" can be decided in $co\mathcal{NP}$ provided that membership in X can be tested in $\mathcal{P}$. We denote the set of all families $H = (h_1, \dots, h_s)$ of s linear hash functions from $\{0,1\}^m$ to $\{0,1\}^k$ by $\mathcal{H}(s, m, k)$.

The following two lemmas due to Sipser give a first approach to estimate the size of a finite set X by using linear hash functions. The first lemma is easily proved by a pigeon-hole argument.

Lemma 1 [15] *No hash family $H \in \mathcal{H}(s,m,k)$ can hash a set $X \subseteq \{0,1\}^m$ of cardinality $|X| > s2^k$.*

The following converse of Lemma 1 is also known as Sipser's "Coding Lemma".

Lemma 2 [15, Coding Lemma] *Let $X \subseteq \{0,1\}^m$ be a set of cardinality $|X| \leq 2^{k-1}$. If we choose a hash family H uniformly at random from $\mathcal{H}(k,m,k)$, then with probability more than $1/2$, H hashes X.*

Thus, the cardinality of a set $X \subseteq \{0,1\}^m$ can be estimated by searching the smallest integer k_0 for which there exists a hash family H from $\mathcal{H}(k_0, m, k_0)$ that hashes X. Then, by Lemmas 1 and 2, for the cardinality of X it holds that $2^{k_0-2} < |X| \leq k_0 2^{k_0}$. By hashing a suitable product X^l instead of X, Stockmeyer [16] showed how to estimate the cardinality of X with arbitrary precision. This well-known result is usually stated in terms of the approximability of functions in $\#\mathcal{P}$ in polynomial time with an oracle in Σ_2^p.

Theorem 1 [16] *For every function f in $\#\mathcal{P}$, there is a function g computable in polynomial-time with a Σ_2^p oracle such that, for all strings $x \in \{0,1\}^*$ and integers $e \geq 1$,*

$$f(x) \leq g(x, 0^e) \leq (1 + 1/e) f(x).$$

Gavaldà [8] extended Sipser's Coding Lemma to the case of a collection $\mathcal{X}$ of exponentially many sets. We use the following refinement of this result.

Lemma 3 [13] *Let $\mathcal{X}$ be a collection of at most 2^n different subsets of $\{0,1\}^m$, each of which has cardinality at most 2^{k-1}. If we choose a hash family H uniformly at random from $\mathcal{H}(s,m,k)$, then with probability at least $1-2^{n+k-s-1}$, H hashes every set $X \in \mathcal{X}$.*

By combining Lemmas 1 and 3 we get the following lemma which allows us to estimate the maximal cardinality of a collection of sets in a similar way as Lemmas 1 and 2 can be used for the cardinality of a single set.

Lemma 4 *Let $\mathcal{X}$ be a collection of at most 2^n subsets of $\{0,1\}^m$. Further, let $a, e, l \geq 1$ be integers with $l \geq 1 + 4ne + 8me^2$ and let $k = 1 + \lceil l \log a \rceil$. Then the following implications hold.*

(i) For any $X \subseteq \{0,1\}^m$, if there exists a hash family $H \in \mathcal{H}(n+k, ml, k)$ that hashes X^l, then $|X| \leq (1+1/e)a$.

(ii) If for every set $X \in \mathcal{X}$, $|X| \leq a$, then some hash family $H \in \mathcal{H}(n+k, ml, k)$ hashes X^l for all sets $X \in \mathcal{X}$.

Proof. To show the first implication, assume that some $H \in \mathcal{H}(n+k, ml, k)$ hashes X^l. We have to show that then $|X| \leq (1+1/e)\,a$. Observe that this trivially holds if $a \geq 2^m$. In the case that $a < 2^m$ it follows by Lemma 1 that only sets $Y \subseteq \{0,1\}^{ml}$ with $|Y| \leq (n+k)2^k$ can be hashed by some $H \in \mathcal{H}(n+k, ml, k)$, and thus

$$\begin{aligned} |X^l| &\leq (n+k)2^k \\ &= (1+n+\lceil l \log a \rceil)2^{1+\lceil l \log a \rceil} \\ &< 4(2+n+l\log a)a^l \\ &< (1+1/e)^l a^l. \end{aligned}$$

For the last inequality we used that $l \geq 1 + 4ne + 8me^2$ and $e \geq 1$:

$$\begin{aligned} (1+1/e)^l &\geq 1 + \underbrace{l/e}_{>8+4n} + l\underbrace{(l-1)/2e^2}_{>4m} \\ &> 9 + 4n + 4lm \\ &> 4(2+n+l\log a), \text{ since } 2^m > a. \end{aligned}$$

This proves (i). To show (ii), assume that $|X| \leq a$, for all $X \in \mathcal{X}$. Since $k = 1 + \lceil l \log a \rceil$, this implies that $|X^l| \leq a^l \leq 2^{k-1}$ for all $X \in \mathcal{X}$. By Lemma 3 (letting $s = n+k$) it follows that a uniformly at random chosen hash family H from $\mathcal{H}(n+k, ml, k)$ hashes with probability at least $1 - 2^{n+k-(n+k)-1} = 1/2$ all the sets X^l, $X \in \mathcal{X}$. □

Now let us return to the question how to compute a good hypothesis. In [6], Lemmas 1 and 2 together with Theorem 1 are used to generate a good hypothesis h

for a given set T of representations as follows. The basic idea is to build h by prefix search. This amounts to decide whether there exists a hypothesis h extending some already determined prefix of h which is good for T, i.e., for all strings x the number of representations in T which agree with h on x is at least $\delta|T|$. By Lemmas 1 and 2, this is essentially equivalent to the fact that there exists a hypothesis h such that no hash family H from a suitable chosen set $\mathcal{H}(k,m,k)$ hashes the set of all representations in T that agree with h on any string x:

$$\exists h \ \forall H \ \forall x : H \text{ does not hash } \{r \in T : r(x) = h(x)\}.$$

Recall that if T can be decided in polynomial time, then the predicate "H does not hash $\{r \in T : r(x) = h(x)\}$" can be decided in $\mathcal{NP}$. This adds a second quantifier alternation to the above formula and thus leads to polynomial-time computation with an oracle in Σ_3^p.

To compute a good hypothesis with an oracle in Σ_2^p instead of Σ_3^p we use Lemma 4 in place of Lemmas 1 and 2 and consider the set of representations in T which *disagree* with h on a given string x, i.e., instead of deciding whether there exists some h such that for all strings x it holds that $|\{r \in T : r(x) = h(x)\}| \geq \delta|T|$, we check the equivalent fact that there exists some h such that for all x it holds that $|\{r \in T : r(x) \neq h(x)\}| \leq (1-\delta)|T|$. The latter is, by Lemma 4, essentially equivalent to saying that there exists some hypothesis h and a hash family H from a suitable chosen set $\mathcal{H}(s,m,k)$ that *simultaneously* hashes all sets $\{r \in T : r(x) \neq h(x)\}$, for all strings x:

$$\exists h \ \exists H \ \forall x : H \text{ hashes } \{r \in T : r(x) \neq h(x)\}.$$

Since the predicate "H hashes $\{r \in T : r(x) = h(x)\}$" can be decided in $co\mathcal{NP}$, this adds no additional quantifier alternation, and thus we can save one alternation compared to the previous approach. An elaboration of this idea yields the following theorem.

Theorem 2 *Let $\mathcal{R}$ be a honest representation class. Then there is a set A decidable in polynomial time relative to an oracle in Σ_2^p such that, for all integers $m,n,s,d \geq 1$ and for every sample S, we have the following two implications.*

- *(i) If $\langle 0^m, 0^n, S, 0^s, 0^d \rangle \in A$ then there exists a representation of size at most s that is $1/(d+1)$-good on $\Sigma^{\leq n}$ for $R_m(S)$.*
- *(ii) If $\langle 0^m, 0^n, S, 0^s, 0^d \rangle \notin A$ then no representation of size at most s is $1/d$-good on $\Sigma^{\leq n}$ for $R_m(S)$.*

Moreover, in the first case it is possible to compute in polynomial time relative to an oracle in Σ_2^p a representation of size at most s which is $1/(d+1)$-good on $\Sigma^{\leq n}$ for $R_m(S)$.

Proof. For notational convenience and w.l.o.g. we assume that $\Sigma = \Delta = \{0,1\}$. Our task is to find out whether there is a representation $h \in R_s$ which is $1/(d+1)$-good on $\Sigma^{\leq n}$ for $R_m(S)$, i.e., h fulfills for all $x \in \Sigma^{\leq n}$ the inequality

$$\frac{|\{r \in R_m(S) : r(x) = h(x)\}|}{|R_m(S)|} > 1/(d+1).$$

This is equivalent to saying that for all $x \in \Sigma^{\leq n}$, h fulfills the inequality

$$\frac{|\{r \in R_m(S) : r(x) \neq h(x)\}|}{|R_m(S)|} < 1 - 1/(d+1).$$

Thus our goal can be restated as follows: find out whether there is a representation $h \in R_s$ such that the collection $\mathcal{X}_h$ defined as

$$\mathcal{X}_h = \{R_m(S \cup \{(x, \neg h(x))\}) : x \in \Sigma^{\leq n}\}$$

contains only sets of size at most $(1 - \frac{1}{d+1})|R_m(S)|$. Lemma 4 provides a tool to estimate the cardinality of all sets in a collection $\mathcal{X}$, provided that all strings in the sets of the collection have the same length. So we use $\hat{r} = r10^{|r|-m}$ to encode any representation $r \in R_m$ by a binary string of length exactly $m+1$ and replace for any sample S the set $R_m(S)$ by the equally sized set $\hat{R}_m(S) = \{\hat{r} : r \in R_m(S)\} \subseteq \{0,1\}^{m+1}$. The collection $\hat{\mathcal{X}}_h$ is defined accordingly.

Now let $\tilde{a}$ be an approximation for the number $|\hat{R}_m(S)|$ with

$$|\hat{R}_m(S)| \leq \tilde{a} \leq (1 + 1/d^3)|\hat{R}_m(S)|.$$

By Theorem 1, $\tilde{a}$ can be computed in polynomial time relative to a Σ_2^p-oracle. We define the set A as

$\langle 0^m, 0^n, S, 0^s, 0^d \rangle \in A \iff$ there exists a representation $h \in R_s$ and a hash family $H \in \mathcal{H}(n+k, ml, k)$ that hashes X^l for all sets $X \in \hat{\mathcal{X}}_h$,

where $l = 1 + 4(n+1)d^3 + 8(m+1)d^6$ and $k = 1 + \lceil l \log(\frac{d-1}{d}\tilde{a}) \rceil$. Since $\tilde{a}$ can be computed in polynomial time relative to a Σ_2^p-oracle, it follows that also A is decidable in polynomial time relative to an oracle in Σ_2^p.

Note that $\hat{\mathcal{X}}_h$ contains no more than 2^{n+1} different sets. Thus, using Lemma 4 and the inequalities $\tilde{a} \leq (1+1/d^3)|R_m(S)|$ and $(1-1/d)(1+1/d^3)^2 < 1-1/(d+1)$ it follows from the assumption $\langle 0^m, 0^n, S, 0^s, 0^d \rangle \in A$ that there exists an $h \in R_s$ with the property that for all $X \in \hat{\mathcal{X}}_h$,

$$|X| \leq (1 + 1/d^3)(\tfrac{d-1}{d}\tilde{a}) < (1 - \tfrac{1}{d+1})|R_m(S)|.$$

Hence, h is a $1/(d+1)$-good hypothesis for $R_m(S)$.

On the other hand, under the assumption $\langle 0^m, 0^n, S, 0^s, 0^d \rangle \notin A$ we can use Lemma 4 and the inequality $\tilde{a} \geq |R_m(S)|$ to conclude that for every $h \in R_s$ there exists some $X \in \hat{\mathcal{X}}_h$ with $|X| > (1 - 1/d)|R_m(S)|$, implying that $|X| < |R_m(S)|/d$. This means that no $h \in R_s$ is $1/d$-good for $R_m(S)$.

Finally, if $\langle 0^m, 0^n, S, 0^s, 0^d \rangle \in A$ then a suitable representation $h \in R_s$ can be computed by prefix-search using the set A' defined as

$\langle 0^m, 0^n, S, 0^s, 0^d, h' \rangle \in A' \iff h'$ is a prefix of some representation $h \in R_s$ such that there is a hash family $H \in \mathcal{H}(n+k, ml, k)$ that hashes X^l for all sets $X \in \hat{\mathcal{X}}_h$.

This completes the proof of the theorem. $\square$

By using a random sampling argument it is not hard to generate a good hypothesis probabilistically (although this hypothesis might not have a small representation in R). As shown in the next theorem, such a hypothesis can also be deterministically computed with the help of an oracle in Σ_2^p.

Theorem 3 *Let $\mathcal{R}$ be a honest representation class and let $m, n \geq 1$ be integers and S be a sample such that $R_m(S)$ is nonempty. Then there is a sequence $r_1, \ldots, r_{10n}$ of $10n$ representations from $R_m(S)$ which can be computed relative to an oracle in Σ_2^p in time polynomial in m, n and $|S|$ such that the concept* $\mathrm{maj}_{[r_1,\ldots,r_{10n}]}$ *defined as*

$$\mathrm{maj}_{[r_1,\ldots,r_{10n}]} = \{\mathrm{x} \in \Sigma^* : |\{1 \leq \mathrm{i} \leq 10\mathrm{n} : \mathrm{r_i(x)} = 1\}| \geq 5\mathrm{n}\}$$

is $(1/5)$-good on $\Sigma^{\leq n}$ for $R_m(S)$.

Proof. Let $r_1, \ldots, r_{10n}$ be independently and uniformly at random chosen representations in $R_m(S)$, let x be some string in $\Sigma^{\leq n}$, and let p be the probability, that for a uniformly at random chosen representation $r \in R_m(S)$, $r(x) \neq \mathrm{maj}_{\mathrm{R_m(S)}}(\mathrm{x})$, where $\mathrm{maj}_{\mathrm{R_m(S)}}(\mathrm{x})$ denotes the majority vote of all representations in $R_m(S)$ on x.

Note that $\mathrm{maj}_{[r_1,\ldots,r_{10n}]}$ can disagree with more than a 3/4-fraction of representations in $R_m(S)$ on x only if $p < 1/4$, and, in particular, $\mathrm{maj}_{[r_1,\ldots,r_{10n}]}(\mathrm{x}) \neq \mathrm{maj}_{\mathrm{R_m(S)}}(\mathrm{x})$. Since the latter implies that at least $5n$ of the randomly chosen $r_1, \ldots, r_{10n}$ disagree with $\mathrm{maj}_{\mathrm{R_m(S)}}$ on x, and since for $q = 1 - p$, $p/q < 1$, it follows that $\mathrm{maj}_{[r_1,\ldots,r_{10n}]}$ disagrees with more than a 3/4-fraction of representations in $R_m(S)$ on x with probability at most

$$\sum_{i=5n}^{10n} \binom{n}{i} p^i q^{10n-i} = \sum_{i=5n}^{10n} \binom{n}{i} (pq)^{5n} (p/q)^{i-5n} < 2^{10n}(pq)^{5n} = (4pq)^{5n} < (3/4)^{5n}.$$

Hence, with probability less than $2^{n+1}(3/4)^{5n} < 1$, there is some string $x \in \Sigma^{\leq n}$ such that $\mathrm{maj}_{[r_1,\ldots,r_{10n}]}$ disagrees with more than a 3/4-fraction of $R_m(S)$ on x.

It follows that with positive probability, $\mathrm{maj}_{[r_1,\ldots,r_{10n}]}$ is 1/4-good on $\Sigma^{\leq n}$ for $R_m(S)$ which, of course, implies the existence of such a sequence $r_1, \ldots, r_{10n}$. Hence, the computation of a sequence of $10n$ representations $r'_1, \ldots, r'_{10n}$ from $R_m(S)$ such that $\mathrm{maj}_{[r'_1,\ldots,r'_{10n}]}$ is 1/5-good on $\Sigma^{\leq n}$ for $R_m(S)$ can be done as described in the proof of Theorem 2. $\square$

4. Learning with Equivalence Queries

In order to show negative results for learning with equivalence queries, Angluin [2] introduced the notion of approximate fingerprints and showed that concept classes that are polynomial-query learnable with equivalence queries do not have approximate fingerprints.

Definition 2 [2, 6] *A representation class $\mathcal{R}$ does not have* approximate fingerprints *if for all polynomials p_1 and p_2, there exists some polynomial q, such that for all $n > 0$, and all sets of concepts $\mathcal{T} \subseteq \mathcal{C}_{p_1(n)}$ containing at least two concepts there exists a concept in $\mathcal{C}_{q(n)}$ which is $1/q(n)$-good on $\Sigma^{\leq p_2(n)}$ for $\mathcal{T}$. Similarly, the representation class $\mathcal{R}$ does not have* approximate fingerprints for its representations *if the above condition holds when sets T of representations are considered instead of sets $\mathcal{T}$ of concepts.*

It is shown in [6] that the lack of approximate fingerprints is also a sufficient condition for a concept class to be polynomial-query learnable with equivalence queries. We will now use Theorem 2 to give a polynomial-time algorithm that asks queries to an oracle in Σ_2^p. For this we have to take into account that the approximate fingerprints property is formulated in terms of goodness for concepts rather than representations. Thus we first show that the approximate fingerprints property can be transferred to the level of representations.

Lemma 5 *If a representation class $\mathcal{R}$ does not have approximate fingerprints, then $\mathcal{R}$ does not have approximate fingerprints for its representations.*

Proof. Fix two polynomials p_1 and p_2, let q be the polynomial as in the definition of approximate fingerprints, and fix some $n > 0$. W.l.o.g. assume that $q(n) \geq p_1(n)$. Then for any subset of concepts $\mathcal{T} \subseteq \mathcal{C}_{p_1(n)}$, even when $\mathcal{T}$ contains only one concept (or when $\mathcal{T}$ is empty), there is a concept in $\mathcal{C}_{q(n)}$ which is $1/q(n)$-good on $\Sigma^{\leq p_2(n)}$ for $\mathcal{T}$.

Fix some set of representations $T \subseteq R_{p_1(n)}$, and let $\mathcal{T} \subseteq \mathcal{C}_{p_1(n)}$ be the corresponding set of concepts represented by T. Now we can successively find $k = q(n) \log |\mathcal{T}|$ hypotheses $h_1, \ldots, h_k$ such that each h_i is $1/q(n)$-good on $\Sigma^{\leq p_2(n)}$ for the subset of concepts $\mathcal{T}_{i-1}$, where $\mathcal{T}_0 = \mathcal{T}$ and

$$\mathcal{T}_i = \{d \in \mathcal{T} : d(x_1) \neq h_1(x_1) \wedge \ldots \wedge d(x_i) \neq h_i(x_i)\},$$

and where each $x_i \in \Sigma^{\leq p_2(n)}$ is chosen such that

$$|T_i| = \min_{x \in \Sigma^{\leq p_2(n)}} |\{r \in T_{i-1} : r(x) = h_i(x)\}|$$

for the corresponding sets of representations $T_0 = T$ and

$$T_i = \{r \in T : r(x_1) \neq h_1(x_1) \wedge \ldots \wedge r(x_i) \neq h_i(x_i)\}.$$

In other words, whereas each pair (h_i, x), $x \in \Sigma^{\leq p_2(n)}$, eliminates at least a $1/q(n)$-fraction of concepts from $\mathcal{T}_{i-1}$, we choose x_i such that the pair (h_i, x_i) eliminates a minimal number of representations from T_{i-1}.

Since each h_i is $1/q(n)$-good on $\Sigma^{\leq p_2(n)}$ for $\mathcal{T}_{i-1}$, we have that $|\mathcal{T}_k| \leq (1 - 1/q(n))^k |\mathcal{T}| < 1$. Thus, all concepts in $\mathcal{T}_k$ and hence also all representations in T_k are eliminated. Therefore, by the pigeon-hole principle, there is some $i \in \{1, \ldots, k\}$ for which

$$|T_{i-1} - T_i| = |\{r \in T_{i-1} : r(x_i) = h_i(x_i)\}| \geq |T|/k.$$

Hence by the choice of x_i it holds that $|\{r \in T_{i-1} : r(x) = h_i(x)\}| \geq |T|/k$ for all strings $x \in \Sigma^{\leq p_2(n)}$. Since $T_{i-1} \subseteq T$, this means that h_i is $1/k$-good on $\Sigma^{\leq p_2(n)}$ for the set of representations T. □

Theorem 4 *Let $\mathcal{R}$ be a honest representation class. If $\mathcal{R}$ does not have approximate fingerprints then $\mathcal{R}$ is polynomial-time query learnable with equivalence queries, where the learner has additionally access to a Σ_2^p-oracle.*

Proof. By Lemma 5 we may assume that $\mathcal{R}$ does not have approximate fingerprints for its representations. Fix polynomials $p_1(n) = p_2(n) = n$, and let q be the polynomial as in Definition 2. The learning algorithm for $\mathcal{R}$ is a variation of the "halving" algorithm and is given in Figure 1. In addition to the sample S we

$S \leftarrow \emptyset$
$l \leftarrow 0$
$m \leftarrow \min\{m' : R_{m'}(S) \neq \emptyset\}$
$n \leftarrow \max\{1, m\}$
loop
  compute a representation $h \in R_{q(n)}$ that is $1/(q(n)+1)$-good on $\Sigma^{\leq n}$ for $R_n(S)$
  if $EQ(h) =$ "YES" **then**
    return h and halt
  else
    let x be the counterexample to h
    if $|x| \leq n$ **then**
      $S \leftarrow S \cup \{(x, \neg h(x))\}$
    else
      compute the least representation d in $R_n(S)$
      if $EQ(d) =$ "YES" **then**
        return d and halt
      else
        let y be the counterexample to d
        $S \leftarrow S \cup \{(y, \neg d(y))\}$
        $l \leftarrow \max\{|x|, |y|\}$
      endif
    endif
  endif
  $m \leftarrow \min\{m' : R_{m'}(S) \neq \emptyset\}$
  $n \leftarrow \max\{l, m\}$
endloop

Fig. 1. Algorithm with equivalence queries

maintain variables l, m, and n, where l keeps the length of the largest counterexample received thus far, m is always set to the minimal size of concepts in $R(S)$ and $n = \max\{l, m\}$. This means that at the beginning of every iteration, $R_n(S)$ is not empty.

In every iteration the main task is to determine a representation $h \in R_{q(n)}$ that is $1/(q(n)+1)$-good on $\Sigma^{\leq n}$ for $R_n(S)$. By the assumption there is always a representation in $R_{q(n)}$ that is $1/q(n)$-good on $\Sigma^{\leq n}$ for $R_n(S)$. Note here that also in the case when $R_n(S)$ contains only a single representation, this is vacuously true. So, by Theorem 2, h can be computed in time polynomial in n and $|S|$ relative to an oracle in Σ_2^p.

We then ask h as an equivalence query. If the answer is negative and if the length of the counterexample x to h is at most n, we expand S by the pair $(x, \neg h(x))$. Because on all strings of length at most n, h agrees with at least a $1/(q(n)+1)$-fraction of $R_n(S)$, we thereby eliminate at least a $1/(q(n)+1)$-fraction of $R_n(S)$.

If the length of the counterexample to h exceeds n then it is possible that the whole hypothesis space $R_n(S)$ disagrees with h on x. Hence, by adding the pair $(x, \neg h(x))$ to S we might not even eliminate a single concept from $R_n(S)$. Instead we proceed by asking an additional equivalence query with the least representation in $R_n(S)$ as the hypothesis and update l. Clearly, this can be done in time polynomial in n and $|S|$ relative to an oracle in $\mathcal{NP}$.

Next we analyze the algorithm. During any iteration of the loop at least one of the following three conditions is fulfilled.

(i) The algorithm terminates.
(ii) At least a $1/(q(n)+1)$-fraction of the current $R_n(S)$ is eliminated.
(iii) The variable n is increased.

Hence, if n is not increased, then after at most polynomial in n additional iterations, either the algorithm terminates, or n is increased because $R_n(S)$ has become empty. It follows that in every stage in the run of the algorithm, the number of preceding iterations is polynomially bounded in the current value of n. Because we never eliminate the target concept, the value of m is always bounded by the size $|c|$ of the target concept c. Consequently, the value of n is always bounded by the maximum of $|c|$ and the length l of the longest counterexample received thus far. Since in every iteration we ask at most two queries of size at most polynomial in n, it follows that the algorithm has polynomial query-complexity.

Furthermore, the additional equivalence queries (with hypothesis d) guarantee that as long as m is smaller than $|c|$, $R_m(S)$ gets empty, and that the target concept c will be eventually queried. Hence, the algorithm exactly identifies $\mathcal{R}$.

Finally, during every iteration we add at most one counterexample to the current sample S, and every iteration can be done in time polynomial in the current values of n and $|S|$ with some oracle in Σ_2^p. Hence, it follows that the time used by the algorithm is polynomially bounded in $|c|+l$. Thus, the algorithm has polynomial-time complexity, implying that $\mathcal{R}$ is polynomial-time query learnable relative to some oracle in Σ_2^p. □

By combining Theorem 4 with the results in [2, 6] we obtain the following corollary.

Corollary 1 *Let $\mathcal{R}$ be a honest representation class. Then the following are equivalent.*

(i) $\mathcal{R}$ does not have approximate fingerprints.
(ii) $\mathcal{R}$ does not have approximate fingerprints for its representations.
(iii) $\mathcal{R}$ is polynomial-query learnable with equivalence queries.
(iv) $\mathcal{R}$ is polynomial-time query learnable with equivalence queries, where the learner has additionally access to a Σ_2^p-oracle.

It is easy to see (using the fact that the majority of a linear number of randomly chosen concepts is 1/4-good with high probability [6]) that boolean circuits do not have approximate fingerprints. Thus the Σ_3^p-oracle used to learn boolean circuits in [6] can be replaced by an oracle in Σ_2^p.

Corollary 2 *Boolean circuits are polynomial-time query learnable with equivalence queries, where the learner has additionally access to a Σ_2^p-oracle.*

5. Learning with Membership and Equivalence Queries

Hellerstein et al. [10] introduced the notion of polynomial certificates and showed that a representation class is polynomial-query learnable with membership and equivalence queries if and only if it has polynomial certificates. For the following definition it is convenient to say that two concepts $c, c' \subseteq \Sigma^*$ are *consistent* over a finite set $X \subseteq \Sigma^*$ if for all $x \in X$, $c(x) = c'(x)$.

Definition 3 [10] *A representation class $\mathcal{R}$ has* polynomial certificates *if there exist two two-variable polynomials p and q, such that for all $m, l > 0$, and for all $f \subseteq \Sigma^*$ with the property that no concept in $\mathcal{C}_{p(m,l)}$ is consistent with f over $\Sigma^{\leq l}$, there exists a subset $Q \subseteq \Sigma^{\leq l}$ with $|Q| \leq q(m,l)$ such that no concept in $\mathcal{C}_m$ is consistent with f over Q.*

The set Q is called a *certificate set* for f since it certifies that the size of f exceeds m.

Using Stockmeyer's result (Theorem 1), Hellerstein et al. give a learning algorithm that runs in polynomial time and asks queries to an oracle in Σ_4^p for any honest representation class with polynomial certificates. Using Theorem 2 we can show that a variant of the algorithm of Hellerstein et al. can be implemented with an oracle in Σ_2^p.

Theorem 5 *Let $\mathcal{R}$ be a honest representation class. If $\mathcal{R}$ has polynomial certificates then $\mathcal{R}$ is polynomial-time query learnable with equivalence and membership queries, where the learner has additionally access to a Σ_2^p-oracle.*

Proof. The algorithm for $\mathcal{R}$ is given in Figure 2. As in Theorem 4 we maintain variables l and m to keep the length l of the largest counterexample received thus far, and the minimal size m of concepts in $R(S)$. In every iteration we first compute a sequence $r_1, \ldots, r_{10l}$ of representations in $R_m(S)$ such that $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10l}]}$ is 1/5-good on $\Sigma^{\leq l}$ for $\mathcal{R}_m(S)$. By Theorem 3 this can be done in time polynomial in m, l and $|S|$. We then distinguish two cases. If there is a concept in $\mathcal{C}_{p(m,l)}$ consistent with $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10l}]}$ over $\Sigma^{\leq l}$, we proceed similar as in Theorem 4. That is, we determine a representation h in $R_m(S)$ for $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10l}]}$, ask h as an equivalence query. If the answer is negative and if the length of the counterexample x is at most l, then by adding the counterexample to S we can eliminate at least a 1/5-fraction of $R_m(S)$. If the length of x is larger than l we ask an additional equivalence query with the least representation in $R_m(S)$ as the hypothesis and update l.

If no concept in $\mathcal{C}_{p(m,l)}$ is consistent with $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10l}]}$ over $\Sigma^{\leq l}$, then the assumption guarantees that there exists a certificate set $Q \subseteq \Sigma^{\leq l}$ for $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10l}]}$ of cardinality at most $q(m,l)$. That is, no concept in $\mathcal{C}_m$ is consistent with $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10l}]}$ over Q. For a given (encoding of a) set $Q \subseteq \Sigma^{\leq l}$ of cardinality at most $q(m,l)$, the question whether no representation in $R_m(S)$ is consistent with $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{6l}]}$ over Q, is in $co\mathcal{NP}$. Thus, we can compute the certificate set Q by a simple prefix-search in time polynomial in m, l, and $|S|$ relative to an oracle in Σ_2^p. Then we ask all strings in Q as membership queries and include the strings together with the answers into S, again eliminating at least a 1/5-fraction of $R_{m,n}(S)$.

In any iteration (exactly) one of the following holds.

$S \leftarrow \emptyset$
$l \leftarrow 0$
$m \leftarrow \min\{m' : R_{m'}(S) \neq \emptyset\}$
loop
    compute a sequence $r_1, \ldots, r_{10l}$ of representations in $\mathcal{R}_m(S)$ such
    that $\mathrm{maj}_{[r_1,\ldots,r_{10l}]}$ is 1/5-good on $\Sigma^{\leq l}$ for $\mathcal{R}_m(S)$
    if there is a concept in $\mathcal{C}_{p(m,l)}$ consistent with $\mathrm{maj}_{[r_1,\ldots,r_{10l}]}$ over $\Sigma^{\leq l}$ **then**
        $h \leftarrow$ representation of $\mathrm{maj}_{[r_1,\ldots,r_{10l}]}$
        if EQ(h)="YES" **then**
            return h and halt
        else
            let x be the counterexample to h
            if $|x| \leq l$ **then**
                $S \leftarrow S \cup \{(x, \neg h(x))\}$
            else
                compute the least representation d in $R_m(S)$
                if $EQ(d) =$ "YES" **then**
                    return d and halt
                else
                    let y be the counterexample to d
                    $S \leftarrow S \cup \{(y, \neg d(y))\}$
                    $l \leftarrow \max\{|x|, |y|\}$
                endif
            endif
        endif
    else
        compute a certificate set $Q \subseteq \Sigma^{\leq l}$ for $\mathrm{maj}_{[r_1,\ldots,r_{10l}]}$ with $|Q| \leq q(m,l)$
        $S \leftarrow S \cup \{(x, MQ(x)) : x \in Q\}$
    endif
    $m \leftarrow \max\{m' : \mathcal{R}_{m'}(S) \neq \emptyset\}$
endloop

Fig. 2. Algorithm with equivalence and membership queries

(i) The algorithm terminates.
(ii) At least a 1/5-fraction of the current $R_m(S)$ is eliminated.
(iii) At least one concept represented in $\mathcal{R}_m(S)$ is eliminated, and l is increased.

So after a polynomial (in $m + l$) number of iterations either the algorithm terminates, or the current hypothesis space $R_m(S)$ becomes empty, and m is increased. Because we never eliminate the target concept c, it follows that the algorithm halts and outputs a representation for c after at most polynomially (in $|c| + l$) many iterations. Since in every iteration we ask at most two equivalence queries or $q(m, l)$ membership queries, this shows that the algorithm identifies $\mathcal{R}$ with polynomial query-complexity.

Moreover, every iteration can be done in time polynomial in m, l, and $|S|$ relative to an oracle in Σ_2^p, and in every iteration we add at most polynomially (in $m + l$) many examples to S. Thus, the algorithm has polynomial-time complexity under an oracle in Σ_2^p. This completes the proof. □

Combining Theorem 5 with the results in [10] we obtain the following corollary.

Corollary 3 *Let $\mathcal{R}$ be a honest representation class. Then the following are equivalent.*

(i) $\mathcal{R}$ has polynomial certificates.
(ii) $\mathcal{R}$ is polynomial-query learnable with equivalence and membership queries.
(iii) $\mathcal{R}$ is polynomial-time query learnable with equivalence and membership queries, where the learner has additionally access to a Σ_2^p-oracle.

6. Learning with Membership Queries

In [9, 10] learnability with polynomially many membership queries is characterized in terms of specifying sets.

Definition 4 [9, 10] *A representation class $\mathcal{R}$ has* polynomial-size specifying sets *if there exists some polynomial q such that for all sufficiently large m and for all concepts $f \subseteq \Sigma^*$, there exists some set $Q \subseteq \Sigma^{\leq q(m)}$ with $|Q| \leq q(m)$ such that at most one concept in $\mathcal{C}_m$ is consistent with f over Q.*
The set Q is called a *specifying set* for f (with respect to $\mathcal{C}_m$).

Theorem 6 *Let $\mathcal{R}$ be a honest representation class. If $\mathcal{R}$ has polynomial-size specifying sets then $\mathcal{R}$ is polynomial-time query learnable with membership queries, where the learner has additionally access to a Σ_2^p-oracle.*

Proof. Let q be the polynomial as in the definition of specifying sets. The algorithm for $\mathcal{R}$ is given in Figure 3. Besides the sample set S, we only maintain the minimal size m of concepts in $R(S)$. In every iteration we first compute a sequence $r_1, \ldots, r_{10m}$ of representations in $R_m(S)$ such that $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10\mathrm{m}}]}$ is 1/5-good on $\Sigma^{\leq m}$ for $\mathcal{R}_m(S)$. By Theorem 3 this can be done in time polynomial in m and $|S|$ relative to some oracle in Σ_2^p. We then determine a specifying set $Q \subseteq \Sigma^{\leq m}$ for $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10\mathrm{m}}]}$ with respect to $\mathcal{C}_m$ of cardinality at most $q(m)$. Then we ask all strings in Q as membership queries. If $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10\mathrm{m}}]}$ agrees with the answers on all strings $x \in Q$, then it follows by the assumption that $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10\mathrm{m}}]}$ is the unique concept left in $\mathcal{C}_m(S)$. So we determine a representation h for $\mathrm{maj}_{[\mathrm{r}_1,\ldots,\mathrm{r}_{10\mathrm{m}}]}$,

$S \leftarrow \emptyset$
$m \leftarrow \min\{m' : R_{m'}(S) \neq \emptyset\}$
loop
  compute a sequence $r_1, \ldots, r_{10m}$ of representations in $R_m(S)$ such that $\mathrm{maj}_{[r_1,\ldots,r_{10m}]}$ is 1/5-good on $\Sigma^{\leq m}$ for $R_m(S)$
  compute a specifying set $Q \subseteq \Sigma^{\leq q(m)}$ for $\mathrm{maj}_{[r_1,\ldots,r_{10m}]}$ with respect to $\mathcal{C}_m$ and with $|Q| \leq q(m)$
  if $(\forall x \in Q) : \mathrm{maj}_{[r_1,\ldots,r_{10m}]}(x) = \mathrm{MQ}(x)$ **then**
    compute a representation h in $R_m(S)$ consistent with $\mathrm{maj}_{[r_1,\ldots,r_{10m}]}$ over Q
    output h and halt
  else
    let x be a string in Q such that $\mathrm{maj}_{[r_1,\ldots,r_{10m}]}(x) \neq \mathrm{MQ}(x)$
    $S \leftarrow S \cup \{(x, MQ(x))\}$
  endif
  $m \leftarrow \min\{m' : R_{m'}(S) \neq \emptyset\}$
endloop

Fig. 3. Algorithm with membership queries

output h and halt. Otherwise there is some string $x \in Q$ on which the answers on x and $\mathrm{maj}_{[r_1,\ldots,r_{10m}]}(x)$ disagree. Adding x together with the answer to S therefore eliminates at least a 1/5-fraction of $R_m(S)$.

Clearly, in every iteration either the algorithm outputs a representation of the target concept and terminates, or at least a 1/5-fraction of $R_m(S)$ is eliminated. It follows that the algorithm identifies the target concept in at most polynomially (in $|c|$) many iterations. Since in every iteration we ask at most polynomially many membership queries, the algorithm exactly identifies $\mathcal{R}$ with polynomial query-complexity.

It remains to show that a specifying set Q for $\mathrm{maj}_{[r_1,\ldots,r_{10m}]}$ with respect to $\mathcal{C}_m$ can be computed in time polynomial in m relative to some oracle in Σ_2^p. First we show that a set $\hat{Q} \subseteq \Sigma^{\leq q(m)}$ is a specifying set for some $f \subseteq \Sigma^*$ with respect to $\mathcal{C}_m$ if and only if at most one concept in $\mathcal{C}_m^{\leq q(m)} = \{c \cap \Sigma^{\leq q(m)} : c \in \mathcal{C}_m\}$ is consistent with f over Q. Clearly, if at most one concept in $\mathcal{C}_m$ is consistent with f over $\hat{Q}$, then at most one concept in $\mathcal{C}_m^{\leq q(m)}$ is consistent with f over $\hat{Q}$. For the reverse implication, assume that a unique concept in $\mathcal{C}_m^{\leq q(m)}$ is consistent with f over $\hat{Q}$, and let c_1 and c_2 be concepts in $\mathcal{C}_m$ such that $c_1 \cap \Sigma^{\leq q(m)} = c_2 \cap \Sigma^{\leq q(m)}$ and both c_1 and c_2 are consistent with f over $\hat{Q}$. By the assumption, also for c_1 there is a specifying set $Q_1 \subseteq \Sigma^{\leq q(m)}$. The concepts c_1 and c_2 are consistent over $\Sigma^{\leq q(m)}$. Thus, in particular, c_1 and c_2 are consistent over Q_1. Because both c_1 and c_2 are in $\mathcal{C}_m$, they must be equal. So c_1 is also the unique concept in $\mathcal{C}_m$ consistent with f over $\hat{Q}$. It follows that $\hat{Q}$ is a specifying set for f if and only if it is not the case that there are representations $r, s \in R_m$, and a string $x \in \Sigma^{\leq q(m)}$ such that $r(x) \neq s(x)$, but both r and s agree with f on all strings of $\hat{Q}$. Hence, for a given (encoding) of the set $\hat{Q}$, the question whether $\hat{Q}$ is a specifying set for $\mathrm{maj}_{[r_1,\ldots,r_{10m}]}$ is in $co\mathcal{NP}$. It follows that the set Q can be computed by prefix-search in polynomial time with the help of an oracle in Σ_2^p. This completes the proof. □

By using the results in [9, 10] we obtain the following corollary.

Corollary 4 *Let $\mathcal{R}$ be a honest representation class. Then the following are equivalent.*

(i) $\mathcal{R}$ has polynomial-size specifying sets.

(ii) $\mathcal{R}$ is polynomial-query learnable with membership queries.

(iii) $\mathcal{R}$ is polynomial-time query learnable with membership queries, where the learner has additionally access to a Σ_2^p-oracle.

Finally, let us mention that it would be very interesting to prove a kind of lower bound for the oracle complexity needed for polynomial-time learning in Angluin's exact learning model (under some reasonable assumption as, for example, that the polynomial hierarchy does not collapse).

Acknowledgements

We thank an anonymous referee for helpful comments.

References

1. D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.
2. D. Angluin. Negative results for equivalence queries. *Machine Learning*, 5(2):121–150, 1990.
3. J. L. Balcázar, J. Díaz, and J. Gabarró. *Structural Complexity I.* EATCS Monographs on Theoretical Computer Science. Springer-Verlag, second edition, 1995.
4. A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information Processing Letters*, 24(6):377–380, 1987.
5. A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.
6. N. Bshouty, R. Cleve, R. Gavaldà, S. Kannan, and C. Tamon. Oracles and queries that are sufficient for exact learning. *Journal of Computer and System Sciences*, 52:421–433, 1996.
7. J. L. Carter and M. N. Wegman. Universal classes of hash functions. *Journal of Computer and System Sciences*, 18:143–154, 1979.
8. R. Gavaldà. Bounding the complexity of advice functions. *Journal of Computer and System Sciences*, 50(3):468–475, 1995.
9. T. Hegedüs. Generalized teaching dimensions and the query complexity of learning. In *Proceedings of the 8th Annual Conference on Computational Learning Theory (COLT'95)*, pages 108–117. ACM Press, July 1995.
10. L. Hellerstein, K. Pillaipakkamnatt, V. Raghavan, and D. Wilkins. How many queries are needed to learn? *Journal of the ACM*, 43(5):840–862, 1996.
11. J. Köbler. Lowness-Eigenschaften und Erlernbarkeit von Booleschen Schaltkreisklassen. Habilitationsschrift, Universität Ulm, 1995.
12. J. Köbler and W. Lindner. Oracles in Σ_2^p are sufficient for exact learning. In *Proc. 8th International Workshop on Algorithmic Learning Theory (ALT'97)*, volume 1316 of *Lecture Notes in Artificial Intelligence*, pages 277–290. Springer-Verlag, 1997.
13. J. Köbler and O. Watanabe. New collapse consequences of NP having small circuits. *SIAM Journal on Computing*, 28(1):311–324, 1999.

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

15. M. Sipser. A complexity theoretic approach to randomness. In *Proc. 15th ACM Symposium on Theory of Computing*, pages 330–335. ACM Press, 1983.

16. L. Stockmeyer. On approximation algorithms for #P. *SIAM Journal on Computing*, 14(4):849–861, 1985.

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