On Polynomial time Constructions of Minimum Height Decision Tree

Waseem Makhoul
On Polynomial time Constructions of Minimum Height Decision Tree

Research Thesis

Submitted in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

Waseem Makhoul

Submitted to the Senate of the Technion — Israel Institute of Technology
Shevat 5778 Haifa February 2018
This research was carried out under the supervision of Prof. Nader Bshouty, in the Faculty of Computer Science.

Some results in this thesis are planned to be published as articles by the author and research collaborators in the leading conference of the field.

Acknowledgements

First of all, I would like to express my gratitude to my advisor, Prof. Nader Bshouty, for his generous assistance, support and guidance throughout this research.

I would also like to thank my parents, my sisters and my brother, for their devoted support, encouragement, concern and invaluable love.

Last but not least, I would like to thank my friends at the department for making the past couple of years an unforgettable period.

The generous financial help of the Technion is gratefully acknowledged.
Contents

List of Figures

Abstract 1

Abbreviations and Notations 3

1 Introduction 7
   1.1 Introduction to the Field and the Model .......................... 7
   1.2 Motivation and the Definition of the Problem ....................... 8
   1.3 Previous and New Results ......................................... 9
   1.4 Outline of the Thesis ............................................. 10

2 Definitions and Preliminary Results 11
   2.1 Notation ............................................................. 11
   2.2 Optimal Algorithm ............................................... 11
   2.3 Extended Teaching Dimension .................................... 12
   2.4 Hitting Set ....................................................... 13
   2.5 Density of a Set ................................................ 13

3 Bounds for OPT 15
   3.1 Lower Bound ..................................................... 15
   3.2 Upper Bounds .................................................... 17

4 Polynomial Time Approximation Algorithm 21

5 Applications to Disjunctions of Predicates 25
   5.1 An Equivalence Relation Over $\mathcal{F}_\lor$ ...................... 25
   5.2 A Partial Order Over $\mathcal{F}_\lor$ and Hasse Diagram .......... 26
   5.3 Witnesses .......................................................... 26
   5.4 The Extended Teaching Dimension of $\mathcal{F}_\lor$ ............... 27
      5.4.1 Teaching Dimension ........................................ 28
      5.4.2 The Value of $\text{ETD}(\mathcal{F}_\lor)$ and the Proof .......... 28
   5.5 Learning Other Classes ......................................... 29
   5.6 Example of Classes .............................................. 29
# List of Figures

3.1 An algorithm that finds the hidden column $a \in A$ .......................... 20

5.1 Hasse diagram of $\text{Ray}_4^2$. The functions are $f_i(x_1, x_2) = [x_1 \geq i]$ and $g_i(x_1, x_2) = [x_2 \geq i]$. .......................................................... 30

5.2 Hasse diagram of terms and monotone terms. ................................. 30

5.3 Hasse diagram when $\mathcal{F} = \{f_1, f_2, f_3, g_1, g_2, g_3, h_1, \ldots, h_5\}$ of functions $\{1, 2, 3\} \times \{1, 2, 3\} \rightarrow \{0, 1\}$ where $f_i(x_1, y_1) = [x_1 \geq i]$, $g_i(x_1, x_2) = [x_2 \geq i]$ and $h_i(x_1, x_2) = [x_1 + x_2 \geq i + 1]$. .................................................. 31
Abstract

A decision tree $T$ in $B_m := \{0, 1\}^m$ is a binary tree where each of its internal nodes is labeled with an integer in $[m] = \{1, 2, \ldots, m\}$, each leaf is labeled with an assignment $a \in B_m$ and each internal node has two outgoing edges that are labeled with 0 and 1, respectively. Let $A \subset \{0, 1\}^m$. We say that $T$ is a decision tree for $A$ if (1) For every $a \in A$ there is one leaf of $T$ that is labeled with $a$. (2) For every path from the root to a leaf with internal nodes labeled with $i_1, i_2, \ldots, i_k \in [m]$, a leaf labeled with $a \in A$ and edges labeled with $\xi_{i_1}, \ldots, \xi_{i_k} \in \{0, 1\}$, $a$ is the only element in $A$ that satisfies $a_{i_j} = \xi_{i_j}$ for all $j = 1, \ldots, k$.

Our goal is to write a polynomial time (in $n := |A|$ and $m$) algorithm that for an input $A \subseteq B_m$ outputs a decision tree for $A$ of minimum depth. This problem has many applications that include, to name a few, computer vision, group testing, exact learning from membership queries and game theory.

Arkin et al. and Moshkov gave a polynomial time $(\ln |A|)$-approximation algorithm (for the depth). The result of Dinur and Steurer for set cover implies that this problem cannot be approximated with ratio $(1 - o(1)) \cdot \ln |A|$, unless $P=NP$. Moskov studied in the combinatorial measure of extended teaching dimension of $A$, $ETD(A)$. He showed that $ETD(A)$ is a lower bound for the depth of the decision tree for $A$ and then gave an exponential time $ETD(A)/\log(ETD(A))$-approximation algorithm.

In this work we further study the $ETD(A)$ measure and a new combinatorial measure, $DEN(A)$, that we call the density of the set $A$. We show that $DEN(A) \leq ETD(A) + 1$. We then give two results. The first result is that the lower bound $ETD(A)$ of Moskov for the depth of the decision tree for $A$ is greater than the bounds that are obtained by the classical technique used in the literature. The second result is a polynomial time $(\ln 2)DEN(A)$-approximation (and therefore $(\ln 2)ETD(A)$-approximation) algorithm for the depth of the decision tree of $A$. We also show that a better approximation ratio implies $P=NP$.

We then apply the above results to learning the class of disjunctions of predicates from membership queries. We show that the ETD of this class is bounded from above by the degree $d$ of its Hasse diagram. We then show that Moshkov algorithm can be run in polynomial time and is $(d/\log d)$-approximation algorithm. This gives optimal algorithms when the degree is constant. For example, learning axis parallel rays over constant dimension space.
# Abbreviations and Notations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T$</td>
<td>binary tree</td>
</tr>
<tr>
<td>AMMRS algorithm</td>
<td>the algorithms of Arkin et al.</td>
</tr>
<tr>
<td>w.l.o.g.</td>
<td>without loss of generality</td>
</tr>
<tr>
<td>log</td>
<td>logarithm base 2</td>
</tr>
<tr>
<td>ln</td>
<td>natural logarithm</td>
</tr>
<tr>
<td>$[m]$</td>
<td>${1, 2, \ldots, m}$</td>
</tr>
<tr>
<td>${0, 1}^m$</td>
<td>the space of all binary columns of size $m$</td>
</tr>
<tr>
<td>$B_m$</td>
<td>the space of all binary columns of size $m$ (${0, 1}^m$)</td>
</tr>
<tr>
<td>$A$</td>
<td>a subset of $B_m$ (a binary matrix)</td>
</tr>
<tr>
<td>$n$</td>
<td>the size of $A$ (the number of columns in $A$)</td>
</tr>
<tr>
<td>$m$</td>
<td>the size of each element in $B_m$</td>
</tr>
<tr>
<td>$OPT(A)$</td>
<td>the minimum height of a decision tree for $A$</td>
</tr>
<tr>
<td>$</td>
<td>A</td>
</tr>
<tr>
<td>$a$</td>
<td>a hidden element in $A$</td>
</tr>
<tr>
<td>$a_i$</td>
<td>the entry $i$ of $A$</td>
</tr>
</tbody>
</table>
\(a^{(j)}\) 
the \(j\)-th element in \(A\)

\(h\) 
an element in \(B_m\)

\(a + h\) 
a bitwise exclusive

\(A + h\) 
\(\{a + h|a \in A\}\)

\(\text{ETD}(A,h)\) 
the minimum size of a specifying set for \(h\)

\(\text{ETD}_z(A)\) 
\(\text{ETD}(A,0)\)

\(\text{ETD}(A)\) 
the extended teaching dimension of \(A\)

\(\text{SETD}(A,h)\) 
the minimum size of a strong specifying set for \(h\)

\(\text{SETD}_z(A)\) 
\(\text{SETD}(A,0)\)

\(\text{SETD}(A)\) 
the strong extended teaching dimension of \(A\)

\(\text{HS}(A)\) 
a hitting set for \(A\)

\(\text{MAJ}(A)\) 
a binary column (in \(B_m\)) which contains the majority

of each entry in \(A\) (0 or 1)

\(\text{MAX}(A)\) 
the maximum number of ones among all entries \(i \in [m]\)

of \(A\)
\( \xi \)  

a binary digit \((\in \{0,1\})\)

\( A_{j,\xi} \)  

the elements in \( A \) for which the entry \( j \) equals \( \xi \)

\( \text{MAMI}(A) \)  

\( \max_j \min(|A_{j,0}|,|A_{j,1}|) \)

\( C \)  

a class which we want to learn

\( \mathcal{F} \)  

a set of boolean functions

\( \lor \)  

the logical AND operation

\( \mathcal{F}_\lor \)  

\( \{\lor_{f \in S} f | S \subseteq \mathcal{F}\} \)

\( F \)  

an element (a function) in \( \mathcal{F}_\lor \)

\( F_1 \equiv F_2 \)  

\( F_1 \) and \( F_2 \) are identical

\( (F_1 = F_2) \)  

\( F_1 \) is logically equal to \( F_2 \)

\( [F] \)  

the equivalence class of \( F \) over (=)

\( \mathcal{F}_\lor^* \)  

the set of equivalence classes of (=)

\( G_F \)  

the representative element of \( F \) in \([F]\)

\( G(\mathcal{F}_\lor) \)  

the set of all representative elements in \( \mathcal{F}_\lor^* \)

\( F_1 \Rightarrow F_2 \)  

\( F_1 \) logically implies \( F_2 \)

\( \bar{G} \)  

the logical negation (complement) function of \( G \)
\( H(\mathcal{F}_v) \) the Hasse diagram of \( G(\mathcal{F}_v) \) for the partial order \( \Rightarrow \)

\( G_{\text{max}} \) the maximum (top) element in \( H(\mathcal{F}_v) \)

\( G_{\text{min}} \) the minimum (bottom) element in \( H(\mathcal{F}_v) \)

\( \text{De}(G) \) the set of all the immediate descendants of \( G \) in \( H(\mathcal{F}_v) \)

\( \text{As}(G) \) the set of all the immediate ascendants of \( G \) in \( H(\mathcal{F}_v) \)

\( \cup \) set union

\( \text{Ne}(G) \) the set of neighbors of \( G \) in \( H(\mathcal{F}_v) \)

\( \text{deg}(G) \) the degree of \( G \)

\( \text{deg}(\mathcal{F}_v) \) the degree of \( \mathcal{F}_v \)

\( \text{lca}(G_1, G_2) \) the lowest common ascendant of \( G_1 \) and \( G_2 \) in \( H(\mathcal{F}_v) \)

\( \text{gcd}(G_1, G_2) \) the greatest common descendant of \( G_1 \) and \( G_2 \) in \( H(\mathcal{F}_v) \)

\( \cap \) set intersection

\( \land \) the logical AND operation

\( \text{As}(G) \land \bar{G} \) \( \{ s \land \bar{G} \mid s \in \text{As}(G) \} \)

\( \text{TD}(C, G) \) the witness size of \( G \) in \( C \)

\( \text{TD}(C) \) the teaching dimension \( C \)
Chapter 1

Introduction

In this chapter we introduce our field of study and the model with which we work. We also provide motivation to our specific studied problem, define it, and provide previous results as well as our new results. Finally, we give an outline for the thesis.

1.1 Introduction to the Field and the Model

Learning is a process that appears in numerous fields in life. In general, it can be described as the process of acquiring new, or modifying and reinforcing existing understanding, knowledge, behaviors, skills, attitudes, values or preferences through studying and experience. It is the ability to predict the unknown based on gained information from past experiences. The ability to learn is possessed by humans, animals, plants and some machines. Machine Learning is the subfield of computer science that gives computers the ability to learn without being explicitly programmed. Computational Learning Theory is a subfield of Artificial Intelligence devoted to studying the design and analysis of machine learning algorithms. It considers the learnability of concept classes and considers their learning algorithms. The focus in Computational Learning Theory is on rigorous mathematical analysis.

The learning process in Computational Learning can be performed using various learning models. Each model has its own rules and targets for learning. Yet, all models share common properties. In all models we have a teacher and a learner. The teacher holds a hidden target function from a known given class of functions. In the learning process the learner constructs queries and presents them as input to the teacher. The teacher then outputs answers to the queries. Sometimes we call the teacher as "adversary", since he usually forces the learner to ask a large number of queries. The purpose of the learner is to ask a minimal number of queries. Sometimes, it is not practical for the learner to ask minimal number of queries. Hence, in many cases the learner searches for an approximation for the minimal number of queries to be asked.
1.2 Motivation and the Definition of the Problem

Consider the following problem: Given an \( n \)-element set \( A \subseteq B_m := \{0,1\}^m \) from some class of sets \( \mathcal{A} \) and a hidden element \( a \in A \). Given an oracle (adversary) that answers queries of the type: “What is the value of \( a_i \)?”. Find a polynomial time algorithm that with an input \( A \), asks minimum number of queries to the oracle and finds the hidden element \( a \). This is equivalent to constructing a minimum height decision tree for \( A \). A decision tree is a binary tree where each internal node is labeled with an index from \([m]\) and each leaf is labeled with an assignment \( a \in B_m \). Each internal node has two outgoing edges one that is labeled with 0 and the other is labeled with 1. A node that is labeled with \( i \) corresponds to the query “Is \( a_i = 0 \)?”. An edge that is labeled with \( \xi \) corresponds to the answer \( \xi \). This decision tree is an algorithm in an obvious way and its height is the worst case complexity of the number of queries. A decision tree \( T \) is said to be a decision tree for \( A \) if the algorithm that corresponds to \( T \) predicts correctly the hidden assignment \( a \in A \). Our goal is to construct a small height decision tree for \( A \subseteq B_m \) in time polynomial in \( m \) and \( n := |A| \). We will denote by \( \text{OPT}(A) \) the minimum height decision tree for \( A \).

In computer vision the problem is related to minimizing the number of “probes” (queries) needed to determine which one of a finite set of geometric figures is present in an image [AMM+98]. In game theory the problem is related to the minimum number of turns required in order to win a guessing game.

This problem is also related to the following problem in exact learning [Aug88]: Given a class \( C \) of boolean functions \( f : X \rightarrow \{0,1\} \). Construct in \( \text{poly}(|C|,|X|) \) time an optimal adaptive algorithm that learns \( C \) from membership queries. This learning problem is equivalent to constructing a minimum height decision tree for the set \( A = \{a^{(i)}|a_j^{(i)} = f_i(x_j)\} \) where \( f_i \) is the \( i \)th function in \( C \) and \( x_j \) is the \( j \)th instance in \( X \).

This problem is also related to the following problem in learning the class of disjunctions of predicates from a set of predicates \( \mathcal{F} \) from membership queries of Bshouty et al. [BDVY17]. Given a set of boolean functions \( \mathcal{F} \), Bshouty et al. present an algorithm for learning \( \mathcal{F}_\wedge := \{\bigvee_{f \in S} f \mid S \subseteq \mathcal{F}\} \) from membership queries. The problem that Bshouty et al. address has practical importance in the field of program synthesis, where the goal is to synthesize a program that meets some requirements. Program synthesis has become popular especially in settings aiming to help end users. In such settings, the requirements are not provided upfront and the synthesizer can only learn them by posing membership queries to the end user. Their work enables such synthesizers to learn the exact requirements while bounding the number of membership queries.
1.3 Previous and New Results

In [AMM+98], Arkin et al. showed that (AMMRS-algorithm) if at every node the decision tree chooses i that partitions the current set (the set of assignments that are consistent to the answers of the queries so far) as evenly as possible, then the height of the tree is within a factor of log |A| from optimal. I.e., log |A|-approximation algorithm. Moshkov [Mos04] analysis shows that this algorithm is (ln |A|)-approximation algorithm. This algorithm runs in polynomial time in \( m \) and |A|.

Hyafil and Rivest, [HR76], show that the problem of constructing a minimum depth decision tree is NP-Hard. The reduction of Laber and Nogueira, [LN04] to set cover with the inapproximability result of Dinur and Steurer [DS14] for set cover implies that it cannot be approximated to a factor of (1 − o(1)) · ln |A| unless P=NP. Therefore, no better approximation ratio can be obtained if no constraint is added to the set A.

Moshkov, [Mos82], studied the extended teaching dimension combinatorial measure, ETD(A), of a set \( A \subseteq B_m \). It is the maximum over all the possible assignments \( b \in B_m \) of the minimum number of indices \( i \in [m] \) in which \( b \) agrees with at most one \( a \in A \). Moshkov showed two results. The first is that ETD(A) is a lower bound for OPT(A).

The second is an exponential time algorithm that asks \((2^\text{ETD}(A))/\log\text{ETD}(A)) \log n\) queries. This gives a (\(\ln 2\)) (\((\ln |A|)/\log\text{ETD}(A)\)) \log n\) -approximation (exponential time) algorithm (since \(\text{OPT}(A) \geq \text{ETD}(A)\)) and at the same time \(2^\text{ETD}(A)/\log\text{ETD}(A)\)-approximation algorithm (since \(\text{OPT}(A) \geq \log |A|\)). Since many interesting classes have small ETD dimension, the latter result gives small approximation ratio but unfortunately Moshkov algorithm runs in exponential time.

In this thesis we further study the ETD measure. We show that any polynomial time \((1 − o(1))\text{ETD}(A)\)-approximation algorithm implies P=NP. Therefore, Moshkov algorithm cannot run in polynomial time unless P=NP. We then show that the above AMMRS-algorithm, [AMM+98], is polynomial time \((\ln 2)\text{ETD}(A)\)-approximation algorithm. This gives a small approximation ratio for classes with small extended teaching dimension.

Another reason for studying the ETD of classes is the following: If you find the ETD of the set \( A \) then you either get a lower bound that is better than the information theoretic lower bound \( \log |A| \) or you get an approximation algorithm with a better ratio than \( \ln |A| \). This is because if \( \text{ETD}(A) < \log |A| \) then the AMMRS-algorithm has a ratio \((\ln 2)\text{ETD}(A)\) that is better than the \( \ln |A| \) ratio and if \( \text{ETD}(A) > \log |A| \) then Moshkov lower bound, \( \text{ETD}(A) \), for \( \text{OPT}(A) \) is better than the information theoretic lower bound \( \log |A| \).

To get the above results, we define a new combinatorial measure called the density \( \text{DEN}(A) \) of the set \( A \). If \( Q = \text{DEN}(A) \) then there is a subset \( B \subseteq A \) such that an adversary can give answers to the queries that eliminate at most \( 1/Q \) fraction of the number of elements in \( B \). This forces the learner to ask at least \( Q \) queries. We then show that \( \text{ETD}(A) \geq \text{DEN}(A) − 1 \). On the other hand, we show that if \( Q = \text{DEN}(A) \)
then a query in the AMMRS-algorithm eliminates at least \((1 - 1/Q)\) fraction of the assignments in \(A\). This gives a polynomial time \((\ln 2)\text{DEN}(A)\)-approximation algorithm which is also a \((\ln 2)(\text{ETD}(A) + 1)\)-approximation algorithm.

In order to compare both algorithms we show that \((\text{ETD}(A) - 1)/\ln |A| \leq \text{DEN}(A) \leq \text{ETD}(A) + 1\) and for random uniform \(A\) (and therefore for almost all \(A\)), with high probability \(\text{DEN}(A) = \Theta(\text{ETD}(A)/\ln |A|)\). Since \(|A| > \text{ETD}(A)\), this shows that AMMRS-algorithm may get a better approximation ratio than Moshkov algorithm.

The inapproximability results follows from the reduction of Laber and Nogueira, \[LN04\] to set cover with the inapproximability result of Dinur and Steurer \[DS14\] and the fact that \(\text{DEN}(A) \leq \text{ETD}(A) + 1 \leq \text{OPT}(A) + 1\).

We then address the class of disjunctions of predicates from a set of predicates \(F\) from membership queries of Bshouty et al. \[BDVY17\]. Given a set of boolean functions \(F\), Bshouty et al. present an algorithm for learning \(F\lor := \{\lor f \in S \mid S \subseteq F\}\) from membership queries. The algorithm they presented, called SPEX algorithm, asks at most \(|F| \cdot \text{OPT}(F\lor)\) membership queries where \(\text{OPT}(F\lor)\) is the minimum worst case number of membership queries for learning \(F\lor\). Bshouty et al. show that given some computational complexity conditions on the set of predicates, \(F\), the algorithm SPEX runs in polynomial time.

We then apply our results to learning the class \(F\lor\) of Bshouty et al. \[BDVY17\]. We show that the ETD of this class is bounded from above by the degree \(d\) of its Hasse diagram. We then show that Moshkov algorithm, for this class, runs in polynomial time and is \((d/\log d)\)-approximation algorithm. Since \(|F| \geq d\) (and in many applications, \(|F| \gg d\)), this improves the \(|F|\)-approximation algorithm SPEX in \[BDVY17\] when the size of Hasse diagram is polynomial. This also gives optimal algorithms when the degree \(d\) is constant. For example, learning axis parallel rays over constant dimension space.

1.4 Outline of the Thesis

The rest of the thesis is organized as follows. In chapter 2 we give some notations, definitions, and preliminary results. In chapter 3 we give upper and lower bounds for \(\text{OPT}\). In chapter 4 we provide our main result, which is a \((\ln 2)\text{ETD}(A)\)-approximation algorithm that runs in a polynomial time. In chapter 5 we apply our results to learning the class of disjunctions of predicates from a set of predicates \(F\) from membership queries \[BDVY17\].

10
Chapter 2

Definitions and Preliminary Results

In this chapter we provide some definitions and preliminary results.

2.1 Notation

Let $B_m = \{0,1\}^m$. Let $A = \{a^{(1)},\ldots,a^{(n)}\} \subseteq B_m$ be an $n$-element set. We will write $|A|$ for the number of elements in $A$. For $h \in B_m$ we define $A + h = \{a + h|a \in A\}$ where $+$ (in the square brackets) is the bitwise exclusive or of elements in $B_m$.

For integer $q$ let $[q] = \{1,2,\ldots,q\}$. Throughout the thesis, $\log x = \log_2 x$.

2.2 Optimal Algorithm

We denote by $\text{OPT}(A)$ the minimum depth of a decision tree for $A$. Our goal is to build a decision tree for $A$ with small depth.

Obviously

$$ \log n \leq \text{OPT}(A) \leq n - 1 $$

(2.1)

where $n := |A|$.

**Lemma 2.2.1.** We have $\text{OPT}(A) = \text{OPT}(A + h)$.

*Proof.* Since $(A + h) + h = A$, it is enough to prove that $\text{OPT}(A + h) \leq \text{OPT}(A)$. Now given a decision tree $T$ for $A$ of depth $\text{OPT}(A)$. For each node, $v$, in $T$ labeled with $j$, such that $h_j = 1$, exchange the labels in their outgoing edges. Then change the label of each leaf labeled with $a$ to $a + h$. It is easy to show that the new tree is a decision tree for $A + h$. 

11
2.3 Extended Teaching Dimension

In this section we define the extended teaching dimension.

Let \( h \in B_m \) be any element. We say that a set \( S \subseteq [m] \) is a specifying set for \( h \) with respect to \( A \) if \( \{|a \in A \mid (\forall i \in S) h_i = a_i\}| \leq 1 \). That is, there is at most one element in \( A \) that is consistent with \( h \) on the entries of \( S \). Denote by \( \text{ETD}(A,h) \) the minimum size of a specifying set for \( h \) with respect to \( A \). The extended teaching dimension of \( A \) is

\[
\text{ETD}(A) = \max_{h \in B_m} \text{ETD}(A,h). \tag{2.2}
\]

We will write \( \text{ETD}(A) \) for \( \text{ETD}(A,0) \). It is easy to see that

\[
\text{ETD}(A,h) = \text{ETD}(A + h) \quad \text{and} \quad \text{ETD}(A) = \text{ETD}(A + h). \tag{2.3}
\]

We say that a set \( S \subseteq [m] \) is a strong specifying set for \( h \) with respect to \( A \) if either \( h \in A \) and \( \{|a \in A \mid (\forall i \in S) h_i = a_i\}| = 1 \), or \( \{|a \in A \mid (\forall i \in S) h_i = a_i\}| = 0 \). That is, if \( h \in A \) then there is exactly one element in \( A \) that is consistent with \( h \) on the entries of \( S \). Otherwise, no element in \( A \) is consistent with \( h \) on \( S \). Denote \( \text{SETD}(A,h) \) the minimum size of a strong specifying set for \( h \) with respect to \( A \). The strong extended teaching dimension of \( A \) is

\[
\text{SETD}(A) = \max_{h \in B_m} \text{SETD}(A,h). \tag{2.4}
\]

We will write \( \text{SETD}(A) \) for \( \text{SETD}(A,0) \). It is easy to see that

\[
\text{SETD}(A,h) = \text{SETD}(A + h) \quad \text{and} \quad \text{SETD}(A) = \text{SETD}(A + h). \tag{2.5}
\]

Obviously, \( \text{ETD}(A,h) \leq \min(m,n-1) \) and \( \text{ETD}(A,h) \leq \text{SETD}(A,h) \leq \min(m,n) \)

We now show

**Lemma 2.3.1.** We have \( \text{ETD}(A,h) \leq \text{SETD}(A,h) \leq \text{ETD}(A,h) + 1 \) and therefore \( \text{ETD}(A) \leq \text{SETD}(A) \leq \text{ETD}(A) + 1 \).

**Proof.** The fact \( \text{ETD}(A,h) \leq \text{SETD}(A,h) \) follows from the definitions. Let \( S \subseteq [m] \) be a specifying set for \( h \) with respect to \( A \). Then for \( T := \{a \in A \mid (\forall i \in S) h_i = a_i\} \) we have \( t := |T| \leq 1 \). If \( t = 0 \) or \( h \in A \) then \( S \) is a strong specifying set for \( h \) with respect to \( A \). If \( t = 1 \) and \( h \notin A \) then for the element \( a \in T \) there is \( j \in [m] \) such that \( a_j \neq h_j \) and then \( S \cup \{j\} \) is a strong specifying set for \( h \) with respect to \( A \). This proves that \( \text{SETD}(A,h) \leq \text{ETD}(A,h) + 1 \).

The other claims follows immediately.

Obviously, for any \( B \subseteq A \)

\[
\text{ETD}(B) \leq \text{ETD}(A), \quad \text{SETD}(B) \leq \text{SETD}(A). \tag{2.6}
\]
2.4 Hitting Set

In this section we define the hitting set for \( A \).

A hitting set for \( A \) is a set \( S \subseteq [m] \) such that for every non-zero element \( a \in A \) there is \( j \in S \) such that \( a_j = 1 \). That is, \( S \) hits every element in \( A \) except the zero element (if it exists). The size of the minimum size hitting set for \( A \) is denoted by \( \text{HS}(A) \).

We now show

**Lemma 2.4.1.** We have \( \text{HS}(A) = \text{SETD}_z(A) \). In particular, \( \text{SETD}(A, h) = \text{HS}(A + h) \) and \( \text{SETD}(A) = \max_{h \in B_m} \text{HS}(A + h) \).

*Proof.* If \( 0 \in A \) then \( \text{SETD}_z(A) \) is the minimum size of a set \( S \) such that \( \{ a \in A \mid (\forall i \in S) a_i = 0 \} = \{0\} \) and if \( 0 \notin A \) then it is the minimum size of a set \( S \) such that \( \{ a \in A \mid (\forall i \in S) a_i = 0 \} = \emptyset \). Therefore the set \( S \) hits all the nonzero elements in \( A \).

The other results follow from (2.5) and the definition of SETD.

2.5 Density of a Set

In this section we define our new measure \( \text{DEN} \) of a set.

Let \( A = \{a^{(1)}, \ldots, a^{(n)}\} \subseteq B_m \). We define \( \text{MAJ}(A) \in B_m \) such that \( \text{MAJ}(A)_i = 1 \) if the number of ones in \( (a^{(1)}_i, \ldots, a^{(n)}_i) \) is greater or equal the number of zeros and \( \text{MAJ}(A)_i = 0 \) otherwise. We denote by \( \text{MAX}(A) \) the maximum number of ones in \( (a^{(1)}_i, \ldots, a^{(n)}_i) \) over all \( i = 1, \ldots, m \). Let

\[
\text{MAMI}(A) = \min_{h \in B_m} \text{MAX}(A + h) = \text{MAX}(A + \text{MAJ}(A)).
\]

(2.7)

For \( j \in [m] \) and \( \xi \in \{0,1\} \) let \( A_{j,\xi} = \{ a \in A \mid a_j = \xi \} \). Then

\[
\text{MAMI}(A) = \max_j \min(|A_{j,0}|, |A_{j,1}|).
\]

(2.8)

We define the density of a set \( A \subseteq B_m \) by

\[
\text{DEN}(A) = \max_{B \subseteq A} \frac{|B| - 1}{\text{MAMI}(B)}.
\]

(2.9)

Notice that since every \( j \in [m] \) can hit at most \( \text{MAX}(A) \) elements in \( A \) we have

\[
\text{HS}(A) \geq \frac{|A| - 1}{\text{MAX}(A)}.
\]

(2.10)
Chapter 3

Bounds for OPT

In this chapter we give upper and lower bounds for OPT.

3.1 Lower Bound

Moshkov results in [Mos82, Heg95] and the information theoretic bound in (2.1) give the following lower bound. We also give the proof for completeness.

Lemma 3.1.1. [Mos82, Heg95] Let \( A \subseteq B_m \) be any set. Then

\[
\text{OPT}(A) \geq \max(\text{ETD}(A), \log |A|).
\]

Proof. The lower bound \( \log |A| \) is the information theoretic lower bound. We now prove the other bound.

Let \( T \) be a decision tree for \( A = \{a^{(1)}, \ldots, a^{(n)}\} \) of minimum depth. Consider the path \( P \) in \( T \) that at each level chooses the edge that is labeled with 0. Let \( S \) be the set of labels in the internal nodes of \( P \) and \( a^{(j)} \) be the label of the leaf of \( P \). Then \( a^{(j)} \) is the only element in \( A \) that satisfies \( a^{(j)}_i = 0 \) for all \( i \in S \). Therefore \( S \) is a specifying set for 0 with respect to \( A \). Thus \( \text{OPT}(A) \geq |S| \geq \text{ETD}_z(A) \). Now, by Lemma 2.2.1, for any \( h \in \{0, 1\}^m \) we have \( \text{OPT}(A) = \text{OPT}(A + h) \geq \text{ETD}_z(A + h) = \text{ETD}(A, h) \) and therefore \( \text{OPT}(A) \geq \max_h \text{ETD}(A, h) = \text{ETD}(A) \).

Many lower bounds in the literature for \( \text{OPT}(A) \) are based on finding a subset \( B \subseteq A \) such that for each query there is an answer that eliminates at most small fraction \( E \) of \( B \). Then \( (|B|-1)/E \) is a lower bound for \( \text{OPT}(A) \). The best possible bound that one can get using this technique is exactly \( \text{DEN}(A) \) (Lemma 3.1.2), the density defined in Section 2.5. Lemma 3.1.3 shows that the lower bound \( \text{ETD}(A) \) for \( \text{OPT}(A) \) exceeds any such bound.

Lemma 3.1.2. We have \( \text{OPT}(A) \geq \text{DEN}(A) \).
Proof. Let $B \subseteq A$ be a set such that

$$\text{DEN}(A) \overset{(2.9)}{=} \frac{|B| - 1}{\text{MAMI}(B)} \overset{(2.7)}{=} \frac{|B| - 1}{\text{MAX}(B + \text{MAJ}(B))}.$$  

For every query $i \in [m]$ (what is “$a_i$”?), the adversary answers $\text{MAJ}(B)_i$. This eliminates at most $\text{MAX}(B + \text{MAJ}(B))$ elements from $B$. Therefore the algorithm is forced to ask at least $\left(\frac{|B| - 1}{\text{MAX}(B + \text{MAJ}(B))}\right)$ queries.

Lemma 3.1.3. We have $\text{ETD}(A) \geq \text{DEN}(A) - 1$.

Proof. By (2.7) and (2.9) there is $B \subseteq A$ such that

$$\text{DEN}(A) \overset{(2.9)}{=} \frac{|B| - 1}{\text{MAMI}(B)} \overset{(2.7)}{=} \frac{|B| - 1}{\text{MAX}(B + h)} \overset{(3.1)}{=} \text{DEN}(A) - 1,$$

where $h = \text{MAJ}(B)$. Then

$$\text{ETD}(A) \overset{(2.6)}{\geq} \text{ETD}(B) \overset{(2.2)}{\geq} \text{ETD}(B, h) \overset{L2.3.1}{\geq} \text{SETD}(B, h) - 1 \overset{L2.4.1}{=} \text{HS}(B + h) - 1 \overset{(3.1)}{=} \text{DEN}(A) - 1. \quad \square$$

Lemma 3.1.4. We have

$$\text{ETD}(A) \leq \ln |A| \cdot \text{DEN}(A) + 1.$$  

Proof. There is $h_0 \in \{0, 1\}^m$ such that

$$\text{ETD}(A) \overset{L2.3.1}{\leq} \text{SETD}(A) \overset{(2.4)}{=} \text{SETD}(A, h_0) \overset{L2.4.1}{=} \text{HS}(A + h_0). \quad (3.2)$$

For any $C \subseteq A$ we have

$$\text{DEN}(C) \overset{(2.9)}{=} \max_{B \subseteq C} \frac{|B| - 1}{\text{MAMI}(B)} \overset{(2.7)}{\geq} \max_{B \subseteq C} \frac{|B| - 1}{\text{MAX}(B + h)} \overset{(3.1)}{=} \frac{|C| - 1}{\text{MAX}(C + h_0)} \overset{2.10}{\geq} \text{DEN}(C),$$

and therefore, for any $C \subseteq A$ we have

$$\text{MAX}(C + h_0) \geq \frac{|C| - 1}{\text{DEN}(C)} \overset{(2.9)}{\geq} \frac{|C + h_0| - 1}{\text{DEN}(A)}. \quad (3.3)$$
We now consider the following sequence of subsets of \( A + h_0, C_0, C_1, \ldots, C_t \) where \( C_0 = A + h_0 \) and the subset \( C_{i+1} \) is defined by \( C_i \) as follows: Since (3.3) is also true for \( C_i \) there is \( j_i \in [m] \) such that \( j_i \) hits at least \((|C_i| - 1)/\text{DEN}(A)\) elements in \( C_i \). Then \( C_{i+1} \) contains all the elements in \( C_i \) that are not hit by \( j_i \). Then

\[
|C_{i+1}| - 1 \leq |C_i| - \frac{|C_i| - 1}{\text{DEN}(A)} = (|C_i| - 1) \left( 1 - \frac{1}{\text{DEN}(A)} \right) - 1.
\]

Therefore

\[
|C_i| \leq (|A| - 1) \left( 1 - \frac{1}{\text{DEN}(A)} \right)^i + 1.
\]

Let \( C_t \) be the first set in this sequence that satisfies \( C_t = \emptyset \) or \( C_t = \{0\} \). Define \( X = \{ j_i | i = 0, 1, \ldots, t-1 \} \). Then \( X \) is a hitting set for \( A + h_0 \) of size \( t \). Therefore, by (3.2) we have

\[
\text{ETD}(A) \leq \text{HS}(A + h_0) \leq t \leq \frac{\ln(|A| - 1)}{\ln \left( 1 - \frac{1}{\text{DEN}(A)} \right)} + 1 \leq \text{DEN}(A) \cdot \ln |A| + 1.
\]

It is also easy to see (by standard analysis using Chernoff Bound) that for a random uniform \( A \), with positive probability, \( \text{DEN}(A) = O(1) \) and \( \text{ETD}(A) = \Theta(\log |A|) \). We will give a proof sketch for this claim. So the bound in Lemma 3.1.4 is asymptotically best possible.

**Lemma 3.1.5.** There is a set \( A \subseteq B_m \) of size \( n \) where \( m = \text{poly}(n) \) such that \( \text{ETD}(A) = \Omega(\log n) \) and \( \text{DEN}(A) = O(1) \).

**Proof.** Consider a random uniform set \( A \subseteq B_m \) of size \( n \). The probability that there are \( k = (\log n)/2 \) entries \( i_1, \ldots, i_k \in [m] \) such that no \( a \in A \) satisfies \( a_{i_1} = a_{i_2} = \cdots = a_{i_k} = 0 \) is

\[
\binom{m}{k} \left( 1 - \frac{1}{2^k} \right)^n \leq \frac{1}{4}.
\]

Therefore, with probability at least \( 3/4 \), \( \text{ETD}(A) \geq k \) and then \( \text{ETD}(A) = \Omega(\log n) \).

The probability that some subset \( B \subseteq A \) of size \( |B| > 100 \) has \( \text{MAMI}(B) \leq |B|/100 \) is at most

\[
2^n \left( \frac{1}{2} \right)^m \leq \frac{1}{4}.
\]

Therefore with probability at least \( 3/4 \), \( \text{MAMI}(B) \geq |B|/100 \) and \( \text{DEN}(A) = O(1) \).

### 3.2 Upper Bounds

Moshkov [Mos82, Heg95] proved the following upper bound. We give here the proof for completeness. It is the same as the proof of Lemma 3.2 in [Heg95]. The proof of this Lemma can be applied to prove Theorem 4.1.
Lemma 3.2.1. [Mos82, Heg95] Let $A \subseteq \{0,1\}^m$ of size $n$. Then

$$\text{OPT}(A) \leq \text{ETD}(A) + \frac{\text{ETD}(A)}{\log \text{ETD}(A)} \log n \leq 2 \cdot \frac{\text{ETD}(A)}{\log \text{ETD}(A)} \log n.$$ 

Proof. Consider the algorithm in Figure 3.1. In Step 3, the algorithm defines a hypothesis that is the bitwise majority of all the vectors in $A^{(i,1)}$. In Step 7 an index $y$ is found that minimizes the size of $A^{(i,k)}(y,f_y) := \{ g \in A^{(i,k)} \mid g_y = h_y \}$.

Suppose the variable $i$ (in the algorithm) gets the values $1, 2, \ldots, t+1$ and for each $1 \leq i \leq t$ the variable $k$ gets the values $0, 1, 2, \ldots, k_i$. Then the number of membership queries asked by the algorithm is $k_1 + \cdots + k_t$. We first prove the following

Claim For $i = 1, \ldots, t-1$ we have

$$|A^{(i+1,1)}| \leq \frac{|A^{(i,1)}|}{\max(2, k_i)}.$$ 

Proof. Since $S$ is a specifying set for $h$, either some $y \in S$ satisfies $h_y \neq a_y$ or $a$ is the only column in $A$ that is consistent with $h$ on $S$. Therefore, since $h = \text{Majority}(A^{(i,1)})$, we have

$$|A^{(i+1,1)}| \leq \frac{|A^{(i,1)}|}{2}. \quad (3.4)$$

Let $D = A^{(i,1)}$ and $D' = A^{(i+1,1)}$. Suppose $y_1, \ldots, y_{k_i}$ are the queries that were asked in the $i$th stage and let $\delta_j = a_{y_j}$ for $j = 1, \ldots, k_i$. Then

$$D' = D_{(y_1, \delta_1), (y_2, \delta_2), \ldots, (y_{k_i}, \delta_{k_i})}$$

and (disjoint union)

$$D = D_{(y_1, \delta_1)} \cup D_{(y_1, \delta_1), (y_2, \delta_2)} \cup \cdots \cup D_{(y_1, \delta_1), (y_2, \delta_2), \ldots, (y_{k_i-1}, \delta_{k_i-1}), (y_{k_i}, \delta_{k_i})} \cup D'.$$

Let $D^{(j)} = D_{(y_1, \delta_1), (y_2, \delta_2), \ldots, (y_{j}, \delta_{j})}$, the set of columns in $D$ that are consistent with the target column on the first $j$ assignments $y_1, \ldots, y_j$. Then

$$D = D_{(y_1, \delta_1)}^{(0)} \cup D_{(y_2, \delta_2)}^{(1)} \cup \cdots \cup D_{(y_{k_i}, \delta_{k_i})}^{(k_i-1)} \cup D'.$$

For $0 \leq j \leq k_i - 2$, the fact that we took $y_{j+1}$ for the $(j+1)$th query and not $y_{k_i}$ implies
that $|D^{(j)}_{(y_{j+1}, h_{y_{j+1}})}| \leq |D^{(j)}_{(y_k, h_{y_k})}|$. Therefore, for $0 \leq j \leq k_i - 2$

$$|D^{(j)}_{(y_{j+1}, \delta_{y_{j+1}})}| = |D^{(j)}_{(y_{j+1}, h_{y_{j+1}})}| \geq |D^{(j)}_{(y_k, h_{y_k})}| \geq |D'|.$$  

Therefore

$$|D| = |D^{(0)}_{(y_1, \delta_1)}| + |D^{(1)}_{(y_2, \delta_2)}| + \cdots + |D^{(k_i - 1)}_{(y_{k_i}, \delta_{k_i})}| + |D'| \geq k_i \cdot |D'|.$$  

With (3.4), the result of the claim follows.

Let $z_i = \max(2, k_i)$. Then

$$1 \leq |A^{(i,1)}| \leq \frac{n}{\prod_{i=1}^{t} z_i}$$

and therefore $\sum_{i=1}^{t-1} \log z_i \leq \log n$. Now for $E \geq 4$ and since $E \leq n$

$$\sum_{i=1}^{t} k_i = k_t + \sum_{i=1}^{t-1} \log z_i \frac{k_i}{\log z_i} \leq k_t + \max_{i} \frac{k_i}{\log z_i} \log n \leq E + \frac{E}{\log E} \log n \leq 2E \log n.$$  

It is also easy to show that the above is also true for $E = 2, 3$.

We now prove the time complexity. Finding a specifying set at each iteration of the While loop takes time $T$ and the number of iterations is at most $\log n$. This takes $T \log n$ time. Now at the first iteration we define an array of length $|S| \leq E$ that contains $|A^{(i,1)}_{(z, h_z)}|$ for each $z \in S$. This takes at most $|A^{(i,1)}| \cdot E$ time. Now if we have such array for $A^{(i,k)}_{(z, h_z)}$, we can find $y$ (in Step 7) in time $E$ and update the array for $A^{(i,k+1)}_{(z, h_z)} = A^{(i,k)}_{(y, h_y)}$ in time $|A^{(i,k)}_{(y, h_y)}| \cdot E$. Therefore the time of the Repeat loop is at most $2|A^{(i,1)}| \cdot E$. Since $|A^{(i+1,1)}| \leq |A^{(i,1)}|/2$, the time of the While loop is at most $4n \cdot E$. This gives the result.

In [Mos82, Heg95], Moshkov gave an example of a $n$-set $A_E \subseteq \{0,1\}^m$ with $\text{ETD}(A_E) = E$ and $\text{OPT}(A_E) = \Omega((E/\log E) \log n)$. So the upper bound in the above lemma is the best possible.
Algorithm: Find the hidden column $a \in A$.

1. $i \leftarrow 1$, $k \leftarrow 0$, $A^{(1,1)} \leftarrow A$.
2. While $|A^{(i,1)}| \geq 2$ do
3. $h \leftarrow \text{Majority}(A^{(i,1)})$
4. Find a specifying set $S$ for $h$ with respect to $A^{(i,1)}$
5. Repeat
6. $k \leftarrow k + 1$.
7. Find $y \leftarrow \arg \min_{z \in S} |A^{(z,h)}^{(i,k)}|$
8. Ask query “What is $a_y$?”
9. $A^{(i,k+1)} \leftarrow A^{(i,k)}_{(y,a_y)}$
10. $S \leftarrow S \setminus \{y\}$.
11. Until ($h_y \neq a_y$ or $|A^{(i,k+1)}| = 1$)
12. $A^{(i+1,1)} \leftarrow A^{(i,k+1)}$, $i \leftarrow i + 1$, $k \leftarrow 0$
13. End While
14. Output the column in $A^{(i,k)}$.

Figure 3.1: An algorithm that finds the hidden column $a \in A$
Chapter 4

Polynomial Time Approximation Algorithm

In this chapter we give our main result, which is a \((\ln 2)\text{ETD}(A)\)-approximation algorithm that runs in a polynomial time.

Given a set \(A \subseteq B_m\). Can one construct an algorithm that finds a hidden \(a \in A\) with \(\text{OPT}(A)\) queries? Obviously, with unlimited computational power this can be done so the question is: How close to \(\text{OPT}(A)\) can one get when polynomial time \(\text{poly}(m, n)\) is allowed for the construction?

An exponential time algorithm follows from the following

\[
\text{OPT}(A) = \min_{i \in [m]} \max(\text{OPT}(A_{i,0}), \text{OPT}(A_{i,1}))
\]

where \(A_{i,\xi} = \{a \in A \mid a_i = \xi\}\). This algorithm runs in time at least \(m! \geq (m/e)^m\). See also [Gar71, AGM93].

Can one give a better exponential time algorithm? In what follows (Theorem 4.1) we use Moshkov [Mos82, Heg95] result (Lemma 3.2.1) to give a better exponential time approximation algorithm. Here we give another simple proof of the Moshkov [Mos82, Heg95] result that in practice uses less number of specifying sets. When the extended teaching dimension is constant, the algorithm is \(O(1)\)-approximation algorithm and runs in polynomial time.

**Theorem 4.1.** Let \(A\) be a class of sets \(A \subseteq B_m\) of size \(n\). If there is an algorithm that for any \(h \in B_m\) and any \(A \in A\) gives a specifying set for \(h\) with respect to \(A\) of size at most \(E\) in time \(T\) then there is an algorithm that for any \(A \in A\) constructs a decision tree for \(A\) of depth at most

\[
E + \frac{E}{\log E} \log n \leq E + \frac{E}{\log E} \text{OPT}(A)
\]

queries and runs in time \(O(T \log n + nm)\).
Proof. of Theorem 4.1 The proof is the same as the proof of Lemma 3.2.1. Here we give another proof.

Consider the following algorithm. After the \(i\)th query, the algorithm defines a set \(A_i \subseteq A\) of all the columns that are consistent with the answers of the queries that were asked so far. Consider any \(0 < \epsilon < 1\). Now the algorithm searches for a \(j \in [m]\) such that
\[
\epsilon |A_i| \leq |\{a \in A_i \mid a_j = 0\}| \leq (1 - \epsilon) |A_i|.
\]
If such \(j \in [m]\) exists then the algorithm asks “What is \(a_j\)?”. Let the answer be \(\xi\). Define \(A_{i+1} = \{a \in A_i \mid a_j = \xi\}\). Obviously, in that case,
\[
|A_{i+1}| \leq (1 - \epsilon) |A_i|.
\]
If no such \(j \in [m]\) exists then the algorithm finds a specifying set \(T_h\) for \(h := \text{Majority}(A_i)\), where “Majority” is the bitwise majority function. Then asks queries “What is \(a_j\)” for all \(j \in T_h\). If the answers are consistent with \(h\) on \(T_h\) then there is a unique column \(c \in A_i\) consistent with the answers and the algorithm outputs the index of this column. Otherwise, there is \(j_0 \in T_h\) such that \(a_{j_0} \neq h_{j_0}\). It is easy to see that in that case
\[
|A_{i+1}| \leq \epsilon |A_i|.
\]
Now when \(\epsilon = \ln E / E\) we get
\[
\text{OPT}(A) \leq \max \left( E \left\lfloor \frac{\log n}{\log (1/\epsilon)} \right\rfloor, \left\lfloor \frac{\log n}{\log (1/(1 - \epsilon))} \right\rfloor \right)
\leq \frac{2E}{\log E} \log n.
\]
The time complexity of this algorithm is \(O(T \log n + mn)\).

In fact one can prove the bound
\[
\text{OPT}(A) \leq \left( \frac{E}{\log E} + \frac{E \log \log E}{\log^2 E} + o \left( \frac{E \log \log E}{\log^2 E} \right) \right) \log n
\]
by substituting \(\epsilon = (\ln E)/(E(1 + \ln \ln E/\ln E))\).

The following result immediately follows from Theorem 4.1.

**Theorem 4.2.** Let \(A \subseteq B_m\) be a \(n\)-set. There is an algorithm that finds the hidden column in time
\[
\binom{m}{\text{ETD}(A)} \cdot \text{ETD}(A) \cdot n \log n
\]
and asks at most
\[
\frac{2 \cdot \text{ETD}(A) \cdot \log n}{\log \text{ETD}(A)} \leq \frac{2 \cdot \min(\text{ETD}(A), \log n)}{\log \text{ETD}(A)} \text{OPT}(A)
\]
queries.

In particular, if $ETD(A)$ is constant then the algorithm is $O(1)$-approximation algorithm that runs in polynomial time.

**Proof.** To find a specifying set for $h$ with respect to $A$ we exhaustively check each $ETD(A)$ row of $A$. Each check takes time $n$. Since the algorithm asks at most $ETD(A) \cdot \log n$ queries, the time complexity is as stated in the Theorem. \qed

Can one do it in $poly(m, n)$ time? Hyafil and Rivest, [HR76], show that the problem of finding $OPT$ is NP-Complete. The reduction of Laber and Nogueira, [LN04], of set cover to this problem with the inapproximability result of Dinur and Steurer [DS14] for set cover implies that it cannot be approximated to $(1 - o(1)) \cdot \ln n$ unless $P=NP$.

In [AMM+98], Arkin et al. showed that (the AMMRS-algorithm) if at the $i$th query the algorithm chooses an index $j$ that partitions the current node set (the elements in $A$ that are consistent with the answers until this node) $A$ as evenly as possible, that is, that maximizes $\min(|\{a \in A | a_j = 0\}|, |\{a \in A | a_j = 1\}|)$, then the query complexity is within a factor of $[\log n]$ from optimal. The AMMRS-algorithm, [AMM+98], runs in time $poly(m, n)$. Moshkov [AMM+98, Mos04] analysis shows that this algorithm is $\ln n$-approximation algorithm and therefore is optimal. In this chapter we will give a simple proof.

In [Mos82, Heg95], Moshkov gave a simple $ETD(A)$-approximation algorithm (Algorithm MEMB-HALVING-1 in [Heg95]). He then gave another algorithm that achieves the query complexity in Lemma 3.2.1 (Algorithm MEMB-HALVING-2 in [Heg95]). This is within a factor of $\frac{2 \cdot \min(ETD(A), \log n)}{\log ETD(A)}$ from optimal. This is better than the ratio $\ln n$, but, unfortunately, both algorithms require finding a minimum size specifying set and the problem of finding a minimum size specifying set for $h$ is NP-Hard, [Shi91, ABCS92, GK95].

Can one achieve a $O(ETD(A))$-approximation. In the following we give a surprising result. We show that the AMMRS-algorithm is $(\ln 2)ETD(A)$-approximation algorithm. We also show that no better ratio can be achieved unless $P=NP$.

**Theorem 4.3.** The AMMRS-algorithm runs in time $poly(mn)$ and finds the hidden element $a \in A$ with at most

$$DEN(A) \cdot \ln(n) \leq \min((\ln 2)DEN(A), \ln n) \cdot OPT(A)$$

$$\leq \min((\ln 2)(ETD(A) + 1), \ln n) \cdot OPT(A)$$

queries.

**Proof.** Let $B$ be any subset of $A$. Then,

$$DEN(B) \geq \frac{|B| - 1}{MAMI(B)}$$

(2.9)
and therefore
\[
\text{MAMI}(B) \geq \frac{|B| - 1}{\text{DEN}(B)} \geq \frac{|B| - 1}{\text{DEN}(A)}.
\]

Since the AMMRS-algorithm chooses at each node in the decision tree the index \( j \) that maximizes \( \min(|B_j,0|,|B_j,1|) \) where \( B_{j,\xi} = \{a \in B|a_j = \xi\} \) and \( B \) is the set of elements in \( A \) that are consistent with the answers until this node, we have
\[
\max(|B_{j,0}|,|B_{j,1}|) - 1 = |B| - 1 - \min(|B_{j,0}|,|B_{j,1}|)
\]
\[
= (2.8) |B| - 1 - \text{MAMI}(B) \leq (|B| - 1) \left(1 - \frac{1}{\text{DEN}(A)}\right).
\]

Therefore, for a node \( v \) of depth \( h \) in the decision tree, the set \( B(v) \) of elements in \( A \) that are consistent with the answers until this node contains at most
\[
(|A| - 1) \left(1 - \frac{1}{\text{DEN}(A)}\right)^h + 1
\]

elements. Therefore the depth of the tree is at most
\[
\text{DEN}(A) \ln |A|.
\]

We now show that our approximation algorithm is optimal

**Theorem 4.4.** Let \( \epsilon \) be any constant. There is no polynomial time algorithm that finds the hidden element with less than \((1 - \epsilon)\text{DEN}(A) \cdot \ln |A|\) unless \( P=NP \).

**Proof.** Suppose such an algorithm exists. Then
\[
(1 - \epsilon)\text{DEN}(A) \ln |A| \leq (1 - \epsilon)\ln |A|\text{OPT}(A).
\]

That is, the algorithm is also \((1 - \epsilon)\ln |A|\)-approximation algorithm. Laber and Nogueira, [LN04] gave a polynomial time algorithm reduction of minimum depth decision tree to set cover and Dinur and Steurer [DS14] show that there is no polynomial time \((1 - o(1)) \cdot \ln |A|\) for set cover unless \( P=NP \). Therefore, such an algorithm implies \( P=NP \).
Chapter 5

Applications to Disjunctions of Predicates

In this chapter we apply our results to learning the class of disjunctions of predicates from a set of predicates \( F \) from membership queries [BDVY17].

Let \( C = \{f_1, \ldots, f_n\} \) be a set of boolean functions \( f_i : X \rightarrow \{0, 1\} \) where \( X = \{x_1, \ldots, x_m\} \). Let \( A_C = \{(f_i(x_1), \ldots, f_i(x_m)) \mid i = 1, \ldots, n\} \). We will write \( \text{OPT}(A_C), \text{ETD}(A_C) \), etc. as \( \text{OPT}(C), \text{ETD}(C) \), etc.

Let \( F \) be a set of boolean functions (predicates) over a domain \( X \). We consider the class of functions \( F \lor := \{\lor f \in S \mid S \subseteq F\} \).

5.1 An Equivalence Relation Over \( F \lor \)

In this section, we present an equivalence relation over \( F \lor \) and define the representatives of the equivalence classes. This enables us in later sections to focus on the representative elements from \( F \lor \). Let \( F \) be a set of boolean functions over the domain \( X \). The equivalence relation \( = \) over \( F \lor \) is defined as follows: two disjunctions \( F_1, F_2 \in F \lor \) are equivalent \( (F_1 = F_2) \) if \( F_1 \) is logically equal to \( F_2 \). In other words, they represent the same function (from \( X \) to \( \{0, 1\} \)). We write \( F_1 \equiv F_2 \) to denote that \( F_1 \) and \( F_2 \) are identical; that is, they have the same representation. For example, consider \( f_1, f_2 : \{0, 1\} \rightarrow \{0, 1\} \) where \( f_1(x) = 1 \) and \( f_2(x) = x \). Then, \( f_1 \lor f_2 = f_1 \) but \( f_1 \lor f_2 \neq f_1 \).

We denote by \( F_\lor^* \) the set of equivalence classes of \( = \) and write each equivalence class as \([F]\), where \( F \in F_\lor \). Notice that if \([F_1] = [F_2]\), then \([F_1 \lor F_2] = [F_1] = [F_2]\). Therefore, for every \([F]\), we can choose the representative element to be \( G_F := \lor f \in S F \) where \( S \subseteq F \) is the maximum size set that satisfies \( \lor S := \lor f \in S f = F \). We denote by \( G(F_\lor) \) the set of all representative elements. Accordingly, \( G(F_\lor) = \{G_F \mid F \in F_\lor\} \). As an example, consider the set \( F \) consisting of four functions \( f_{11}, f_{12}, f_{21}, f_{22} : \{1, 2\}^2 \rightarrow \{0, 1\} \) where \( f_{ij}(x_1, x_2) = [x_i \geq j] \) where \([x_i \geq j] = 1 \) if \( x_i \geq j \) and 0 otherwise. There are \( 2^4 = 16 \) elements in \( \text{Ray}_2^2 := F_\lor \) and five representative functions in \( G(F_\lor) \):
\( G(F_\vee) = \{ f_{11} \lor f_{12} \lor f_{21} \lor f_{22}, f_{12} \lor f_{22}, f_{12}, f_{22}, 0 \} \) (where 0 is the zero function).

### 5.2 A Partial Order Over \( F_\vee \) and Hasse Diagram

In this section, we define a partial order over \( F_\vee \) and present related definitions. The partial order, denoted by \( \Rightarrow \), is defined as follows: \( F_1 \Rightarrow F_2 \) if \( F_1 \) logically implies \( F_2 \). Consider the Hasse diagram \( H(F_\vee) \) of \( G(F_\vee) \) for this partial order. The maximum (top) element in the diagram is \( G_{\text{max}} := \lor_{f \in F} f \). The minimum (bottom) element is \( G_{\text{min}} := \lor_{f \notin F} f \), i.e., the zero function. Figures 5.2 and 5.3 shows an illustration of the Hasse diagram.

In a Hasse diagram, \( G_1 \) is a descendant (resp., ascendant) of \( G_2 \) if there is a (nonempty) downward path from \( G_2 \) to \( G_1 \) (resp., from \( G_1 \) to \( G_2 \)), i.e., \( G_1 \Rightarrow G_2 \) (resp., \( G_2 \Rightarrow G_1 \)) and \( G_1 \neq G_2 \). \( G_1 \) is an immediate descendant of \( G_2 \) in \( H(F_\vee) \) if \( G_1 \Rightarrow G_2 \), \( G_1 \neq G_2 \) and there is no \( G \in G(F_\vee) \) such that \( G \neq G_1 \), \( G \neq G_2 \) and \( G_1 \Rightarrow G \Rightarrow G_2 \). \( G_1 \) is an immediate ascendant of \( G_2 \) if \( G_2 \) is an immediate descendant of \( G_1 \).

We denote by \( \text{De}(G) \) and \( \text{As}(G) \) the sets of all the immediate descendants and immediate ascendants of \( G \), respectively. The neighbours set of \( G \) is \( \text{Ne}(G) = \text{De}(G) \cup \text{As}(G) \). We further denote by \( \text{DE}(G) \) and \( \text{AS}(G) \) the sets of all \( G \)'s descendants and ascendants, respectively.

**Definition 5.2.1.** The degree of \( G \) is \( \deg(G) = |\text{Ne}(G)| \) and the degree \( \deg(F_\vee) \) of \( F_\vee \) is \( \max_{G \in G(F_\vee)} \deg(G) \).

For \( G_1 \) and \( G_2 \), we define their lowest common ascendent (resp., greatest common descendant) \( G = \text{lca}(G_1, G_2) \) (resp., \( G = \text{gcd}(G_1, G_2) \)) to be the minimum (resp., maximum) element in \( \text{AS}(G_1) \cap \text{AS}(G_2) \) (resp., \( \text{DE}(G_1) \cap \text{DE}(G_2) \)).

The following result is from [BDVY17]

**Lemma 5.2.2.** Let \( G_1, G_2 \in G(F_\vee) \). Then, \( \text{lca}(G_1, G_2) = G_1 \lor G_2 \).

In particular, if \( G_1, G_2 \) are two distinct immediate descendants of \( G \), then \( G_1 \lor G_2 = G \).

For \( G \in G(F_\vee) \), we denote its logical negation (complement) function by \( \bar{G} \).

### 5.3 Witnesses

In this subsection we define the term witness. Let \( G_1 \) and \( G_2 \) be elements in \( G(F_\vee) \). An element \( a \in X \) is a witness for \( G_1 \) and \( G_2 \) if \( G_1(a) \neq G_2(a) \).

For a class of boolean functions \( C \) over a domain \( X \) and a function \( G \in C \) we say that a set of elements \( W \subseteq X \) is a witness set for \( G \) in \( C \) if for every \( G' \in C \) and \( G' \neq G \) there is a witness in \( W \) for \( G \) and \( G' \).
5.4 The Extended Teaching Dimension of $\mathcal{F}_\vee$

In this section we prove

**Lemma 5.4.1.** For every $h : X \rightarrow \{0, 1\}$ if $h G_{\text{max}}$ then $\text{ETD}(\mathcal{F}_\vee, h) = 1$. Otherwise, there is $G \in G(\mathcal{F}_\vee)$ such that

$$\text{ETD}(\mathcal{F}_\vee, h) \leq |\text{De}(G)| + \text{HS}(\text{As}(G) \wedge \bar{G}) \leq |\text{Ne}(G)| = \deg(G)$$

where $\text{As}(G) \wedge \bar{G} = \{s \wedge \bar{G} \mid s \in \text{As}(G)\}$. In particular,

$$\text{ETD}(\mathcal{F}_\vee) \leq \max_{G \in G(\mathcal{F}_\vee)} (|\text{De}(G)| + \text{HS}(\text{As}(G) \wedge \bar{G})) \leq \deg(\mathcal{F}_\vee).$$

**Proof.** Let $h : X \rightarrow \{0, 1\}$ be any function. If $h G_{\text{max}}$ then there is an assignment $a$ that satisfies $h(a) = 1$ and $G_{\text{max}}(a) = 0$. Since for all $G \in G(\mathcal{F}_\vee)$, $G \Rightarrow G_{\text{max}}$ we have $G(a) = 0$. Therefore, the set $\{a\}$ is a specifying set for $h$ with respect to $\mathcal{F}_\vee$ and $\text{ETD}(\mathcal{F}_\vee, h) = 1$.

Let $h \Rightarrow G_{\text{max}}$. Consider any $G \in G(\mathcal{F}_\vee)$ such that $h \Rightarrow G$ and for every immediate descendant $G'$ of $G$ we have $h G'$. Now for every immediate descendant $G'$ of $G$ find an assignment $a$ such that $G'(a) = 0$ and $h(a) = 1$. Then $a$ is a witness for $h$ and $G'$. Therefore, $a$ is also a witness for $h$ and every descendant of $G'$. Let $A$ be the set of all such assignments, i.e., for every descendant of $G$ one witness. Then $|A| \leq |\text{De}(G)|$ and $A$ is a witness set for $h$ and all the descendants of $G$. We note here that if $h = 0$ then $G = G_{\text{min}}$ which has no immediate descendants and then $A = \emptyset$.

Consider a hitting set $B$ for $\text{As}(G) \wedge \bar{G}$ of size $\text{HS}(\text{As}(G) \wedge \bar{G})$. Now for every immediate ascendant $G''$ of $G$ find an assignment $b \in B$ such that $G''(b) \wedge \bar{G}(b) = 1$. Then $G''(b) = 1$ and $G(b) = 0$. Since $G(b) = 0$ we have $h(b) = 0$ and then $b$ is a witness for $h$ and $G''$. Therefore, $b$ is also a witness for $h$ and every ascendant of $G''$. Thus $B$ is a witness set for $h$ in all the ascendants of $G$.

Let $G_0$ be any element in $G(\mathcal{F}_\vee)$ (that is not a descendant or an ascendant of $G$). Consider $G_1 = \text{lca}(G, G_0)$. By Lemma 5.2.2, we have $G_1 = G \vee G_0$. Since $G_1$ is an ascendant of $G$ there is a witness $a \in B$ such that $G_1(a) = 1$ and $G(a) = 0$. Then $G_0(a) = 1$, $h(a) = 0$ and $a$ is a witness of $h$ and $G_0$. Therefore $A \cup B$ is a specifying set for $h$ with respect to $G(\mathcal{F}_\vee)$. Since for every $F \in \mathcal{F}_\vee$ we have $F = G_F \in G(\mathcal{F}_\vee)$, $A \cup B$ is also a specifying set for $h$ with respect to $\mathcal{F}_\vee$.

Since

$$\text{ETD}(\mathcal{F}_\vee, h) \leq |A| + |B| \leq |\text{De}(G)| + \text{HS}(\text{As}(G) \wedge \bar{G})$$

the result follows.

We could have replaced $|\text{De}(G)|$ by $\text{HS}(\overline{\text{De}(G)} \wedge G)$, but the upcoming Lemma 5.4.3 shows that they are both equal.

The following result is from [BDVY17].
Lemma 5.4.2. Let $\text{De}(G) = \{G_1, G_2, \ldots, G_t\}$ be the set of immediate descendants of $G$. If $a$ is a witness for $G_1$ and $G$, then $a$ is not a witness for $G_i$ and $G$ for all $i > 1$. That is, $G_1(a) = 0$, $G(a) = 1$, and $G_2(a) = \cdots = G_t(a) = 1$.

In the following subsections we find $\text{ETD}(\mathcal{F}_\vee)$ exactly. We prove

$$\text{ETD}(\mathcal{F}_\vee) = \max_{G \in \mathcal{G}(\mathcal{F}_\vee)} |\text{De}(G)| + \text{HS}(\text{As}(G) \land \bar{G}).$$

5.4.1 Teaching Dimension

The minimum size of a witness set for $G$ in $C$ is called the witness size and is denoted by $\text{TD}(C, G)$. The value

$$\text{TD}(C) := \max_{G \in C} \text{TD}(C, G)$$

is called the teaching dimension of $C$, [GK95, GRS, SM]. Obviously,

$$\text{ETD}(C, G) \geq \text{TD}(C, G), \quad \text{and} \quad \text{ETD}(C) \geq \text{TD}(C).$$

5.4.2 The Value of $\text{ETD}(\mathcal{F}_\vee)$ and the Proof

Lemma 5.4.3. For every $G \in \mathcal{F}_\vee$ we have

$$\text{TD}(\mathcal{F}_\vee, G) \geq |\text{De}(G)| + \text{HS}(\text{As}(G) \land \bar{G}).$$

In particular,

$$\text{ETD}(\mathcal{F}_\vee) = \text{TD}(\mathcal{F}_\vee) = \max_{G \in \mathcal{G}(\mathcal{F}_\vee)} \left( |\text{De}(G)| + \text{HS}(\text{As}(G) \land \bar{G}) \right).$$

Proof. Let $B$ be a witness set for $G$ in $\text{Ne}(G)$. Take any $G' \in \text{De}(G)$. Then there is $a \in B$ such that $G'(a) = 0$ and $G(a) = 1$. Since for any ascendent $G''$ of $G$ we have $G''(a) = 1$, $a$ is not a witness to $G$ and any of its ascendants. By Lemma 5.4.2, $a$ cannot be a witness to any other descendent. In the similar way, a witness for an ascendent of $G$ and $G$ cannot be a witness for any descendent of $G$ and $G$. Therefore,

$$\text{TD}(\mathcal{F}_\vee, G) \geq \text{TD}(\text{Ne}(G), G) = \text{TD}(\text{De}(G), G) + \text{TD}(\text{As}(G), G) = |\text{De}(G)| + \text{TD}(\text{As}(G), G). \quad (5.1)$$

Now let $S$ be a witness set for $G$ in $\text{As}(G)$. Then for every $G'' \in \text{As}(G)$ there is $a \in S$ such that $G''(a) = 1$ and $G(a) = 0$ which is equivalent to $G''(a) \land \bar{G}(a) = 1$. Therefore,

$$\text{TD}(\text{As}(G), G) \geq \text{HS}(\text{As}(G) \land \bar{G}).$$

This with (5.1) gives the result.
The following result follows immediately from the proof of Lemma 5.4.1

**Lemma 5.4.4.** For any \( h : X \rightarrow \{0, 1\} \), a specifying set for \( h \) with respect to \( F \) of size \( \deg(F) \) can be found in time \( O(nm) \).

By Theorem 4.1 we have

**Theorem 5.1.** There is an algorithm that learns \( F \) in time \( O(nm) \) and asks at most
\[
\deg(F) + \frac{\deg(F)}{\log \deg(F)} \log n \leq \left( \frac{\deg(F)}{\log \deg(F)} + 1 \right) \text{OPT}(F)
\]
member queries.

### 5.5 Learning Other Classes

If a specifying set of small size cannot be found in polynomial time then from Theorem 4.2, 4.3 and Lemma 5.4.1, we have

**Theorem 5.2.** For a class \( C \) we have

1. There is an algorithm that learns \( C \) in time
\[
\left( \frac{m}{\deg(C)} \right) \cdot \text{ETD}(C) \cdot n \log n
\]
and asks at most
\[
\frac{2 \cdot \text{ETD}(C) \cdot \log n}{\log \text{ETD}(C)} \leq \frac{2 \cdot \min(\text{ETD}(C), \log n)}{\log \text{ETD}(C)} \text{OPT}(C)
\]
member queries.

In particular, when \( \text{ETD}(C) \) is constant the algorithm runs in polynomial time and its query complexity is (asymptotically) optimal.

2. There is an algorithm that learns \( C \) in time \( \text{poly}(nm) \) and asks at most
\[
\text{DEN}(C) \cdot \ln(n) \leq \min((\ln 2)\text{DEN}(C), \ln n) \cdot \text{OPT}(C)
\]
\[
\leq \min((\ln 2)(\text{ETD}(C) + 1), \ln n) \cdot \text{OPT}(C)
\]
member queries.

### 5.6 Example of Classes

Define the class \( \text{Ray}_n^m \). The functions are \( f_{i_1, i_2, \ldots, i_m}(x_1, \ldots, x_m) : \{0, 1\}^m \rightarrow \{0, 1\} \) where
\[
f_{i_1, i_2, \ldots, i_m}(x_1, \ldots, x_m) = \bigwedge_{j=1}^m [x_j \geq i_j].
\]
It is easy to see that this class contains \( O(n^m) \) functions and its Hasse degree is \( 2m \). See \( \text{Ray}_4^2 \) in Figure 5.1.

See figure 5.3 for another example of \( F \) with Hasse degree 3.
Figure 5.1: Hasse diagram of $\text{Ray}^2_4$. The functions are $f_i(x_1, x_2) = [x_1 \geq i]$ and $g_i(x_1, x_2) = [x_2 \geq i]$.

Figure 5.2: Hasse diagram of terms and monotone terms.
Figure 5.3: Hasse diagram when $\mathcal{F} = \{f_1, f_2, f_3, g_1, g_2, g_3, h_1, h_2, h_3, h_4, h_5\}$ of functions $\{1, 2, 3\} \times \{1, 2, 3\} \to \{0, 1\}$ where $f_i(x_1, y_1) = [x_1 \geq i]$, $g_i(x_1, x_2) = [x_2 \geq i]$ and $h_i(x_1, x_2) = [x_1 + x_2 \geq i + 1]$. 

Technion - Computer Science Department - M.Sc. Thesis  MSC-2018-11 - 2018
Bibliography


[GRS] Sally A. Goldman, Ronald L. Rivest, and Robert E. Schapire. Learning binary relations and total orders (extended abstract). In 30th Annual


There is another reason to study the \( ETD \) value of a class: if we find the \( ETD \) of a class \( A \), then we either find a better lower bound than the known theoretical lower bound \( \log(|A|) \), or we get an algorithm with a better approximation than \( \log(|A|) \). This is because if \( ETD(A) \leq \log(|A|) \), then the AMMRS algorithm has an approximation of \( (\log 2) \cdot ETD(A) \), which is better than \( \log(|A|) \), and if \( ETD(A) > \log(|A|) \), then the Moshkov algorithm for \( \text{OPT}(A) \) gets a lower bound of the class for \( \text{OPT}(A) \), which is better than the known theoretical lower bound.

On the other hand, there is another inapproximability, that is, the result that it is impossible to get an algorithm that runs in polynomial time and has an approximation of \( (1 - o(1)) \cdot ETD(A) \) unless \( P = \text{NP} \). This result follows directly from the reduction of Laber and Nogueira, for the cover class, and also from the inapproximability result of Dinur and Steurer for the cover class, and the fact that \( \text{DEN}(A) \leq ETD(A) + 1 \leq \text{OPT}(A) + 1 \).
We have the following inequality: \( f : X \rightarrow \{0, 1\} \) where \( X \) is a random variable and \( C \) is a constant. We want to find a function \( g : \mathbb{R} \rightarrow \mathbb{R} \) such that \( g(x) = \log x \) for all \( x \geq 1 \).

Consider the following problem: given a set of points \( X \) and a set of lines \( L \), find a line that minimizes the maximum distance to any point in \( X \) and maximizes the minimum distance to any line in \( L \).

We define the extended teaching dimension (ETD) as follows: \( \text{ETD}(A) = \min \{ n \in \mathbb{N} \mid A \subseteq B_n \} \).

Now, let's consider the case where \( A \subseteq B_m \) and \( m > n \). We want to show that \( \text{ETD}(A) \leq \frac{\log m}{\log n} \).

For any set \( A \subseteq B_m \), we have \( \text{ETD}(A) \leq \log m \) and \( \log n \leq \log m \).

Therefore, we conclude that \( \text{ETD}(A) \leq \frac{\log m}{\log n} \).

Moreover, if \( A \subseteq B_m \) and \( m \) is sufficiently large, then \( \text{ETD}(A) \) is close to \( n \) with high probability.

In summary, we have shown that the extended teaching dimension is a useful tool for analyzing the complexity of teaching a concept to a learner. It provides a measure of how difficult it is to convey the concept to the learner and to what extent the learner understands it.

References:


Technion - Computer Science Department - M.Sc. Thesis MSC-2018-11 - 2018
Theorization

Learning is a process that appears in various fields of life. In general, one can describe learning as the process of acquisition, expansion, or improvement of knowledge, understandings, abilities, stances, skills, behaviors, values, or preferences through research and experimentation. In this way, learning can be said to be a change and growth in any field.

Learning is the ability to predict the unknown based on knowledge gained from previous experiences. The ability to learn is a capability that humans, animals, plants, and various machines also have. Machine learning is a sub-field of computer science that grants computers the ability to learn without being explicitly programmed. The theory of computational learning is a sub-field of artificial intelligence that deals with the study of planning and analysis of algorithms in machine learning. The emphasis in learning computational is on mathematical refinement and rigor.

The learning process in computational learning can be performed using various models of learning. Each model has its rules and laws for learning and also its goals for learning. However, all these models of learning share a few common traits. In each of these models, we have a teacher on one hand and a learner on the other.

The teacher conceals a certain function of a known class of functions. The learning process is performed by the learner building questions and sending them as input to the teacher (oracle). As a response, the teacher returns answers to the learner as output.

Sometimes, we consider the teacher as the rival, because it will cause the learner to ask as many questions as possible. The goal of the learner is to ask the minimum number of questions.

Sometimes, it is impossible to build a model that enables the learner to ask the minimum number of questions, and therefore, in many cases, we try to find a solution that resembles the number of minimum questions that the learner should ask.

We are dealing with the following problem: Given a set of n elements in a set A, δ ∈ {0, 1}^m, in the context of the learning, we want to find an algorithm that runs in polynomial time, which, given the input A, asks the minimum number of questions and finds the element δ. This is equivalent to building a decision tree for A with a minimum height. A decision tree is a binary tree, where each internal node is labeled with an index from \[m\].

Theoretical

A set \(A \subseteq B_m \equiv \{0, 1\}^m\) is a match if

\[\alpha \in A \Rightarrow m^2 \equiv \sum_{i=1}^{m} \alpha_i \]
המחקרו בוצע בהנחייתו של פרופ' נאדר בשושי, בפקולטה למדעי המחשב בטכנון.

תודה

ראשית כל ברצוני ללבוש את מחברו התודה של למתנה של, פרופ', נאדר בשושי, על התמיכה הנדיבה, התמיכה וההדרכה במחלקה של המחקרה. אני רוצה להודות גם להוריו של, לאחיותיו של, ולאחיו של, על התמיכה והשכרה, העידון, האכפתיות והאהבה רבת הערך.

לבסוף ברצוני ל랩ור לрабור של, מחקרה של, את שכרתי ואchersי לрабרו של, בלתי נשכחת.

אני מודה בטכנון על התמיכה הנדיבה במהלך עלייתו של המחקרה במשרה.

ארכיון ל糧ינן על התרומת והזדמנות במשרה והשורת המחקרי.
אלגוריתמים לבניית עצי החלטה בגובה מינימלי במקם פלינומר

זбирן על מחקר

לשם פטנט חלקי של תרומת קבלת התואר
מגיסטר למדעי מחשב המשמש

וסים מח'ול

ורם מח'ול

 Downtown Tel Aviv – הפוליטכנית לישראל
ארה תשע"ח בחודש מרץ 2018
אלגוריתמים לבניית עצים החלטה בגובה מינימלי במחם פולינומי

וסים מוחלט

Technion - Computer Science Department - M.Sc. Thesis MSC-2018-11 - 2018