Algorithms and Complexity Results for Learning and Big Data

BY

Ádám D. Lelkes
B.Sc., Budapest University of Technology and Economics, 2012
M.S., University of Illinois at Chicago, 2014

THESIS

Submitted as partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mathematics
in the Graduate College of the
University of Illinois at Chicago, 2017

Chicago, Illinois

Defense Committee:

György Turán, Chair and Advisor
Lev Reyzin, Advisor
Shmuel Friedland
Lisa Hellerstein, NYU Tandon School of Engineering
Robert Sloan, Department of Computer Science

To my parents and my grandmother / Szüleimnek és nagymamámnak

Acknowledgments

I had a very enjoyable and productive four years at UIC, which would not have been possible without my two amazing advisors, Lev Reyzin and György Turán. I would like to thank them for their guidance and support in every aspect of my graduate studies and research and for always being available when I had questions. Gyuri's humility, infinite patience, and meticulous attention to detail, as well as the breadth and depth of his knowledge, set an example for me to aspire to. Lev's energy and enthusiasm for research and his effectiveness at doing it always inspired me; the hours I spent in Lev's office were often the most productive hours of my week. Both Gyuri and Lev served as role models for me both as researchers and as people.

Also, I would like to thank Gyuri and his wife Rózsa for their hospitality. They invited me to their home a countless number of times, which made time in Chicago much more pleasant.

I would like to thank my defense committee: Shmuel Friedland, Lisa Hellerstein, Lev Reyzin, Robert Sloan, and György Turán. Special thanks to Lisa for the many helpful comments about my dissertation.

This dissertation also wouldn't have been possible without two of my fellow graduate students, Ben Fish and Jeremy Kun, with whom I worked together on many research projects, and thanks to whom eating lunch was never boring.

I would also like to thank Mehryar Mohri for inviting me to NYU for my final

year and Lisa Hellerstein for many research discussions.

In my final year of graduate studies I was supported by the Dean's Scholar Award of UIC's Graduate College.

From my pre-UIC years, I would like to specifically thank my undergraduate thesis advisor, Lajos Rónyai, who introduced me to the world of mathematical research. His ability to choose a research problem which is suitable for an undergraduate student, and his patience and skill for guiding him through the process of research that led to publishable results in the course of a mere semester, still amaze me every time I think about it.

From high school through graduate school, I had many teachers who inspired me and helped me grow as a mathematician as a person. Instead of listing them individually, thereby risking offending anyone omitted from the list, I would like to thank them all in the hope that they know who they are and that I remember them and their support.

Finally, I would like to thank my family for their continuous support and encouragement. Without them none of this would be possible.

Contribution of Authors

Chapter 3 represents the published manuscript [37]: Benjamin Fish, Jeremy Kun, Ádám D. Lelkes, Lev Reyzin, György Turán: On the Computational Complexity of MapReduce. *International Symposium on Distributed Computing (DISC)* 2015: 1-15. This work was also published in Jeremy Kun's thesis [67].

Chapter 4 represents the published manuscript [68]: Ádám D. Lelkes, Lev Reyzin: Interactive Clustering of Linear Classes and Cryptographic Lower Bounds. *International Conference on Algorithmic Learning Theory (ALT)* 2015: 165-176.

Chapter 5 represents the published manuscript [36]: Benjamin Fish, Jeremy Kun, Ádám D. Lelkes: A Confidence-Based Approach for Balancing Fairness and Accuracy. *2016 SIAM International Conference on Data Mining (SDM)*: 144-152.

Some parts of Chapter 2 also originate from the first two papers ([37, 68]); these parts were moved to Chapter 2 in order to improve the organization of this thesis.

In all of these papers, all the work, including the literature review, the formulation of the definitions and theorems, the design of the algorithms, proving the theorems, writing the manuscript and, in the case of the last paper, the implementation of the algorithms and the design and execution of the experiments, was done jointly with the other coauthors.

Table of Contents

List of Figures

List of Tables

Summary

This thesis focuses on problems in the theory and practice of machine learning and big data. We will explore the complexity-theoretic properties of MapReduce, one of the most ubiquitous distributed computing frameworks for big data, give new algorithms and prove computational hardness results for a model of clustering, and study the issue of fairness in machine learning applications.

In our study of MapReduce, we address some of the central questions that computational complexity theory asks about models of computation. After giving a detailed and precise formalization of MapReduce as a model of computation, based on the work of Karloff et al. [61], we compare it to classical Turing machines, and show that languages which can be decided by a Turing machine using sublogarithmic space can also be decided by a constant-round MapReduce computation. In the second half of the chapter, we turn our attention to the question of whether an increased number of rounds or an increased amount of computation time per processor leads to strictly more computational power. We answer this question in the affirmative, proving a hierarchy theorem for MapReduce computations, conditioned on the Exponential Time Hypothesis.

We will also study an interactive model of clustering introduced by Balcan and Blum [11]. In this framework of clustering, we give algorithms for clustering linear functionals and hyperplanes, and give computational hardness results that show that other concept classes, including deterministic finite automata, constant-depth threshold circuits, and Boolean formulas, are not possible to efficiently cluster if standard cryptographic assumptions hold.

Finally, we address the issue of fairness in machine learning. We propose a novel approach for modifying three popular machine learning algorithms, AdaBoost, logistic regression, and support vector machines, to eliminate bias against a protected group. We empirically compare our method to previous approaches in the literature as well as various baseline algorithms by evaluating them on various real-world datasets, and also give theoretical justification for its performance. We also propose a new measure of fairness for machine learning classifiers, and demonstrate that it can help distinguish between naive and more sophisticated approaches even in the cases when measuring error and bias is not sufficient.

1

Introduction

In the last few decades, the higher availability and decreasing cost of computational power and storage, along with significant innovations in methods for processing large data sets, introduced the era of big data. According to some estimates, the amount of data generated every day is in the exabytes (10^{18} bytes) range.

Processing large amounts of data poses many challenges, to which both theorists and practitioners are constantly trying to find solutions. Innovation has been driven from both sides. Some of the biggest drivers of growth in the practice of big data have been novel engineering solutions such as MapReduce (which we will study in Chapter 3). Also, many of the most successful methods started as *ad hoc* solutions which were based on some intuitive understanding of big data phenomena, but lacked solid theoretical underpinnings. On the other hand, there have also been examples of new areas of practice started by a theoretical insight.

A prominent such example is boosting. Boosting started out as a question to a purely theoretical question about the nature of PAC learning, but through successive stages of theoretical research, the solution evolved into a meta-algorithm that not only solved the original theoretical question, but also proved immensely useful for practical applications [81, 40]. Indeed, for a long time the state of the art face detection algorithm was based on boosting [93]. (We will describe both PAC learning and boosting in Chapter 2).

In many cases, there is even a back and forth between theory and practice: theoretical insights lead to new algorithms which, when applied in practice, exhibit behavior that is not fully explainable by the theory. This in turn drives more theoretical research which can lead to yet better algorithms for practice, and so on.

This interplay between theory and practice inspired most of the research presented in this thesis. In the cases of MapReduce and interactive clustering, we will explore frameworks which are ubiquitous in practice but have lacked a satisfactory theory. In the case of fairness in machine learning, we will address an important social consideration that has long been overlooked by machine learning practitioners; our proposed methods, although of a practical nature, build on results in learning theory.

1.1 MapReduce

MapReduce [31] is a framework of distributed computing which makes it easy for software engineers to design systems that process large amounts of data by abstracting away the low-level details of distributed computing and presenting a clean logical view to the engineer.

Parallel and distributed computing, in particular in the case of MapReduce, is an example of an area of computer science where theory has not been able to keep up with practice. Models of parallel computing in complexity theory, many of them formulated before large-scale distributed computing became practically feasible, often fail to describe real-world distributed computing systems. For instance, the well-studied model of PRAM computations [39], in which a large number of processors share the same memory, bears little resemblance to how distributed computing is solved in practice. Conversely, MapReduce, perhaps the currently most popular framework for large-scale distributed computing, has attracted little attention from the complexity theory community. Most theoretical work on MapReduce has focused on designing efficient algorithms; the high-level complexity-theoretic properties of MapReduce have remained largely unexplored.

In Chapter 3, building upon the model and preliminary results of Karloff et al. [61], we begin the complexity-theoretic study of MapReduce by asking some of the most fundamental questions about MapReduce: how does its computational power compare to other models? Does increasing the bounds on computational resources in the model lets it solve a strictly larger set of problems? As for the former question, we give a partial answer by comparing MapReduce to space-bounded complexity classes, proving that MapReduce computations can simulate Turing machines which only use a sublogarithmic amount of space. We will also give evidence which suggests that proving stronger results than this might require significant theoretical breakthroughs. The latter question will be answered in the affirmative, assuming one believes the Existential Time Hypothesis (ETH) to hold. Conditioned on the ETH, we prove a hierarchy theorem which, although less sharp than the classical hierarchy theorems, demonstrates that either increasing the number of rounds or the amount of computation time for each processor strictly increases the set of problems that the MapReduce computation can solve.

1.2 Interactive Clustering

After MapReduce we will turn our attention to clustering. Clustering is one of the most important paradigms in data analysis, but it famously lacks precise theoretical foundations. This has led to such a diverse variety of approaches that it is difficult to even define what the term "clustering" means, other than the vague goal of somehow grouping data points into sets according to some not necessarily well-defined objective.

Approaches to clustering which are based on a well-defined objective function, but no strong assumptions on the data, usually run into computational barriers, most typically by turning out to be NP-hard. In many cases, these problems are not only NP-hard to solve optimally, but also hard to approximate. This NP-hardness is in turn often circumvented by heuristics which try to optimize the objective function but do not have any theoretical guarantees. Other methods make strong assumptions about the distribution generating the data, which assumptions are unlikely to hold for real-world data sets.

Although there have been a variety of approaches to provide a theoretical foundation for clustering, there is still no consensus in the community about the right direction. In this thesis, we will focus on one specific approach, that of introducing limited supervision into clustering. Clustering is usually thought of as a completely unsupervised form of learning, which is what makes it hard to formulate an adequate theory for it. However, there are good reasons to depend on some limited form of supervision in the task of clustering. How to introduce supervision is another question which has many possible answers, including PAC-like frameworks [8] and query-based models [9, 11].

We will study the interactive clustering model of Balcan and Blum [11]. In this model, which we will describe in detail in Chapter 4, the clustering algorithm has access to a teacher that has a correct clustering in mind. The goal is to find this clustering with a small number of queries of a specific limited form. The model is defined in a way such that the teacher, even though she has to answer these queries, has to do exponentially less communication than she would need to if she were to communicate the assignment of each point to a cluster. In this model, we give algorithms for two classes of clustering problems, one in which the the clusters are given by a linear functional over a finite field (of which parity functions are a special case), and one in which the points are in a Euclidean space and the clusters are hyperplanes.

In addition to these positive results, we also prove computational lower bounds for interactive clustering. These lower bounds, like most lower bounds for machine learning problems, are based on cryptographic hardness assumptions. Also, the reader who is familiar with hardness results for PAC learning will not be surprised to see that the concept classes for which we prove our lower bounds include deterministic finite automata, constant-depth threshold circuits, and Boolean formulas. Although our lower bounds and the classical PAC lower bounds do share a similar flavor, there is a fundamental conceptual difference which we will explain in detail in Chapter 4.

1.3 Fairness in Machine Learning

Finally, in Chapter 5, we address a question which is not strictly theoretical in its nature, but is also an example of the study of big data applications lagging behind the practice. In this last chapter we will address the issue of fairness in machine learning applications. As machine learning enters more and more areas of our lives, from lending to education to policing, there is a growing concern that outcomes of decisions made by machines can be discriminatory against protected groups. This concern is well founded: one can find extensive documentation of biases in machine learning systems in the literature, e.g. [4, 20, 88]. Nevertheless, amid the rapid competition in machine learning, practitioners have been busy working on getting better and better results, and spent less time on worrying about unintended consequences. Only recently has the machine learning community shown serious interest in making sure that methods work fairly for everyone.

Our contribution to this area is twofold: first, we introduce a new method for eliminating bias from three of the most popular supervised learning algorithms, namely AdaBoost, logistic regression, and support vector machines. Our method has several advantages. Not only does it outperform most previous methods on several data sets, but it also lets the user transparently and efficiently quantify the trade-off between classification error (as measured on a biased data set) and bias.

Also, we address another question: how can we measure the fairness of a machine learning algorithm? Discrimination occurs because our data has inherent bias. Because of this, we do not have access to an unbiased ground truth that we could use to measure the true bias of an algorithm. Indeed, if we had such unbiased data, we would not be facing the issue of discrimination in machine learning in the first place. Our proposed solution to this problem is to synthetically generate this unbiased ground truth by adding a new random binary attribute, and then introducing bias against one half of the population defined by this attribute. We demonstrate that the measure of bias that this method yields gives useful information about the fairness properties of machine learning algorithms that other, simpler fairness measures do not.

1.4 Organization of the Thesis

Finally, before we begin with the definitions, let us briefly describe the organization of this thesis.

- In Chapter 2, we review definitions and theorems from complexity theory, cryptography, and learning theory that we will be using throughout the thesis.

- In Chapter 3, we study the complexity-theoretic properties of MapReduce computations.

- In Chapter 4, we give algorithms and computational hardness results for a model of interactive clustering.

- In Chapter 5, we study the issue of fairness in machine learning.

Chapter 2 contains the basic definitions and theorems that can be found in standard graduate-level textbooks for complexity theory and learning theory. In each of the subsequent chapters, we refer to these definitions and theorems, but review the background and previous work related to the specific models and problems we discuss in the given chapter. Also, during the time between the publication of the papers on which these chapters are based and the writing of this thesis, there have been new results on these topics, some of which build on our work. We review these subsequent results at the end of each chapter.

2
Background

Parts of this chapter were published in the papers

- Benjamin Fish, Jeremy Kun, Ádám D. Lelkes, Lev Reyzin, György Turán: On the Computational Complexity of MapReduce. *International Symposium on Distributed Computing (DISC)* 2015: 1-15; and

- Ádám D. Lelkes, Lev Reyzin: Interactive Clustering of Linear Classes and Cryptographic Lower Bounds. *International Conference on Algorithmic Learning Theory (ALT)* 2015: 165-176;

which are presented in Chapters 3 and 4, respectively.

2.1 Computational Complexity Theory

We will use notions from complexity theory throughout this thesis. In this section we will give a short summary of the most important concepts and introduce the notation we will use in the later chapters. For a more detailed introduction to computational complexity theory, we refer the reader to [6].

2.1.1 Complexity Classes

The computational model we will use is the Turing machine and its variants, the probabilistic and the nondeterministic Turing machines. We will not define the Turing machine here; a precise definition and a detailed discussion of its most important properties can be found in [6]. However, we will briefly discuss oracle Turing machines. An oracle Turing machine is a Turing machine with a special oracle tape on which it can write the input of a fixed oracle function and in one step obtain the output. We will use T^f to denote an oracle Turing machine T with access to a function f as the oracle. For sets, we will use T^L to denote an oracle Turing machine T with oracle access to the indicator function of a set L.

Complexity classes are sets of *decision problems* that are solvable in a given computational model under certain resource limitations. We will generally assume that inputs to such decision problems are represented as binary strings; we represent decision problems as subsets of the set $\{0,1\}^*$ of all finite binary strings. We will also refer to decision problems as *languages*. A language is *decided* by a Turing machine if the machine accepts an input string if and only if it is a member of the language.

Definition 1 (TIME and NTIME)**.** *A language L is in* TIME$(T(n))$ *for some sequence $T : \mathbb{N} \to \mathbb{N}$ if there is a Turing machine running in time $O(T(n))$ which decides L. Similarly, L is in* NTIME$(T(n))$ *if there is a nondeterministic Turing machine running in time $O(T(n))$ which decides L.*

Now we can define the two most important complexity classes.

Definition 2 (P and NP)**.** $\mathrm{P} = \bigcup_{c=1}^{\infty} \mathrm{TIME}(n^c)$. *Similarly,* $\mathrm{NP} = \bigcup_{c=1}^{\infty} \mathrm{NTIME}(n^c)$.

Definition 3 (NP-hardness)**.** *A language H is* NP-hard *if for every $L \in$* NP*, there is an oracle Turing machine T^H with oracle access to H which runs in polynomial time and decides L. A language is* NP-complete *if it is in* NP *and it is* NP-*hard.*

After time, space is the second most important computational resource that we will consider. The space analogues of the previous complexity classes follow.

Definition 4 (SPACE and PSPACE)**.** *A language L is in* SPACE$(S(n))$ *for some sequence S if there is a Turing machine which decides L using $O(S(n))$ space.* $\mathrm{PSPACE} = \bigcup_{c=1}^{\infty} \mathrm{SPACE}(n^c)$.

We remark that although P = NP is central open problem of complexity theory, the relationship between deterministic and nondeterministic space complexity classes has been well understood since Walter Savitch's 1970 paper [80].

We will also study classes of languages solvable under simultaneous time and space restrictions.

Definition 5 (TISP)**.** *A language L is in* $\mathrm{TISP}(T(n), S(n))$ *or a pair of sequences* $S, T : \mathbb{N} \to \mathbb{N}$ *if there is a Turing machine which decides L using $O(T(n))$ time and $O(S(n))$ space.*

Note that it is believed that generally $\mathrm{TISP}(T(n), S(n)) \neq \mathrm{TIME}(T(n)) \cap \mathrm{SPACE}(S(n))$. The complexity class TISP is studied in the context of time-space trade-offs (see e.g. [38, 95]).

We will also refer to circuit classes which we will now define.

Definition 6 (Boolean circuit)**.** *A* Boolean circuit *is a directed acyclic graph in which the vertices with no incoming edges represent input variables, and other vertices are called* gates *and labeled with one of the logical operators AND, OR, and NOT, and there is exactly one vertex with no outgoing edges. (Sometimes, depending on the circuit class, other Boolean functions besides AND, OR, and NOT are allowed as gate operators, too.)*

The value of an input gate is defined to be equal to the value of the corresponding input variable; the value of the other gates are defined recursively as the result of the application of the logical operator labeling the gate to the values of the vertices connected to the gate.

The value of the vertex with no outgoing edges is called the output *of the circuit.*

The number of incoming edges for a gate is called the fan-in *of the gate. The* depth *of a circuit is the length of the longest directed path in it.*

We now proceed to define two circuit complexity classes.

Definition 7 (AC and TC)**.** *For every $i \in \mathbb{N}$, the class* AC^i *is the set of languages L such that there is a sequence C_n of circuits with AND, OR, and NOT gates such* *that for every $x \in \{0,1\}^*$, the output of $C_{|x|}$ on input x is* 1 *if and only if $x \in L$ and the number of gates in C_n is $O(n^c)$ for some constant c and the depth of C_n is $O(\log^i n)$. The class* AC *is defined as* $\mathrm{AC} = \bigcup_{i=1}^{\infty} \mathrm{AC}^i$.

TC^i *and* TC *are defined analogously with the difference that there is an additional set of allowed gate types (Boolean functions computed by gates) which is called* threshold gates*. A threshold gate with threshold t has value* 1 *if and only if the sum of its inputs is at least t.*

2.1.2 Hierarchy Theorems

One of the central goals of complexity theory is the separation of complexity classes; i.e., proving that two complexity classes are not equal. The most basic separation results are the *hierarchy theorems* which show that allowing a Turing machine to use more of a computational resource lets it solve problems that are unsolvable with less of the same resource. Resources for which hierarchy theorems have been proven include time, space, and nondeterministic time. There are other resources, such as probabilistic time, for which the existence of a hierarchy is still an open problem. To illustrate the idea of a hierarchy theorem, we now state the earliest hierarchy theorem and sketch its proof.

Definition 8 (Time-constructibility)**.** *We call a sequence T* time-constructible *if for all n, $T(n)$ can be computed in time $O(T(n))$.*

Theorem 1 (Time hierarchy theorem [46])**.** *For any pair of time-constructible sequences T and T' such that $T(n)\log T(n) = o(T'(n))$ it is the case that* $\mathrm{TIME}(T(n)) \subsetneq \mathrm{TIME}(T'(n))$.

Proof. We only gave the basic idea of the proof. The proof is based on *diagonalization*, essentially the same technique as used by Cantor in his proof that the set of all infinite sequences of bits is uncountable, or the argument used to prove the undecidability of the halting problem.

Let us fix a binary representation of Turing machines. (It is assumed that every binary string describes a Turing machine, and every Turing machine has infinitely many such encodings.) Consider the Turing machine M which on input x simulates the execution of the Turing machine represented by x for $T(n)$ steps and, if the simulated machine outputs an answer in this time, outputs the opposite answer.

The language L decided by this machine is then by construction in $\mathrm{TIME}(T'(n))$. On the other hand, no Turing machine running in $O(T(n))$ time can decide L: to see that, assume for contradiction there is such a machine M' which runs in $cT(n)$ time for some constant c. Then, there exists a long enough string x representing M' such that $M(x) = 1 - M'(x)$, a contradiction. □

We omitted several details of the proof, including the observation that the overhead introduced by the simulation is at most logarithmic in the input size; hence the logarithmic gap required by the theorem.

An essentially identical argument can be used to prove the analogous hierarchy theorem for space. A similar, albeit somewhat more complex, argument works for proving a hierarchy theorem for nondeterministic time [26].

However, much less is known about simultaneous time and space complexity bounds. In particular, there is no time hierarchy theorem for fixed space; i.e., it is not known that $\mathrm{TISP}(T(n), S(n)) \subsetneq \mathrm{TISP}(T'(n), S(n))$ for an appropriate gap between $T(n)$ and $T'(n)$. The existence of such a hierarchy is mentioned as an open problem in the monograph of Wagner and Wechsung [94]. In Chapter 3 we will prove a conditional hierarchy theorem for TISP, assuming that a conjecture called the Exponential Time Hypothesis is true.

2.1.3 The Exponential Time Hypothesis

The *Exponential Time Hypothesis* (ETH) states that 3-SAT, the canonical NP-complete problem, is not only not decidable in polynomial time, but it actually requires exponential time. 3-SAT is the problem of deciding whether a Boolean formula in 3-conjunctive normal form (i.e., a conjunction of clauses such that each clause contains at most three variables or their negations) is satisfiable. A formula is satisfiable if there is an assignment of true or false values to the variables such that the value of the formula becomes true.

Conjecture 1 (Exponential Time Hypothesis [49, 50])**.** *There exists a constant $c > 0$ such that 3-SAT $\notin \mathrm{TIME}(2^{cn})$.*

This hypothesis and its strong version have been used to prove conditional lower bounds for specific hard problems like vertex cover, and for algorithms in the context of fixed parameter tractability (see, e.g., the survey of Lokshtanov, Marx and Saurabh [71]). The first open problem mentioned in [71] is to relate ETH to some other known complexity theoretic hypotheses.

2.2 Cryptography

In Chapter 4, we will use tools from the theory of cryptography to provide computational hardness results. In this section, we will review the necessary definitions and theorems from cryptography.

2.2.1 Basic Cryptographic Primitives

We start with the most important cryptographic primitive, the one-way function.

Definition 9 (One-way function [96])**.** *A polynomial-time computable function $f : \{0,1\}^* \to \{0,1\}^*$ is* one-way *if for every probabilistic polynomial-time algorithm A, every positive integer α, and every sufficiently large integer n it holds that*

$$\Pr(f(A(f(x))) = f(x)) < n^{-\alpha}$$

where x is chosen uniformly at random from $\{0,1\}^n$.

It is conjectured that one-way functions exist. There are one-way function candidates (i.e., functions believed to be one-way) based on several problems that are believed to be computationally hard, such as factoring or the subset sum problem. The existence of one-way functions is a stronger conjecture than P $\neq$ NP, but it is necessary and sufficient for many other cryptographic primitives.

Definition 10 (Pseudorandom generator [96])**.** *A polynomial-time computable function $G : \{0,1\}^* \to \{0,1\}^*$ is a* pseudorandom generator *of stretch $S(n)$ if for all $x \in \{0,1\}^*$, $|G(x)| = S(|x|)$ and for every probabilistic polynomial-time* *algorithm D, every positive integer α, and every sufficiently large integer n it holds that*

$$|\Pr(D(G(x)) = 1) - \Pr(D(y) = 1)| < n^{-\alpha}$$

where x is chosen uniformly at random from $\{0,1\}^n$ and y is chosen uniformly at random from $\{0,1\}^{S(n)}$.

The existence of one-way functions implies that pseudorandom generators of polynomial stretch exist, too [47]. We will also use a related pseudorandom object, a pseudorandom function family. Also, in Chapter 4, we will need these cryptographic constructions to be resilient against attackers with superpolynomial power; therefore we also add a hardness parameter to the definition which provides a specific time bound for the attacker.

In the following definition we use superscript notation to denote *oracle Turing machines* (see discussion at the beginning of the chapter.)

Definition 11 (Pseudorandom function family [42])**.** *A* pseudorandom function family *of hardness $h(n)$ is a sequence F_n of sets of efficiently computable functions $A_n \to B_n$ such that*

- *there is a probabilistic* poly(n)*-time algorithm that on input 1^n returns a uniformly randomly chosen element of F_n, and*
- *for every $h(n)$ time-bounded probabilistic algorithm D with oracle access to a function $A_n \to B_n$ and every $\alpha > 0$ it holds that*

$$|\Pr(D^{f_n}(1^n) = 1) - \Pr(D^{r_n}(1^n) = 1)| < n^{-\alpha}$$

where f_n is chosen uniformly randomly from F_n, r_n is chosen uniformly at random from the set of all functions $A_n \to B_n$.

It often makes it easier to think about a pseudorandom function family by representing it as a sequence of *keyed pseudorandom functions*, where the elements of F_n correspond to different keys; i.e., $F_n = \{f_K : K \in \{0,1\}^n\}$. In this representation, the sampling algorithm can simply sample a key from the uniform distribution on $\{0,1\}^n$ and return the corresponding function.

The existence of one-way functions implies the existence of pseudorandom function families of polynomial hardness.

2.3 Learning Theory

In this section we give a brief introduction to the concepts from learning theory that we will use in the later chapters. For more background on learning theory, we refer the reader to [72] and [85].

2.3.1 PAC Learning

We begin with a description of the Probably Approximately Correct (PAC) learning framework. A learning task is given by an *instance space* X of all possible examples and a *concept class* C of functions $X \to \{-1,1\}$. The learner is given an i.i.d. random sample from an arbitrary unknown distribution D on X, along with the *labels* assigned to the points in the sample by an unknown concept $c \in C$. The goal of the learner is to output a *hypothesis* that with high probability will be close to the correct target concept under the unknown domain distribution. In the so-called *proper* model of learning, this hypothesis is required to be a member of C; the learner is called *improper* if the output hypothesis is not in C.

We assume that X comes with a fixed representation; we will use n to denote the maximum length of a representation of an element in X. We similarly assume that a representation of C is fixed as well; $size(c)$ will denote the length of the representation of a concept $c \in C$.

Definition 12 (PAC learning [90]). *An algorithm A* PAC learns *a concept class C if for all $\epsilon, \delta > 0$, for all distributions D on X, and for all possible target concepts $c \in C$, if given an i.i.d labeled sample of size $m > \mathrm{poly}(1/\epsilon, 1/\delta, n, size(c))$, A outputs a hypothesis function $h : X \to \{-1, 1\}$ such that*

$$\Pr\left(\Pr_{x \sim D}(h(x) \neq c(x)) \leq \epsilon\right) \geq 1 - \delta$$

where the outer probability is taken over the random sample and the internal randomness of A.

The following combinatorial dimension of concept classes characterizes learnability:

Definition 13 (VC-dimension [92]). *A set $S \subseteq X$ is* shattered *by a concept class C on X if different concepts in C can produce all $2^{|S|}$ possible labelings of S. The* VC-dimension *of C is the maximum cardinality of shattered sets (and ∞ if arbitrarily large sets can be shattered).*

A concept class is PAC learnable if and only if its VC-dimension is finite [19], and the sample complexity of PAC learning, i.e., the minimum sample size m

such there there is an algorithm which PAC learns C using a sample of size m, is $\Theta\left(\frac{1}{\epsilon}\left(d + \log\left(\frac{1}{\delta}\right)\right)\right)$ where d denotes the VC-dimension of C [19, 45].

Also, for such concept classes there is a learning algorithm which is guaranteed to PAC learn the class: output a concept in the class which has minimal error on the sample. This algorithm is called *Empirical Risk Minimization (ERM)*.

In the above definition of PAC learning we assumed that the points are labeled by a function that belongs to the concept class to be learned or, in other words, that the concept class C contains a function which has perfect accuracy on the data. This (so-called *realizability*) assumption is arguably unrealistic for many real-world learning problems. We therefore introduce a slightly different model of learning, in which there is no assumption made about the distribution on the labeled sample. In this setting we clearly cannot hope to achieve arbitrarily low error in the worst case, therefore the learning goal is relaxed: our new goal is to achieve an accuracy which is arbitrarily close to the best classifier in the concept class.

Definition 14 (Agnostic PAC learning [63])**.** *An algorithm A is an* agnostic PAC learning algorithm *for a concept class C if for all $\epsilon, \delta > 0$, for all distributions D on $X \times \{-1, 1\}$, and for all concepts $c \in C$, if given an i.i.d labeled sample of size $m > \text{poly}(1/\epsilon, 1/\delta, n, size(c))$, A outputs a hypothesis function $h : X \to \{-1, 1\}$ such that*

$$\Pr\left(\Pr_{(x,y)\sim D}(h(x) \neq y) \leq \Pr_{(x,y)\sim D}(c(x) \neq y) + \epsilon\right) \geq 1 - \delta$$

where the outer probability is taken over the random sample and the internal randomness of A.

It turns out that VC-dimension characterizes learnability in this more difficult setting as well; in particular, a concept class is agnostic PAC learnable if and only if it is PAC learnable (i.e., if it has finite VC-dimension). Moreover, ERM is an agnostic PAC learning algorithm for any such class.

2.3.2 Boosting

One might ask if more concept classes become learnable if, instead of requiring the learning algorithm to achieve an arbitrarily low error (ϵ for any $\epsilon > 0$), we only require it to perform slightly better than random guessing. This seemingly weaker model of learning is called *weak learning.*

Definition 15 (Weak learning [62])**.** *An algorithm A is a* weak learner *for a concept class C if there exists a $\gamma > 0$ such that for all $\delta > 0$, for all distributions D on X, and for all possible target concepts $c \in C$, if given an i.i.d labeled sample of size $m > \mathrm{poly}(1/\epsilon, 1/\delta, n, size(c))$, A outputs a hypothesis function $h : X \to \{-1, 1\}$ such that*

$$\Pr\left(\Pr_{x\sim D}(h(x) \neq c(x)) \leq \frac{1}{2} - \gamma\right) \geq 1 - \delta$$

where the outer probability is taken over the random sample and the internal randomness of A.

Although this definition might seem strictly weaker than (strong) PAC learning but, as first proved by Schapire [81], it is equivalent to it: a concept class is PAC learnable if and only if it is weakly learnable.

Moreover, we will now present a meta-algorithm called *AdaBoost* which, given black-box access to any weak learning algorithm for a concept class C will PAC learn C.

Algorithm 1 AdaBoost [40]

```
for $i = 1$ to $m$ do
    $D_1(i) = \frac{1}{m}$
end for
for $t = 1$ to $T$ do
    $h_t$ = hypothesis output by weak learner given distribution $D_t$ on the sample
    $\epsilon_t = \sum_{i=1}^{m} D_t(i)(1 - \delta_{h_t(\mathbf{x}_i),y_i})$
    $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$
    $Z_t = 2\sqrt{\epsilon_t(1 - \epsilon_t)}$
    for $i = 1$ to $m$ do
        $D_{t+1}(i) = \frac{D_t(i)e^{-h_t(\mathbf{x}_i)y_i}}{Z_t}$
    end for
end for
$g = \sum_{t=1}^{T} \alpha_t h_t$
$h = \mathrm{sgn} \circ g$
return $h$
```

For more background on AdaBoost and the theory of boosting in general, we refer the reader to [82].

2.3.3 Convex Surrogate Loss Functions

Even though ERM is a good PAC learning algorithm for any class with finite VC-dimension even in the agnostic case, unfortunately for many important concept classes, including those of hyperplanes and closed balls, ERM cannot be implemented efficiently in the agnostic setting [18].

In this section we will be concerned with the problem of learning hyperplanes. In this setting, the instance space is a real Euclidean space and the concepts are hyperplanes: points on one side of the hyperplane are labeled 1 and points on the other side are labeled 0. For notational convenience, we will identify these concepts with the normal vectors of the hyperplanes, which we will denote as w.

To circumvent the above-mentioned computational hardness results, one can relax the learning goal by changing the loss function. In the definition of (agnostic) PAC learning, the goal was to minimize the expectation of $\ell_{0-1}(h,(x,y)) = 1 - \delta_{y,h(x)}$ where the expectation is taken over the unknown data distribution and δ denotes the Kronecker delta. This function is called the *0-1 loss* function. For the task of learning hyperplanes, we can replace this function by different functions which upper bound it and are easier two optimize. We will introduce two such surrogate loss functions here.

Definition 16 (Hinge loss and logistic loss)**.**

The hinge loss *is* $\ell_{hinge}(w,(x,y)) = \max\{1 - y \cdot \langle w, x\rangle, 0\}$.

The logistic loss *is* $\ell_{logistic}(w,(x,y)) = \ln(1 + e^{-y\langle w,x\rangle})$.

The expectation of the loss function over the data distribution is called *risk*. Note that since both of these functions are greater than or equal to the 0-1 loss, the risk of a hypothesis under any of these loss functions is an upper bound for the misclassification error.

For reasons which we will not discuss here, it is usually not the risk function which is minimized, but a linear combination of the risk function and the L_2

norm of the vector w. (Adding this term in the optimization problem is called *regularization*. The optimization problem is called *Regularized Risk Minimization*.) For the loss functions introduced above, this is a computationally tractable optimization problem.

The method of optimizing regularized hinge loss is called *Support Vector Machine (SVM)*; learning algorithms based on optimizing the logistic loss are called *logistic regression*.

2.3.4 The Kernel Trick

Both SVM and logistic regression can be reformulated into equivalent optimization problems in which the objective function can be expressed as a function of the pairwise inner products of data points and it does not directly depend on the points themselves.

The advantage of this formulation (which we will not present here) is that the standard Euclidean inner product can be replaced in it by an inner product of an implicit transformation of the points. More precisely, for a mapping ψ that maps the instance space $\mathbb{R}^d$ to a different inner product space, the inner product $\langle x, y \rangle$ in the optimization problem can be replaced by $K(x, y) = \langle \psi(x), \psi(y) \rangle$. This so-called *kernel function* can be expressed as a function of x and y without a need to explicitly compute the map ψ, making the optimization task more efficient.

Also, by the so-called *representer theorem* [64, 84], the hyperplane w minimizing the empirical (regularized) risk under either of the above-mentioned loss functions is a linear combination of the data points in the sample. Thus w too can often be concisely represented even if the implicit mapping ψ maps $\mathbb{R}^d$ to a space of much higher dimensionality. This technique is called the *kernel trick*.

We will use the following kernels:

Definition 17 (Kernels)**.**

Linear kernel*:* $K(x, y) = \langle x, y \rangle$*.*

Polynomial kernel*:* $K(x, y) = (1 + \langle x, y \rangle)^d$ *(for some integer parameter d).*

Gaussian kernel*:* $K(x, y) = e^{\frac{-\|x-y\|^2}{2}}$*.*

For more on convex learning problems and kernel methods, we again refer the reader to [72] and [85].

3

The Computational Complexity of MapReduce

This chapter was previously published as Benjamin Fish, Jeremy Kun, Ádám D. Lelkes, Lev Reyzin, György Turán: On the Computational Complexity of MapReduce. *International Symposium on Distributed Computing (DISC)* 2015: 1-15.

3.1 Introduction

MapReduce is a programming model originally developed to separate algorithm design from the engineering challenges of massively distributed computing. A programmer can separately implement a “map” function and a “reduce” function that satisfy certain constraints, and the underlying MapReduce technology handles all the communication, load balancing, fault tolerance, and scaling. MapReduce frameworks and their variants have been successfully deployed in industry by Google [31], Yahoo! [87], and many others.

MapReduce offers a unique and novel model of parallel computation because it alternates parallel and sequential steps, and imposes sharp constraints on communication and random access to the data. This distinguishes MapReduce from classical theoretical models of parallel computation and this, along with its popoularity in industry, is a strong motivation to study the theoretical power of MapReduce. From a theoretical standpoint we ask how MapReduce relates to established complexity classes. From a practical standpoint we ask which problems can be efficiently modeled using MapReduce and which cannot.

In 2010 Karloff et al. [61] initiated a principled theoretical study of MapReduce, providing the definition of the complexity class MRC and comparing it with the classical PRAM models of parallel computing. But to our knowledge, since this initial paper, almost all of the work on MapReduce has focused on algorithmic issues.

In this chapter we prove a result that establishes a connection between MapReduce and space-bounded computation on classical Turing machines. Another traditional question asked by complexity theory is whether increasing the resource bound on a certain computational resource strictly increases the set of solvable problems. Such so-called hierarchy theorems exist for time and space on deterministic and non-deterministic Turing machines, among other settings. In this chapter we prove conditional hierarchy theorems for MapReduce rounds and time.

First we lay a more precise theoretical foundation for studying MapReduce computations (Section 3.3). In particular, we observe that Karloff et al.'s definitions are non-uniform, allowing the complexity class to contain undecidable languages. We reformulate the definition of [61] to make a uniform model and to more finely track the parameters involved. In addition, we point out that our results hold for other important models of parallel computations, including Valiant's Bulk-Synchronous Processing (BSP) model [91] and the Massively Parallel Communication (MPC) model of Beame et al [16]. (Section 3.3.2). We then prove two main theorems: SPACE($o(\log n)$) has constant-round MapReduce computations (Section 3.4) and, conditioned on a version of the Exponential Time Hypothesis, there are strict hierarchies within MRC. In particular, sufficiently increasing time or number of rounds increases the power of MRC (Section 3.5).

Our sub-logarithmic space result is achieved by a direct simulation, using a two-round protocol that localizes state-to-state transitions to the section of the input being simulated, combining the sections in the second round. It is a major open problem whether undirected graph connectivity (a canonical logarithmic-space problem) has a constant-round MapReduce algorithm, and our result is the most general that can be proven without a breakthrough on graph connectivity. Our hierarchy theorem involves proving a conditional time hierarchy within linear space achieved by a padding argument, along with proving a time-and-space upper and lower bounds on simulating MRC machines within P. To the best of our knowledge our hierarchy theorem is the first of its kind. We conclude with a discussion and open questions raised by our work (Section 3.6).

3.2 Background and Previous Work

3.2.1 MapReduce

The MapReduce protocol can be roughly described as follows. The input data is given as a list of key-value pairs, and over a series of rounds two things happen per round: a "mapper" is applied to each key-value pair independently (in parallel), and then for each distinct key a "reducer" is applied to all corresponding values for a group of keys. The canonical example is counting word frequencies with a two-round MapReduce protocol. The inputs are (index, word) pairs, the first mapper maps $(k, v) \mapsto (v, k)$, and the first reducer computes the sum of the word frequencies for the given key. In the second round the mapper sends all data to a single processor via $(k, n_k) \mapsto (1, (k, n_k))$, and the second processor formats the output appropriately.

One of the primary challenges in MapReduce is data locality. MapReduce was designed for processing massive data sets, so MapReduce programs require that every reducer only has access to a substantially sublinear portion of the input, and the strict modularization prohibits reducers from communicating within a round.

All communication happens indirectly through mappers, which are limited in power by the independence requirement. Finally, it's understood in practice that a critical quantity to optimize for is the number of rounds [61], so algorithms which cannot avoid a large number of rounds are considered inefficient and unsuitable for MapReduce.

There are a number of MapReduce-like models in the literature, including the MRC model of Karloff et al. [61], the "mud" algorithms of Feldman et al. [34], Valiant's BSP model [91], the MPC model of Beame et al. [16], and extensions or generalizations of these, e.g. [43]. The MRC class of Karloff et al. is the closest to existing MapReduce computations, and is also among the most restrictive in terms of how it handles communication and tracks the computational power of individual processors. In their influential paper [61], Karloff et al. display the algorithmic power of MRC, and prove that MapReduce algorithms can simulate CREW PRAMs which use subquadratic total memory and processors. It is worth noting that the work of Karloff et al. did not include comparisons to the standard (non-parallel) complexity classes, which is the aim of the present work.

Since [61], there has been extensive work in developing efficient algorithms in MapReduce-like frameworks. For example, Kumar et al. [66] analyze a sampling technique allowing them to translate sequential greedy algorithms into log-round MapReduce algorithms with a small loss of quality. Farahat et al. [33] investigate the potential for sparsifying distributed data using random projections. Kamara and Raykova [55] develop a homomorphic encryption scheme for MapReduce. And much work has been done on graph problems such as connectivity, matchings, sorting, and searching [43]. Chu et al. [25] demonstrate the potential to express any statistical-query learning algorithm in MapReduce. Finally, Sarma et al. [79] explore the relationship between communication costs and the degree to which a computation is parallel in one-round MapReduce problems. Many of these papers pose general upper and lower bounds on MapReduce computations as an open problem, and to the best of our knowledge our results are the first to do so with classical complexity classes.

The study of MapReduce has resulted in a wealth of new and novel algorithms, many of which run faster than their counterparts in classical PRAM models. As such, a more detailed study of the theoretical power of MapReduce is warranted. This chapter contributes to this by establishing a more precise definition of the MapReduce complexity class, proving that it contains sublogarithmic deterministic space, and showing the existence of certain kinds of hierarchies.

3.2.2 Complexity

From a complexity-theory viewpoint, MapReduce is unique in that it combines bounds on time, space and communication. Each of these bounds would be very weak on its own: the total time available to processors is polynomial; the total space and communication are slightly less than quadratic. In particular, even though arranging the communication between processors is one of the most difficult parts of designing MapReduce algorithms, classical results from communication complexity do not apply since the total communication available is more than linear. These innocent-looking bounds lead to serious restrictions when combined, as demonstrated by the fact that it is unknown whether constant-round MRC machines can decide graph connectivity (the best known result achieves a logarithmic number of rounds with high probability [61]), although it is solvable using only logarithmic space on a deterministic Turing machine.

We relate the MRC model to more classical complexity classes by studying simultaneous time-space bounds. We show in Lemma 2 that ETH implies directly a time-space trade-off statement involving time-space complexity classes. This statement is not a well-known complexity theoretic hypothesis, although it is related to the existence of a time hierarchy with a fixed space bound. In fact, as detailed in Section 3.5, a hypothesis weaker than ETH is sufficient for the lemma. The relative strengths of ETH, the weaker hypothesis, and the statement of the lemma seem to be unknown.

3.3 Models

In this section we introduce the model we will use in this chapter, a uniform version of Karloff's MapReduce Class (MRC), and contrast MRC to other models of parallel computation, such as Valiant's Bulk-Synchronous Parallel (BSP) model, for which our results also hold.

3.3.1 MapReduce and MRC

The central piece of data in MRC is the key-value pair, which we denote by a pair of strings $\langle k, v \rangle$, where k is the key and v is the value. An input to an MRC machine is a set of key-value pairs $\langle k_i, v_i \rangle_{i=1}^{N}$ with a total size of $n = \sum_{i=1}^{N} |k_i| + |v_i|$.

The definitions in this subsection are adapted from [61].

Definition 18. *A* mapper μ *is a Turing machine*[*] *which accepts as input a single key-value pair* $\langle k, v\rangle$ *and produces a multiset of key-value pairs* $\langle k'_1, v'_1\rangle, \ldots, \langle k'_s, v'_s\rangle$.

Definition 19. *A* reducer ρ *is a Turing machine which accepts as input a key* k *and a list of values* $\langle v_1, \ldots, v_m\rangle$*, and produces as output a multiset of key-value pairs* $\langle k, v'_1\rangle, \ldots, \langle k, v'_M\rangle$*, with all of the keys equal to the input key* k.

Definition 20. *For a decision problem, an input string* $x \in \{0,1\}^*$ *to an MRC machine is the set of pairs* $\langle i, x_i\rangle_{i=1}^n$ *describing the index and value of each bit. We will denote by* $\langle x\rangle$ *the set* $\langle i, x_i\rangle$.

An MRC machine operates in rounds. In each round, a set of mappers running in parallel first process all the key-value pairs. Then the pairs are partitioned (by a mechanism called "shuffle and sort" that is not considered part of the runtime of an MRC machine) so that each reducer only receives key-value pairs for a single key. Then the reducers process their data in parallel, and the results are merged to form the multiset of key-value pairs for the next round. More formally:

Definition 21. *An* R-round MRC machine *is an alternating sequence of mappers and reducers* $M = (\mu_1, \rho_1, \ldots, \mu_R, \rho_R)$*. The execution of the machine is as follows. For each* $r = 1, \ldots, R$*:*

1. *Let* U_{r-1} *be the multiset of key-value pairs generated by round* $r-1$ *(or the input pairs when* $r = 1$*). Apply* μ_r *to each key-value pair of* U_{r-1} *to get the multiset* $V_r = \bigcup_{\langle k,v\rangle \in U_{r-1}} \mu_r(k, v)$.

[*] The definitions of [61] were for RAMs. However, because we wish to relate MapReduce to classical complexity classes, we reformulate the definitions here in terms of Turing machines.

2. *Shuffle-and-sort groups the values by key. Call each of the pieces* $V_{k,r} = (k, (v_{k,1}, \dots, v_{k,s_k}))$.

3. *Assign a different copy of reducer* ρ_r *to each* $V_{k,r}$ *(run in parallel) and set* $U_r = \bigcup_k \rho_r(V_{k,r})$.

The output is the final set of key-value pairs. For decision problems, we define M to accept $\langle x \rangle$ if in the final round $U_R = \emptyset$. Equivalently we may give each reducer a special accept state and say the machine accepts if at any time any reducer enters the accept state. We say M *decides* a language L if it accepts $\langle x \rangle$ if and only if $x \in L$.

The central caveat that makes MRC an interesting class is that the reducers have space constraints that are sublinear in the size of the input string. In other words, no sequential computation may happen that has random access to the entire input. Thinking of the reducers as processors, cooperation between reducers is obtained not by message passing or shared memory, but rather across rounds in which there is a global communication step.

In the MRC model we use in this chapter, we require that every mapper and reducer arise as separate runs of the same Turing machine M. Our Turing machine $M(m, r, R, y)$ will accept as input the current round number r, a bit m denoting whether to run the r-th map or reduce function, the total number of rounds R, and the corresponding input y. Equivalently, we can imagine a sequence of mappers and reducers in each round $\mu_1, \rho_1, \mu_2, \rho_2, \dots$, where the descriptions of the μ_i, ρ_i are computable in polynomial time in $\log i$.

Definition 22 (Uniform Deterministic MRC)**.** *A language* L *is said to be in*

MRC$[f(n), g(n)]$ *if there is a constant* $0 < c < 1$, *an* $O(n^c)$*-space and* $O(g(n))$*-time Turing machine* $M(m, r, n, y)$, *and an* $R = O(f(n))$, *such that for all* $x \in \{0,1\}^n$, *the following holds.*

1. *Letting* $\mu_r = M(1, r, n, -), \rho_r = M(0, r, n, -)$, *the MRC machine* $M_R = (\mu_1, \rho_1, \ldots, \mu_R, \rho_R)$ *accepts* x *if and only if* $x \in L$.

2. *Each* μ_r *outputs* $O(n^c)$ *distinct keys.*

This definition closely hews to practical MapReduce computations: $f(n)$ represents the number of times global communication has to be performed, $g(n)$ represents the time each processor gets, and sublinear space bounds in terms of $n = |x|$ ensure that the size of the data on each processor is smaller than the full input.

Remark 1. *By* $M(1, r, n, -)$, *we mean that the tape of* M *is initialized by the string* $\langle 1, r, n \rangle$. *In particular, this prohibits an MRC algorithm from having* $2^{\Omega(n)}$ *rounds; the space constraints would prohibit it from storing the round number.*

Remark 2. *Note that a polynomial time Turing machine with sufficient time can trivially simulate a uniform MRC machine. All that is required is for the machine to perform the key grouping manually, and run the MRC machine as a subroutine. As such,* MRC$[poly(n), poly(n)] \subseteq P$. *We give a more precise computation of the amount of overhead required in the proof of Lemma 3.*

Definition 23. *Define by* MRC^i *the union of uniform MRC classes*

$$\mathrm{MRC}^i = \bigcup_{k \in \mathbb{N}} \mathrm{MRC}[\log^i(n), n^k].$$

So in particular $\mathrm{MRC}^0 = \bigcup_{k \in \mathbb{N}} \mathrm{MRC}[1, n^k]$.

A complexity class is generally called uniform if the descriptions of the machines solving problems in it do not depend on the input length. Classical complexity classes defined by Turing machines with resource bounds, such as P, NP, and L, are uniform. On the other hand, circuit complexity classes are naturally nonuniform since a fixed Boolean circuit can only accept inputs of a single length. There is ambiguity about the uniformity of MRC as defined in [61]. Since we wish to relate the MRC model to classical complexity classes such as P and L, making sure that the model is uniform is crucial. Indeed, innocuous-seeming changes to the definitions above introduce nonuniformity.

We will now show that the original MRC definition of [61] allows MRC machines to decide undecidable languages. This definition required a polylogarithmic number of rounds, and also allowed completely different MapReduce machines for different rounds. For simplicity's sake, we will allow a linear number of rounds, and use our notation $\mathrm{MRC}[f(n), g(n)]$ to denote an MRC machine that operates in $O(f(n))$ rounds and each processor gets $O(g(n))$ time per round. In particular, we show that nonuniform $\mathrm{MRC}[n, \sqrt{n}]$ accepts all unary languages, i.e. languages of the form $L \subseteq \{1^n \mid n \in \mathbb{N}\}$.

Lemma 1. *Let L be a unary language. Then L is in nonuniform* $\mathrm{MRC}[n, \sqrt{n}]$.

Proof. We define the mappers and reducers as follows. Let μ_1 distribute the input as contiguous blocks of $\sqrt{n}$ bits, ρ_1 compute the length of its input, μ_2 send the counts to a single processor, and ρ_2 add up the counts, i.e. find $n = |x|$ where x is the input. Now the input data is reduced to one key-value pair $\langle \star, n \rangle$. Then let ρ_i

for $i \geq 3$ be the reducer that on input $\langle \star, i-3 \rangle$ accepts if and only if $1^{i-3} \in L$ and otherwise outputs the input. Let μ_i for $i \geq 3$ send the input to a single processor. Then ρ_{n+3} will accept iff x is in L. Note that ρ_1, ρ_2 take $O(\sqrt{n})$ time, and all other mappers and reducers take $O(1)$ time. All mappers and reducers are also in SPACE($\sqrt{n}$). □

In particular, Lemma 1 implies that nonuniform MRC[$n, \sqrt{n}$] contains the unary version of the halting problem. A more careful analysis shows all unary languages are even in MRC[$\log n, \sqrt{n}$], by having ρ_{i+3} check 2^i strings for membership in L.

3.3.2 Other Models of Parallel Computation

Several other models of parallel computation have been introduced, including the BSP model of Valiant [91] and the MPC model of Beame et al. [16]. The main difference between BSP and MapReduce is that in the BSP model the key-value pairs and the shuffling steps needed to redistribute them are replaced with point-to-point messages. Similarly to [61], in Valiant's paper [91] there is also ambiguity about the uniformity of the model. In this chapter, when we refer to BSP we mean a uniform deterministic version of the model. For completeness, we include the exact definition here.

A BSP machine with p processors is a sequence $(M_1, \ldots, M_p)$ of p Turing machines which on any input, output a list $((j_1, y_1), (j_2, y_2), \ldots, (j_m, y_m))$ of messages to be sent to other processors in the next round. Specifically, message y_k is sent to prcessor j_k. A BSP machine operates in rounds as follows. In the first round the input is partitioned into equal-sized pieces $x_{1,0}, \ldots, x_{p,0}$ and distributed arbitrarily to the processors. Then for rounds $r = 1, \dots, R$,

1. Each processor i takes $x_{i,r}$ as input and computes some number s_i of messages $M_i(x_{i,r}) = \{(j_{i,k}, y_{i,k}) : k = 1, \dots, s_i\}$.

2. Set $x_{i,r+1}$ to be the set of all messages sent to i (as with MRC's shuffle-and-sort, this is not considered part of processor i's runtime).

We say the machine *accepts* a string x if any machine accepts at any point before round R finishes. We now define uniform deterministic BSP analogously to MRC.

Definition 24 (Uniform Deterministic BSP)**.** *A language L is said to be in* $\mathrm{BSP}[f(n), g(n)]$ *if there is a constant* $0 < c < 1$*, an $O(n^c)$-space and $O(g(n))$-time Turing machine $M(p, y)$, and an $R = O(f(n))$, such that for all $x \in \{0,1\}^n$, the following holds: letting $M_i = M(i, -)$, the BSP machine $M = (M_1, M_2, \dots, M_{n^c})$ accepts x in R rounds if and only if $x \in L$.*

Remark 3. *As with MRC, we count the size and number of each message as part of the space bound of the machine generating/receiving the messages. Differing slightly from Valiant, we do not provide persistent memory for each processor. Instead, persistent memory can be simulated by processors sending messages to themselves. This is without loss of generality since we are not concerned with the cost of sending individual messages.*

Goodrich et al. [43] and Pace [74] showed that MapReduce computations can be simulated in the BSP model and vice versa, with only a constant blow-up in the computational resources needed. This implies that our theorems about MapReduce automatically apply to BSP.

Similarly, the MPC model uses point-to-point messages and Beame et al.'s paper [16] does not discuss the uniformity of the model. The main distinguishing characteristic of the MPC model is that it introduces the number of processors p as an explicit parameter. Setting $p = O(n^c)$, our results will also hold in this model.

There are other variants of these models, including the model that Andoni et. al. [2] uses, which follows the MPC model but also introduces the additional constraint that total space used across each round must be no more than $O(n)$. It is straightforward to check that the proofs of our results never use more than $O(n)$ space, implying that our results hold even under this more restrictive model.

3.4 Space Complexity Classes in MRC^0

In this section we prove that small space classes are contained in constant-round MRC. Again, the results in this section also hold for other similar models of parallel computation, including the BSP model and the MPC model. First, we prove that the class REGULAR of regular languages is in MRC^0. It is well known that SPACE($O(1)$) = REGULAR [86], and so this result can be viewed as a warm-up to the theorem that SPACE($o(\log n)$) $\subseteq$ MRC^0. Indeed, both proofs share the same flavor, which we sketch before proceeding to the details.

We wish to show that any given DFA can be simulated by an MRC^0 machine. The simulation works as follows: in the first round each parallel processor receives a contiguous portion of the input string and constructs a state transition function using the data of the globally known DFA. Though only the processor with the beginning of the string knows the true state of the machine during its portion of the input, all processors can still compute the *entire* table of state-to-state transitions for the given portion of input. In the second round, one processor collects the transition tables and chains together the computations, and this step requires only the first bit of input and the list of tables.

We can count up the space and time requirements to prove the following theorem.

Theorem 2. REGULAR $\subsetneq$ MRC^0

Proof. Let L be a regular language and D a deterministic finite automaton recognizing L. Define the first mapper so that the j^{th} processor has the bits from $\lfloor j\sqrt{n}\rfloor$ to $\lfloor (j+1)\sqrt{n}\rfloor - 1$. This means we have $K = O(\sqrt{n})$ processors in the first round. Because the description of D is independent of the size of the input string, we also assume each processor has access to the relevant set of states S and the transition function $t : S \times \{0,1\} \to S$.

We now define ρ_1. Fix a processor j and call its portion of the input y. The processor constructs a table T_j of size at most $|S|^2 = O(1)$ by simulating D on y starting from all possible states and recording the state at the end of the simulation. It then passes T_j to the single processor in the second round.

In the second round the sole processor has K tables T_j. Treating T_j as a function mapping states to states, this processor computes $q = T_K(\dots T_2(T_1(\mathit{initial})))$ where *initial* denotes the initial state of D, and accepts if and only if q is an accepting state. This requires $O(\sqrt{n})$ space and time and proves containment. To show this is strict, inspect the prototypical problem of deciding whether the majority of bits in the input are 1's. □

Remark 4. *While the definition of* MRC^0 *includes languages with time complexity* $O(n^k)$ *for all* $k \geq 0$*, our Theorem 2 is more efficient than the definition implies: we show that regular languages can be computed in* MRC^0 *in time and space* $O(\sqrt{n})$*, with the option of a trade-off between time* n^ε *and space* $n^{1-\varepsilon}$*.*

One specific application of this result is that for any given regular expression, a two-round MapReduce computation can decide if a string matches that regular expression, even if the string is so long that any one machine can only store n^ϵ bits of it.

We now move on to prove SPACE$(o(\log n)) \subseteq$ MRC^0. It is worth noting that this is a strictly stronger statement than Theorem 2. That is, REGULAR = SPACE$(O(1)) \subsetneq$ SPACE$(o(\log n))$. Several non-trivial examples of languages that witness the strictness of this containment are given in [89].

The proof is very similar to the proof of Theorem 2: Instead of the processors computing the entire table of state-to-state transitions of a DFA, the processors now compute the entire table of all transitions possible among the configurations of the work tape of a Turing machine that uses $o(\log n)$ space.

Theorem 3. SPACE$(o(\log n)) \subseteq$ MRC^0.

Proof. Let L be a language in SPACE$(o(\log n))$ and T a Turing machine recognizing L in polynomial time and $o(\log(n))$ space, with a read/write work tape

W. Define the first mapper so that the j^{th} processor has the bits from $\lfloor j\sqrt{n}\rfloor$ to $\lfloor (j+1)\sqrt{n}\rfloor - 1$. Let $\mathcal{C}$ be the set of all possible configurations of W and let S be the states of T. Since the size of S is independent of the input, we can assume that each processor has the transition function of T stored on it.

Now we define ρ_1 as follows: Each processor j constructs the graph of a function $T_j : \mathcal{C} \times \{L, R\} \times S \to \mathcal{C} \times \{L, R\} \times S$, which simulates T when the read head starts on either the left or right side of the jth $\sqrt{n}$ bits of the input and W is in some configuration from $\mathcal{C}$. It outputs whether the read head leaves the y portion of the read tape on the left side, the right side, or else accepts or rejects. To compute the graph of T_j, processor j simulates T using its transition function, which takes polynomial time.

Next we show that the graph of T_j can be stored on processor j by showing it can be stored in $O(\sqrt{n})$ space. Since W is by assumption size $o(\log n)$, each entry of the table is $o(\log n)$, so there are $2^{o(\log n)}$ possible configurations for the tape symbols. There are also $o(\log n)$ possible positions for the read/write head, and a constant number of states T could be in. Hence $|\mathcal{C}| = 2^{o(\log n)} o(\log n) = o(n^{1/3})$. Then processor j can store the graph of T_j as a table of size $O(n^{1/3})$.

The second map function μ_2 sends each T_j (there are $\sqrt{n}$ of them) to a single processor. Each is size $O(n^{1/3})$, and there are $\sqrt{n}$ of them, so a single processor can store all the tables. Using these tables, the final reduce function can now simulate T from starting state to either the accept or reject state by computing $q = T_k^*(\ldots T_2^*(T_1^*(\emptyset, L, \mathit{initial})))$ for some k, where $\emptyset$ denotes the initial configuration of T, *initial* is the initial state of T, and q is either in the accept or reject state.

Note T_j^* is the modification of T_j such that if $T_j(x)$ outputs L, then $T_j^*(x)$ outputs R and vice versa. This is necessary because if the read head leaves the j^{th} $\sqrt{n}$ bits to the right, it enters the $j+1^{\text{th}}$ $\sqrt{n}$ bits from the left, and vice versa. Finally, the reducer accepts if and only if q is in an accept state.

This algorithm successfully simulates T, which decides L, and only takes a constant number of rounds, proving containment. □

3.5 Hierarchy Theorems

In this section we prove two main results (Theorems 4 and 5) about hierarchies within MRC relating to increases in time and rounds. They imply that allowing MRC machines sufficiently more time or rounds strictly increases the computing power of the machines. The first theorem states that for all α, β there are problems $L \notin \text{MRC}[n^\alpha, n^\beta]$ which can be decided by *constant time* MRC machines when given enough extra rounds.

Theorem 4. *Suppose the ETH holds with constant c. Then for every $\alpha, \beta \in \mathbb{N}$ there exists a $\gamma = O(\alpha + \beta)$ such that*

$$\text{MRC}[n^\gamma, 1] \not\subseteq \text{MRC}[n^\alpha, n^\beta].$$

The second theorem is analogous for time, and says that there are problems $L \notin \text{MRC}[n^\alpha, n^\beta]$ that can be decided by a *one round* MRC machine given enough extra time.

Theorem 5. *Suppose the ETH holds with constant c. Then for every $\alpha, \beta \in \mathbb{N}$ there exists a $\gamma = O(\alpha + \beta)$ such that*

$$\mathrm{MRC}[1, n^\gamma] \not\subseteq \mathrm{MRC}[n^\alpha, n^\beta].$$

As both of these theorems depend on the ETH, we first prove a complexity-theoretic lemma that uses the ETH to give a time-hierarchy within linear space TISP. Recall that TISP is the complexity class defined by simultaneous time and space bounds. The lemma can also be described as a time-space trade-off. For some $b > a$ we prove the existence of a language that can be decided by a Turing machine with simultaneous $O(n^b)$ time and linear space, but cannot be decided by a Turing machine in time $O(n^a)$ even without any space restrictions. It is widely believed such languages exist for *exponential* time classes (for example, TQBF, the language of true quantified Boolean formulas, is a linear space language which is PSPACE-complete). We ask whether such trade-offs can be extended to polynomial time classes, and this lemma shows that indeed this is the case.

Lemma 2. *Suppose that the ETH holds with constant c. Then for any positive integer a there exists a positive integer $b > a$ such that*

$$\mathrm{TIME}(n^a) \not\subseteq \mathrm{TISP}(n^b, n).$$

Proof. By the ETH, 3-SAT $\in \mathrm{TISP}(2^n, n) \setminus \mathrm{TIME}(2^{cn})$. Let $b := \lceil \frac{a}{c} \rceil + 2$, $\delta := \frac{1}{2}(\frac{1}{b} + \frac{c}{a})$. Pad 3-SAT with $2^{\delta n}$ zeros and call this language L, i.e. let $L := \{x0^{2^{\delta|x|}} \mid x \in \text{3-SAT}\}$. Let $N := n + 2^{\delta n}$. Then $L \in \mathrm{TISP}(N^b, N)$ since $N^b > 2^n$. On the other hand, assume for contradiction that $L \in \mathrm{TIME}(N^a)$. Then, since $N^a < 2^{cn}$, it follows that 3-SAT $\in \mathrm{TIME}(2^{cn})$, contradicting the ETH. □

There are a few interesting complexity-theoretic remarks about the above proof. First, the starting language does not need to be 3-SAT, as the only assumption we needed was its hypothesized time lower bound. We could relax the assumption to the hypothesis that there exists a $c > 0$ such that TQBF, the PSPACE-complete language of true quantified Boolean formulas, requires 2^{cn} time, or further still to the following complexity hypothesis.

Conjecture 2. *There exist* c', c *satisfying* $0 < c' < c < 1$ *such that* $\mathrm{TISP}(2^n, 2^{c'n}) \setminus \mathrm{TIME}(2^{cn}) \neq \emptyset$.

Second, since $\mathrm{TISP}(n^a, n) \subseteq \mathrm{TIME}(n^a)$, this conditionally proves the existence of a hierarchy within $\mathrm{TISP}(\mathrm{poly}(n), n)$. We note that finding time hierarchies in fixed-space complexity classes was posed as an open question by [94], and so removing the hypothesis or replacing it with a weaker one is an interesting open problem.

Using this lemma we can prove Theorems 4 and 5. The proof of Theorem 4 depends on the following lemma.

Lemma 3. *For every* $\alpha, \beta \in \mathbb{N}$ *the following holds:*

$$\mathrm{TISP}(n^\alpha, n) \subseteq \mathrm{MRC}[n^\alpha, 1] \ \subseteq \mathrm{MRC}[n^\alpha, n^\beta] \subseteq \mathrm{TISP}(n^{\alpha+\beta+2}, n^2).$$

Proof. The first inequality follows from a simulation argument similar to the proof of Theorem 3. The MRC machine will simulate the $\mathrm{TISP}(n^\alpha, n)$ machine by making one step per round, with the tape (including the possible extra space needed on the work tape) distributed among the processors. The position of the tape is passed between the processors from round to round. It takes constant time to simulate one step of the $\mathrm{TISP}(n^\alpha, n)$ machine, thus in n^α rounds we can simulate all steps. Also, since the machine uses only linear space, the simulation can be done with $O(\sqrt{n})$ processors using $O(\sqrt{n})$ space each. The second inequality is trivial.

The third inequality is proven as follows. Let $T(n) = n^{\alpha+\beta+2}$. We first show that any language in $\mathrm{MRC}[n^\alpha, n^\beta]$ can be simulated in time $O(T(n))$, i.e. $\mathrm{MRC}[n^\alpha, n^\beta] \subseteq \mathrm{TIME}(T(n))$. The r-th round is simulated by applying μ_r to each key-value pair in sequence, shuffle-and-sorting the new key-value pairs, and then applying ρ_r to each appropriate group of key-value pairs sequentially. Indeed, $M(m, r, n, -)$ can be simulated naturally by keeping track of m and r, and adding n to the tape at the beginning of the simulation. Each application of μ_r takes $O(n^\beta)$ time, for a total of $O(n^{\beta+1})$ time. Since each mapper outputs no more than $O(n^c)$ keys, and each mapper and reducer is in $\mathrm{SPACE}(O(n^c))$, there are no more than $O(n^2)$ keys to sort. Then shuffle-and-sorting takes $O(n^2 \log n)$ time, and the applications of ρ_r also take $O(n^{\beta+1})$ time. So a round takes $O(n^{\beta+1} + n^2 \log n)$ time. Note that keeping track of m,r, and n takes no more than the above time. So over $O(n^\alpha)$ rounds, the simulation takes $O(n^{\alpha+\beta+1} + n^{\alpha+2} \log(n)) = O(T(n))$ time. □

Now we prove Theorem 4.

Proof. By Lemma 2, there is a language L in $\mathrm{TISP}(n^\gamma, n) \backslash \mathrm{TIME}(n^{\alpha+\beta+2})$ for some

γ. By Lemma 3, $L \in \mathrm{MRC}[n^\gamma, 1]$. On the other hand, because $L \notin \mathrm{TIME}(n^{\alpha+\beta+2})$ and $\mathrm{MRC}[n^\alpha, n^\beta] \subseteq \mathrm{TIME}(n^{\alpha+\beta+2})$, we can conclude that $L \notin \mathrm{MRC}[n^\alpha, n^\beta]$. □

Next, we prove Theorem 5 using a padding argument.

Proof. Let $T(n) = n^{\alpha+\beta+2}$ as in Lemma 3. By Lemma 2, there is a γ such that $\mathrm{TISP}(n^\gamma, n) \setminus \mathrm{TIME}(T(n^2))$ is nonempty. Let L be a language from this set. Pad L with n^2 zeros, and call this new language L', i.e. let $L' = \{x0^{|x|^2} \mid x \in L\}$. Let $N = n + n^2$. There is an $\mathrm{MRC}[1, N^\gamma]$ algorithm to decide L': the first mapper discards all the key-value pairs except those in the first n, and sends all remaining pairs to a single reducer. The space consumed by all pairs is $O(n) = O(\sqrt{N})$. This reducer decides L, which is possible since $L \in \mathrm{TISP}(n^\gamma, n)$. We now claim L' is not in $\mathrm{MRC}[N^\alpha, N^\beta]$. If it were, then L' would be in $\mathrm{TIME}(T(N))$. A Turing machine that decides L' in $T(N)$ time can be modified to decide L in $T(N)$ time: pad the input string with n^2 ones and use the decider for L'. This shows L is in $\mathrm{TIME}(T(n^2))$, a contradiction. □

We conclude by noting explicitly that Theorems 4, 5 give proper hierarchies within MRC, and that proving certain stronger hierarchies imply the separation of L and P.

Corollary 1. *Suppose the ETH. For every α, β there exist $\mu > \alpha$ and $\nu > \beta$ such that*

$$\mathrm{MRC}[n^\alpha, n^\beta] \subsetneq \mathrm{MRC}[n^\mu, n^\beta]$$

and

$$\mathrm{MRC}[n^\alpha, n^\beta] \subsetneq \mathrm{MRC}[n^\alpha, n^\nu].$$

Proof. By Theorem 5, there is some $\mu > \alpha$ such that $\mathrm{MRC}[n^\mu, 1] \not\subseteq \mathrm{MRC}[n^\alpha, n^\beta]$. It is immediate that $\mathrm{MRC}[n^\alpha, n^\beta] \subseteq \mathrm{MRC}[n^\mu, n^\beta]$ and also that $\mathrm{MRC}[n^\mu, 1] \subseteq \mathrm{MRC}[n^\mu, n^\beta]$. So $\mathrm{MRC}[n^\alpha, n^\beta] \neq \mathrm{MRC}[n^\mu, n^\beta]$. The proof of the second claim is similar. □

Corollary 2. *If* $\mathrm{MRC}[\mathrm{poly}(n), 1] \subsetneq \mathrm{MRC}[\mathrm{poly}(n), \mathrm{poly}(n)]$*, then it follows that* $\mathrm{L} \neq \mathrm{P}$.

Proof.

$$\begin{aligned} \mathrm{L} &\subseteq \mathrm{TISP}(\mathrm{poly}(n), \log n) \subseteq \mathrm{TISP}(\mathrm{poly}(n), n) \subseteq \mathrm{MRC}[\mathrm{poly}(n), 1] \\ &\subseteq \mathrm{MRC}[\mathrm{poly}(n), \mathrm{poly}(n)] \subseteq \mathrm{P}. \end{aligned}$$

The first containment is well known, the third follows from Lemma 3, and the rest are trivial. □

Corollary 2 is interesting because if any of the containments in the proof are shown to be proper, then $\mathrm{L} \neq \mathrm{P}$. Moreover, if we provide MRC with a polynomial number of rounds, Corollary 2 says that determining whether time provides substantially more power is at least as hard as separating L from P. On the other hand, it does not rule out the possibility that $\mathrm{MRC}[\mathrm{poly}(n), \mathrm{poly}(n)] = \mathrm{P}$, or even that $\mathrm{MRC}[\mathrm{poly}(n), 1] = \mathrm{P}$.

3.6 Conclusion

In this chapter we established the first general connections between MapReduce and classical complexity classes, and showed the conditional existence of a hierarchy within MapReduce. Our results also apply to variants of MapReduce, most notably Valiant's BSP model.

Our work suggests some natural open problems. How does MapReduce relate to other complexity classes, such as the circuit class uniform AC^0? Can one improve the bounds from Corollary 1 or remove the dependence on Hypothesis 2? Does Lemma 2 imply Hypothesis 2? Can one give explicit hierarchies for space or time alone, e.g. $\mathrm{MRC}[n^\alpha, \mathrm{poly}(n)] \subsetneq \mathrm{MRC}[n^\mu, \mathrm{poly}(n)]$?

We also ask whether $\mathrm{MRC}[\mathrm{poly}(n), \mathrm{poly}(n)] = \mathrm{P}$. In other words, if a problem has an efficient solution, does it have one with using data locality? A negative answer implies $\mathrm{L} \neq \mathrm{P}$ which is a major open problem in complexity theory, and a positive answer would likely provide new and valuable algorithmic insights. Finally, while we have focused on the relationship between rounds and time, there are also implicit parameters for the amount of (sublinear) space per processor, and the (sublinear) number of processors per round. A natural complexity question is to ask what the relationship between all four parameters are.

3.6.1 Subsequent Results

In a paper by Roughgarden et al. [78], published after our paper [37], lower bounds are proved for a different but related model of MapReduce. In this model, which the authors named s-Shuffle, the computational limits of the MapReduce processors are abstracted away and the focus is on the communication patterns and the amount of bits received by each machine. In particular, the machines are arranged in a layered circuit-like structure where the "fan-in" of each machine is limited to at most s bits. Our setting would correspond to $s = O(n^c)$ for $c < 1$.

Since the computational power of each node is unrestricted in the s-Shuffle model, it is strictly more powerful than our MRC model; in particular, it is proved in the paper that any function on n-bit inputs can be computed in $\lceil \log_s n \rceil$ rounds. Nevertheless, the authors can prove a lower bound against this model by showing that r-round s-Shuffle computations can be represented by polynomials of degree at most s^r. Equivalently, if some output bit of a function cannot be represented by a polynomial of degree at most d, that yields a round lower bound of $\lceil \log_s d \rceil$.

To use this theorem to prove lower bounds for specific problems, the authors prove asymptotic polynomial degree lower bounds for monotone graph properties in general, and exact bounds for several versions of the connectivity problem. Since these bounds are polynomial in n, for $s = n^{\Omega(1)}$ the resulting round lower bounds are at most constant; thus these results do not resolve the open problem about the round complexity of connectivity in the MRC model. However, if the fan-in is restricted to $s = n^{o(1)}$, these theorems yield superconstant round lower bounds.

Perhaps even more interestingly, Roughgarden et al. [78] prove a barrier for stronger round lower bounds. In particular, they show that a superconstant round lower bounds for connectivity for a reasonable model of MapReduce computations in which the fan-in restriction is only $n^{\Omega(1)}$ and there is a polynomial number of machines, would imply a separation between NC^1 and P, a longstanding open problem in complexity theory. For more details, we refer the reader to [78].

4

Interactive Clustering

This chapter was previously published as Ádám D. Lelkes, Lev Reyzin: Interactive Clustering of Linear Classes and Cryptographic Lower Bounds. *International Conference on Algorithmic Learning Theory (ALT)* 2015: 165-176.

4.1 Introduction

In this chapter we consider the interactive clustering model proposed by Balcan and Blum [11]. This clustering (and learning) model allows the algorithm to issue proposed explicit clusterings to an oracle, which replies by requesting two of the proposed clusters "merge" or that an impure cluster be "split." This model captures an interactive learning scenario, where one party has a target clustering in mind and communicates this information via these simple requests.

Balcan and Blum [11] give the example of a human helping a computer cluster news articles by topic by indicating to the computer which proposed different clusters are really about the same topic and which need to be split. Another motivating example is computer-aided image segmentation, where an algorithm can propose image segmentations to a human, who can show the computer which clusters need to be "fixed up" – this is likely to be much simpler than having the human segment the image manually.

Many interesting results are already known for this model [10, 11], including the learnability of various concept classes and some generic, though inefficient, algorithms (for an overview, see Sect. 4.2.2).

In this chapter we extend the theory of interactive clustering. Among our main results:

- We show efficient algorithms for clustering parities and, more generally, linear functionals over finite fields – parities are a concept class of central importance in most models of learning. (Section 4.3.1)

- We also give an efficient algorithm for clustering hyperplanes, a generalization of linear functionals over $\mathbb{R}^d$. These capture a large and important set of concept classes whose efficient clusterability was not known in this model. (Section 4.3.2)

- We prove lower bounds for the interactive clustering model under plausible cryptographic assumptions, further illustrating the richness of this model. (Section 4.4)

4.2 Background and Previous Work

4.2.1 The Model

In this section we describe the interactive clustering model of Balcan and Blum [11]. In this model of clustering, no distributional assumptions are made about the data; instead, it is assumed that the teacher knows the target clustering, but it is infeasible for him to label each data point by hand. Thus the goal of the learner is to learn the target clustering by making a small number of queries to the teacher. In this respect, the model is similar to the foundational query learning models introduced by Angluin [3]. (As a consequence, the classes we consider in this chapter might be more familiar from the theory of query learning than from the usual models of clustering.)

More specifically, the learner is given a sample S of m points, and knows the number of target clusters which is denoted as k. The target clustering is an element of a concept class C. In each round, the learner presents a hypothesis clustering to the teacher. The answer of the teacher to this query is one of the following: either that the hypothesis clustering is correct, or a `split` or `merge` request. If this hypothesis is incorrect, that means that at least one of the following two cases has to hold: either there are *impure* hypothesis clusters, i.e. hypothesis clusters which contain points from more than one target cluster, or there are more than one distinct hypothesis clusters that are subsets of the same cluster. In the first case, the teacher can issue a `split` request to an impure cluster, in the second case the teacher can issue a `merge` request to two clusters that are both subsets of the same target cluster. A `split` request only communicates the information that the the given hypothesis cluster is not pure. It does not provide any additional information. If there are several valid possibilities for `split` or `merge` requests, the teacher can arbitrarily choose one of them.

Definition 25. *An interactive clustering algorithm is called* ***efficient*** *if it runs in* $O(\text{poly}(k, m, \log |C|))$ *time and makes* $O(\text{poly}(k, \log m, \log |C|))$ *queries.*

Observe that allowing the learner to make m queries would make the clustering task trivial: by starting from the all singleton hypothesis clustering and merging clusters according to the teacher's requests, the target clustering can be found in at most m rounds.

4.2.2 Previous Work

Extensive research on clustering has yielded a plethora of important theoretical results, including traditional hardness results [44, 52], approximation algorithms [5, 7, 14, 24, 65, 30], and generative models [21, 28]. More recently researchers have examined properties of data that imply various notions of "clusterability" [1]. An ongoing research direction has been to find models that capture real-world behavior and success of clustering algorithms, in which many foundational open problems remain [17].

Inspired by models where clusterings satisfy certain natural relations with the data, e.g. [12], Balcan and Blum [11] introduced the notion of interactive clustering we consider in this chapter – the data assumption here, of course, is that a "teacher" has a clustering in mind that the data satisfies, while the algorithm is aware of the space of possible clusterings.

In addition to defining the interactive clustering model, Balcan and Blum [11] gave some of the first results for it. In particular, they showed how to efficiently cluster intervals, disjunctions, and conjunctions (the latter only for constant k). Moreover, they gave a general, but inefficient, version space algorithm for clustering any finite concept class using $O(k^3 \log |C|)$ queries. They also gave a lower bound that showed efficient clustering was not possible if if the algorithm is required to be proper, i.e. produce k-clusterings to the teacher. These results contrast with our cryptographic lower bounds, which hold for arbitrary hypothesis clusterings.

Awasthi and Zadeh [10] later improved the generic bound of $O(k^3 \log |C|)$ to $O(k \log |C|)$ queries using a simpler version space algorithm. They presented an algorithm for clustering axis-aligned rectangles.

Awasthi and Zadeh [10] also introduced a noisy variant of this model. In the noisy version, split requests are still only issued for impure clusters, but `merge` requests might have "noise": a `merge` request might be issued if at least an η fraction of the points from both hypothesis clusters belong to the same target cluster. Alternatively, a stricter version of the noisy model allows arbitrary noise: the teacher might issue a `merge` request for two clusters even if they both have only one point from some target cluster. Awasthi and Zadeh [10] gave an example of a concept class that cannot be learned with arbitrary noise, and presented an algorithm for clustering intervals in the η noise model. To the best of our knowledge, our algorithm for clustering linear functionals over finite fields, presented in Sect. 4.3.1, is the first algorithm for clustering a nontrivial concept class under arbitrary noise.

Other interactive models of clustering have, of course, also been considered [15, 29]. In this chapter, however, we keep our analysis to the Balcan and Blum [11] interactive model.

4.3 Interactive Clustering Algorithms

4.3.1 Clustering Linear Functionals

In this section we present an algorithm for clustering linear functionals over finite fields. That is, the instance space is $X = GF(q)^n$ for some prime power q and positive integer n, where $GF(q)$ denotes the finite field of order q. The concept class is the dual space $(GF(q)^n)^*$ of linear operations mapping from $GF(q)^n$ to $GF(q)$. Thus the number of clusters is $k = q$. Recall that every linear functional in $(GF(q)^n)^*$ is of the form $v \mapsto x \cdot v$, thus clustering linear functionals is equivalent to learning this unknown vector x. For the special case of $q = 2$, we get the concept class of parity functions over $\{0,1\}^n$, where there are two classes/clusters (for the positively and negatively labeled points).

The idea of the algorithm is the following: in each round we output the largest sets of elements that are already known to be pure, thus forcing the teacher to make a `merge` request. A `merge` request for two clusters will yield a linear equation for the target vector which is independent from all previously learned equations.

We use a graph on the data points to keep track of the learned linear equations. Since the algorithm learns an independent equation in each round, it finds the target vector in at most n rounds. The description of the algorithm follows.

Algorithm 2 Cluster-Functional

```
initialize $G = (V, \emptyset)$, with $|V| = m$, each vertex corresponding an element from the sample.
initialize $Q = \emptyset$.
repeat
  find the connected components of $G$ and output them as clusters.
  on a merge request to two clusters:
  for each pair $a, b$ of points in the union do
    if $(a - b) \cdot x = 0$ is independent from all equations in $Q$ then
      add $(a - b) \cdot x = 0$ to $Q$.
    end if
  end for
  for each non-edge $(a, b)$, add $(a, b)$ to $G$ if $(a - b) \cdot x = 0$ follows from the equations in $Q$.
until the target clustering is found
```

Theorem 6. *Algorithm 2 finds the target clustering using at most n queries and $O(m^2n^4)$ time. Moreover, the query complexity of the algorithm is optimal: every clustering algorithm needs to make at least n queries to find the target clustering.*

Proof. We claim that in each round we learn a linear equation that is independent from all previously learned equations, thus in n rounds we learn the target functional.

Assume for contradiction that there is a round where no independent equations are added. All hypothesis clusters are pure by construction so they can never be split. If two clusters are merged, then let us pick an element a from one of them and b from the other. Then $(a-b)\cdot x = 0$ has to be independent from Q since otherwise the edge (a, b) would have been added in a previous round and the two elements would thus belong to the same cluster.

Thus after at most n rounds G will consist of k marked cliques which will give the correct clustering. Finding the connected components and outputting the hypothesis clusters takes linear time. To update the graph, $O(m^2)$ Gaussian elimination steps are needed. Hence the total running time is $O(m^2n^4)$.

To show that at least n queries are necessary, notice that `merge` and `split` requests are equivalent to linear equations and inequalities, respectively. Since the dimension of the dual space is n, after less than n queries there are at least two linearly independent linear functionals, and therefore at least two different clusterings, that are consistent with all the queries. $\square$

Observe that for $q > 2$ this is in fact an efficient implementation of the generic halving algorithm of Awasthi and Zadeh [10]. Every subset of elements is either known to be pure, in which case it is consistent with the entire version space, or is possibly impure, in which case a `split` request would imply that the target vector satisfies a disjunction of linear equations. Thus in the latter case the set is consistent with at most a $\frac{1}{q} < \frac{1}{2}$ fraction of the version space. (We call a set of point *consistent* with a clustering, if under that clustering, this set is pure or, in other words, a subset of a cluster in the given clustering.)

There are two other notable properties of the algorithm. One is that it works without modification in the noisy setting of Awasthi and Zadeh [10]: if any pair of elements from two pure sets belong to the same target cluster, then it follows immediately by linearity that both sets are subsets of this target cluster.

The other notable property is that the algorithm never outputs impure hypothesis clusters. This is because it is always the case that every subset of the sample is either known to be pure, or otherwise it is consistent with at most half of the version space. Any concept class that has a similar gap property can be clustered using only pure clusters in the hypotheses. The following remark formalizes this idea.

Remark 5. *Consider the following generic algorithm: in each round, output the maximal subsets of S that are known to be pure, i.e. are consistent with the entire version space. The teacher cannot issue a* `split` *request since every hypothesis cluster is pure. If there is an $\varepsilon > 0$ such that in each round every subset $h \subseteq S$ of the sample is consistent with either the entire version space or at most a $(1 - \varepsilon)$ fraction of the version space, then on a* `merge` *request, by the maximality of the hypothesis clusters, we can eliminate an ε fraction of the version space. Therefore this algorithm finds the target clustering after $k \log_{\frac{1}{1-\varepsilon}} |C|$ queries using only pure clusters in the hypotheses.*

4.3.2 Efficient Clustering of Hyperplanes

Now we turn to a natural generalization of linear functions over $\mathbb{R}^d$, k hyperplanes. Clustering geometric concept classes was one of the proposed open problems by Awasthi and Zadeh [10]; hyperplanes are an example of a very natural geometric concept class. The data are points in $\mathbb{R}^d$ and they are clustered $(d-1)$-dimensional affine subspaces. Every point is assumed to lie on exactly one of k hyperplanes.

First, observe that this is a nontrivial interactive clustering problem: even for $d = 2$ the cardinality of the concept class can be exponentially large as a function of k. For example, let k be an odd integer, and consider $m - 2(k-1)$ points on a line and $2(k-1)$ points arranged as vertices of n squares such that no two edges are on the same line. Then it is easy to see that the number of different possible clusterings is at least 3^k. Hence, if $k = \omega(\text{polylog}(m))$, the target clustering cannot be efficiently found by the halving algorithm of Awasthi and Zadeh [10]: since the cardinality of the initial version space is superpolynomial in m, the algorithm cannot keep track of the version space in polynomial time.

Nevertheless, the case of $d = 2$ can be solved by the following trivial algorithm: start with the all-singleton hypothesis, and on a `merge` request, merge all the points that are on the line going through the two points. This algorithm will find the target clustering after k queries. However, this idea does not even generalize to $d = 3$: the teacher might repeatedly tell the learner to merge pairs of points that define parallel lines. In this case, it is not immediately clear which pairs of lines span the planes of the target clustering, and there can be a linear number of such parallel lines.

On the other hand, in the case of $d = 3$, coplanar lines either have to be in the same target cluster, or they all have to be in different clusters. Therefore if we have $k + 1$ coplanar lines, by the pigeonhole principle we know that the plane containing them has to be one of the target planes. Moreover, since all points are clustered by the k planes, it follows by the pigeonhole principle that after $k^2 + 1$ merge requests for singleton pairs we will get $k+1$ coplanar lines. This observation gives an algorithm of query complexity $O(k^3)$, although it is not immediately clear how the coplanar lines can be found efficiently.

Algorithm 3, described below, is an efficient clustering algorithm based on a similar idea which works for arbitrary dimension.

Algorithm 3 Cluster-Hyperplanes

```
let $H = S$.
for $i = 1$ to $d - 1$ do
    for each affine subspace $F$ of dimension $i$ do
        if at least $k^i + 1$ elements of $H$ are subsets of $F$ then
            replace these elements in $H$ by $F$.
        end if
    end for
end for
repeat
    output elements of $H$ as hypothesis clusters.
    on a merge request, merge the two clusters in $H$.
until the target clustering is found
```

Theorem 7. *Algorithm 3 finds the target clustering using at most $O(k^{d+1})$ queries and $O(d \cdot m^{d+1})$ time.*

Proof. We claim that in each iteration of the for loop, it holds for every F that every subset of $k^{i-1} + 1$ elements of H that lie on F spans F. The proof is by induction. For $i = 1$ this is clear: all pairs of points on a line span the line. Assume that the claim holds for $i - 1$. Consider $k^{i-1} + 1$ elements of H on an affine subspace F of dimension i. If they spanned an affine subspace of dimension less than i, then they would have been merged in a previous iteration. Hence they have to span F.

Now if $k^i + 1$ elements of H lie on an i-dimensional affine subspace F for $i < d$, then they have to be in the same target cluster. If they were not, no hyperplane could contain more than k^{i-1} of the elements, and therefore the k target hyperplanes could cover at most k^i elements contained by F, which contradicts the assumption that all points belong to a target cluster.

Hence, at the start of the repeat loop there can be at most k^{d+1} elements in H: if there were more than $k^{d+1} + 1$ elements in H, by the pigeonhole principle there would be a target cluster containing $k^d + 1$ of them. However, this is not possible since those $k^d + 1$ elements would have been merged previously.

Therefore in the repeat loop we only need k^{d+1} queries to find the target clustering. In each iteration of the outer for loop, we have to consider every affine subspace of a certain dimension. Since every at most $(d-1)$-dimensional subspace is defined by d points, there are at most $\binom{m}{d}$ subspaces. For each of them, we have to count the elements that are contained by them, this takes m time. Thus the total running time is $O\left(d \cdot \binom{m}{d} \cdot m\right) = O(d \cdot m^{d+1})$. □

Hence, for constant d, this is an efficient clustering algorithm.

4.4 Cryptographic Lower Bounds for Interactive Clustering

In this section, we show cryptographic lower bounds for interactive clustering. In particular, we prove that, under plausible cryptographic assumptions, the class of constant-depth polynomial-size threshold circuits and polynomial-size Boolean formulas are not learnable in the interactive clustering model. These lower bounds further go to show the richness of this model, which allows for both positive and negative clusterability results.

It was first observed by Valiant [90] that the existence of certain cryptographic primitives implies unlearnability results. Later, Kearns and Valiant [62] showed that, assuming the intractability of specific problems such as inverting the RSA function, some natural concept classes, for example the class of constant-depth threshold circuits, are not efficiently PAC learnable.

The hardness results for PAC learning are based on the following observation: if f is a trapdoor one-way function, and there is an efficient learning algorithm which, after seeing polynomially many labeled examples of the form $(f(x), x)$, can predict the correct label $f^{-1}(y)$ of a new unlabeled data point y, then that learning algorithm by definition breaks the one-way function f.

This observation doesn't apply to interactive clustering since here the learner doesn't have to make predictions about new examples and the teacher can give information about any of the elements in the sample. Indeed, if the learner were allowed to make a linear number of queries to the teacher, the clustering task would be computationally trivial. Instead, our proofs are based on the following counting argument: if the concept class is exponentially large in the size of the sample, then there is an immediate information-theoretic exponential lower bound on the required number of queries; therefore on average a learner would have to make an exponential number of queries to learn a randomly chosen clustering. If there exist certain pseudorandom objects, then one can construct concept classes of subexponential size such that a randomly chosen concept from the smaller class is computationally indistinguishable from a randomly chosen concept from the exponential-size class. However, on the smaller concept class the learner is only allowed to make a subexponential number of queries; consequently, this smaller class is not efficiently learnable.

We will use the following information-theoretic lower bound to prove our hardness result.

Lemma 4. *For $k = 2$, every clustering algorithm has to make at least $\Omega\left(\frac{\log|C|}{\log m}\right)$ queries to find the target clustering.*

Proof. There are $\log|C|$ bits are needed to describe the clustering. To each query, the answer is `split` or `merge` and the identifier of at most two clusters. Since there are at most m clusters in any hypothesis, this means that the teacher gives at most $2\log m + 1$ bits of information per query. Thus the required number of queries is $\Omega\left(\frac{\log|C|}{\log m}\right)$. □

We remark that Theorem 9 of Balcan and Blum [11] implies a worst-case lower bound of $\Omega(\log|C|)$. However, this weaker bound of $\Omega\left(\frac{\log|C|}{\log m}\right)$ holds for teachers that are not necessarily adversarial.

To prove our lower bounds, we will use cryptographic primitives that we introduced in Chapter 2. Recall that the existence of pseudorandom function families that can fool any polynomial time-bounded distinguishers is implied by the existence of one-way functions. Unfortunately, this hardness does not seem enough to imply a lower bound for interactive clustering for the following reason. If we take a sample of size m from $\{0,1\}^n$, then if $m = O(\text{poly}(n))$, the learner is allowed to make m queries which makes the clustering problem trivial. On the other hand, if m is superpolynomial in n, the learner is allowed to take superpolynomial time, therefore it might break pseudorandom functions that can only fool polynomial-time adversaries.

However, if there exist pseudorandom functions that can fool distinguishers that have slightly superpolynomial time, a hardness result for interactive clustering follows. Candidates for pseudorandom functions or permutations used in cryptographic practice are usually conjectured to have this property.

Theorem 8. *If there exist strongly pseudorandom functions that can fool distinguishers which have $n^{\omega(1)}$ time, then there exists a concept class C which is not learnable in the interactive clustering model with* $\text{poly}(\log m, \log |C|)$ *queries and* $\text{poly}(m, \log C)$ *time.*

Proof. Let $f_K : \{0,1\}^n \to \{0,1\}$ be a keyed pseudorandom function that can fool distinguishers which have $t(n)$ time for some time-constructible $t(n) = n^{\omega(1)}$. Without loss of generality, assume that $t(n) = o(2^n)$. Let us fix a time-constructible function $m(n)$ such that $m(n) = n^{\omega(1)}$ and $\text{poly}(m(n)) = o(t(n))$. Let S be a subset of $\{0,1\}^n$ of cardinality $m = m(n)$ and let $k = 2$. Let U_n be the set of all functions $\{0,1\}^n \to \{0,1\}$, $F_n = \{f_K : K \in \{0,1\}^n\}$.

Let us assume for contradiction that there is an efficient interactive clustering algorithm A for the concept class $C = F_n$. Since $|C| = 2^n$, this learner has to make at most $\text{poly}(n, \log m(n)) = \text{poly}(n)$ queries and has $\text{poly}(n, m(n)) = \text{poly}(m(n))$ time. Let us assume that the learner finds the target clustering after $O(n^\alpha)$ queries.

Let B be the following algorithm: given oracle access to a function $f : \{0,1\}^n \rightarrow \{0,1\}$, pick a sample S of size $m = m(n)$ from $\{0,1\}^n$, label the sample vectors according to the value of f, and simulate the learner A for at most $n^{\alpha+1}$ queries. Accept if the learner finds the target clustering and reject otherwise.

Since $\text{poly}(m(n)) = o(t(n))$, B runs in time $t(n)$. If f is chosen from F_n, B will accept with probability 1. On the other hand, if f is chosen from U_n, then since $|U_n| = 2^{2^n}$, by Lemma 4, we have a query lower bound of $\frac{\log |U_n|}{\log m} = \frac{2^n}{\log m(n)} = \omega(n^{\alpha+1})$. Therefore after $n^{\alpha+1}$ queries there are at least two different clusterings in the version space, therefore B will reject with probability at least $\frac{1}{2}$. This contradicts the $t(n)$-hardness of f_K. □

Naor and Reingold [73] constructed pseudorandom functions with one-bit output that are not only as secure as factoring Blum integers, but also computable by TC^0 circuits. Since $\log |TC^0| = \text{poly}(n)$, this, together with Theorem 8, implies the following corollary:

Corollary 3. *If factoring Blum integers is hard for $h(n)$-time bounded algorithms for some $h(n) = n^{\omega(1)}$ then the class TC^0 of constant-depth polynomial-size threshold circuits and the class of polynomial-size Boolean formulas are not learnable in the interactive clustering model.*

Proof. By Theorem 8, learning a pseudorandom function family of superpolynomial hardness is hard in the interactive clustering model. If factoring Blum integers is superpolynomially hard, then by the construction of Naor and Reingold [73], TC^0 contains such a pseudorandom function family. Furthermore,

$\log |TC^0| = \mathrm{poly}(n)$, the learner is still only allowed to have $\mathrm{poly}(n, \log m)$ queries and $\mathrm{poly}(n, m)$ time, therefore the Theorem 8 also applies to TC^0. In fact, this holds for TC^0 circuits of size at most n^α for some constant α (determined by the size of the circuits implementing the pseudorandom function). The set of languages computable by TC^0 circuits of size n^α is in turn a subset of the languages computable by Boolean formulas of size at most n^β for some other constant β. Thus our cryptographic lower bound also holds for polynomial-sized Boolean formulas. □

Remark 6. *After Naor and Reingold's first construction of pseudorandom functions in TC^0, several others showed that it is possible to construct even more efficient PRFs, or PRFs based on different, possibly weaker cryptographic assumptions. For example, we refer the reader to the work of Lewko and Waters [69] for a construction under the so-called "decisional k-linear assumption" which is weaker than the assumption of Naor and Reingold [73], or to Banerjee et al. [13] for a construction based on the "learning with errors" problem, against which there is no known attack by efficient quantum algorithms.*

Kearns and Valiant [62] used the results of Pitt and Warmuth [75] about prediction-preserving reductions to show that in the PAC model, their cryptographic hardness result for NC^1 circuits also implies the intractability of learning DFAs. Despite the fact that the problem of interactive clustering is fundamentally different from prediction problems, we show that the ideas of Pitt and Warmuth [75] can be applied to show that DFAs are hard to learn in this model as well. We use the following theorem:

Theorem 9 (Pitt and Warmuth [75])**.** *Let k be a fixed positive constant. If T is a single-tape Turing machine of size at most s that runs in space at most $k \log n$ on inputs of length n, then there exist polynomials p and q such that for all positive integers n there exists a DFA M of size $q(s, n)$ such that M accepts $g(w) = 1^{|w|}0w^{p(|w|,s,n)}$ if and only if T accepts w.*

This theorem implies a hardness result for interactive clustering.

Corollary 4. *If there are $n^{\omega(1)}$-hard pseudorandom function families computable in logarithmic space, then polynomial-size DFAs are not efficiently learnable in the interactive clustering model.*

Proof. Let $f_K : \{0,1\}^n \to \{0,1\}$ be an $n^{\omega(1)}$-hard keyed pseudorandom function. If $S \subset \{0,1\}^n$ has cardinality $m(n)$ as defined in Theorem 8 and the concept class is $\{f_K : K \in \{0,1\}^n\}$, the interactive clustering task is hard.

For all $K \in \{0,1\}^n$, let T_K be a Turing machine of size at most s that runs in space $k \log n$ and, given w as an input, computes $f_K(w)$. It is easy to see that there exist functions g, p and q defined as in Theorem 9 that work for T_K for all K. Consider the sample $S' = g(S)$ and the concept class C of DFAs of size $q(s, n)$. Since $|S'| = m(n)$ and $\log |C| = \text{poly}(n)$, the hardness result of Theorem 8 holds here as well. □

4.5 Conclusion

In this chapter we studied a model of clustering with interactive feedback. We presented efficient clustering algorithms for linear functionals over finite fields, of which parity functions are a special case, and hyperplanes in $\mathbb{R}^d$, thereby showing that these two natural problems are learnable in the model. On the other hand, we also demonstrated that under standard cryptographic assumptions, constant-depth polynomial-size threshold circuits, polynomial-size Boolean formulas, and polynomial-size deterministic finite automata are not learnable.

We propose the following open problems.

1. It would be interesting to see if the exponential dependence on d in the complexity of Algorithm 3 for clustering hyperplanes can be reduced.

2. Although for half-spaces in fixed dimension, the general version space algorithm of [10] is efficient because of the polynomial size of the version space, designing more efficient interactive clustering algorithms for half-spaces and Voronoi partitions remains a natural and important open problem.

4.5.1 Subsequent Results

Models of clustering aided by user interaction continues to be of interest to the clustering research community, mainly as one of the several approaches towards giving provable guarantees for clustering algorithms, classifying clustering problems by their complexity, and bypassing worst-case hardness results.

Let us mention one example of a paper on this broader topic which was published after our paper [68]. Ashtiani et al. [9] introduced a related query-aided clustering model, in which the clustering algorithm can make queries to an oracle. In these queries, the algorithm can ask whether two points belong to the same cluster or not. These queries are then used to efficiently solve k-means problems satisfying certain margin conditions which otherwise would be NP-hard.

More generally, the question of how interaction or other forms of weak supervision can help build a more adequate theory of clustering remains an exciting direction for future research.

5

Balancing Fairness and Accuracy in Supervised Learning

This chapter was previously published as Benjamin Fish, Jeremy Kun, Ádám D. Lelkes: A Confidence-Based Approach for Balancing Fairness and Accuracy. *2016 SIAM International Conference on Data Mining (SDM)*: 144-152.

5.1 Introduction

Machine learning algorithms assume an increasingly large role in making decisions across many different areas of industry, finance, and government, from facial recognition and social network analysis to self-driving cars to data-based approaches in commerce, education, and policing. The decisions made by algorithms in these domains directly affect individual people, and not always for the better. Consequently, there has been a growing concern that machine learning algorithms, which are often poorly understood by those that use them, make discriminatory decisions.

If the data used for training the algorithm is biased, a machine learning algorithm will learn the bias and perpetuate discriminatory decisions against groups that are protected by law, even in the absence of “discriminatory intent” by the designers. A typical example is an algorithm serving predatory ads to protected groups. Such issues resulted in a 2014 report from the US Executive Office [76] which voiced concerns about discrimination in machine learning. The primary question we study in this chapter is

> How can we maintain high accuracy of a learning algorithm while reducing discriminatory biases?

In this chapter we will focus on the issue of biased training data, which is one of the several possible causes of discriminatory outcomes in machine learning. In this setting, we have a protected attribute (e.g. race or gender) which we assert should be independent from the target attribute. For example, if the goal is to decide creditworthiness for loans and the protected attribute is gender, a classifier’s prediction should not correlate with an applicant’s gender. We say that the classifier achieves *statistical parity* if the protected subgroup is as likely as the broader population to have a given label.

Of course, there might be situations where the target label depends on legitimate factors that correlate with the protected attribute. For example, if the protected attribute is gender and the target label is income, some argue that lower salaries for women can be partly explained by the fact that on average, men work longer hours than women. In this chapter we assume that this is not the case. The issue of "explainable discrimination" in machine learning was studied in [58].

In our setting, since we only have biased data, we cannot evaluate our classifiers against an unbiased ground truth. In particular only a biased classifier could achieve perfect accuracy; to achieve statistical parity in general one must be willing to reduce accuracy. Hence the natural goal is to find a classifier that achieves statistical parity while minimizing error, or more generally to study the trade-off between bias and accuracy so as to make favorable trade-offs.

Our first contribution in this chapter is a method for optimizing this trade-off which we call the *shifted decision boundary* (SDB). SDB is a generic method based on the theory of margins [27, 83], and it can be combined with any learning algorithm that produces a measure of confidence in its prediction (Section 5.3.1). In particular we combine SDB with boosting, support vector machines, and logistic regression, and it performs comparably to or outperforms previous algorithms in the fair learning literature. See Section 5.5 for its empirical evaluation. We also give a theorem based on the analysis in [83] bounding the loss of accuracy for SDB under weighted majority schemes (Section 5.3.4). SDB makes the assumptions on the bias explicit and transparent, so that the trade-off can be understood without a detailed understanding of the algorithm.

Unfortunately, studying the bias-error trade-off is an incomplete picture of the fairness of an algorithm. The shortcomings were discussed in [32], e.g., in terms of how an adversary could achieve statistical parity while still targeting the protected group unfairly. We demonstrate these shortcomings in action even in the absence of adversarial manipulation. Among other methods, we show that modifying a classifier by randomly flipping certain output labels with a certain probability already outperforms much of the prior fairness literature in both accuracy and bias. Such a naive algorithm is obviously unfair because the relabeling is independent of the classification task. Our second contribution is a measure of fairness that addresses this shortcoming, which we call *resilience to random bias*. We define it in Section 5.4 and demonstrate that it distinguishes well between our naive baseline algorithms and SDB.

5.2 Background and Previous Work

5.2.1 Existing Notions of Fairness

The study of fairness in machine learning is young, but there has been a lot of disparate work studying notions of what it means for data to be fair. Finding the "right" definition of fairness is a major challenge; see the extensive survey of [77] for a detailed discussion. Two prominent definitions of fairness that have emerged are *statistical parity* and *k-nearest-neighbor consistency.*

Statistical parity: Let D be a distribution over a set of labeled examples X with labels $l : X \to \{-1, 1\}$ and a protected subset $S \subset X$. The *bias* of l with respect to D is defined as the difference in probability of an example in S having label 1 and the probability of an example in S^C having label 1, i.e.

$$B(D,S) = \Pr_{x \sim D|_{S^C}}[l(x) = 1] - \Pr_{x \sim D|_S}[l(x) = 1].$$

The bias of a hypothesis h is the same quantity with $h(x)$ replacing $l(x)$. If a hypothesis has low bias in absolute value we say it achieves *statistical parity.* Note that S represents the group we wish to protect from discrimination, and the bias represents the degree to which they have been discriminated against. The sign of bias indicates whether S or S^C is discriminated against. A similar statistical measure called *disparate impact* was introduced and studied by Friedler et al. [35] based on the "80% rule" used in United States hiring law.

Dwork et al. [32] point out that statistical parity is only a measure of population-wide fairness. They provide a laundry list of ways one could achieve statistical parity while still exhibiting serious and unlawful discrimination. In particular, one can achieve statistical parity by flipping the labels of a certain number of arbitrarily chosen members of the disadvantaged group, regardless of the relation between the individuals and the classification task. In our experiments we show this already outperforms some of the leading algorithms in the fairness literature.

Despite this, it is important to study the ability for learning algorithms to achieve statistical parity. For example, it might be reasonable to flip the labels of the "most qualified" individuals of the disadvantaged group who are classified negatively. Some previous approaches assume the existence of a ranking or metric on individuals, or try to learn this ranking from data [56, 32]. By contrast, our

SDB achieves statistical parity without the need for such a ranking.

kNN-consistency: The second notion, due to [32], calls a classifier "individually fair" if it classifies similar individuals similarly. They use k-nearest-neighbor to measure the consistency of labels of similar individuals. Note that "closeness" is defined with respect to a metric chosen as part of the data cleaning and feature selection process. By contrast SDB does not require a metric on individuals.

5.2.2 Previous Work on Fair Algorithms

Learning algorithms studied previously in the context of fairness include naive Bayes [22], decision trees [57], and logistic regression [59]. To the best of our knowledge we are the first to study boosting and SVM in this context, and our confidence-based analysis is new for both these and logistic regression.

The two main approaches in the literature are massaging and regularization. Massaging means changing the biased data set before training to remove the bias in the hope that the learning algorithm trained on the now unbiased data will be fair. Massaging is done in the previous literature based on a ranking learned from the biased data [56]. The regularization approach consists of adding a regularizer to an optimization objective which penalizes the classifier for discrimination [60]. While SDB can be thought of as a post-processing regularization, it does so in a way that makes the trade-off between bias and accuracy transparent and easily controlled.

There are two other notable approaches in the fairness literature. The first, introduced in [32], is a framework for maximizing the utility of a classification with the constraint that similar people be treated similarly. One shortcoming of this approach is that it relies on a metric on the data that tells us the similarity of individuals with respect to the classification task. Moreover, the work in [32] suggests that learning a suitably fair similarity metric from the data is as hard as the original problem of finding a fair classifier. Our SDB method does not require such a metric.

The "Learning Fair Representations" method of Zemel et al. [97] formulates the problem of fairness in terms of intermediate representations: the goal is to find a representation of the data which preserves as much information as possible from the original data while simultaneously obfuscating membership in the protected class. Given that in this chapter we seek to make explicit the trade-off between bias and accuracy, we will not be able to hide membership in the protected class as Zemel et al. seeks to do. Rather, we align with the central thesis of [32], that knowing the protected feature is useful to promote fairness.

5.2.3 Margins

The theory of margins has provided a deep, foundational explanation for the generalization properties of algorithms such as AdaBoost and soft-margin SVMs [27, 83]. A hypothesis $f : X \to [-1, 1]$ induces a *margin* for a labeled example $\text{margin}_f(x, y) = y \cdot f(x)$, where $x \in X$ is a data point and $y \in \{-1, 1\}$ is the correct label for x. The sign of the margin is positive if and only if f correctly labels x, and the magnitude indicates how confident f is in its prediction.

As an example of the power of margins, we quote a celebrated theorem on the generalization accuracy of weighted majority voting schemes in PAC-learning. Here a weighted majority vote is a function $f(x) = \sum_{i=1}^{N} \alpha_i h_i(x)$ for some hypotheses $h_i \in H$ and $\alpha_i \geq 0, \sum_i \alpha_i = 1$.

Theorem 10 (Schapire et al. [83])**.** *Let D be a distribution over $X \times \{-1, 1\}$ and S be a sample of m examples chosen i.i.d. at random according to D. Let H be a set of hypotheses of VC-dimension d. Then for any $\delta > 0$, with probability at least $1 - \delta$* every *weighted majority voting scheme satisfies the following for every $\theta > 0$:*

$$\Pr_D[yf(x) \leq 0] \leq \Pr_S[yf(x) \leq \theta] + \\ O\left(\frac{1}{\sqrt{m}}\left(\frac{d\log^2(m/d)}{\theta^2} + \log(1/\delta)\right)^{1/2}\right)$$

In other words, the generalization error is bounded by the probability of a small margin *on the sample.* One can go on to show AdaBoost [82], a popular algorithm that produces a weighted voting scheme, performs well in this respect. Recall that the output of AdaBoost is a hypothesis which outputs the sign of a weighted majority vote $\sum_i \alpha_i, h_i(x)$. Rather than measure the margin we measure the *signed confidence* of the boosting hypothesis on an unlabeled example x as

$$\text{conf}(\mathbf{x}) = \frac{\sum_{i=1}^{T} \alpha_i h_i(\mathbf{x})}{\sum_{i=1}^{T} \alpha_i}.$$

The magnitude of the confidence measures the agreement of the voters in their classification of an example.

The theoretical work on margins for boosting suggests that examples with small confidence are more likely to have incorrect labels than examples with large confidence. For example, we display in Figure 5.1 the signed confidence values for all examples and incorrectly predicted examples respectively. The incorrect examples have confidence centered around zero. One can leverage this for fairness by flipping negative labels of members of the protected class with a small confidence value. This is a rough sketch of the SDB method. The empirical results of SDB suggest that SDB achieves statistical parity with relatively little loss in accuracy. Indeed, we state a similar guarantee to Theorem 10 in Section 5.3.4 that solidifies this intuition.

The idea of a signed confidence generalizes nicely to other machine learning algorithms. We study support vector machines (SVM) which have a natural geometric notion of margin, and logistic regression which outputs a confidence in its prediction. For background on SVM, logistic regression, and AdaBoost, see [85].

5.2.4 Interpretations of Signed Confidence

Here we state how signed confidence is defined for each of the learning methods.

AdaBoost

Boosting algorithms work by combining *base hypotheses*, “rules of thumb” that have a fixed edge over random guessing, into highly accurate predictors. In each round, a boosting algorithm finds the base hypothesis that achieves the smallest weighted error on the sample. It then increases the weights of the incorrectly classified examples, thus forcing the base learner to improve the classification of difficult examples. In this chapter we study AdaBoost, a ubiquitous boosting

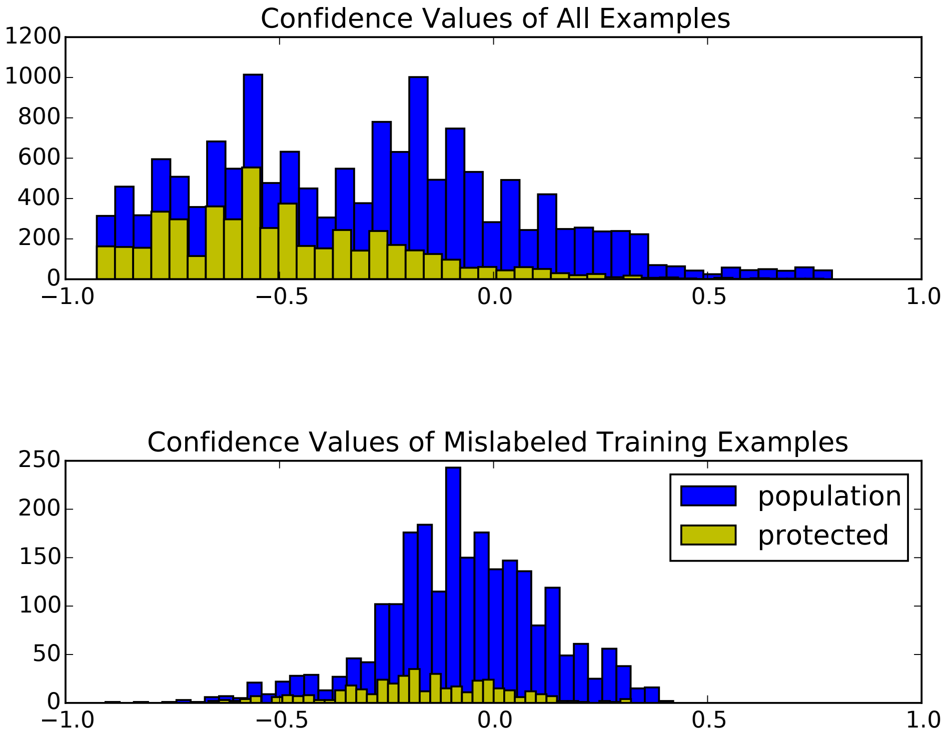


Figure 5.1: Histogram of boosting confidences for the Census data set. The top histogram shows the distribution of confidence values for the entire data set, and the bottom shows the confidence for only mislabeled examples. The vast majority of women are classified as -1. Comparing incorrectly classified women to correctly classified women, a larger proportion of the examples corresponding to incorrectly classified women are close to the decison boundary.

algorithm. For more on boosting, we refer the reader to [82].

Let H be a set of base classifiers, and let $(\alpha_t, h_t)_{t=1}^{T}$ be the weights and hypotheses output by AdaBoost after T rounds. The signed confidence of the hypothesis is $\mathrm{conf}(\mathbf{x}) = \frac{\sum_{i=1}^{T} \alpha_i h_i(\mathbf{x})}{\sum_{i=1}^{T} \alpha_i}$. In all of our experiments we boost decision stumps for $T = 20$ rounds.

SVM

The soft-margin SVM of Cortes and Vapnik [27] outputs a maximum margin hyperplane $\mathbf{w}$ in a high-dimensional space implicitly defined by a kernel K, and $\mathbf{w}$ can be expressed implicitly as a linear combination of the input vectors, say $\mathbf{w}'$. We define the confidence as the distance of a point from the separating hyperplane, i.e. $\text{conf}(\mathbf{x}) = K(\mathbf{w}', \mathbf{x})$. For the Census Income and Singles data sets we use the standard Gaussian kernel, and for the German data set we use a linear kernel (the data sets are described in Section 5.5).

Logistic Regression

Logistic regression is often used to assign probabilities to class labels; in this chapter we will use it as a binary classifier and disregard the probabilities. (In other words, we pick the label with the higher probability.) In this setting, the classifier output by logistic regression has the form $h(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle)$ and the vector $\mathbf{w}$ is found by empirical risk minimization (ERM) with the standard logistic loss $\ell(\mathbf{w}, (\mathbf{x}, y)) = \log(1 + e^{-y\langle \mathbf{w}, \mathbf{x} \rangle})$ and L_2 regularization. Here we define the confidence of logistic regression simply as the value that the classifier takes before rounding: $\text{conf}(\mathbf{x}) = \phi(\langle \mathbf{w}, \mathbf{x} \rangle)$, where $\phi(z) = \frac{1}{1+e^{-z}}$ is the logistic function.

5.3 Fair Learning Algorithms

5.3.1 Shifted Decision Boundary

In this section we define our methods. In what follows X is a labeled data set, $l(x)$ are the given labels, and $S \subset X$ is the protected group. We further assume that members of S are less likely than S^C to have label 1. First we describe our proposed method, called *shifted decision boundary* (SDB), and then we describe three techniques we use for baseline comparisons (in addition to comparing to previous literature).

Let $\mathrm{conf} : X \to [-1, 1]$ be a function corresponding to a classifier $h(x) = \mathrm{sign}(\mathrm{conf}(x))$, and define the *decision boundary shift of λ for S* as the classifier $h_\lambda : X \to \{-1, 1\}$, defined as

$$h_\lambda(x) = \begin{cases} 1 & \text{if } x \in S, \mathrm{conf}(x) \geq -\lambda \\ \mathrm{sign}(\mathrm{conf}(x)) & \text{otherwise.} \end{cases}$$

The SDB algorithm accepts as input confidences conf and finds the minimal error decision boundary shift for S that achieves statistical parity. That is, given conf and $\varepsilon > 0$, it produces a value λ such that h_λ has minimal error subject to achieving statistical parity up to bias ε.

5.3.2 Naive Baseline Algorithms

We define two naive baseline methods which are intended to be both baseline comparisons for our SDB algorithm and illustrations of the shortcomings of the bias-error trade-off.

Similarly to SDB, the *random relabeling* (RR) algorithm modifies a given hypothesis h by flipping labels. In particular, RR computes the probability p for which, if members of S with label -1 under h are flipped by h' to $+1$ randomly and independently with probability p, the bias of h' is zero in expectation. The classifier h' is then defined as the randomized classifier that flips members of S with label -1 with probability p and otherwise is the same as h.

Next, we define *random massaging* (RM). Massaging strategies, introduced by [56], involve eliminating the bias of the training data by modifying the labels of data points, and then training a classifier on this data in the hope that the statistical parity of the training data will generalize to the test set as well. In our experiment, we massage the data randomly; i.e. we flip the labels of S from -1 to $+1$ independently at random with the probability needed to achieve statistical parity in expectation, as in RR.

As we have already noted, these two baseline methods perform comparably to much of the previous literature in both bias and error. This illustrates that the semantics of *why* an algorithm achieves statistical parity is crucial part of its evaluation. As such, these two baselines can be useful for any analysis that measures bias and accuracy. Moreover, they can be used to determine the suitability of a new proposed measure of fairness.

5.3.3 Fair Weak Learning

Finally, we include a method which is based on a natural idea but is empirically suboptimal to SDB. Recall that boosting works by combining weak learners into a "strong" classifier. It is natural to ask whether boosting keeps the fairness properties of the weak learners. Weak learners used in practice, such as decision stumps, have very low complexity, therefore it is easy to impose fairness constraints on them. In our *fair weak learning* (FWL) baseline we replace a standard boosting weak learner with one which tries to minimize a linear combination of error and bias and run the resulting boosting algorithm unchanged. The weak learner we use computes the decision stump which minimizes the sum of label error and bias of its induced hypothesis.

5.3.4 Theoretical Properties of SDB

Because the SDB method only flips the labels of examples with small signed confidence, margin theory implies that it will not increase the error too much. We formalize this precisely below. This theorem, a direct corollary of Theorem 10, provides strong theoretical justification for our SDB method. To the best of our knowledge, SDB is the first empirically tested method for fair learning that has any specific guarantees for its accuracy.

Informally, the theorem says that when a majority voting scheme is post-processed by the SDB technique, the resulting hypothesis maintains the generalization accuracy bounds in terms of the margin on the sample when the shift is small ($\lambda \leq \theta$). But as the shift grows, the error bound increases proportionally to the fraction of the protected population that has large enough negative margins (i.e., in $[-\lambda, -\theta]$).

Theorem 11. *Let X be finite and D, S, m, H, and d be as in Theorem 10. Let $T \subset S$ be the subset of the sample in the protected class. Let $\delta > 0$. Let $\operatorname{err}(m)$ be the tail error function from Theorem 10. For any $A \subset X$ let $A_{\lambda,\theta} = \{a \in A : -\lambda \le \operatorname{conf}(a) \le -\theta\}$.*

Then with probability at least $1 - \delta$, every function h_λ post-processed by SDB with weighted majority vote $\operatorname{conf}(x)$ and shift $\lambda > 0$ satisfies the following for every $\theta > 0$:

$$\begin{aligned}
\Pr_D[yh_\lambda(x) \le 0] \le{} & \Pr_{T_{\lambda,\theta}}[y \cdot \operatorname{conf}(x) \ge -\theta] \Pr_S[x \in T_{\lambda,\theta}] \\
& + \Pr_{S - T_{\lambda,\theta}}[y \cdot \operatorname{conf}(x) \le \theta] \Pr_S[x \notin T_{\lambda,\theta}] \\
& + \max(\operatorname{err}(|T_{\lambda,\theta}|), \operatorname{err}(|T^C_{\lambda,\theta}|))
\end{aligned}$$

Proof. The bound follows by conditioning on the event that h_λ flips the label, noticing $-\operatorname{conf}(x)$ is also a majority function, and applying Theorem 10 twice. □

5.4 Resilience to Random Bias

One of the biggest challenges for designers of fair learning algorithms is the lack of good measures of fairness. The most popular measures are statistical measures of bias such as statistical parity. As Dwork et al. [32] have pointed out, statistical parity fails to capture all important aspects of fairness. In particular, it is easy to achieve statistical parity simply by flipping the labels of an arbitrary set of individuals in the protected class. A real-world example would be giving a raise to a random group of women to eliminate the gender disparity in wages. The root cause of this problem is that one does not have access to reliable (unbiased) ground truth labels. We propose to compensate for this by evaluating algorithms on synthetic bias. In doing this we make transparent the *kind* of bias a claimed "fair" algorithm protects against, and we can accurately measure its resilience to said bias.

We introduce a new notion of fairness called *resilience to random bias* (RRB). Informally we introduce a new, random feature which has no correlation with the target attribute, and then we introduce bias against individuals who have a certain value for this new feature. We call an algorithm fair if it can recover the original, unbiased labels. For RRB in particular, the synthetic bias is i.i.d. random against the protected group.

Certainly, in practice, bias may not be of this form and we do not pretend that this notion captures all forms of bias. Rather, this notion seeks to model a comparatively mild form of bias – if an algorithm cannot recover from this type of random bias against a protected class then there is little reason to think it can handle other types of bias. In other words, we propose this as a minimally necessary condition but not necessarily a sufficient condition for individual fairness. Relating our RRB measure more formally to other notions of individual fairness is left for future work.

We formally define RRB as follows. Let X be a set of examples and D be a distribution over examples, with $l : X \to \{-1, 1\}$ a target labeling function.

We first define a randomized process mapping $(X, D, l) \to (\tilde{X}, \tilde{D}, \tilde{l})$. Let $\tilde{X} = X \times \{-1, 1\}$ and $\tilde{D}$ be the distribution on $\tilde{X}$ which is independently D on the X coordinate and uniform on the $\{-1, 1\}$ coordinate. Denote by $\tilde{X}_0 = \{(x, b) \in \tilde{X} \mid b = 0\}$ and call this the *protected set*. Finally, $\tilde{l}(x, b)$ is *fixed* to either $l(x)$ or $-l(x)$ independently at random for each $(x, b) \in \tilde{X}$ according to the following:

$$\Pr[\tilde{l}(x, b) = l(x)] = \begin{cases} 1 & \text{if } b = 1 \text{ or } l(x) = -1 \\ 1 - \eta & \text{if } b = 0 \text{ and } l(x) = 1 \end{cases}.$$

In other words, the positive labels of a randomly chosen protected subgroup are flipped to negative independently at random with probability η. We emphasize that the process mapping $l \mapsto \tilde{l}$ is randomized, but the map $\tilde{l}(x, b)$ itself is fixed and deterministic. So an algorithm which queries labels from $\tilde{l}$ is given consistent answers. Now we define the resilience to random bias as follows:

Definition 26. *Let $(X, D, l), (\tilde{X}, \tilde{D}, \tilde{l})$ be as above. Let $h = A(\tilde{D}, \tilde{l})$ be the output classifier of a learning algorithm A when given biased data as input. The* resilience to random bias *(RRB) of A with respect to (X, D, l) and discrimination rate $0 \leq \eta < 1/2$, denoted* $\mathrm{RRB}_\eta(A)$, *is*

$$\mathrm{RRB}_\eta(A) = \Pr_{\tilde{D}}[h(x, b) = l(x) \mid b = 0, l(x) = 1]$$

Similarly to calculating statistical parity, RRB is estimated on a fixed data set by simulating the process described above and outputing an empirical average.

5.5 Empirical Evaluation

We measure our methods on label error, statistical parity, and RRB with $\eta = 0.2$. In all of our experiments we split the data sets randomly into training, test, and model-selection subsets, and we output the average of 10 experiments.

5.5.1 Datasets

The Census Income data set [70], extracted from the 1994 Census database, contains demographic information about 48842 American adults. The prediction task is to determine whether a person earns over \$50K a year. The data set contains $16,192$ females (33%) and $32,650$ males. Note 30.38% of men and 10.93% of women reported earnings of more than \$50K, therefore the bias of the data set is 19.45%.

The German credit data set [70] contains financial information about 1000 individuals who are classified into groups of good and bad credit risk. The "good" credit group contains 699 individuals. Following the work of [56], we consider age as the protected attribute with a cut-off at 25. Only 59% of the younger people are considered good credit risk, whereas of the 25 or older group, 72% are creditworthy, making the bias 13%.

In the Singles data set, extracted from the marketing data set of [48] by taking all respondents who identified as "single," the goal is to predict if annual income is greater than \$25K from 13 other demographic attributes. The protected attribute is gender. The data set contains $3,653$ data points, $1,756$ (48%) of which belong to the protected group. 34% of the data set has a positive label. The bias is 9.8%.

5.5.2 Results and Analysis

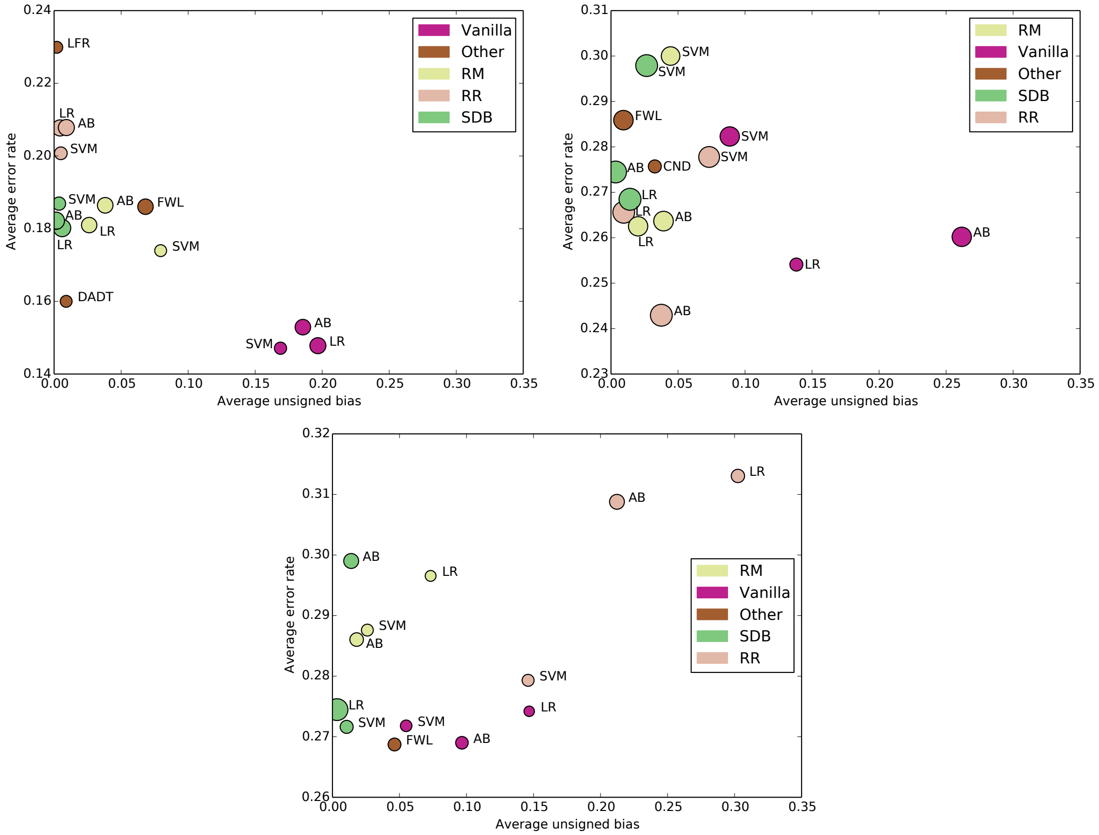


Figure 5.2: A summary of our experimental results for our three data sets, from left to right: Census Income, German, Singles. Labels show which learning algorithm is used and the colors give which method for achieving fairness was used. The parameters of each algorithm were chosen to minimize bias. The size of a point is proportional to the RRB of the learner (only for those algorithms for which we have the RRB numbers), where larger dots mean there is a larger probability of correcting a label.

In this section we state our experimental results. They are summarized in Figure 5.2 for the Census Income, German, and Singles data sets, and the full set of numbers are in Tables 5.2, 5.3, and 5.4 respectively. For comparison, we also included the numbers for the Learning Fair Representations (LFR) method of [97]

Method	Census	German	Singles
SVM	0.2702	0.6756	0.2424
SVM (RR)	0.2821	0.7827	0.2588
SVM (RM)	0.2545	0.6232	0.2552
SVM (SDB)	**0.3172**	**0.8619**	**0.3064**
LR	0.4647	0.3070	0.1971
LR (RR)	0.4696	0.8564	0.3213
LR (RM)	0.4282	0.6741	0.2117
LR (SDB)	**0.5402**	**0.8687**	**0.8596**
AB	0.4372	0.6774	0.2864
AB (RR)	0.4661	**0.8629**	0.3996
AB (RM)	0.4410	0.6965	0.3325
AB (SDB)	**0.5461**	0.8596	**0.4027**
AB (FWL)	0.5174	0.6879	0.2971

Table 5.1: The RRB numbers for each of our methods and baselines. In each column and section the largest values are shown in bold, and they are almost always SDB.

for the Census Income data set, for Classification with No Discrimination (CND) method of [56], and for the Discrimination Aware Decision Tree (DADT) technique of [57] (specifically we use the numbers for the "IGC+IGS_Relab" method). In [97] the authors implemented three other learning algorithms, these are unregularized logistic regression, Fair Naive-Bayes [56], and Regularized Logistic Regression [60]. These methods all had errors above 20% on the Census data set and so we omit them for brevity. In [57] the authors implemented variations on the decision tree learning scheme, and the one we include has the highest accuracy, though they are all closely comparable. We reported all biases as unsigned. We were unable to access implementations of the prior authors' algorithms, so we were not able to reproduce their results or measure their algorithms with respect to RRB.

First, we display the trade-offs made explicit by our SDB method in Figure 5.3,

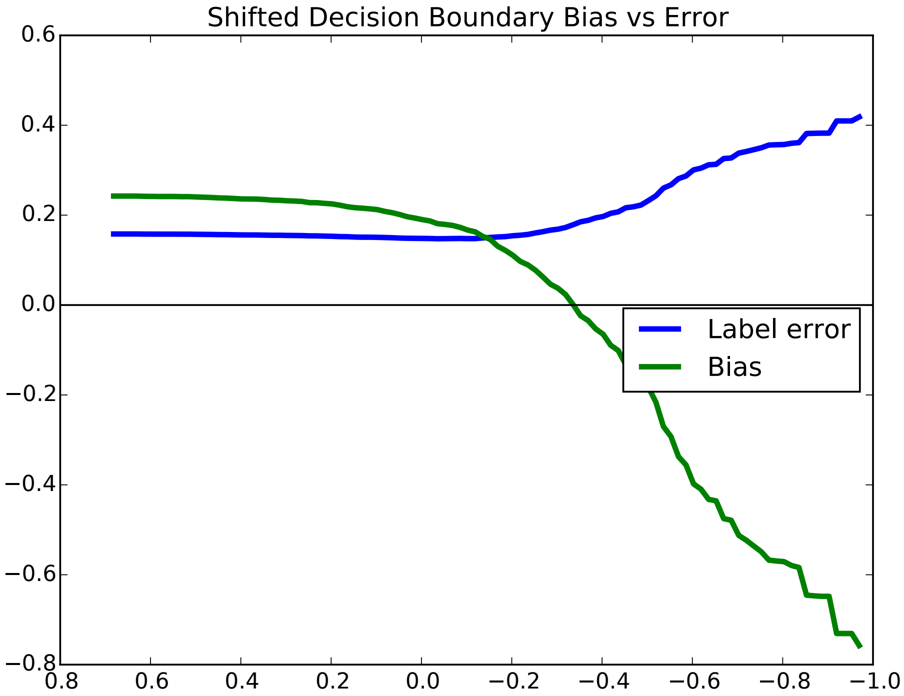
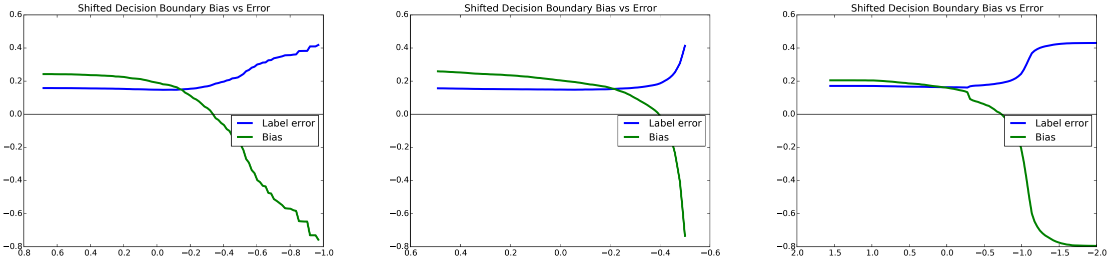


Figure 5.3: Trade-off between (signed) bias and error for SDB with AdaBoost on the Census Income data. The horizontal axis is the threshold used for SDB.

which shows an example of the rate at which error increases as bias goes to zero.

For the Census Income data set, the three SDB techniques outperform the baselines and outperform all the prior literature except for DADT. Both SDB algorithms achieve statistical parity with about 18% error. Moreover, these two SDB algorithms have the highest RRB, while SVM appears to overfit the random bias introduced by RRB more than the other algorithms. While DADT appears to achieve lower label error and comparable bias, we note that the standard deviation of the bias reported in [57] is 0.015 while for SDB (on the Census Income data set) the standard deviations are at least one order of magnitude smaller.



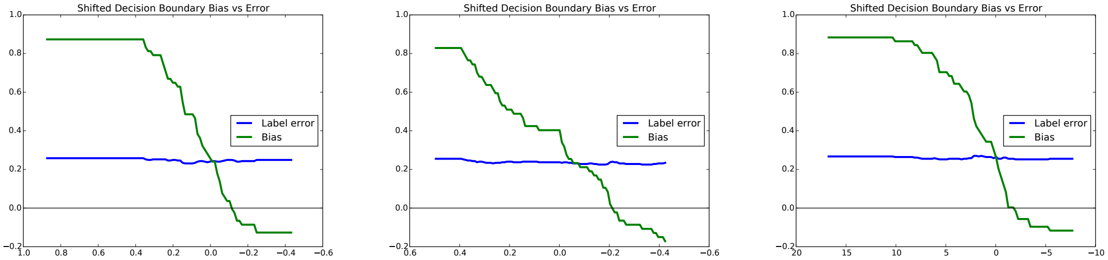
(a) Boosting **(b)** Logistic Regression **(c)** SVM

Figure 5.4: Trade-off between (signed) bias and error for SDB on the Census Income data. The horizontal axis is the threshold used for SDB.

The singles data set shows a similar pattern, with SDB combined with logistic regression outperforming all other baselines. Note that in the instances where the baselines perform comparably to SDB, SDB tends to have a much larger resilience to random bias.

The German data set is more puzzling. While two of the SDB techniques outperform the prior literature by a moderate margin, they do not outperform random relabeling or random massaging by a significant margin (and these baselines already outperform CND). Another curious observation is, as Figure 5.5 shows, label error stays constant as the decision boundary is shifted. In addition, we used a linear kernel for SVM on the German data set because we observed clear overfitting with a Gaussian kernel.

Note again the difference in SVM kernels between the data sets. The Gaussian kernel performs well for the Census Income and Singles data set. However, in the case of the German data set, which is the smallest of the three, with the Gaussian kernel every point becomes a support vector. This is not only a clear sign of overfitting but it also makes SDB useless since the model gives the same



(a) Boosting

(b) Logistic Regression

(c) SVM

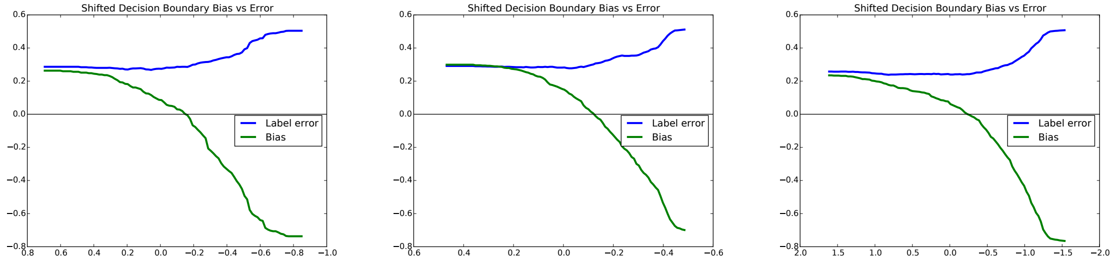
Figure 5.5: Trade-off between (signed) bias and error for SDB on the German data. The horizontal axis is the threshold used for SDB.

confidence for almost every data point.

These facts seem to be evidence that the German data set (which has only about a thousand records) is too small to draw a significant conclusion. We nevertheless include it here for completeness and to show comparison with the previous literature.

Fair weak learning (FWL) does empirically reduce bias but does not achieve statistical parity in two of the three data sets. FWL performs worse on either label error or bias on each of the data sets and the trade-off between label error and bias cannot easily be controlled. It also does not seem easy to control this trade-off using either random massaging and random relabeling.

One notable advantage of SDB is that the trade-off between label error and bias can be controlled *after* training. To decide how much bias and error we want to allow, we do not have to pick a hyper-parameter before training the algorithm, unlike for most other fair learning methods. This means that the computational cost of choosing the best point on the trade-off curve is very low, and the trade-off is transparent.



(a) Boosting **(b)** Logistic Regression **(c)** SVM

Figure 5.6: Trade-off between (signed) bias and error for SDB on the Singles data. The horizontal axis is the threshold used for SDB.

The results also highlight the usefulness of RRB as a measure of fairness. The RRB values across all data sets and algorithms we studied are in Table 5.1. In cases where random relabeling or random massaging performs comparably to SDB, the RRB measure is able to distinguish them, giving a lower score to the less reasonable baselines and a higher score to SDB. This suggests that the performance of fair learning algorithms should not be evaluated solely by their accuracy and bias.

5.6 Conclusion

In this chapter, we introduced a general method for balancing discrimination and label error. This method, which we call shifted decision boundary (SDB), is applicable to any learning algorithm which has an efficiently computable measure of confidence. We studied three such algorithms – AdaBoost, support vector machines, and linear regression – compared our methods to other methods proposed in the earlier literature and our own baselines, and empirically evaluated our methods' performances in terms of their resilience to random bias.

Our method, in addition to outperforming much of the previous literature, has several other desirable properties. Unlike most other fair learning algorithms, SDB applied to AdaBoost has theoretical bounds on generalization error. Also, since the margin shift can be specified after the original learner has been trained on the data, a practitioner can easily evaluate the trade-off between error and bias and choose the most desirable point on the trade-off curve. This makes SDB a fast and transparent way to study the fairness properties of an algorithm.

Our resilience to random bias (RRB) measure is a novel approach to evaluate the fairness of a learning algorithm. Although i.i.d. random bias is a simplified model of real-world discrimination, we posit that any algorithm which can be considered fair must be fair with respect to RRB. Moreover, RRB generalizes to an arbitrary distribution over the input data, and one could adapt it to well-studied models of bias in social science.

5.6.1 Subsequent Results

The topic of fairness of machine learning has been receiving rapidly increasing interest from the machine learning community. There have been so many important papers published on this topic after our paper [36] that it would be nearly impossible to list, let alone discuss, them all.

The study of fair algorithms has been extended from the setting of classical supervised learning to other learning paradigms, such as online decision making [53], bandit learning [54], Markovian settings [51], and word embeddings [23, 20].

	SVM	SVM (RR)	SVM (SDB)	SVM (RM)	LFR [97]
label error	0.1471 (5.7e-17)	0.2007 (0.002)	0.1869 (0.004)	0.1740 (0.003)	0.2299
bias	0.1689 (5.7e-17)	0.0050 (0.003)	0.0036 (0.009)	0.0795 (0.010)	0.0020
RRB	0.2702 (0.014)	0.2926 (0.004)	0.3172 (0.025)	0.2545 (0.007)	n/a
	LR	LR (RR)	LR (SDB)	LR (RM)	DADT [57]
label error	0.1478 (4.8e-04)	0.2077 (0.004)	0.1802 (0.002)	0.1810 (0.003)	0.1600
bias	0.1968 (0.003)	0.0044 (0.006)	0.0060 (0.011)	0.0262 (0.008)	0.0090 (0.015)
RRB	0.4647 (0.013)	0.4696 (0.009)	0.5402 (0.011)	0.4282 (0.019)	n/a
	AdaBoost	AB (RR)	AB (SDB)	AB (RM)	AB (FWL)
label error	0.1529 (0.002)	0.2078 (0.004)	0.1822 (0.005)	0.1864 (0.004)	0.1860 (0.004)
bias	0.1856 (0.012)	0.0091 (0.006)	0.0013 (0.007)	0.0381 (0.013)	0.0682 (0.004)
RRB	0.4372 (0.032)	0.4661 (0.019)	0.5461 (0.015)	0.4410 (0.013)	0.4321 (0.016)

Table 5.2: A summary of our experimental results for the Census Income data for relabeling, massaging, and the fair weak learner. The threshold for SDB was chosen to achieve perfect statistical parity on the training data. Standard deviations are reported in parentheses when known.

Also, the question of what it means for a supervised learning algorithm to be fair and how to quantify discrimination continues to be studied. A notable paper on this question is Friedler et al. [41]. This paper approaches the problem by separating different spaces in which the data about individuals is explicitly or implicitly represented, and making the often unstated assumptions about the interactions between these spaces explicit. By this they create a common vocabulary in which to discuss and compare different notions of fairness; our RRB measure is mentioned as a particular example of one of the paradigms listed in the paper.

With all these results, the study of fairness in machine learning remains in an early stage, and one can expect this topic to provide machine learning researchers, data scientists, and social scientists with many exciting and difficult questions for a long time.

	SVM	SVM (RR)	SVM (SDB)	SVM (RM)	CND [56]
label error	0.2823 (0)	0.2778 (0.025)	0.2979 (0.022)	0.3000 (0.017)	0.2757
bias	0.0886 (4.2e-17)	0.0732 (0.066)	0.0266 (0.085)	0.0445 (0.028)	0.0327
RRB	0.6756 (0.081)	0.7827 (0.054)	0.8619 (0.041)	0.6232 (0.070)	n/a
	LR	LR (RR)	LR (SDB)	LR (RM)	
label error	0.2541 (0.005)	0.2656 (0.020)	0.2685 (0.021)	0.2625 (0.011)	
bias	0.1383 (0.014)	0.0095 (0.064)	0.0142 (0.219)	0.0202 (0.566)	
RRB	0.3070 (0.067)	0.8564 (0.045)	0.8687 (0.042)	0.6741 (0.045)	
	AdaBoost	AB (RR)	AB (SDB)	AB (RM)	AB (FWL)
label error	0.2602 (0.009)	0.2429 (0.010)	0.2745 (0.010)	0.2637 (0.019)	0.2859 (0.016)
bias	0.2617 (0.272)	0.0376 (0.044)	0.0034 (0.064)	0.0391 (0.023)	0.0093 (0.035)
RRB	0.6774 (0.219)	0.8629 (0.051)	0.8596 (0.067)	0.6965 (0.037)	0.6879 (0.042)

Table 5.3: A summary of our experimental results for the German data for relabeling, massaging, and the fair weak learner. The threshold for SDB was chosen to achieve perfect statistical parity on the training data. On this data set SVM was run with a linear kernel. Standard deviations are reported in parentheses when known.

	SVM	SVM (RR)	SVM (SDB)	SVM (RM)	
label error	0.2718 (5.7e-17)	0.2793 (0.009)	0.2716 (0.013)	0.2876 (0.015)	
bias	0.0550 (1.4e-17)	0.1460 (0.017)	0.0106 (0.035)	0.0260 (0.047)	
RRB	0.2424 (0.045)	0.2588 (0.009)	0.3064 (0.042)	0.2552 (0.032)	
	LR	LR (RR)	LR (SDB)	LR (RM)	
label error	0.2742 (1.14e-16)	0.3130 (0.011)	0.2745 (0.010)	0.2966 (0.008)	
bias	0.1468 (9.99e-18)	0.3025 (0.040)	0.0034 (0.640)	0.0732 (0.024)	
RRB	0.1971 (0.036)	0.3213 (0.035)	0.8596 (0.067)	0.2117 (0.036)	
	AdaBoost	AB (RR)	AB (SDB)	AB (RM)	AB (FWL)
label error	0.2690 (0.004)	0.3088 (0.009)	0.2990 (0.008)	0.2860 (0.019)	0.2687 (0.008)
bias	0.0966 (0.020)	0.2123 (0.013)	0.0140 (0.017)	0.0180 (0.037)	0.0463 (0.016)
RRB	0.2864 (0.057)	0.3996 (0.105)	0.4027 (0.061)	0.3325 (0.060)	0.2971 (0.028)

Table 5.4: A summary of our experimental results for the Singles data for relabeling, massaging, and the fair weak learner. The threshold for SDB was chosen to achieve perfect statistical parity on the training data. Standard deviations are reported in parentheses when known.

Cited Literature

[1] Margareta Ackerman and Shai Ben-David. Clusterability: A theoretical study. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS 2009, Clearwater Beach, Florida, USA, April 16-18, 2009*, pages 1–8, 2009.

[2] Alexandr Andoni, Aleksandar Nikolov, Krzysztof Onak, and Grigory Yaroslavtsev. Parallel algorithms for geometric graph problems. In *STOC*, pages 574–583, 2014.

[3] Dana Angluin. Queries and concept learning. *Machine learning*, 2(4):319–342, 1988.

[4] Julia Angwin, Jeff Larson, Surya Mattu, and Lauren Kirchner. Machine bias: There's software used across the country to predict future criminals. and it's biased against blacks. *ProPublica, May*, 23, 2016.

[5] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for euclidean k-medians and related problems. In *STOC*, pages 106–113, 1998.

[6] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach.* Cambridge University Press, 2009.

[7] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pandit. Local search heuristics for k-median and facility location problems. *SIAM J. Comput.*, 33(3):544–562, 2004.

[8] Hassan Ashtiani and Shai Ben-David. Representation learning for clustering: a statistical framework. In *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*, pages 82–91. AUAI Press, 2015.

[9] Hassan Ashtiani, Shrinu Kushagra, and Shai Ben-David. Clustering with same-cluster queries. *arXiv preprint arXiv:1606.02404*, 2016.

[10] Pranjal Awasthi and Reza B. Zadeh. Supervised clustering. In *Advances in Neural Information Processing Systems*, pages 91–99, 2010.

[11] Maria-Florina Balcan and Avrim Blum. Clustering with interactive feedback. In *Algorithmic Learning Theory*, pages 316–328. Springer, 2008.

[12] Maria-Florina Balcan, Avrim Blum, and Santosh Vempala. A discriminative framework for clustering via similarity functions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 671–680, 2008.

[13] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *Advances in Cryptology–EUROCRYPT 2012*, pages 719–737. Springer, 2012.

[14] Y. Bartal, M. Charikar, and D. Raz. Approximating min-sum k-clustering in metric spaces. In *STOC*, pages 11–20, 2001.

[15] Sugato Basu, Arindam Banerjee, and Raymond J. Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA, April 22-24, 2004*, pages 333–344, 2004.

[16] Paul Beame, Paraschos Koutris, and Dan Suciu. Communication steps for parallel query processing. In *PODS*, pages 273–284, 2013.

[17] Shai Ben-David. Computational feasibility of clustering under clusterability assumptions. *CoRR*, abs/1501.00437, 2015.

[18] Shai Ben-David, Nadav Eiron, and Philip M. Long. On the difficulty of approximately maximizing agreements. *Journal of Computer and System Sciences*, 66(3):496–514, 2003.

[19] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM (JACM)*, 36(4):929–965, 1989.

[20] Tolga Bolukbasi, Kai-Wei Chang, James Y. Zou, Venkatesh Saligrama, and Adam T. Kalai. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*, pages 4349–4357, 2016.

[21] S. Charles Brubaker and Santosh Vempala. Isotropic PCA and affine-invariant clustering. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008, October 25-28, 2008, Philadelphia, PA, USA*, pages 551–560, 2008.

[22] Toon Calders and Sicco Verwer. Three naive Bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, 21(2):277–292, 2010.

[23] Aylin Caliskan-Islam, Joanna J Bryson, and Arvind Narayanan. Semantics derived automatically from language corpora necessarily contain human biases. *arXiv preprint arXiv:1608.07187*, 2016.

[24] M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem. *J. Comput. Syst. Sci.*, 65(1):129–149, 2002.

[25] Cheng-Tao Chu, Sang Kyun Kim, Yi-An Lin, YuanYuan Yu, Gary R. Bradski, Andrew Y. Ng, and Kunle Olukotun. Map-Reduce for machine learning on multicore. In *NIPS*, pages 281–288, 2006.

[26] Stephen A. Cook. A hierarchy for nondeterministic time complexity. *Journal of Computer and System Sciences*, 7(4):343–353, 1973.

[27] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[28] Anirban Dasgupta, John E. Hopcroft, Ravi Kannan, and Pradipta Prometheus Mitra. Spectral clustering by recursive partitioning. In *Algorithms - ESA 2006, 14th Annual European Symposium, Zurich, Switzerland, September 11-13, 2006, Proceedings*, pages 256–267, 2006.

[29] Sajib Dasgupta and Vincent Ng. Which clustering do you want? Inducing your ideal clustering with minimal feedback. *J. Artif. Intell. Res. (JAIR)*, 39:581–632, 2010.

[30] W. F. de la Vega, M. Karpinski, C. Kenyon, and Y. Rabani. Approximation schemes for clustering problems. In *STOC*, pages 50–58, 2003.

[31] Jeffrey Dean and Sanjay Ghemawat. MapReduce: simplified data processing on large clusters. *Commun. ACM*, 51(1):107–113, 2008.

[32] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 214–226. ACM, 2012.

[33] Ahmed K. Farahat, Ahmed Elgohary, Ali Ghodsi, and Mohamed S. Kamel. Distributed column subset selection on MapReduce. In *ICDM*, pages 171–180, 2013.

[34] Jon Feldman, S. Muthukrishnan, Anastasios Sidiropoulos, Clifford Stein, and Zoya Svitkina. On distributing symmetric streaming computations. *ACM Transactions on Algorithms*, 6(4), 2010.

[35] Michael Feldman, Sorelle A. Friedler, John Moeller, Carlos Scheidegger, and Suresh Venkatasubramanian. Certifying and removing disparate impact. *Proceedings of the 21th ACM SIGKDD Intl. Conference on Knowledge Discovery and Data Mining*, pages 259–268, 2015.

[36] Benjamin Fish, Jeremy Kun, and Ádám D. Lelkes. A confidence-based approach for balancing fairness and accuracy. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 144–152. SIAM, 2016.

[37] Benjamin Fish, Jeremy Kun, Ádám D. Lelkes, Lev Reyzin, and György Turán. On the computational complexity of MapReduce. In *International Symposium on Distributed Computing*, pages 1–15. Springer, 2015.

[38] Lance Fortnow. Time-space tradeoffs for satisfiability. *J. Comput. Syst. Sci.*, 60(2):337–353, 2000.

[39] Steven Fortune and James Wyllie. Parallelism in random access machines. In *Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 114–118. ACM, 1978.

[40] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

[41] Sorelle A. Friedler, Carlos Scheidegger, and Suresh Venkatasubramanian. On the (im)possibility of fairness. *arXiv preprint arXiv:1609.07236*, 2016.

[42] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *J. ACM*, 33(4):792–807, August 1986.

[43] Michael T. Goodrich, Nodari Sitchinava, and Qin Zhang. Sorting, searching, and simulation in the MapReduce framework. In *ISAAC*, pages 374–383, 2011.

[44] S. Guha and S. Khuller. Greedy strikes back: Improved facility location algorithms. *J. Algorithms*, 31(1):228–248, 1999.

[45] Steve Hanneke. The optimal sample complexity of PAC learning. *Journal of Machine Learning Research*, 17(38):1–15, 2016.

[46] Juris Hartmanis and Richard E. Stearns. On the computational complexity of algorithms. *Transactions of the American Mathematical Society*, 117:285–306, 1965.

[47] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[48] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*, volume 2. Springer, 2009.

[49] Russell Impagliazzo and Ramamohan Paturi. The complexity of k-SAT. *2012 IEEE 27th Conference on Computational Complexity*, 0:237, 1999.

[50] Russell Impagliazzo, Ramamohan Paturi, and Francis Zane. Which problems have strongly exponential complexity? *J. Comput. Syst. Sci.*, 63(4):512–530, 2001.

[51] Shahin Jabbari, Matthew Joseph, Michael Kearns, Jamie Morgenstern, and Aaron Roth. Fair learning in Markovian environments. *arXiv preprint arXiv:1611.03071*, 2016.

[52] K. Jain, M. Mahdian, and A. Saberi. A new greedy approach for facility location problems. In *STOC*, pages 731–740. ACM, 2002.

[53] Matthew Joseph, Michael Kearns, Jamie Morgenstern, Seth Neel, and Aaron Roth. Rawlsian fairness for machine learning. *arXiv preprint arXiv:1610.09559*, 2016.

[54] Matthew Joseph, Michael Kearns, Jamie Morgenstern, and Aaron Roth. Fairness in learning: Classic and contextual bandits. *arXiv preprint arXiv:1605.07139*, 2016.

[55] Seny Kamara and Mariana Raykova. Parallel homomorphic encryption. In *Financial Cryptography Workshops*, pages 213–225, 2013.

[56] Faisal Kamiran and Toon Calders. Classifying without discriminating. In *2nd Intl. Conference on Computer, Control and Communication, 2009.*, pages 1–6. IEEE, 2009.

[57] Faisal Kamiran, Toon Calders, and Mykola Pechenizkiy. Discrimination aware decision tree learning. In *2010 IEEE 10th Intl. Conference on Data Mining (ICDM)*, pages 869–874. IEEE, 2010.

[58] Faisal Kamiran, Indrė Žliobaitė, and Toon Calders. Quantifying explainable discrimination and removing illegal discrimination in automated decision making. *Knowledge and Information Systems*, 35(3):613–644, 2013.

[59] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-aware classifier with prejudice remover regularizer. In *Machine Learning and Knowledge Discovery in Databases*, pages 35–50. Springer, 2012.

[60] Toshihiro Kamishima, Shotaro Akaho, and Jun Sakuma. Fairness-aware learning through regularization approach. In *Data Mining Workshops (ICDMW), 2011 IEEE 11th Intl. Conference on*, pages 643–650. IEEE, 2011.

[61] Howard Karloff, Siddharth Suri, and Sergei Vassilvitskii. A model of computation for MapReduce. In *SODA '10*, pages 938–948, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

[62] Michael Kearns and Leslie Valiant. Cryptographic limitations on learning Boolean formulae and finite automata. *Journal of the ACM (JACM)*, 41(1):67–95, 1994.

[63] Michael J. Kearns, Robert E. Schapire, and Linda M. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

[64] George S. Kimeldorf and Grace Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 41(2):495–502, 1970.

[65] A. Kumar, Y. Sabharwal, and S. Sen. Linear-time approximation schemes for clustering problems in any dimensions. *J. ACM*, 57(2), 2010.

[66] Ravi Kumar, Benjamin Moseley, Sergei Vassilvitskii, and Andrea Vattani. Fast greedy algorithms in MapReduce and streaming. In *SPAA '13*, pages 1–10, New York, NY, USA, 2013. ACM.

[67] Jeremy Kun. *Graphs, New Models, and Complexity.* PhD thesis, University of Illinois at Chicago, 2016.

[68] Ádám D. Lelkes and Lev Reyzin. Interactive clustering of linear classes and cryptographic lower bounds. In *Proceedings of the 2016 SIAM International Conference on Data Mining*, pages 165–176. SIAM, 2016.

[69] Allison B. Lewko and Brent Waters. Efficient pseudorandom functions from the decisional linear assumption and weaker variants. In *Proceedings of the 16th ACM conference on Computer and communications security*, pages 112–120. ACM, 2009.

[70] M. Lichman. UCI machine learning repository, 2013.

[71] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

[72] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of machine learning.* MIT press, 2012.

[73] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *Journal of the ACM (JACM)*, 51(2):231–262, 2004.

[74] Matthew Felice Pace. BSP vs MapReduce. In *Proceedings of the International Conference on Computational Science, ICCS 2012, Omaha, Nebraska, USA, 4-6 June, 2012*, pages 246–255, 2012.

[75] Leonard Pitt and Manfred K. Warmuth. Prediction-preserving reducibility. *Journal of Computer and System Sciences*, 41(3):430–467, 1990.

[76] John Podesta, Penny Pritzker, Ernest J. Moniz, John Holdren, and Jeffrey Zients. Big data: Seizing opportunities, preserving values, 2014.

[77] Andrea Romei and Salvatore Ruggieri. A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review*, 29:582–638, 11 2014.

[78] Tim Roughgarden, Sergei Vassilvitskii, and Joshua R. Wang. Shuffles and circuits (on lower bounds for modern parallel computation). In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures*, pages 1–12. ACM, 2016.

[79] Anish Das Sarma, Foto N. Afrati, Semih Salihoglu, and Jeffrey D. Ullman. Upper and lower bounds on the cost of a Map-Reduce computation. In *PVLDB'13*, pages 277–288. VLDB Endowment, 2013.

[80] Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of computer and system sciences*, 4(2):177–192, 1970.

[81] Robert E. Schapire. The strength of weak learnability. *Machine learning*, 5(2):197–227, 1990.

[82] Robert E. Schapire and Yoav Freund. *Boosting: Foundations and Algorithms.* MIT Press, 2012.

[83] Robert E. Schapire, Yoav Freund, Peter Bartlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics*, pages 1651–1686, 1998.

[84] Bernhard Schölkopf, Ralf Herbrich, and Alex J. Smola. A generalized representer theorem. In *International Conference on Computational Learning Theory*, pages 416–426. Springer, 2001.

[85] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge University Press, 2014.

[86] J. C. Shepherdson. The reduction of two-way automata to one-way automata. *IBM J. Res. Dev.*, 3(2):198–200, April 1959.

[87] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler. The Hadoop distributed file system. In Mohammed G. Khatib, Xubin He, and Michael Factor, editors, *MSST*, pages 1–10. IEEE Computer Society, 2010.

[88] Latanya Sweeney. Discrimination in online ad delivery. *Queue*, 11(3):10, 2013.

[89] A. Szepietowski. *Turing Machines with Sublogarithmic Space*. Ernst Schering Research Foundation Workshops. Springer, 1994.

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

[91] Leslie G. Valiant. A bridging model for parallel computation. *Commun. ACM*, 33(8):103–111, 1990.

[92] Vladimir Vapnik and Alexey Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

[93] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2001.

[94] K. Wagner and G. Wechsung. *Computational Complexity*. Mathematics and its Applications. Springer, 1986.

[95] Ryan Williams. Time-space tradeoffs for counting NP solutions modulo integers. *Computational Complexity*, 17(2):179–219, 2008.

[96] Andrew C. Yao. Theory and application of trapdoor functions. In *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*, pages 80–91. IEEE, 1982.

[97] Rich Zemel, Yu Wu, Kevin Swersky, Toni Pitassi, and Cynthia Dwork. Learning fair representations. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 325–333, 2013.

Appendix

This appendix contains copies of the copyright agreements signed with the publishers of the papers reproduced in this thesis. The agreement with SIAM was filled out and signed electronically; the blank form is reproduced here.

Consent to Publish

Lecture Notes in Computer Science

Title of the Book or Conference Name: 2015 International Symposium on Distributed Computing

Volume Editor(s): Yoram Moses

Title of the Contribution: On the Computational Complexity of MapReduce

Author(s) Name(s): Benjamin Fish, Jeremy Kun, Adam Lelkes, Lev Reyzin, Gyorgy Turan

Corresponding Author's Name, Address, Affiliation and Email: Jeremy Kun
851 S. Morgan St. Chicago, IL 60607.
University of Illinois at Chicago, jkun2@uic.edu

When Author is more than one person the expression "Author" as used in this agreement will apply collectively unless otherwise indicated.

§ 1 Rights Granted

Author hereby grants and assigns to Springer International Publishing AG, Cham (hereinafter called Springer) the exclusive, sole, permanent, world-wide, transferable, sub-licensable and unlimited right to reproduce, publish, distribute, transmit, make available or otherwise communicate to the public, translate, publicly perform, archive, store, lease or lend and sell the Contribution or parts thereof individually or together with other works in any language, in all revisions and versions (including soft cover, book club and collected editions, anthologies, advance printing, reprints or print to order, microfilm editions, audiograms and videograms), in all forms and media of expression including in electronic form (including offline and online use, push or pull technologies, use in databases and networks for display, print and storing on any and all stationary or portable end-user devices, e.g. text readers, audio, video or interactive devices, and for use in multimedia or interactive versions as well as for the display or transmission of the Contribution or parts thereof in data networks or seach engines), in whole, in part or in abridged form, in each case as now known or developed in the future, including the right to grant further time-limited or permanent rights. For the purposes of use in electronic forms, Springer may adjust the Contribution to the respective form of use and include links or otherwise combine it with other works. For the avoidance of doubt, Springer has the right to permit others to use individual illustrations and may use the Contribution for advertising purposes.

The copyright of the Contribution will be held in the name of Springer. Springer may take, either in its own name or in that of copyright holder, any necessary steps to protect these rights against infringement by third parties. It will have the copyright notice inserted into all editions of the Contribution according to the provisions of the Universal Copyright Convention (UCC) and dutifully take care of all formalities in this connection in the name of the copyright holder.

§ 2 Regulations for Authors under Special Copyright Law

The parties acknowledge that there may be no basis for claim of copyright in the United States to a Contribution prepared by an officer or employee of the United States government as part of that person's official duties. If the Contribution was performed under a United States government contract, but Author is not a United States government employee, Springer grants the United States government royalty-free permission to reproduce all or part of the Contribution and to authorize others to do so for United States government purposes.

If the Contribution was prepared or published by or under the direction or control of Her Majesty (i.e., the constitutional monarch of the Commonwealth realm) or any Crown government department, the copyright in the Contribution shall, subject to any agreement with Author, belong to Her Majesty.

If the Contribution was created by an employee of the European Union or the European Atomic Energy Community (EU/Euratom) in the performance of their duties, the regulation 31/EEC, 11/EAEC (Staff Regulations) applies, and copyright in the Contribution shall, subject to the Publication Framework Agreement (EC Plug), belong to the European Union or the European Atomic Energy Community.

If Author is an officer or employee of the United States government, of the Crown, or of EU/Euratom, reference will be made to this status on the signature page.

§ 3 Rights Retained by Author

Author retains, in addition to uses permitted by law, the right to communicate the content of the Contribution to other scientists, to share the Contribution with them in manuscript form, to perform or present the Contribution or to use the content for non-commercial internal and educational purposes, provided the Springer publication is mentioned as the original source of publication in any printed or electronic materials. Author retains the right to republish the Contribution in any collection consisting solely of Author's own works without charge subject to ensuring that the publication by Springer is properly credited and that the relevant copyright notice is repeated verbatim.

Author may self-archive an author-created version of his/her Contribution on his/her own website and/or the repository of Author's department or faculty. Author may also deposit this version on his/her funder's or funder's designated repository at the funder's request or as a result of a legal obligation. He/she may not use the publisher's PDF version, which is posted on SpringerLink and other Springer websites, for the purpose of self-archiving or deposit. Furthermore, Author may only post his/her own version, provided acknowledgment is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be accompanied by the following text: "The final publication is available at link.springer.com".

Prior versions of the Contribution published on non-commercial pre-print servers like ArXiv/CoRR and HAL can remain on these servers and/or can be updated with Author's accepted version. The final published version (in pdf or html/xml format) cannot be used for this purpose. Acknowledgment needs to be given to the final publication and a link must be inserted to the published Contribution on Springer's website, accompanied by the text "The final publication is available at link.springer.com".

Author retains the right to use his/her Contribution for his/her further scientific career by including the final published paper in his/her dissertation or doctoral thesis provided acknowledgment is given to the original source of publication. Author also retains the right to use, without having to pay a fee and without having to inform the publisher, parts of the Contribution (e.g. illustrations) for inclusion in future work, and to publish a substantially revised version (at least 30% new content) elsewhere, provided that the original Springer Contribution is properly cited.

§ 4 Warranties

Author warrants that the Contribution is original except for such excerpts from copyrighted works (including illustrations, tables, animations and text quotations) as may be included with the permission of the copyright holder thereof, in which case(s) Author is required to obtain written permission to the extent necessary and to indicate the precise sources of the excerpts in the manuscript. Author is also requested to store the signed permission forms and to make them available to Springer if required.

Author warrants that he/she is entitled to grant the rights in accordance with Clause 1 "Rights Granted", that he/she has not assigned such rights to third parties, that the Contribution has not heretofore been published in whole or in part, that the Contribution contains no libelous statements and does not infringe on any copyright, trademark, patent, statutory right or proprietary right of others, including rights obtained through licenses; and that Author will indemnify Springer against any costs, expenses or damages for which Springer may become liable as a result of any breach of this warranty.

§ 5 Delivery of the Work and Publication

Author agrees to deliver to the responsible Volume Editor (for conferences, usually one of the Program Chairs), on a date to be agreed upon, the manuscript created according to the Springer Instructions for Authors. Springer will undertake the reproduction and distribution of the Contribution at its own expense and risk. After submission of the Consent to Publish form Signed by the Corresponding Author, changes of authorship, or in the order of the authors listed, will not be accepted by Springer.

§ 6 Author's Discount

Author is entitled to purchase for his/her personal use (directly from Springer) the Work or other books published by Springer at a discount of 33 1/3% off the list price as long as there is a contractual arrangement between Author and Springer and subject to applicable book price regulation. Resale of such copies or of free copies is not permitted.

§ 7 Governing Law and Jurisdiction

This agreement shall be governed by, and shall be construed in accordance with, the laws of Switzerland. The courts of Zug, Switzerland shall have the exclusive jurisdiction.

Corresponding Author signs for and accepts responsibility for releasing this material on behalf of any and all Co-authors.

Signature of Corresponding Author: **Date:** 2015-08-15

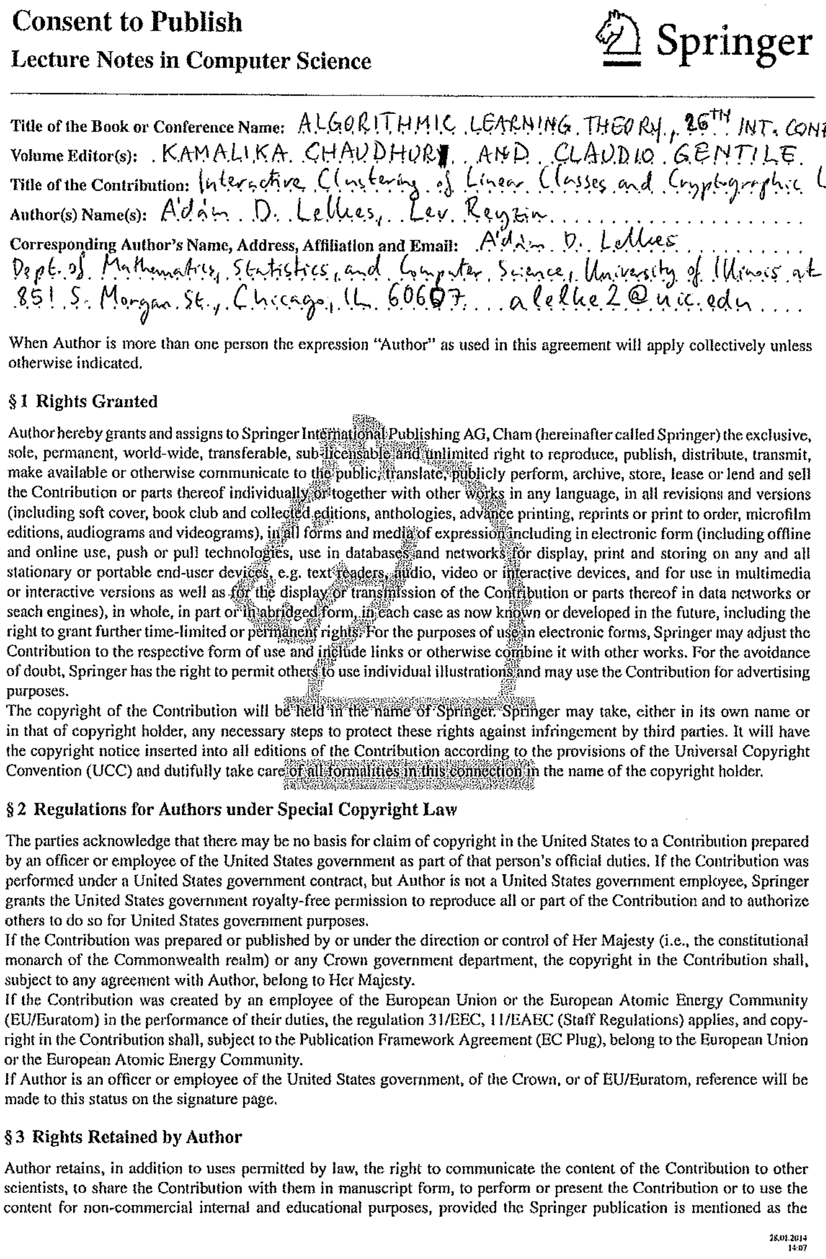
. . . .

□ I'm an employee of the US Government and transfer the rights to the extent transferable
(Title 17 §105 U.S.C. applies)

□ I'm an employee of the Crown and copyright on the Contribution belongs to Her Majesty

□ I'm an employee of the EU or Euratom and copyright on the Contribution belongs to EU or Euratom

15.03.2013
11:29



Consent to Publish

Lecture Notes in Computer Science

Springer

Title of the Book or Conference Name: ALGORITHMIC LEARNING THEORY, 26TH INT. CONFERENCE
Volume Editor(s): KAMALIKA CHAUDHURI AND CLAUDIO GENTILE
Title of the Contribution: Interactive Clustering of Linear Classes and Cryptographic Lower Bounds
Author(s) Name(s): Ádám D. Lelkes, Lev Reyzin
Corresponding Author's Name, Address, Affiliation and Email: Ádám D. Lelkes
Dept. of Mathematics, Statistics, and Computer Science, University of Illinois at Chicago
851 S. Morgan St., Chicago, IL 60607. alelke2@uic.edu

When Author is more than one person the expression "Author" as used in this agreement will apply collectively unless otherwise indicated.

§ 1 Rights Granted

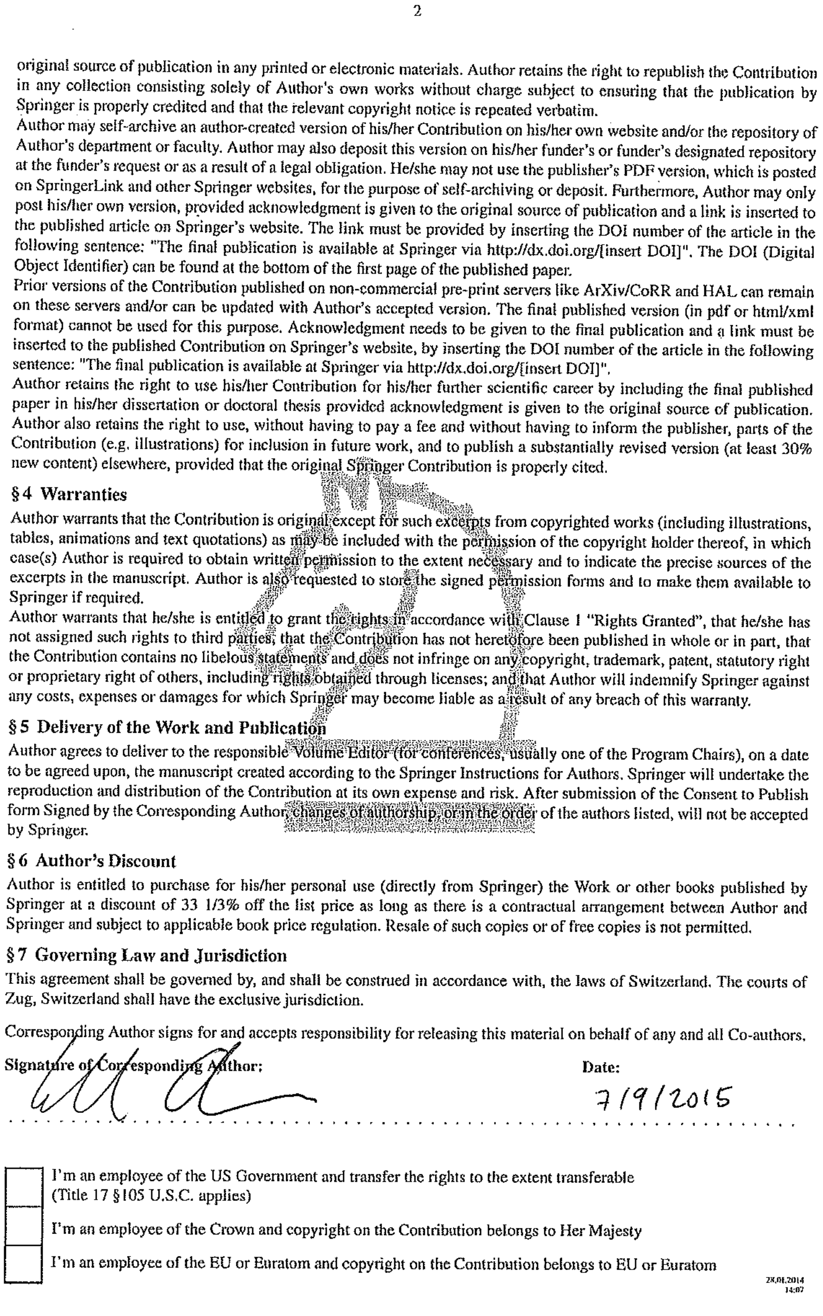
Author hereby grants and assigns to Springer International Publishing AG, Cham (hereinafter called Springer) the exclusive, sole, permanent, world-wide, transferable, sub-licensable and unlimited right to reproduce, publish, distribute, transmit, make available or otherwise communicate to the public, translate, publicly perform, archive, store, lease or lend and sell the Contribution or parts thereof individually or together with other works in any language, in all revisions and versions (including soft cover, book club and collected editions, anthologies, advance printing, reprints or print to order, microfilm editions, audiograms and videograms), in all forms and media of expression including in electronic form (including offline and online use, push or pull technologies, use in databases and networks for display, print and storing on any and all stationary or portable end-user devices, e.g. text readers, audio, video or interactive devices, and for use in multimedia or interactive versions as well as for the display or transmission of the Contribution or parts thereof in data networks or seach engines), in whole, in part or in abridged form, in each case as now known or developed in the future, including the right to grant further time-limited or permanent rights. For the purposes of use in electronic forms, Springer may adjust the Contribution to the respective form of use and include links or otherwise combine it with other works. For the avoidance of doubt, Springer has the right to permit others to use individual illustrations and may use the Contribution for advertising purposes.
The copyright of the Contribution will be held in the name of Springer. Springer may take, either in its own name or in that of copyright holder, any necessary steps to protect these rights against infringement by third parties. It will have the copyright notice inserted into all editions of the Contribution according to the provisions of the Universal Copyright Convention (UCC) and dutifully take care of all formalities in this connection in the name of the copyright holder.

§ 2 Regulations for Authors under Special Copyright Law

The parties acknowledge that there may be no basis for claim of copyright in the United States to a Contribution prepared by an officer or employee of the United States government as part of that person's official duties. If the Contribution was performed under a United States government contract, but Author is not a United States government employee, Springer grants the United States government royalty-free permission to reproduce all or part of the Contribution and to authorize others to do so for United States government purposes.
If the Contribution was prepared or published by or under the direction or control of Her Majesty (i.e., the constitutional monarch of the Commonwealth realm) or any Crown government department, the copyright in the Contribution shall, subject to any agreement with Author, belong to Her Majesty.
If the Contribution was created by an employee of the European Union or the European Atomic Energy Community (EU/Euratom) in the performance of their duties, the regulation 31/EEC, 11/EAEC (Staff Regulations) applies, and copyright in the Contribution shall, subject to the Publication Framework Agreement (EC Plug), belong to the European Union or the European Atomic Energy Community.
If Author is an officer or employee of the United States government, of the Crown, or of EU/Euratom, reference will be made to this status on the signature page.

§ 3 Rights Retained by Author

Author retains, in addition to uses permitted by law, the right to communicate the content of the Contribution to other scientists, to share the Contribution with them in manuscript form, to perform or present the Contribution or to use the content for non-commercial internal and educational purposes, provided the Springer publication is mentioned as the



original source of publication in any printed or electronic materials. Author retains the right to republish the Contribution in any collection consisting solely of Author's own works without charge subject to ensuring that the publication by Springer is properly credited and that the relevant copyright notice is repeated verbatim.
Author may self-archive an author-created version of his/her Contribution on his/her own website and/or the repository of Author's department or faculty. Author may also deposit this version on his/her funder's or funder's designated repository at the funder's request or as a result of a legal obligation. He/she may not use the publisher's PDF version, which is posted on SpringerLink and other Springer websites, for the purpose of self-archiving or deposit. Furthermore, Author may only post his/her own version, provided acknowledgment is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be provided by inserting the DOI number of the article in the following sentence: "The final publication is available at Springer via http://dx.doi.org/[insert DOI]". The DOI (Digital Object Identifier) can be found at the bottom of the first page of the published paper.
Prior versions of the Contribution published on non-commercial pre-print servers like ArXiv/CoRR and HAL can remain on these servers and/or can be updated with Author's accepted version. The final published version (in pdf or html/xml format) cannot be used for this purpose. Acknowledgment needs to be given to the final publication and a link must be inserted to the published Contribution on Springer's website, by inserting the DOI number of the article in the following sentence: "The final publication is available at Springer via http://dx.doi.org/[insert DOI]".
Author retains the right to use his/her Contribution for his/her further scientific career by including the final published paper in his/her dissertation or doctoral thesis provided acknowledgment is given to the original source of publication. Author also retains the right to use, without having to pay a fee and without having to inform the publisher, parts of the Contribution (e.g. illustrations) for inclusion in future work, and to publish a substantially revised version (at least 30% new content) elsewhere, provided that the original Springer Contribution is properly cited.

§ 4 Warranties

Author warrants that the Contribution is original except for such excerpts from copyrighted works (including illustrations, tables, animations and text quotations) as may be included with the permission of the copyright holder thereof, in which case(s) Author is required to obtain written permission to the extent necessary and to indicate the precise sources of the excerpts in the manuscript. Author is also requested to store the signed permission forms and to make them available to Springer if required.
Author warrants that he/she is entitled to grant the rights in accordance with Clause 1 "Rights Granted", that he/she has not assigned such rights to third parties, that the Contribution has not heretofore been published in whole or in part, that the Contribution contains no libelous statements and does not infringe on any copyright, trademark, patent, statutory right or proprietary right of others, including rights obtained through licenses; and that Author will indemnify Springer against any costs, expenses or damages for which Springer may become liable as a result of any breach of this warranty.

§ 5 Delivery of the Work and Publication

Author agrees to deliver to the responsible Volume Editor (for conferences, usually one of the Program Chairs), on a date to be agreed upon, the manuscript created according to the Springer Instructions for Authors. Springer will undertake the reproduction and distribution of the Contribution at its own expense and risk. After submission of the Consent to Publish form Signed by the Corresponding Author, changes of authorship, or in the order of the authors listed, will not be accepted by Springer.

§ 6 Author's Discount

Author is entitled to purchase for his/her personal use (directly from Springer) the Work or other books published by Springer at a discount of 33 1/3% off the list price as long as there is a contractual arrangement between Author and Springer and subject to applicable book price regulation. Resale of such copies or of free copies is not permitted.

§ 7 Governing Law and Jurisdiction

This agreement shall be governed by, and shall be construed in accordance with, the laws of Switzerland. The courts of Zug, Switzerland shall have the exclusive jurisdiction.

Corresponding Author signs for and accepts responsibility for releasing this material on behalf of any and all Co-authors.

Signature of Corresponding Author: **Date:** 7/9/2015

□ I'm an employee of the US Government and transfer the rights to the extent transferable (Title 17 § 105 U.S.C. applies)

□ I'm an employee of the Crown and copyright on the Contribution belongs to Her Majesty

□ I'm an employee of the EU or Euratom and copyright on the Contribution belongs to EU or Euratom

28.01.2014
14:07

In order for SIAM to include your paper in the 2016 SIAM International Conference on Data Mining proceedings, the following Copyright Transfer Agreement must be agreed to during the paper upload process.

COPYRIGHT TRANSFER AGREEMENT

Title of Paper:

Author(s):

Copyright to this paper is hereby irrevocably assigned to SIAM for publication in the **2016 SIAM International Conference on Data Mining (SDM16)**, May 5 -7, 2016 at the Hilton Miami Downtown, Miami, Florida, USA. SIAM has sole use for distribution in all forms and media, such as microfilm and anthologies, except that the author(s) or, in the case of a "work made for hire," the employer will retain:

> The right to use all or part of the content of the paper in future works of the author(s), including author's teaching, technical collaborations, conference presentations, lectures, or other scholarly works and professional activities as well as to the extent the fair use provisions of the U.S. Copyright Act permit. If the copyright is granted to SIAM, then the proper notice of the SIAM's copyright should be provided.
>
> The right to post the final draft of the paper on noncommercial pre-print serves like arXiv.org.
>
> The right to post the final version of the paper on the author's personal web site and on the web server of the author's institution, provided the proper notice of the SIAM's copyright is included and that no separate or additional fees are collected for access to or distribution of the paper.
>
> The right to refuse permission to third parties to republish all or part of the paper or translation thereof.

It is affirmed that neither this paper nor portions of it have been published elsewhere and that a copyright transfer agreement has not been signed permitting the publication of a similar paper in a journal or elsewhere. For multi-author works, the signing author agrees to notify all co-authors of his/her action.

Transfer of Copyright to the Publisher

SIAM strongly recommends this option. This transfer of copyright provides SIAM the legal basis not only to publish and to distribute the work, but also to pursue infringements of copyright (such as plagiarism and other forms of unauthorized use) and to grant permissions for the legitimate uses of the work by third parties. This option should not be selected if the work was prepared by a government office or employee as part of his or her official duties.

[__] By selecting the box at left, the Author hereby irrevocably assigns, conveys and transfers the copyright to the Work to SIAM. SIAM shall have sole rights of distribution and/or publication of the work in all forms and media, throughout the world, except for those rights given to the Author above.

[__] By selecting the box at left, the Author DOES NOT assign, convey and transfer the

copyright to the Work to SIAM. Please list in whose name copyright should
appear here: ____

Work Made for Hire
[__] Check here if signature is on behalf of employer in the event article is “work made for hire.”

Previously Published
Check here if portions have been published elsewhere and enclose appropriate credits and permissions to republish.
[__] Yes (if yes, expand site so authors can enclose credits and permissions)
[__] No

Alternate Copyright
[__] Check here to submit an alternative copyright. Send copyright to [*meeting manager e-mail*]using subject line “SIAM – [*proceedings* acronym] – Alternate Copyright – LAST NAME.”

Vita

Education

B.Sc., Budapest University of Technology and Economics, Budapest, Hungary, 2012

M.S., University of Illinois at Chicago, Chicago, Illinois, 2014

Ph.D., University of Illinois at Chicago, Chicago, Illinois, 2017

Honors and Awards

Dean's Scholar Fellowship, 2016–2017

Chicago Consular Corps Scholarship, 2015

BUTE Faculty of Sciences, Scientific Student Conference, Discrete Mathematics Section, 1st Prize and Special Presidential Award, 2011

BUTE Mathematics Competition, 3rd Prize, 2011

Employment

Software Engineering Intern, *Google Inc.*, Pittsburgh, Pennsylvania, Summer 2016

Software Engineering Intern, *Google Inc.*, Pittsburgh, Pennsylvania, Summer 2015

Research Scientist Intern, *Amazon.com, Inc.*, Seattle, Washington, Summer 2014

Teaching Assistant, *University of Illinois at Chicago*, Chicago, Illinois,
Fall 2013–Spring 2016

Teaching Assistant, *The University of Iowa*, Iowa City, Iowa,
Fall 2012–Spring 2013

Papers

A Confidence-Based Approach for Balancing Fairness and Accuracy. With Benjamin Fish and Jeremy Kun. In Proceedings of the 2016 SIAM International Conference on Data Mining (SDM 2016).

On the Computational Complexity of MapReduce. With Benjamin Fish, Jeremy Kun, Lev Reyzin and György Turán. In Proceedings of the 29th International Symposium on Distributed Computing (DISC 2015).

Interactive Clustering of Linear Classes and Cryptographic Lower Bounds. With Lev Reyzin. In Proceedings of the 26th International Conference on Algorithmic Learning Theory (ALT 2015).

Fair boosting: A case study. With Benjamin Fish and Jeremy Kun. ICML 2015 Workshop on Fairness, Accountability, and Transparency in Machine Learning (FAT ML 2015).

Network installation under convex costs. With Alexander Gutfraind, Jeremy Kun and Lev Reyzin. Journal of Complex Networks **4.2** (2016): 177–186.

Biclique coverings, rectifier networks and the cost of ε-removal. With Szabolcs Iván, Judit Nagy-György, Balázs Szörényi and György Turán. In Proceedings of the 16th International Workshop on Descriptional Complexity of Formal Systems (DFCS 2014).

Improved algorithms for splitting full matrix algebras. With Gábor Ivanyos and Lajos Rónyai. JP Journal of Algebra, Number Theory and Applications **28** (2013), 141–156.

Patents

Control method for a cooling system with variable cooling power and cooling system. European Patent Office, EP17155305.0 (patent pending)

Control method for an electrically excited motor and inverter. United States Patent and Trademark Office, US 15/413,433 (patent pending)

Control method for a converter and converter. German Patent and Trademark Office, DE 10 2016 004 282.6 (patent pending)

Control method for a cooling system with variable cooling power and cooling system. German Patent and Trademark Office, DE 10 2016 001 824.0 (patent pending)

Control method for an electromagnetic motor and inverter. German Patent and Trademark Office, DE 10 2016 000 743.5 (patent pending)

Professional Activities

Program committee member for the 24th International Conference on Machine Learning (ICML 2017)

Conference reviewer for the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), the 25th International Joint Conference on Artificial Intelligence (IJCAI 2016), and the 26th International Conference on Algorithmic Learning Theory (ALT 2015)

Reviewer for Mathematical Reviews