Lower Bounds in Computational Complexity from Information Theory, Algebra and Combinatorics

Sivaramakrishnan Natarajan Ramamoorthy

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2020

Reading Committee:

Anup Rao, Chair

Paul Beame

Dan Suciu

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Lower Bounds in Computational Complexity
from Information Theory, Algebra and Combinatorics

Sivaramakrishnan Natarajan Ramamoorthy

Chair of the Supervisory Committee:
Associate Professor Anup Rao
Paul G. Allen School of Computer Science and Engineering

In this thesis, we study basic lower bound questions in communication complexity, data structures and depth-2 threshold circuits, and prove lower bounds in these models by devising new techniques in information theory, algebra and combinatorics.

Communication Complexity: A central open problem in communication complexity is to determine whether the messages exchanged by two parties can be compressed if we know that the amount of information revealed by the parties about their inputs is small. We consider the compression question when the information revealed by one of the parties is much less than the information revealed by the other. In this setting, we prove two new improved compression schemes.

Data Structures: Our contribution to data structure lower bounds is threefold:
(a) Consider the Vector-Matrix-Vector problem, in which the data structure stores a $\sqrt{n} \times \sqrt{n}$ bit matrix and provides an algorithm to compute $u^\intercal M v \pmod 2$ for $\sqrt{n}$-bit vectors u, v. We prove new static data structure lower bounds for this problem, which improve upon the previous work of Chattopadhyay, Koucký, Loff, and Mukhopadhyay by a factor of $\log n$. Our proof uses a new technique by combining the discrepancy method from communication

complexity with a modification of cell sampling. This technique turns out to be more general, and can be used to prove strong lower bounds for data structures that err and have a binary query output.

(b) We show new connections between systematic linear data structures, linear data structures and matrix rigidity. Specifically, we prove the equivalence between systematic linear data structures and set rigidity, a relaxation of matrix rigidity that was defined by Alon, Panigrahy and Yekhanin. This equivalence not only sheds light on the difficulty of proving strong lower bounds against data structures but also suggests candidate rigid sets from data structures. We also use this equivalence to relate linear data structures and rigidity.

(c) We study data structures that maintain a set from $\{1, 2, \ldots, n\}$, allow insertion of new elements and report the median, minimum or predecessors of the set. In particular, we prove that if one of the operations of the data structure is non-adaptive and each cell in memory stores $O(\log n)$ bits, then some operation must take time $\Omega(\log n / \log \log n)$. This bound nearly matches the guarantees of binary search trees, whose insertions and predecessor operations can be made non-adaptive. Our lower bounds are obtained via the sunflower lemma from combinatorics.

Balancing Sets and Depth-2 Threshold Circuits: Majority and threshold circuits are important sub-classes of Boolean circuits. Kulikov and Podoslkii asked the question of finding the minimum fan-in required to compute the majority of n-bits using a depth-2 majority circuit. We identify a connection between this circuit question and Galvin's balancing sets problem from combinatorics, a well studied discrepancy-type question that was initiated by the work of Frankl and Rödl. We use this finding to prove tight bounds for both the circuit question and Galvin's problem. The proofs use polynomials over finite fields.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to thank Anup Rao for being an amazing advisor and for teaching me how to do mathematics. He gave me all the freedom I could have hoped for in terms of my research directions. Most importantly, he was always available to listen to my ideas and was generous about sharing his own. Anup has always had my back and he has been a pivotal part of my Ph.D. journey. I would also like to thank Paul Beame for his helpful advice, meticulous feedback on my writing and service on all my committees. I also want to thank Dan Suciu and Rekha Thomas for serving on my doctoral committee.

My experiences, both social and academic, in the UW Theory Group have been incredibly valuable. Thank you to Makrand Sinha, who I have known since the start of my Ph.D., for listening to all my random ideas, collaborating on numerous occasions, taking me climbing and bouldering, and introducing me to many coffee shops and fancy restaurants. Conference and workshop visits would not have been nearly as much fun without him around. Thanks to Cyrus Rashtchian for his collaboration and his fancy parties. I will always remember the conversations with Kira Goldner about food from different parts of the world and will always cherish chatting about research and Indian food recipes with Swati Padmanabhan. Summers on campus were definitely more fun and relaxing with the bike rides and canoe trips with Swati, Kuikui Liu and Sami Davies. Conversations with my office mates, Jeffrey Hon, Alireza Rezaei and Farzam Ebrahimnejad, are something I have always looked forward to. Thanks to Rahul Kidambi for being a good friend and innumerable conversations on so many

random topics. I am lucky to recently have found such good friends in Siddharth Iyer and Ashrujit Ghoshal.

Outside of my research, the University of Washington provided me with countless opportunities to learn, teach, travel and grow during my program. I would like to thank the Theory Group faculty for the fantastic courses, Anup and Stefano Tessaro for giving me the opportunity to lecture in their classes, the Theory Group students for exploring new concepts in the field with me, and Melody Kadenko and Elise deGoede Dorough for helping me navigate funding and administrative hurdles. Each of you has been instrumental in shaping my time in this program.

Many thanks to Amir Yehudayoff and Pavel Hrubeš for hosting me at the Technion and the Institute of Mathematics of the Czech Academy of Sciences, respectively. Special thanks to Shay Moran for putting me up at Yagur. I learned a lot during both visits, not only about mathematics but about the world. I thank Anup for arranging visits to The Henri Poincaré Institute and Banff International Research Station. Thank you to Omri Weinstein for inviting me to Columbia University and Sajin Koroth for hosting me at Simon Fraser University.

Outside of UW, I was lucky to be in the company of my amazing friends Amey Khanolkar, Kartik Iyer, Rajan Pawar, Armando Diaz Tolentino and Praveen Sekar. I will cherish the board game nights, potlucks, hiking challenging trails, camping trips, visits to national parks, and heated agreements about anything and everything that filled our days. Amey and Kartik were wonderful roommates. Exploring biking trails and coffee shops around the city with Armando, Amey and Kartik, cooking Pav Bhaji with excessive spices and butter at Rajan's place, and visits to Annapurna Cafe and India Bistro with Praveen have been some of my favorite moments. Thank you to Sambuddha Roy for his valuable advice, the countless restaurant visits for a Chole Bhature and his fantastic friendship. I also thank Ravi Kiran Raman, Navneeth Nair,

Rajesh Sridhar and Ashwin Mohandas, Meghana Bande and Madhavi Yenugula for the fun road trips.

Most of the work in this thesis was carried out in wonderful coffee shops in Seattle. Thank you to Cafe Solstice, Trabant, Slate, Caffé Vita, Capitol Coffee Works, Lighthouse Roasters, Milstead and Caffe Ladro for their excellent coffee and amicable environment to think in.

Last but not the least, I'm grateful to my Amma, Appa and Paati for their unwavering support.

DEDICATION

In memory of my grandfather, A. Ramamoorthy.

Chapter 1

INTRODUCTION

Understanding the limitations of concrete computational models is fundamental to computer science. At a high level, we want to understand if there are inherent obstructions in many algorithmic tasks preventing the design of efficient algorithms. For some tasks, the answer is easy, but for many more, our understanding is far from complete. We know that the median of n numbers can be computed in $O(n)$ time and a running time of $\Omega(n)$ must be incurred. On the other hand, it is still not known if the traveling salesman problem has an efficient polynomial time algorithm.

Showing limits to computation is often referred to as proving *lower bounds* in computational models. To elaborate further, consider the basic algorithmic task of sorting numbers. Merge Sort, Timsort and Heap Sort are some algorithms that sort n numbers in $O(n \log n)$ time. They are comparison based algorithms; in other words, all the operations of the algorithm depend only on the outcome of whether a number in the input is less than or equal to another number in the input. All the aforementioned algorithms make $\Theta(n \log n)$ comparisons. It is now well known that any comparison based algorithm that sorts n numbers must make at least $\Omega(n \log n)$ comparisons. This is an instance of a lower bound on the number of comparisons for sorting against comparison based algorithms. In general, we want to understand the minimum amount of *resources* required to solve a *specific task* in some *computational model*.

In this thesis, the computational models we study are communication complexity, data structures and depth-2 threshold circuits. We develop new techniques in information theory, algebra and combinatorics to prove new lower bounds for basic computational tasks. Additionally, we present lower bounds for data structures computing linear functions, which

elucidate (and sometimes simplify) many previously known techniques for proving such lower bounds (see Sections 4.1 and 4.2 in Chapter 4). We now discuss each of these models in detail along with the contributions of this thesis.

1.1 Communication Complexity

Communication complexity, introduced by Yao [105], is the study of the number of bits of exchange between two parties needed to compute a joint function of their inputs. As an example problem, each party receives an n-bit vector from $\{0,1\}^n$ as input and their goal is to communicate to check whether the inner product between their inputs is odd or even. This model is central owing to the ability to cast lower bound questions in various other computational models as questions in communication complexity. Moreover, we know a rich collection of techniques to prove lower bounds in communication complexity. In fact, many of the (best) known lower bounds in other computational models like data structures, streaming algorithms and Boolean circuits are derived from communication complexity.

The key parameter of interest is the number of bits of exchange between the two parties. Since we want this to be as little as possible, a natural attempt is for the parties to identify the *wasteful* bits exchanged to reduce the amount of communication. For example, if a bit is repeated multiple times, then all its repetitions are wasteful. Shannon's seminal work [94] lays the mathematical foundations to formalize what is wasteful by defining the amount of useful *information* carried in a message. In the case when the second party receives no input and the first party wants to communicate their input, Shannon defined the notion of *entropy* to capture the amount of information that the message carries, and he showed that the first party can *compress* the message to its entropy. In other words, the first party can communicate an equivalent set of messages with expected length at most one plus the entropy. Moreover, he proved that entropy is a lower limit, as all the relevant information has to be communicated. In the case when both parties receive inputs but only one party communicates, the information content of the message is captured by Shannon's mutual information between the first party's input and the message conditioned on the second party's

input. These results set the stage to explore what it means to compress when both parties communicate. This question of compressing protocols was posed by Chakrabarti, Shi, Wirth and Yao [29] (see also [17, 1, 92]).

We now state the protocol compression question more formally. Let X and Y be the random variables denoting the inputs to the parties, where the underlying joint distribution on X and Y is known to both parties. On inputs X and Y, the two parties exchange bits, where each bit communicated by a party depends on their input and the bits exchanged so far. Let M denote the random variable for the bits exchanged between the parties, which we refer to as the message of the protocol. A protocol refers to the function that determines each bit in M. We want to know if there is another communication protocol on inputs X and Y and message M' such that (a) $\mathbb{E}\left[|M'|\right] \ll \mathbb{E}\left[|M|\right]$, and (b) there exists a map g and the ℓ_1 distance between the distributions of M and $g(M')$ is bounded by a small constant less than $1/2$. We say that a protocol with message M simulates another with message M' if property (b) from above is satisfied.

Naturally, we want $\mathbb{E}\left[|M'|\right]$ as small as possible; however, it is not clear if there exist any non-trivial upper or lower bounds on $\mathbb{E}\left[|M'|\right]$. Building on [29, 17], Barak, Braverman, Chen and Rao [18] defined the notion of *internal information* for protocols. It turns out that the internal information of a protocol is a lower bound on $\mathbb{E}\left[|M'|\right]$, and it is viewed as a generalization of Shannon's mutual information. This is the amount of information that is revealed by the parties of a protocol π about their inputs: let $I^A_\pi = I(M; X|Y)$ denote the information revealed by the first party, and $I^B_\pi = I(M; Y|X)$ denote the information revealed by the second party. Then the *internal information cost* is defined to be $I_\pi = I^A_\pi + I^B_\pi$. To elaborate further, consider a protocol in which Alice sends her input X to Bob. In this protocol, the amount of information revealed by Alice is $I(X; M|Y) = H(X|Y)$; in other words, Bob learns X, which amounts to $H(X|Y)$ bits of information. The internal information cost turns out to be the most interesting for applications to lower bounds.

Equipped with these definitions, the compression question for communication complexity asks if the messages of a protocol can be compressed when its internal information cost is

low. Let π be a protocol with information I_π, where C_π is the number of bits exchanged in any of its execution. [18] showed that π can be simulated with $O\left(\sqrt{C_\pi \cdot I_\pi} \cdot \log C_\pi\right)$ bits of communication. Braverman [23] (also see [25]) designed a simulation with communication $2^{O(I_\pi)}$. Efficient simulations are known when the inputs X and Y are independent [18, 62, 95].

Apart from information-theoretic motivations, another reason to study protocol compression comes from connections to proving lower bounds via the *direct sum* question. The direct sum problem suggests a general strategy to prove lower bounds: How does the number of bits of communication to compute k independent copies of a function scales with the number of bits of communication required to compute the function on a single copy? Barak, Braverman, Chen and Rao [18] used their compression protocol to show that if C is the minimum number of bits required to compute a function, then $\gtrsim \sqrt{k} \cdot C$ bits of communication is required to compute k independent copies of the same function.

Our Contribution. We generalize and strengthen several of these results, in the case that $I_\pi^B \ll I_\pi^A$ and $I_\pi^B \ll C_\pi$. This case is interesting in part because many lower bounds for data structures involve proving lower bounds on so-called *lopsided* problems, problems where the optimal lower bound on communication for one party is much smaller than for the other party (see [72], [80], [14], [71], [70], [22], [58], [83]). Indeed, our techniques allow us to reprove an important and well known theorem of Pătraşcu [80] giving a tight lower bound on the communication complexity of lopsided disjointness. We shall elaborate more on this connection while discussing data structures.

Let π be a protocol with internal information $\left(I_\pi^A, I_\pi^B\right)$ in which at most C_π bits are exchanged in any execution. Our first compression is an analogue to [18], which guarantees that π can be simulated using $O\left(I_\pi^A + \sqrt[4]{C_\pi^3 \cdot I_\pi^B} \cdot \log C_\pi\right)$ bits. Specifically, this is better than [18] when $I_\pi^A \gg C_\pi^{3/4}$ and $I_\pi^B \ll C_\pi^{1/4}$. Our second compression is analogous to [25] - we show that π can be simulated with communication at most $I_\pi^A \cdot 2^{O\left(I_\pi^B\right)}$. Particularly, if $I_\pi^B = O(1)$, then the protocol can be optimally compressed, a result that was not known prior to this work. We also use our second compression to obtain a new *rectangle* lower bound,

which is used to simplify the result of Pătraşcu [80] on the communication complexity of lopsided disjointness. All the above mentioned results are joint work with Rao [74].

1.2 Data Structures

Data structures are ubiquitous algorithmic primitives that are crucial to the design of efficient algorithms. It is well known that data structures play a key role in the implementation of Dijkstra's shortest path algorithm, Kruskal's and Prim's algorithms for finding the minimum spanning tree, and many other algorithms. Informally, a data structure succinctly stores the input *data* in memory as a collection of *cells* and provides efficient algorithms to modify the data and compute a function of the data. Union-Find is an example of a data structure for graph algorithms, which supports the addition of new edges and allows checking whether pairs of vertices are connected.

Formally, for a data structure in the cell-probe model of Yao [106], the data $x \in D$ is stored in memory using s cells, where each cell is w bits long (s and w are referred to as the *space* and *word length* respectively). The data structure is equipped with a query and an update algorithm: (a) The query algorithm on input $q \in Q$ accesses a subset of cells in memory to compute a query function. The query time is defined to be the maximum number of accesses by the query algorithm on any data and query; (b) The update algorithm brings about changes to the underlying data by accessing and rewriting a subset of the cells. The update time is defined to be the maximum number of memory accesses and rewrites by the update algorithm on any data and update. Data structures that support both queries and updates are referred to as *dynamic* data structures, and the ones that only support queries but not any changes to the data are called *static* data structures. For example, in the case of Union-Find, D corresponds to all graphs on n vertices, Q corresponds to pairs of vertices and the updates correspond to adding an edge.

The basic question here is to understand the trade-offs between the space, word length, query time and update time. Proving such trade-offs is usually challenging. Several works have developed techniques to prove lower bounds on various data structure problems in both

the static setting [2, 70, 72, 71, 82, 83, 80, 77, 65, 32] and the dynamic setting [43, 81, 78, 79, 66, 33, 109, 102, 67]. In the static setting, two well known techniques for data structure lower bounds are via Miltersen's [70] (also see [72]) reduction to communication complexity and cell sampling due to Panigrahi, Talwar and Weider [77] and Larsen [66]. In the dynamic setting, all known lower bounds use the chronogram technique of Fredman and Saks [43] in conjunction with ideas from the static setting. We will discuss all of them in more detail in Chapters 4 and 6.

We now briefly discuss how the reduction to communication complexity [70] is connected to asymmetric communication problems discussed in the previous section. In the reduction, one party receives the data $x \in D$ and the other party receives the query $q \in Q$; the parties then simulate the query algorithm to compute the query function, where the party with q requests the contents at specific memory locations by communicating the indices (or the addresses) and the party with x responds with the contents. This implies that efficient data structures will lead to efficient protocols. Typically, $|D| \gg |Q|$, exhibiting an implicit asymmetry, not only in the number of bits communicated by each party but also on the information revealed by each party in this simulation.

Our contributions are both to static and dynamic data structures. We begin by discussing the Vector-Matrix-Vector problem in the static setting.

1.2.1 Vector-Matrix-Vector Problem

Let n be a square natural number and $\mathbb{F}_2$ be the field with 2 elements. In the Vector-Matrix-Vector problem, the input data is given by a matrix $M \in \mathbb{F}_2^{\sqrt{n} \times \sqrt{n}}$ and the query set Q consists of pairs $(u, v) \in \mathbb{F}_2^{\sqrt{n}} \times \mathbb{F}_2^{\sqrt{n}}$. The data structure stores M and its query algorithm computes $u^\intercal M v$. (Equivalently, the query set can be defined to be the set of all rank one matrices and the query algorithm on a rank one matrix $uv^\intercal$ as input computes $\mathsf{Trace}\,((uv^\intercal)M)$.) Observe that $|Q| = 2^{2\sqrt{n}}$. The Boolean semiring version of this problem has received much recent attention due to connections to the online Matrix-Vector multiplication conjecture [54]. Moreover, this problem has motivated the study of data structures for a super-polynomial

number of queries, even when the output is binary [30, 32]. Other prior work has either studied binary output problems with $\mathsf{poly}(n)$ queries (see e.g. [80, 83]) or achieved better lower bounds by looking at multi-output problems (see e.g. [33, 65]). In general, the Vector-Matrix-Vector problem is a good testbed for proving better data structure lower bounds because linear algebraic tools could provide new insights.

A straightforward data structure stores M itself using n bits, and the query algorithm computes $u^\intercal Mv$ by accessing all coordinates of M. Observe that the space and query time are both n. On the other end of the spectrum, the data structure can store the answers to all possible queries, and the query algorithm will just access the corresponding bit to produce the output. In this case, the space is $2^{2\sqrt{n}}$ and the query time is 1. Artazarov, Dinic, Kronrod and Faradzev [15] combined the above two strategies to obtain a data structure with space $n^3/\log^2 n$ and query time $n/\log^2 n$. Fix $k = \log n$ and $k' = \sqrt{n}/\log n$, and for simplicity assume that $\log n$ divides $\sqrt{n}$. For (u, v), let $u_1, \ldots, u_{k'}$ and $v_1, \ldots, v_{k'}$ be projections of u and v to k length consecutive intervals. Similarly, let $M_{1,1}, \ldots, M_{1,k'}, \ldots, M_{k',1}, \ldots, M_{k',k'} \in \mathbb{F}_2^{k\times k}$ be disjoint submatrices of M such that

$$u^\intercal Mv = \sum_{i=1,j=1}^{k'} (u_i)^\intercal M_{i,j} v_j.$$

For every i, j, we store $a^\intercal M_{i,j} b$ for all possible a, b. It is easy to see that the space is at most $(k')^2 \cdot 2^{2k} = n^3/\log^2 n$. The query algorithm on (u, v) computes $u^\intercal Mv$ by computing each term in $\sum_{i=1,j=1}^{k'} (u_i)^\intercal M_{i,j} v_j$. Since computing each term requires access to only one bit, the query time is at most $(k')^2 = n/\log^2 n$. This data structure achieves the best known trade-off between the space and query time when $w = 1$. Throughout this discussion, we have assumed that $w = 1$, but the data structures can be generalized for an arbitrary w by packing w bits in each cell.

From the point of view of lower bounds, the previously best known bound is due to Chattopadhyay, Koucký, Loff, and Mukhopadhyay [32]. Moreover, their lower bound holds for a randomized model in which the query algorithm has access to a random tape and can err in its computation. For constants c and c', they prove that for every M and (u, v), if the

query algorithm computes $u^\intercal Mv$ correctly with probability at least $\frac{1}{2} + \frac{1}{2^{c'\sqrt{n}}}$, then the query time is at least $\min\left\{\frac{c\sqrt{n}}{\log \frac{sw}{\sqrt{n}}}, \frac{cn}{w}\right\}$.

Our Contribution. We prove a new lower bound for the vector-matrix-vector problem in the high error regime. We show that for every M and (u, v), if the query algorithm is correct with probability at least $\frac{1}{2} + \frac{1}{2^{\sqrt{n}/64}}$, then the query time must be at least

$$\min\left\{\frac{c\sqrt{n}}{\log \frac{s\alpha}{n}}, \frac{cn}{\alpha}\right\},$$

where $0 < c \leq 1/36$ is a universal constant and $\alpha := 2(w + \log \frac{sw}{n})$. Our bound is strictly better than the bound in [32]. For example, in the linear space regime, when $s = O(n)$ and $w = O(1)$, we show that $t = \Omega(\sqrt{n})$, while the prior result gives only $t = \Omega(\sqrt{n}/\log n)$. This is joint work with Rashtchian [76].

Even when $w = 1$ and $s = O(n)$, it is a major open problem to improve the query time lower bound to $\omega(\sqrt{n})$. It is also open to prove better lower bounds against linear data structures, where the data structure stores linear functions of M and the query algorithm's computation is a linear function of the bits accessed. This leads to our next discussion on linear data structures in the static setting.

1.2.2 Linear Data Structures and Systematic Linear Data Structures

The *inner product problem* motivates the study of linear data structures. The task is to preprocess an n-bit vector x to compute inner products $\langle q, x\rangle$ over $\mathbb{F}_2$ for queries $q \in Q$, where $Q \subseteq \mathbb{F}_2^n$ is the *query set* and $\mathbb{F}_2$ is the field with 2 elements. This problem generalizes the Prefix-Sum problem [47] and the Vector-Matrix-Vector problem [30, 68]. They are important from the perspective of developing techniques to be able to prove stronger lower bounds. Though they are a restriction of the general cell-probe model, the highest known lower bounds in the linear model match the highest known lower bounds in the cell-probe model.

Formally, the input data is $x \in \mathbb{F}_2^n$ and the query set is given by $Q \subseteq \mathbb{F}_2^n$. Moreover, $w = 1$ and the data structure stores x as $\langle x, v_1\rangle, \ldots, \langle x, v_s\rangle$, for some $v_1, \ldots, v_s \in \mathbb{F}_2^n$. The

query algorithm on $q \in Q$ accesses a subset of the stored bits to compute $\langle q, x \rangle$. The query time is defined to be the maximum number of accesses by the query algorithm on any q and x. It is without loss of generality that the query algorithm's computation is a linear function of the accessed bits [59]. In other words, the query algorithm finds a minimum sized set $\{i_1, \ldots, i_k\}$ such that $q = v_{i_1} + \cdots + v_{i_k}$ and computes $\langle q, x \rangle$ using the identity $\langle q, x \rangle = \langle x, v_{i_1} \rangle + \cdots + \langle x, v_{i_k} \rangle$. The linear data structure model was introduced by Fredman [44, 45] (also see [107]) and is sometimes called the *semigroup arithmetic* model.

A simple counting argument will show that $t \geq \Omega\left(\frac{\log |Q|}{\log s}\right)$, where t is the query time of the data structure. Indeed, the number of vectors generated by taking $k \leq t$ linear combinations among $v_1, \ldots, v_s$ is at most $\binom{s}{t} \cdot 2^t$. At the same time, these vectors generated must cover Q. Hence, $\binom{s}{t} \cdot 2^t \geq |Q|$, implying the desired inequality. The best known lower bound on the query time t for an *explicit* query set Q is

$$t \geq \Omega\left(\frac{\log |Q|}{\log \frac{s}{n}}\right), \tag{1.1}$$

where Q is defined to be explicit if every element in Q can be computed in $\mathsf{poly}(n)$ time. Note that the bound in Eq. (1.1) is, at best, only better by a $\log n$ factor when compared to the counting lower bound. In contrast, if $s = O(n)$, $|Q| \geq n^2$ and Q is chosen uniformly at random, then the query time must be at least $\Omega\left(n/\log n\right)$ with high probability.

In an attempt to explain the difficulty in improving the bound in Eq. (1.1), recently, multiple connections between data structures and circuits have arisen [27, 34, 36, 101]. The premise of these results is that hard problems for data structures may shed new light on *rigid* matrices and circuits. A matrix is rigid if it is far in Hamming distance from low rank matrices.

We take a similar angle, and we prove an equivalence between the systematic linear model and rectangular rigidity, a notion that is very close to matrix rigidity. This model may only store x verbatim along with a small number $r \ll n$ of *redundant* bits, which are the evaluations of r linear functions of x. To compute $\langle q, x \rangle$ for $q \in Q$, the query algorithm must output a linear function of these r bits along with any t bits of x, where t is the *query time*.

We motivate this model with a simple upper bound. Suppose that the query set Q happens to be close to an r-dimensional subspace U. More precisely, assume that $d_H(q, U) \leq t$ for any $q \in Q$, where $d_H(q, U) := \min_{u \in U} d_H(q, u)$ and $d_H(q, u)$ denotes the Hamming distance. The systematic linear model will store r bits that correspond to inner products between x and some r vectors that form a basis for U. The query algorithm computes $\langle q, x \rangle$ by invoking the identity $\langle q, x \rangle = \langle u, x \rangle + \langle q - u, x \rangle$, using any vector $u \in U$ with $d_H(q, u) \leq t$. Indeed, the r precomputed bits suffice to determine $\langle u, x \rangle$, and at most t bits of x are needed to calculate $\langle q - u, x \rangle$.

Our Contribution. It turns out that the requirement of Q being close to a small dimensional subspace exactly corresponds to the notion of rigid sets, defined by Alon, Panigrahy and Yekhanin [11]. Our result shows that an efficient algorithm exists in the above model if and only if the query set is not rigid in their sense. Conversely, it is possible to derive new rigidity lower bounds by proving lower bounds for the systematic linear model. Formally we prove that any systematic linear data structure for Q with redundancy r has query time at least t if and only if for any r-dimensional subspace U, there is a point in Q at a Hamming distance of at least t from U.

As an application of our framework, we provide new results for the Vector-Matrix-Vector problem in the systematic linear model. Recall that the query set consists of $\sqrt{n} \times \sqrt{n}$ matrices with rank one. We lower bound the rigidity of this set when its elements are viewed as vectors, and consequently, we obtain a query time lower bound of $\Omega(n^{3/2}/r)$ for the systematic linear model with redundancy $r \geq \sqrt{n}$. Any asymptotically better lower bounds for this problem (in the systematic linear model) would directly imply that this query set is rigid with better parameters than the currently known results for explicit matrices [8, 11]. The results mentioned here are joint work with Rashtchian [76]

Dvir, Golovnev, and Weinstein also demonstrate a connection between rigidity and linear data structures [36]. For a query set $Q \subseteq \mathbb{F}_2^n$, they show that a query time lower bound of $\omega(\log |Q| \cdot \log n)$ for linear data structures with space $O(n)$ leads to a semi-explicit rigid

set. When $|Q| = m$, their result uses a $\mathsf{poly}(m)$ time algorithm that requires access to an NP oracle to produce the rigid set. Compared to their work, our connection preserves explicitness and offers a two-way equivalence via the systematic linear model. In particular, when $c'n \leq r \leq cn$ for constants $c', c < 1$, a lower bound of $t = \omega(\log m)$ in the systematic linear model implies that Q is rigid with better parameters than known results. Their work requires a lower bound of $t = \omega(\log m \log n)$ against the linear model, and the resulting set is not explicit. Our results also extend to show that linear data structure lower bounds lead to explicit rigid matrices. However, compared to the work of Dvir, Golovnev, and Weinstein, we require stronger lower bounds to achieve new rigidity parameters.

We now proceed to discuss dynamic data structures in the non-adaptive setting.

1.2.3 Data Structures for Maintaining Sets of Numbers

Data structures that maintain a set of numbers S and allow for quickly computing basic statistics like minimum, median or predecessors of the set are very useful. The median is the middle number of the set in sorted order, and the predecessor of a number x is the largest element in S that is at most x. For example, computing such statistics efficiently play a vital role in Prim's and Kruskal's algorithms for computing the minimum spanning tree. More generally, they play a key role in the performance of greedy algorithms.

The data structure stores a set S of numbers from $\{1, 2, \ldots, n\}$. The updates correspond to inserting numbers from $\{1, 2, \ldots, n\}$ to the underlying set and the queries correspond to computing either the minimum of S or the median of S or the predecessor of $x \in \{1, 2, \ldots, n\}$ in S. Here we are interested in understanding the trade-off between the time for performing different operations. For example, if we maintain the set S by storing its indicator vector (with $w = 1$), then elements can be inserted (and also deleted) from the set in time 1, but computing the minimum, median or predecessors of the set could take time $\Omega(n)$ in the worst case. However, if we maintained the set by storing the size of the set and its elements in sorted order (with $w = \log n$), then the minimum and median can be computed in time 2, the predecessors can be computed in time $O(\log n)$, but inserting elements into the set

would take time $\Omega(n)$. Binary search trees are a well-known data structure that maintain sets and allow one to compute the minimum, median and predecessors in time $O(\log n)$, when $w = \log n$. One can also use a very clever data structure due to van Emde Boas [100] that brings down the time required for all operations to $O(\log \log n)$, when $w = \log n$. The Fusion trees data structure of Fredman and Willard [46] takes $O(\log n / \log w)$ time for all operations.

We consider *non-adaptive* data structures solving the tasks discussed above. In general, for an update or query operation, the set of cells accessed by the algorithm can be viewed incrementally, in which the location of each access is not only a function of the input to the operation but also the contents of the previously accessed cells. Non-adaptivity is a straightforward restriction where the locations of all the accesses depend only on the input to the operation and not on the contents of the cells. The updates or insertions in a binary search tree can be performed in a non-adaptive fashion. Non-adaptive data structures tend to be simple and faster in practice. This is because a practical implementation can load all of the cells required to perform the operation into a local cache in a single step, rather than having to fetch cells from the memory multiple times.

Our Contribution. Let t_{ins} be the time for insertions, t_{med} be the time to compute the median and t_{pred} be the time to compute predecessors. We prove new lower bounds on non-adaptive data structures that allow for computing the median, minimum, and predecessors.

(a) If the data structure performs non-adaptive deletions and minimum computations, then at least one operation must take $\Omega\left(\frac{\log n}{\log \log n + \log w}\right)$ time.

(b) If the data structure supports non-adaptive insert operations, then $t_{\mathsf{med}} \geq \Omega\left(\frac{n^{\frac{1}{t_{\mathsf{ins}}+1}}}{w^2 \cdot t_{\mathsf{ins}}^2}\right)$. As a corollary, we get that $\max\{t_{\mathsf{ins}}, t_{\mathsf{med}}\} \geq \Omega\left(\frac{\log n}{\log \log n + \log w}\right)$.

(c) If the predecessor computations are non-adaptive, then either $t_{\mathsf{pred}} \geq \Omega\left(\frac{\log n}{\log \log n + \log w}\right)$ or $t_{\mathsf{ins}} \geq \Omega\left(n^{\frac{1}{2(t_{\mathsf{pred}}+1)}}\right)$.

All our lower bounds are obtained using the sunflower lemma of Erdős and Rado [40], and are joint work with Rao [75].

1.3 Balancing Sets and Depth-2 Threshold Circuits

Balancing set families are families of proper non-empty subsets of a finite universe that satisfy a *discrepancy* type property. They are well studied objects in combinatorics [42, 38, 6, 53, 10, 52], and they have found many applications in computer science [6, 90, 57, 10, 52], specifically to obtain lower bounds on the size of arithmetic circuits. Here, we prove lower bounds on the size of such families and then use them to prove lower bounds on depth-2 *majority* and *threshold circuits* that compute the majority and *weighted threshold* functions.

For a positive integer n, let $[n]$ denote the set $\{1, 2, \ldots, n\}$. Proper non-empty subsets $S_1, \ldots, S_k \subset [n]$ are called a balancing family if for every $X \subset [n]$ of size $n/2$ there is an $i \in [k]$ such that $|S_i \cap X| = |S_i|/2$. The existence of such families of small size was first shown by Galvin [42, 38]. In his construction, n is assumed to be divisible by 4. Each set in the family has size $n/2$ and the family is given by

$$S_1 = \{1, 2, \ldots, n/2\}, S_2 = \{2, 3, \ldots, n/2 + 1\}, \ldots, S_{n/2} = \{n/2 + 1, n/2 + 2, \ldots, n\}.$$

For any $X \subset [n]$ of size $n/2$, we claim that there is an $i \in [n/2]$ such that $|S_i \cap X| = n/4$. Indeed, first observe that $|S_1 \cap X| + |S_{n/2} \cap X| = n/2$. Second, for each $j \in [n/2 - 1]$, $||S_j \cap X| - |S_{j+1} \cap X|| \in \{-1, 0, 1\}$. If $|S_1 \cap X| = n/4$, then we are done. Otherwise, if $|S_1 \cap X| < n/4$, then the first observation implies that $|S_{n/2} \cap X| > n/4$. Therefore, by the second observation, there must be a $j \in \{2, 3, \ldots, n/2 - 1\}$ such that $|S_j \cap X| = n/4$. An analogous argument holds when $|S_1 \cap X| > n/4$.

Our Contribution. The goal is to understand how small can k be for balancing set families to exist. Let $n = 2p$ for a prime p. We prove that if $S_1, \ldots, S_k$ are a balancing family, then $k \geq n/2 = p$. Moreover, there is a balancing family with $n/2 = p$ sets. Previously, Alon, Kumar and Volk [10] showed that $k \geq \Omega(n)$. For arbitrary values of $n \geq 4$, we prove that

$k \geq n/2 - o(n)$. We also prove new lower bounds on other notions of balancing sets defined in the literature. In addition, we define a new notion of *unbalancing sets* and prove tight bounds on the size of unbalancing sets – see Chapter 7 for more details.

Balancing set families are closely related to depth-2 majority circuits computing the majority of n-bits. Let $\mathsf{MAJ}(x_1, \ldots, x_n)$ denote the majority of n-bits. A depth-2 majority circuit is one that computes $\mathsf{MAJ}(\mathsf{MAJ}(A_1), \ldots, \mathsf{MAJ}(A_{k'}))$, where each input A_i corresponds to a subset of the input bits $x_1, \ldots, x_n$ and some number of constants in $\{0, 1\}$. The top fan-in of the circuit is defined to be k' and the bottom fan-in of the circuit is the maximum size of the inputs $A_1, \ldots, A_{k'}$. One can prove that if n is even and there is a balancing family with k sets, then there is a depth-2 majority circuit computing majority of n-bits with top fan-in at most $2k + 2$. Indeed, let $S_1, \ldots, S_k$ be a balancing set family. Define k majority gates, each on variables indexed by S_i, another k majority gates, each on variables indexed by $[n] \setminus S_i$ and finally 2 majority gates, each with 0 as input. The top majority gate, with fan-in $2k + 2$, reads these $2k + 2$ gates. We now explain why this circuit correctly computes the majority function. For each S_i, let $S_i^c = [n] \setminus S_i$, and define $X = \{i \mid x_i = 1\}$. Observe that $|S_i \cap X| + |S_i^c \cap X| = |X|$. Moreover, if $|X| < n/2$, then only one among $|S_i \cap X| \geq |S_i|/2$ and $|S_i^c \cap X| \geq |S_i^c|/2$ holds. In other words, if the number of ones in the input is less than $n/2$, only k of the bottom majority gates output 1. When $|X| = n/2$, by the property of balancing sets, there is an i such that $|S_i \cap X| = |S_i|/2$ and $|S_i^c \cap X| = |S_i^c|/2$. So, at least $k + 1$ of the bottom majority gates output 1. It is the same outcome when $|X| > n/2$.

$\mathsf{MAJ}(\mathsf{MAJ}(x_1, \ldots, x_n))$ computes the majority of n-bits, where the top fan-in is 1 and the bottom fan-in is n. This naturally leads to the question of what the top fan-in must be when the bottom fan-in is at most $n - 1$; a priori, it is not clear if depth-2 majority circuits with both top and bottom fan-in at most $n - 1$ can compute the majority of n-bits. Kulikov and Podolskii [63] asked the following question: What is the minimum fan-in required to compute majority using a depth-2 majority circuit? They showed that any depth-2 circuit computing majority must have fan-in at least $\Omega\left(n^{0.7}\right)$. Following them, Engle, Garg, Makina and Rao [37] proved that the fan-in must be at least $(n^{0.8})$ if no gate has constants as input.

Amano [12] showed the existence of a depth-2 circuit that computes majority with top fan-in $n/2+2$ and bottom fan-in $n-2$.

Our Contribution. We prove that in any depth-2 majority circuit computing the majority of n-bits, either the top fan-in or the bottom fan-in must be at least $n/2-o(n)$. We also show new lower bounds on the top fan-in of depth-2 threshold circuits with unbounded bottom fan-in computing a weighted threshold function.

The results discussed in this section are joint work with Hrubeš, Rao and Yehudayoff [56]. All our lower bounds on balancing sets, depth-2 majority circuits and depth-2 threshold circuits are proved using the polynomial method. Specifically, our lower bounds crucially use a lower bound on the degree of a special class of polynomials: Let p be a prime and let f be a polynomial over $\mathbb{F}_p$ on $2p$ variables, where $\mathbb{F}_p$ is the field with p elements. Let f be such that (a) f vanishes on all $x \in \{0,1\}^p$ with p zeros and (b) f is non-zero on the all zeros input. Then, the degree of f is at least p.

Outline

The rest of the thesis is organized as follows. Chapter 2 sets up all necessary notation, definitions and proposition used in the chapters that follow. Notation and preliminaries specific to each chapter is discussed in the respective ones. Our results on communication complexity and asymmetric compression is in Chapter 3. Chapter 4 has the proof of the lower bound on the Vector-Matrix-Matrix problem, along with a discussion of all known static data structure lower bounds by applying them to the *Vector-Vector* and the Vector-Matrix-Vector problems. The connections between different linear data structure models and rigidity is presented in Chapter 5 and the non-adaptive dynamic data structure lower bounds are in Chapter 6. Chapter 7 contains our results on balancing sets, unbalancing sets and depth-2 threshold circuits. Finally, we conclude with some open questions in Chapter 8.

Chapter 2

PRELIMINARIES

We begin by defining the notation used in this thesis. Unless otherwise stated, logarithms are computed base two. For a natural number ℓ, $[\ell]$ denotes the set $\{1, 2, \ldots, \ell\}$. Given $a = a_1, a_2, \ldots, a_n$, we write $a_{\leq i}$ to denote $a_1, \ldots, a_i$. We define $a_{>i}$ and $a_{\leq i}$ similarly. For a prime p, $\mathbb{F}_p$ denotes the finite field on p elements. When $x \in \{0,1\}^n$ or $x \in \mathbb{F}_p^n$, $x[i]$ denotes its i-th coordinate, for $i \in [n]$. In general, if x is an n length vector, $x[i]$ is its i-th coordinate. Similarly, for a $n \times n'$ matrix M, $M[i,j]$ denotes the entry indexed by the i-th row and the j-th column.

Basic Inequalities

Recall two standard binomial estimates:

Proposition 2.1. *For integers $0 \leq k \leq \ell$,*

(a) $\log \binom{\ell}{k} \leq k \cdot \log \frac{e\ell}{k}$.

(b) if $k \leq \ell/16$, then $\sum_{i=0}^{k} \binom{\ell}{i} \leq 2^{\ell/4}$.

Proposition 2.2. *For $x \in [0, 1/2]$,* $\log(1/(1-x)) \leq 3x$.

Proof. Let $T = \ln(1/(1-x))$, where ln is the logarithm function to the base e. The Taylor expansion of $\ln(1/(1-x))$ gives,

$$T = \ln(1/(1-x)) = x + x^2/2 + x^3/3 + \cdots = x + x(x/2 + x^2/3 + x^3/4 + \cdots) \leq x + x \cdot T.$$

This implies, $T \leq x/(1-x)$. Since, $x \leq 1/2$, $T \leq 2x$. Now,

$$\log(1/(1-x)) = \ln(1/(1-x))/\ln(2) \leq 1.5 \ln(1/(1-x)) \leq 3x,$$

which follows from the fact that $1/\ln(2) \leq 1.5$ □

Proposition 2.3. *For every $a, b \geq 1$ and $c > 2$, if $a \log ab \geq c$, then $a \geq \frac{c}{\log c + \log b}$.*

Proof. Suppose that $a < \frac{c}{\log c + \log b}$. We then have,

$$a \log ab < \frac{c}{\log c + \log b} \cdot (\log b + \log c - \log(\log c + \log b)) < c,$$

where the last inequality follows from the fact that $c > 2$. This contradicts $a \log ab \geq c$, and therefore, $a \geq \frac{c}{\log c + \log b}$. □

Probability Basics

Random variables are denoted by capital letters and values they attain are denoted by lowercase letters. For example, A may be a random variable and then a denotes a value A may attain and we may consider the event $A = a$. We use the notation $p(a)$ to denote both the distribution on the variable a, and the number $\Pr[A = a]$. The meaning will be clear from context. We write $p(a|b)$ to denote either the distribution of A conditioned on the event $B = b$, or the number $\Pr[A = a|B = b]$. Again, the meaning will be clear from context. Given a distribution $p(a, b, c, d)$, we write $p(a, b, c)$ to denote the marginal distribution on the variables a, b, c (or the corresponding probability). We often write $p(ab)$ instead of $p(a, b)$ for conciseness of notation. If W is an event, we write $p(W)$ to denote its probability according to p. If W is an event that is a function of a random variable A, then $\Pr_a[W]$ denotes the probability of W according to the distribution on A. Moreover, if W is an event, then W^c denotes its complement. We denote by $\mathbb{E}_a[g(a)]$ the expected value of $g(a)$ with respect to a distributed according to its distribution p. If p is not explicitly specified, then $\mathbb{E}_a[g(a)]$ is the expected value of $g(a)$ with respect to a being uniformly distributed.

For two distributions p, q, we write $|p(a) - q(a)|$ to denote the ℓ_1 distance between the distributions p and q. We write $p \overset{\epsilon}{\approx} q$ if $|p - q| \leq \epsilon$.

Proposition 2.4. *Let $p(x), q(x)$ be two distributions and F be an event such that $p(x|F) \overset{\epsilon}{\approx} q(x)$. Then if $p(F) \geq 1 - \gamma$, we have $p(x) \overset{\epsilon+2\gamma}{\approx} q(x)$.*

Proof. The ℓ_1 distance between p, q can be expressed as $2\max_T(p(T) - q(T))$, where the maximum is taken over all subsets of the support of $p(x)$. Let T be the maximizer. Then

$$\begin{aligned}|p(x) - q(x)| = 2(p(T) - q(T)) &= (2(p(T|F)p(F) + p(F^c)p(T|F^c)) - q(T)) \\ &\leq 2(p(T|F) - q(T)) + 2p(F^c) \\ &\leq \epsilon + 2\gamma,\end{aligned}$$

as required. □

Proposition 2.5. *Let $p(x), q(x)$ be two distributions and F be an event such that $p(x) \overset{\epsilon}{\approx} q(x)$. Then if $p(F) \geq 1 - \gamma \geq 3/4$, we have $p(x|F) \overset{\epsilon+4\gamma}{\approx} q(x)$.*

Proof. The ℓ_1 distance between $p(x|F), q(x)$ can be expressed as $2\max_T p(T|F) - q(T)$, where the maximum is taken over all subsets of the support of $p(x)$. Let T be the maximizer. Then

$$|p(x|F) - q(x)| = 2(p(T|F) - q(T)) = 2(p(T, F)/p(F) - q(T)) \leq 2(p(T)/p(F) - q(T)).$$

Since $p(F) \geq 1 - \gamma$, we get that

$$2(p(T)/p(F) - q(T)) \leq 2(p(T)/(1 - \gamma) - q(T)) \leq 2(p(T)(1 + 2\gamma) - q(T)) \leq \epsilon + 4\gamma,$$

as required. □

The following proposition relates the bias of a binary random variable to the bias of the sum of n its independent copies.

Proposition 2.6. *Let $0 \leq \epsilon \leq 1$, n be a natural number and X be a binary random variable such that $\Pr[X = 0] = \frac{1}{2} \cdot (1 + \epsilon)$. If $X_1, \ldots, X_n$ are identically and independently distributed according to X, then*

$$\Pr\left[X_1 + \ldots + X_n = 0 \pmod 2\right] = \frac{1}{2} \cdot \left(1 + \epsilon^n\right).$$

Proof. We prove it by induction on n. For the base case, take $n = 1$, which is true by the premise of the proposition. The induction hypothesis states that the proposition is true for

$n-1$, and we will now prove it for n. We have,

$$\begin{aligned}\Pr\left[X_1+\ldots+X_n=0 \pmod 2\right] &= \sum_{c=0}^{1}\Pr\left[X_1+\ldots+X_{n-1}=c \pmod 2\right]\cdot\Pr[X_n=c]\\ &= \frac{1}{2}\left(1+\epsilon^{n-1}\right)\cdot\frac{1}{2}\left(1+\epsilon\right)+\frac{1}{2}\left(1-\epsilon^{n-1}\right)\cdot\frac{1}{2}\left(1-\epsilon\right)\\ &= \frac{1}{2}\cdot\left(1+\epsilon^{n}\right),\end{aligned}$$

where the second equality follows from the induction hypothesis. □

Information Theory Basics

The *entropy* of a discrete random variable A, is defined to be $\mathsf{H}_p(A)=\sum_a p(a)\cdot\log\frac{1}{p(a)}$, where we take $0\cdot\log(1/0)=0$. For $0\le x\le 1$, the binary entropy function is defined to be

$$\mathsf{h}(x)=x\log\frac{1}{x}+(1-x)\log\frac{1}{1-x}.$$

For two random variables A, B, the entropy of A conditioned on B is defined as

$$\mathsf{H}_p(A|B)=\sum_{a,b}p(ab)\cdot\log\frac{1}{p(a|b)}.$$

This is always a non-negative quantity and is at most $\log|\mathsf{Supp}(A)|$. When the underlying distribution p is clear from the context, we sometimes drop it from the notation.

The entropy satisfies some useful properties:

Proposition 2.7 (Chain Rule)**.** $\mathsf{H}(A_1A_2|B)=\mathsf{H}(A_1|B)+\mathsf{H}(A_2|BA_1)$.

Proposition 2.8 (Subadditivity)**.** $\mathsf{H}(A_1A_2|B)\le\mathsf{H}(A_1|B)+\mathsf{H}(A_2|B)$.

The *divergence* between two distributions is defined to be

$$\mathbb{D}\left(\frac{p}{\!\!=\!\!}\,q\right)=\sum_a p(a)\cdot\log\frac{p(a)}{q(a)}.$$

The *mutual information* between two random variables A,B, conditioned on C is defined to be

$$I_p(A;B|C)=\sum_{a,b,c}p(abc)\cdot\log\frac{p(abc)}{p(a|c)p(b|c)}.$$

This is always a non-negative quantity and is at most $\log |\mathsf{Supp}(A)|$. When the underlying distribution p is clear from the context, we sometimes omit it from the notation. Mutual information satisfies the chain rule:

Proposition 2.9 (Chain Rule)**.** $I(A_1A_2; B|C) = I(A_1; B|C) + I(A_2; B|A_1C)$.

Pinsker's inequality bounds the ℓ_1 distance in terms of the divergence:

Proposition 2.10 (Pinsker's Inequality)**.** $\mathbb{D}\left(\frac{p}{\!=\!}{q}\right) \geq \frac{1}{2} \cdot |p - q|^2$.

An alternate formulation is as follows:

Proposition 2.11 (Alternate Pinsker's)**.** $\mathbb{E}_{p(bc)} \left[|p(a|bc) - p(a|c)|\right] \leq \sqrt{I(A; B|C)}$.

The chain rule easily gives the following inequality:

Proposition 2.12 (Data Processing Inequality)**.** *If the random variable A determines B, then $I(A; C) \geq I(B; C)$.*

Proposition 2.13. *[48] Let $p(ab)$ be a distribution and $q(a)$ be another. Then*

$$\mathop{\mathbb{E}}_{p(b)} \left[\mathbb{D}\left(\frac{p(a|b)}{\!=\!}{p(a)}\right)\right] \leq \mathop{\mathbb{E}}_{p(b)} \left[\mathbb{D}\left(\frac{p(a|b)}{\!=\!}{q(a)}\right)\right].$$

We shall sometimes deal with distribution on strings of variable length. We have the following proposition, which follows from Shannon's source coding theorem:

Proposition 2.14. *Let B, C be random variables. Suppose A is a random variable supported on binary strings of length up to n, such that no string in the support of A is a prefix of another string in the support of A. Then $I(A; B|C) \leq \mathsf{H}(A) \leq \mathbb{E}\left[|A|\right]$.*

Chapter 3

ASYMMETRIC COMPRESSION

Recall that we are interested in the problem of compressing the communication between two parties when the information revealed by them is small. More specifically, in any protocol, we consider the amount of information revealed by each party separately and prove new compression schemes when the information revealed by one party is much less than the information revealed by the other party and the total communication. In this chapter, two such simulations that achieve better compression in this setting are discussed. Moreover, this simulation is used to prove a new rectangle lower bound, which is then used to show a lower bound on the communication complexity of lopsided disjointness.

We begin with a formal treatment of communication complexity. In communication complexity, we have two parties, Alice and Bob, with inputs X and Y respectively, where the inputs are drawn from a distribution that is known to both parties. Alice and Bob both have access to a public random string R, in addition their own private random strings. The *message* or *transcript* of the protocol is denoted by $M = M_1, M_2, \ldots$, where each $M_i \in \{0, 1\}$ is sent either by Alice or by Bob. If M_i is communicated by Alice, then M_i is a function of $X, R, M_{<i}$ and Alice's private random string. Analogously, if M_i is sent by Bob, then M_i is a function of $Y, R, M_{<i}$ and Bob's private random string. In any reference to a protocol, we mean the function that determines each bit in the transcript.

Given a protocol π that operates on inputs X, Y drawn from a distribution μ using public randomness[1] R and message M, we write $\pi(xymr)$ to denote the joint distribution of these

[1]In this dissertation we define protocols where the public randomness is sampled from a continuous (i.e. non-discrete) set. Nevertheless, we often treat the randomness as if it were supported on a discrete set, for example by taking the sum over the set rather than the integral. This simplifies notation throughout our proofs, and does not affect correctness in any way, since all of our public randomness can be approximated to arbitrary accuracy by sufficiently dense finite sets.

variables. We write $\|\pi\|$ to denote the *communication complexity* of π, namely the maximum number of bits that may be exchanged by the protocol in any execution. The maximum number of alternations between messages sent by Alice and those sent by Bob is called the number of *rounds* of the protocol. For a more involved introduction to communication complexity, we refer the reader to the books [64, 89].

Our work relies on ways to measure the information complexity of a protocol. The two well known notions of information cost are

(a) **External Information:** Chakrabarti, Shi, Wirth and Yao defined the information cost of a protocol π to be $I(XY; MR)$. In words, this is the amount of information an external observer learns about the inputs to the parties from the message. Barak, Braverman, Chen and Rao [18] called this measure the *external* information cost of a protocol, which we will write as I^{ext}_{π}.

(b) **Internal Information:** Barak, Braverman, Chen and Rao [18] identified another measure of information called the *internal* information cost of the protocol. The *internal information cost* of a protocol π is defined to be $I_{\pi} = I_{\pi}(X; M|YR) + I_{\pi}(Y; M|XR)$. This quantity is the sum of the information learned by Alice about Bob's input from the message, $I^{B}_{\pi} = I_{\pi}(Y; M|XR)$, and the information learned by Bob about Alice's input from the message, $I^{A}_{\pi} = I_{\pi}(X; M|YR)$. We will sometimes say that the internal information is (I^{A}, I^{B}) when we want to consider the values of both quantities instead of the sum.

Since each of the parties already knows one of the inputs, the internal information cost is never more than the external information cost, with equality when the inputs to the parties are independent of each other. Formally,

Proposition 3.1. *If π is a protocol with communication complexity C_{π}, internal information I_{π} and external information I^{ext}_{π}, then $I_{\pi} \leq I^{\mathsf{ext}}_{\pi} \leq C_{\pi}$.*

Here, we only consider the internal information cost for protocols. Internal information cost turns out to be the most interesting for applications to lower bounds. Indeed, Braverman and Rao [24] showed that the internal information cost required to compute a function is exactly equal to the amortized communication complexity of the function – this quantity has a very natural interpretation, seemingly independent of information theory.

There has been a lot of interest on protocol compression in the past decade. [18] proved that if the protocol π has external information cost I^{ext}_{π} and communication C_{π}, then one can simulate the protocol with $O\left(I^{\mathsf{ext}}_{\pi} \log C_{\pi}\right)$ bits of communication, which is optimal up to the factor of $\log C_{\pi}$. For internal information, [18] showed that any protocol π can be simulated in $O\left(\sqrt{I_{\pi} C_{\pi}} \cdot \log C_{\pi}\right)$ bits of communication. If we wish the simulation to not have a dependence on the communication of the original protocol, Braverman [23] showed that one can carry out the simulation using communication complexity $2^{O(I_{\pi})}$ (see also [25, 61]), a result that was subsequently proven to be tight by Ganor, Kol and Raz [48, 49] (see also [88]). In addition, Braverman and Weinstein [25] (see also [61]) showed that if a function is computed by π, then the space of inputs must contain a nearly *monochromatic rectangle* of density $2^{-O(I_{\pi})}$, a fact that can be used to prove lower bounds on the information complexity of computing functions.

When X and Y are independent, [18] proved that any protocol π with internal information cost I_{π} and communication C_{π} can be simulated by another protocol with $O\left(I_{\pi} \cdot \log C_{\pi}\right)$ bits of communication. Kol [62] improved this bound to $O\left(I^2_{\pi} \cdot \mathsf{polylog}\left(I_{\pi}\right)\right)$, and later Shersstov [95] obtained a near optimal bound of $O\left(I_{\pi} \cdot \log^2 I_{\pi}\right)$. In the setting of bounded round communication, Braverman and Rao [24] showed that a single message can be compressed to its internal information, giving a protocol that can simulate any r-round protocol with internal information cost I using $I + O(r)$ bits of communication.

Before stating our results, we introduce some new definitions. Let $q(x, y, a)$ be an arbitrary distribution. We say that a protocol π *δ-simulates* q, if there is a function g and a

function h such that

$$\pi(x, y, g(x, r, m), h(y, r, m)) \overset{\delta}{\approx} q(x, y, a, a),$$

in which $q(x, y, a, a)$ is the distribution on 4-tuples (x, y, a, a) where (x, y, a) are distributed according to q. Thus if π δ-simulates q, the protocol allows the parties to sample a according to $q(a|xy)$. If λ is a protocol with inputs x, y, public randomness r' and messages m', we say that π δ-simulates λ if π δ-simulates $\lambda(x, y, (r', m'))$. We say that π simulates λ if π δ-simulates λ for a constant δ.

We shall sometimes refer to the *expected communication*, or *expected number of rounds* of a protocol π. We note here that one can always use a bound on the expected communication or number of rounds to get a bound on the worst case communication, via the following proposition:

Proposition 3.2. *If π has expected communication c, then it can be γ-simulated by a protocol with communication c/γ.*

All our results in this chapter are joint work with Rao [74]. Our first theorem is somewhat analogous to the result of [18]. We show

Theorem 3.1. *Every protocol π can be ϵ-simulated by a protocol with expected communication $O\left(I_\pi^A + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log(1/\epsilon) + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log \|\pi\|\right)$.*

Our second result is an analogue of [23]:

Theorem 3.2. *π can be simulated in communication complexity $I_\pi^A \cdot 2^{O(I_\pi^B)}$.*

Theorem 3.2 shows that when the information revealed by one of the parties is a constant, the communication is within a constant factor of the information. As a corollary to Theorem 3.2, we show the following.

Corollary 3.1. *If π computes $f(x, y)$, then there exists a rectangle $S \times T$ such that*

$$\Pr[x \in S] \geq 2^{-O(I_\pi^A)}, \qquad \Pr[y \in T | x \in S] \geq 2^{-O(I_\pi^B)},$$

and there exists a constant $c \in \{0,1\}$ *such that*

$$\Pr[f(x,y) = c | (x,y) \in S \times T] \geq 2/3.$$

One can view Corollary 3.1 as defining an asymmetric notion of discrepancy, and showing that if the information complexity is small, then the discrepancy must be large. Corollary 3.1 is a useful tool to prove lower bounds on lopsided problems. We illustrate this by using the ideas going into Corollary 3.1 to give optimal lower bounds on the communication complexity of lopsided disjointness (a bound first proved by Pătraşcu [80]).

The Table 3.1 summarizes the simulation results discussed thus far. The proof of Theorem 3.1 is in Section 3.1 and the proof of Theorem 3.2 is in Section 3.2. Section 3.3 has the proof of Corollary 3.1 along with the lower bound for lopsided disjointness.

Reference	Communication Complexity of the Simulation
[18]	$O\left(\sqrt{I \cdot C} \log C\right)$
[23]	$2^{O(I)}$
[24]	$I + O\left(\sqrt{r \cdot I} + 1\right) + r \log(1/\epsilon)$
Theorem 3.1	$O\left(I^A + \sqrt[4]{C^3 \cdot I^B} \cdot \log(1/\epsilon) + \sqrt[4]{C^3 \cdot I^B} \cdot \log C\right)$
Theorem 3.2	$I^A \cdot 2^{O\left(I^B\right)}$
[18]	$O\left(I \cdot \log C\right)$ (when inputs are independent)
[62]	$O\left(I^2 \cdot \mathsf{polylog}(I)\right)$ (when inputs are independent)
[95]	$O\left(I \cdot \log^2 I\right)$ (when inputs are independent)

Table 3.1: Known bounds on the complexity of simulating r-round protocols with communication C and information $\left(I^A, I^B\right)$. In addition, $I = I^A + I^B$.

Our proofs use the following two results from past work.

Theorem 3.3 ([24]). *For every $\epsilon > 0$, if π is a protocol with internal information cost I and r rounds in expectation, then π can be ϵ-simulated with expected communication $I + O(\sqrt{r \cdot I} + 1) + r \log(1/\epsilon)$.*

Theorem 3.4 ([24]). *For any f, μ, ϵ, let π be the protocol computing f on n independent pairs of inputs, each drawn from the distribution μ and probability of error is at most ϵ on each pair, then there exists a protocol τ computing f on a single input pair with communication $||\tau|| = ||\pi||$, information $I^A_\tau \leq \frac{I^A_\pi}{n}$, $I^B_\tau \leq \frac{I^B_\pi}{n}$ and probability of error at most ϵ.*

3.1 Compressing Protocols with Asymmetric Information - I

In this section, we present the proof of Theorem 3.1. We compress the given protocol in two steps. In the first step, we convert the protocol π into a bounded round protocol, while controlling its internal information cost. In the second step, we apply Theorem 3.3 to conclude the proof. The first step is captured by the following theorem:

Theorem 3.5 (Bounded round simulation). *Given any protocol π and a parameter k, there exists a protocol that 0-simulates π with $\sqrt{I^B_\pi \cdot \|\pi\|} + \frac{\|\pi\|}{k}$ number of rounds in expectation, and internal information at most $\frac{\|\pi\| \log \|\pi\|}{k} + k\sqrt{I^B_\pi \cdot \|\pi\|} + I^A_\pi + 2 \log \|\pi\| \sqrt{\|\pi\| \cdot I^B_\pi} + 3$.*

Before presenting the proof of this key theorem, we state its immediate corollary and use it to finish the proof of Theorem 3.1.

Corollary 3.2. *Given any protocol π, there exists a protocol that 0-simulates π with $2 \cdot \sqrt[4]{\|\pi\|^3 \cdot I^B_\pi}$ number of rounds in expectation, and internal information at most*

$$I^A_\pi + \sqrt[4]{\|\pi\|^3 \cdot I^B_\pi} + \sqrt[4]{\|\pi\|^3 \cdot I^B_\pi} \cdot \log \|\pi\| + 2 \cdot \sqrt{\|\pi\| \cdot I^B_\pi} \cdot \log \|\pi\| + 3.$$

Proof. Set $k = \sqrt[4]{\|\pi\| / I^B_\pi}$. By Theorem 3.5, we get that the expected number of rounds of the simulation is at most

$$\sqrt{I^B_\pi \cdot \|\pi\|} + \sqrt[4]{\|\pi\|^3 \cdot I^B_\pi} \leq 2 \sqrt[4]{\|\pi\|^3 \cdot I^B_\pi},$$

which uses the inequality $I_\pi^B \le \|\pi\|$, and the internal information is at most

$$I_\pi^A + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log \|\pi\| + 2 \cdot \sqrt{\|\pi\| \cdot I_\pi^B} \cdot \log \|\pi\| + 3. \qquad \square$$

Proof of Theorem 3.1. Applying Theorem 3.3 to the simulation guaranteed by Corollary 3.2 gives a simulation with communication bounded by

$$O\left(I_\pi^A + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log(1/\epsilon) + \sqrt[4]{\|\pi\|^3 \cdot I_\pi^B} \cdot \log \|\pi\| + \sqrt{\|\pi\| \cdot I_\pi^B} \cdot \log \|\pi\|\right),$$

as required in Theorem 3.1. $\square$

Proof of Theorem 3.5

We use the protocol τ given in Figure 3.1[2] to simulate π. Let M denote the output of τ. Then we claim that the distribution of m is correct:

Lemma 3.1. $\tau(xyrm) = \pi(xyrm)$.

Proof. It is clear that $\tau(xyr) = \pi(xyr)$. For each $i \in [\|\pi\|]$, if m_i is to be sent by Alice in π, then $m_i = 1$ exactly when $\rho_i < \pi(m_i|xrm_{<i})$, and if m_i is to be sent by Bob in π, then $m_i = 1$ exactly when $\rho_i < \pi(m_i|yrm_{<i})$. Thus, if m_i is to be sent by Alice in π, $\tau(m_i|xyrm_{<i}) = \pi(m_i|xrm_{\le i}) = \pi(m_i|xyrm_{\le i})$. On the other hand, if m_i is to be sent by Bob in π, then $\tau(m_i|xyrm_{<i}) = \pi(m_i|yrm_{<i}) = \pi(m_i|xyrm_{<i})$. Thus

$$\tau(xyrm) = \tau(xyr) \cdot \prod_{i=1}^{\|\pi\|} \tau(m_i|xyrm_{<i}) = \pi(xyr) \cdot \prod_{i=1}^{\|\pi\|} \pi(m_i|xyrm_{<i}) = \pi(xyrm). \qquad \square$$

Let L denote the number of *mistake* indices j reported to Alice by Bob in τ. Then we have:

Lemma 3.2. *The number of rounds in τ is at most $\|\pi\|/k + L$*

[2]Here we do not bother optimizing the communication of τ, since the communication will be eventually optimized via Theorem 3.3.

Input: x, y, the inputs to π. A parameter k.

Output: m, r distributed according to $\pi(mr|xy)$.

Public Randomness: The public randomness r of π, as well as an additional sequence of uniformly random numbers $r' = \rho_1, \ldots, \rho_{\|\pi\|} \in [0, 1]$.

Let m be the empty string;
while $|m| < \|\pi\|$ **do**
    Set $t = |m|$;
    for $i = t + 1, \ldots, \min\{k + t, \|\pi\|\}$ **do**
        if *m_i is sent by Bob in π* **then** Alice checks if $\rho_i < \pi(m_i = 1|xrm_{<i})$, and sets $m_i = 1$ if this is the case. Otherwise she sets $m_i = 0$;
        else Alice samples m_i privately according to the distribution $\pi(m_i|xrm_{<i})$;
    end
    Alice sends the current transcript m to Bob;
    Bob computes the smallest index $j \in [\min\{k + t, \|\pi\|\}]$ such that m_j would have been sent by Bob in π and ρ_j lies in the interval between $\pi(m_j = 1|xrm_{<j})$ and $\pi(m_j = 1|yrm_{<j})$. Bob can check this using ρ_j, m, y, r;
    Bob sends j to Alice, or reports that there is no such j;
    Alice corrects m if such j is found, by flipping the bit m_j, and truncating $m = m_{\leq j}$;
end
return m;

Figure 3.1: Protocol τ simulating π

Proof. There are L rounds where Alice needs to truncate m. In every other round, at least k new bits of the messages of π are sampled, so there can be at most $\|\pi\|/k$ additional rounds of communication. □

Given the last lemma, we can bound the number of rounds in the protocol by bounding L:

Lemma 3.3. $\mathbb{E}[L] \leq \sqrt{I_\pi^B \cdot \|\pi\|}$.

Proof. Let L_i denote the indicator random variable for the event that the i-th index of the message is corrected by Bob in τ; so $L = \sum_{i=1}^{\|\pi\|} L_i$. If the i-th message is sent by Alice, then $L_i = 0$. On the other hand, if it is sent by Bob, by Proposition 2.11, we get

$$\begin{aligned}
\tau(L_i = 1) &= \mathop{\mathbb{E}}_{xyrm_{<i}} [|\pi(m_i|xrm_{<i}) - \pi(m_i|yrm_{<i})|] \\
&= \mathop{\mathbb{E}}_{xyrm_{<i}} [|\pi(m_i|xrm_{<i}) - \pi(m_i|xyrm_{<i})|] \\
&\leq \sqrt{I(M_i; Y|XRM_{<i})}.
\end{aligned}$$

The penultimate inequality is true since m_i is sampled by Bob in π, hence is independent of x conditioned on $y, r, m_{<i}$. Therefore, by the linearity of expectation and Cauchy Schwartz inequality, we get

$$\begin{aligned}
\mathbb{E}[L] &\leq \sum_{i=1}^{\|\pi\|} \sqrt{I(M_i; Y|XRM_{<i})} \\
&\leq \sqrt{\|\pi\| \cdot \sum_{i=1}^{\|\pi\|} I(M_i; Y|XRM_{<i})} \\
&= \sqrt{\|\pi\| \cdot I(M; Y|XR)},
\end{aligned}$$

where the last equality is an application of Proposition 2.9. □

Lemma 3.3 and Lemma 3.2 together imply that the expected number of rounds in τ is at most $\sqrt{I_\pi^B \cdot \|\pi\|} + \|\pi\|/k$. It only remains to bound the internal information of τ:

Lemma 3.4. *The internal information of τ is at most*

$$\frac{\|\pi\| \log \|\pi\|}{k} + k\sqrt{I_\pi^B \cdot \|\pi\|} + I_\pi^A + 2\log\|\pi\|\sqrt{\|\pi\| \cdot I_\pi^B} + 3$$

Proof. Recall that R' denotes the sequence of numbers $\rho_1, \ldots, \rho_{\|\pi\|}$. The public randomness of τ consists of R, R'. Let Z denote the messages exchanged in the protocol τ. Let Z_A denote the bits sent by Alice, and Z_B denote the bits sent by Bob. Then the information learnt by Alice can be expressed as

$$I_\tau(Z_A Z_B; Y|XRR') = I_\tau(Z_B; Y|XRR') + I_\tau(Z_A; Y|XRR'Z_B), \tag{3.1}$$

by the chain rule. The second term of Eq. (3.1) is 0, since Alice's messages are independent of Y, given Bob's messages and Alice's inputs. For the first term, we use Proposition 2.14 to bound it by $\mathbb{E}[|Z_B|]$. The total number of rounds of the protocol is at most $L + \|\pi\|/k$, since every round where there is no mistake must simulate at least k messages from the protocol. Thus $\mathbb{E}[|Z_B|] \leq (\mathbb{E}[L] + \|\pi\|/k)\log\|\pi\| \leq \sqrt{I_\pi^B \cdot \|\pi\|}\log\|\pi\| + \frac{\|\pi\|\log\|\pi\|}{k}$, by Lemma 3.3.

Next we bound the information learnt by Bob in τ. Using Proposition 2.9, we get

$$I_\tau(Z; X|YRR') = \sum_{i=1}^{|Z|} I_\tau(Z_i; X|Z_{<i}YRR') = \mathop{\mathbb{E}}_{xyrr'z}\left[\sum_{i=1}^{|z|} \mathbb{D}\left(\frac{z_i|xyrr'z_{<i}}{z_i|yrr'z_{<i}}\right)\right]. \tag{3.2}$$

Claim 3.1. *If $m_{<j}$ represents the messages of π sampled by the simulation τ at the point the messages $z_{<i}$ were sent, then*

$$\mathop{\mathbb{E}}_{r'|xyrz_{<i}}\left[\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})}\right)\right] \begin{cases} = 0 & \text{if Bob sends } z_i, \\ \leq 1 & \text{if } z_i \text{ is discarded}, \\ \leq \mathbb{D}\left(\dfrac{\pi(m_j|xyrm_{<j})}{\pi(m_j|yrm_{<j})}\right) & \text{if Alice sends } m_j, \\ \leq \sqrt{\mathbb{D}\left(\dfrac{\pi(m_j|xyrm_{<j})}{\pi(m_j|xrm_{<j})}\right)\log\|\pi\| + \frac{3}{\|\pi\|}} & \text{if Bob sends } m_j \end{cases}$$

Proof. When z_i is sent by Bob, both distributions are the same, so the divergence is 0.

To prove the remaining cases, we apply Proposition 2.13. When z_i is to be discarded, set $q(z_i|yrr'z_{<i})$ to be the uniform distribution on bits. When q is the uniform distribution on the bits, $\mathbb{D}\left(\frac{p}{\!\!=\!\!}\,q\right) = 1 - \mathsf{h}(p(0)) \leq 1$. Since $\mathbb{D}\left(\frac{p}{\!\!=\!\!}\,q\right) \leq 1$ for any p, the bound follows using Proposition 2.13. When m_j is sent by Alice in π, observe that the distribution of $\tau(z_i|xyrr'z_{<i})$ is exactly the same as the distribution of $\pi(m_j|xyrm_{<j})$. Set $q(z_i|yrr'z_{<i}) = \pi(m_j|yrm_{<j})$. The bound follows by Proposition 2.13. For the last case, observe that in τ, z_i is determined by $xyrr'm_{<j}$, since $z_i = 1$ exactly when $\rho_i < \pi(m_j = 1|xrm_{<j})$. Set

$$b(\rho_i, y, r, r', m_{<j}) = \begin{cases} 1 & \text{if } \rho_i < \pi(m_j = 1|yrm_{<j}), \\ 0 & \text{otherwise} \end{cases}$$

and

$$q(z_i|yrr'z_{<i}) = \begin{cases} 1 - 1/\|\pi\| & \text{if } b(\rho_i, y, r, r', m_{<j}) = z_i, \\ 1/\|\pi\| & \text{otherwise.} \end{cases}$$

When ρ_i is in between $\pi(m_j = 1|yrm_{<j})$ and $\pi(m_j = 1|xrm_{<j})$,

$$\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\!\!=\!\!}\,q(z_i|yrr'z_{<i})\right) = \log\left(\frac{1}{(1/\|\pi\|)}\right) = \log\|\pi\|,$$

for all z_i with positive probability. When ρ_i is not in between those two quantities,

$$\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\!\!=\!\!}\,q(z_i|yrr'z_{<i})\right) \leq \log\frac{1}{1 - 1/\|\pi\|} \leq 3/\|\pi\|,$$

which follows from Proposition 2.2 and the assumption that $\|\pi\| > 2$. It is safe to assume that $\|\pi\| > 2$, as the compression is trivial for protocols with communication at most 2.

The probability of the first case is at most $\sqrt{\mathbb{D}\left(\frac{\pi(m_j|xyrm_{<j})}{\!\!=\!\!}\,\pi(m_j|yrm_{<j})\right)}$, by Proposition 2.10. □

Now given Z, call i *good* if the message Z_i does not correspond to a mistake and is not discarded by the simulation. Let G denote the set of good indices. We have,

$$\begin{aligned}
\mathop{\mathbb{E}}_{xyrr'z}\left[\sum_{i=1}^{|z|}\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})}\right)\right] &= \mathop{\mathbb{E}}_{xyrz}\left[\sum_{i=1}^{|z|}\mathop{\mathbb{E}}_{r'|xyrz_{<i}}\left[\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})}\right)\right]\right] \\
&= \mathop{\mathbb{E}}_{xyrz}\left[\sum_{i\in G}\mathop{\mathbb{E}}_{r'|xyrz_{<i}}\left[\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})}\right)\right]\right] \\
&\quad + \mathop{\mathbb{E}}_{xyrz}\left[\sum_{i\notin G}\mathop{\mathbb{E}}_{r'|xyrz_{<i}}\left[\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})}\right)\right]\right].
\end{aligned}$$

For every Z, at most $k \cdot L$ indices are discarded. This is because, at most k indices are discarded for every round with a mistake. Then by Claim 3.1,

$$\mathop{\mathbb{E}}_{xyrz}\left[\sum_{i\notin G}\mathop{\mathbb{E}}_{r'|xyrz_{<i}}\left[\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})}\right)\right]\right] \le k\cdot\mathbb{E}\left[L\right].$$

For every index $i \in G$, by Claim 3.1

$$\begin{aligned}
&\mathop{\mathbb{E}}_{xyrz}\left[\sum_{i\in G}\mathop{\mathbb{E}}_{r'|xyrz_{<i}}\left[\mathbb{D}\left(\frac{\tau(z_i|xyrr'z_{<i})}{\tau(z_i|yrr'z_{<i})}\right)\right]\right] \\
&\le \mathop{\mathbb{E}}_{\pi(mxyr)}\left[\mathbb{D}\left(\frac{\pi(m_j|xyrm_{<j})}{\pi(m_j|yrm_{<j})}\right)\right] + \mathop{\mathbb{E}}_{\pi(mxyr)}\left[\sum_{j=1}^{\|\pi\|}\sqrt{\mathbb{D}\left(\frac{\pi(m_j|xyrm_{<j})}{\pi(m_j|xrm_{<j})}\right)}\log\|\pi\| + \frac{3}{\|\pi\|}\right] \\
&= \mathop{\mathbb{E}}_{\pi(mxyr)}\left[\mathbb{D}\left(\frac{\pi(m_j|xyrm_{<j})}{\pi(m_j|yrm_{<j})}\right)\right] + \log\|\pi\|\mathop{\mathbb{E}}_{\pi(mxyr)}\left[\sum_{j=1}^{\|\pi\|}\sqrt{\mathbb{D}\left(\frac{\pi(m_j|xyr)}{\pi(m_j|xr)}\right)}\right] + 3 \\
&\le \mathop{\mathbb{E}}_{\pi(mxyr)}\left[\mathbb{D}\left(\frac{\pi(m_j|xyrm_{<j})}{\pi(m_j|yrm_{<j})}\right)\right] + \log\|\pi\|\sqrt{\|\pi\|\cdot\mathop{\mathbb{E}}_{\pi(mxyr)}\left[\sum_{j=1}^{\|\pi\|}\mathbb{D}\left(\frac{\pi(m_j|xyr)}{\pi(m_j|xr)}\right)\right]} + 3 \\
&= I^A_\pi + \log\|\pi\|\sqrt{\|\pi\|\cdot I^B_\pi} + 3,
\end{aligned}$$

where the penultimate inequality follows from an application of Cauchy Schwartz inequality, and the last inequality follows from the definitions of I^A_π and I^B_π. □

3.2 Compressing Protocols with Asymmetric Information - II

In this section, we prove Theorem 3.2, whose proof crucially uses the following theorem.

Theorem 3.6. *Let U be a finite set. Let $p_A, p_B, q_A, q_B : U \to [0,1]$ be such that $\forall z \in U$, $\mu(z) = p_A(z)q_B(z)$, $p(z) = p_A(z)p_B(z)$ and $q(z) = q_A(z)q_B(z)$ are distributions. There exists a randomized protocol with inputs p_A, p_B to Alice and q_A, q_B to Bob, such that*

1. *Both Alice and Bob either accept and compute (possibly different) samples $z \in U$, or abort the protocol.*
2. $\Pr[\textit{Both parties accept}] \geq 2^{-O\left(\mathbb{D}\left(\frac{\mu}{q}\right)+1\right)}$.
3. *Given that both parties accept, the distribution of their samples is 0.01-close in ℓ_1 distance to the distribution where both parties sample the same sample from $\mu(z)$.*

We first present the proof of Theorem 3.6 and then prove Theorem 3.2 in Section 3.2.1.

Proof. Let $I^B = \mathbb{D}\left(\frac{\mu}{p}\right)$ and $I^A = \mathbb{D}\left(\frac{\mu}{q}\right)$. Figure 3.2 describes the randomized protocol promised by the theorem. The first item of the theorem is straightforward from the definition of the sampling procedure in Figure 3.2.

In the proof of the remaining two items of the theorem, the following definition is useful.

$$\mathcal{G} = \left\{ z \;\middle|\; 2^{1000(I^B+1)} \cdot p(z) \geq \mu(z), 2^{1000(I^A+1)} \cdot q(z) \geq \mu(z) \right\}.$$

We need a simple claim about $\mathcal{G}$ that was proved in [23].

Claim 3.2. $\mu(G) \geq 0.998$.

To prove the second item of the theorem, we have to lower bound the probability that both parties accept (see Lemma 3.8). For the same, we use the equality

$$\Pr[\text{Both parties accept}] = \Pr[i^* \text{ is defined}] \cdot \Pr[\text{Both parties accept} | i^* \text{ is defined}],$$

Simulation π

Public randomness: A sequence of $L = 10 \cdot |U| \cdot 2^{1000(I^A+1)}$ tuples $(z_i, a_i, b_i) \in U \times \left[0, 2^{1000(I^A+1)}\right] \times \left[0, 2^{1000(I^B+1)}\right]$, for $i = 1, 2, \ldots, L$, and a random function $h : [L] \to \left[2^{2000(I^A+1)}\right]$.

1. Alice computes the set $\mathcal{A} = \left\{ i \;\middle|\; a_i \leq p_A(z_i), b_i \leq p_B(z_i) \cdot 2^{1000(I^B+1)} \right\}$, and Bob computes the set $\mathcal{B} = \left\{ i \;\middle|\; a_i \leq q_A(z_i) \cdot 2^{1000(I^A+1)}, b_i \leq q_B(z_i) \right\}$.

2. Alice computes i^*, the smallest element of $\mathcal{A}$.

3. Alice sends $h(i^*)$ to Bob.

4. If there is a unique $i \in \mathcal{B}$ such that $h(i) = h(i^*)$, Bob accepts and assumes that the outcome of the protocol is z_i. Otherwise Bob aborts.

Figure 3.2: The sampling procedure

and lower bound each term on the right. We first lower bound $\Pr[i^* \text{ is defined}]$.

Lemma 3.5. $\Pr[i^* \textit{ is defined}] \geq 1 - e^{-10}$.

Proof. For each i, we have

$$\Pr[i \in \mathcal{A}] = \sum_{z \in U} \frac{p_A(z) p_B(z)}{|U| \cdot 2^{1000(I^A+1)}} = \frac{1}{|U| \cdot 2^{1000(I^A+1)}} \sum_{z \in U} p_A(z) p_B(z) = \frac{1}{|U| \cdot 2^{1000(I^A+1)}},$$

where the last equality follows from the fact that p is a distribution. The probability that i^* is not defined is exactly equal to the probability that $i \notin \mathcal{A}$ for all $1 \leq i \leq L$. Thus,

$$\Pr[i^* \text{ not defined}] = \left(1 - \frac{1}{|U| \cdot 2^{1000(I^A+1)}}\right)^L \leq e^{-\frac{L}{|U| \cdot 2^{1000(I^A+1)}}} = e^{-10},$$

where the first inequality uses the fact that for every $x \geq 0$, $(1-x)^n \leq e^{-xn}$. □

We need the following lemma before lower bounding the probability that both parties accept conditioned on i^* being defined.

Lemma 3.6. *For $z \in U$,*

$$\Pr[z_{i^*} = z \ \&\ i^* \in \mathcal{B} | i^* \text{ is defined}] \leq \frac{\mu(z)}{2^{1000(I^B+1)}},$$

with equality when $z \in \mathcal{G}$.

Proof. By definition,

$$\Pr[z_{i^*} = z \ \&\ i^* \in \mathcal{B} | i^* \text{ is defined}] = \Pr[z_{i^*} = z | i^* \text{ is defined}] \cdot \Pr[i^* \in \mathcal{B} | z_{i^*} = z]. \tag{3.3}$$

We have

$$\begin{aligned} \Pr[z_{i^*} = z | i^* \text{ is defined}] &= \frac{p_A(z)p_B(z)2^{-1000(I^A+1)}}{\sum_{z \in U} p_A(z)p_B(z)2^{-1000(I^A+1)}} \\ &= p_A(z)p_B(z). \end{aligned} \tag{3.4}$$

Let us now analyze $\Pr[i^* \in \mathcal{B} | z_{i^*} = z]$. If $z_{i^*} = z$, we have $a_{i^*} \leq p_A(z), b_{i^*} \leq p_B(z)2^{1000(I^B+1)}$. Thus $\Pr[i^* \in \mathcal{B} | z_{i^*} = z]$ is exactly

$$\begin{aligned} \Pr[i^* \in \mathcal{B} | z_i^* = z] &= \min\left\{\frac{q_B(z)}{2^{1000(I^B+1)}p_B(z)}, 1\right\} \cdot \min\left\{\frac{2^{1000(I^A+1)}q_A(z)}{p_A(z)}, 1\right\} \\ &\leq \frac{q_B(z)}{2^{1000(I^B+1)}p_B(z)}. \end{aligned} \tag{3.5}$$

Equality holds in Eq. (3.5) when $z \in \mathcal{G}$, since for such z,

$$\frac{q_B(z)}{2^{1000(I^B+1)}p_B(z)} \leq 1 \qquad \text{and} \qquad \frac{2^{1000(I^A+1)}q_A(z)}{p_A(z)} \geq 1.$$

Therefore, using Eqs. (3.3) to (3.5)

$$\Pr[z_{i^*} = z \ \&\ i^* \in \mathcal{B}] \leq p_A(z)p_B(z) \cdot \frac{q_B(z)}{2^{1000(I^B+1)}p_B(z)} = \mu(z)/2^{1000(I^B+1)},$$

with equality for $z \in \mathcal{G}$. □

When i^* is defined, let E denote the event that i^* is the only possible index that is in $\mathcal{B}$ and consistent with the message Bob receives, namely: $\forall i \neq i^*, i \in \mathcal{B} \Rightarrow h(i) \neq h(i^*)$. Then we have

Lemma 3.7. $\Pr[\textit{Both parties accept} | i^* \textit{ is defined}] \geq \frac{8}{10} \cdot 2^{-1000(I^B+1)}$

Proof. Observe that

$$\Pr[\text{Both parties accept} | i^* \text{ is defined}] \geq \Pr[i^* \in \mathcal{B} | i^* \text{ is defined}] - \Pr[E^c | i^* \text{ is defined}].$$

We claim

Claim 3.3. *(a)* $\Pr[E^c | i^* \textit{ is defined}] \leq \frac{L \cdot 2^{-2000(I^A+1)-1000(I^B+1)}}{|U| \Pr[i^* \textit{ is defined}]}$.

(b) $\Pr[i^* \in \mathcal{B} | i^* \textit{ is defined}] \geq \mu(\mathcal{G}) \cdot 2^{-1000(I^B+1)}$.

The above claim implies that

$$\begin{aligned}
\Pr[\text{Both parties accept} | i^* \text{ is defined}] &\geq \mu(\mathcal{G})/2^{1000(I^B+1)} - \frac{10}{1-e^{-10}} \cdot 2^{-1000(I^A+1)-1000(I^B+1)} \\
&\geq \frac{9}{10} \cdot 2^{-1000(I^B+1)} - \frac{10}{1-e^{-10}} \cdot 2^{-1000(I^A+1)-1000(I^B+1)} \\
&> \frac{8}{10} \cdot 2^{-1000(I^B+1)},
\end{aligned}$$

where the second inequality follows from Claim 3.2 and the last inequality follows from $I^A \geq 0$. □

Proof of Claim 3.3. (a) Conditioned on i^* being defined, the probability that $i \in \mathcal{B}$ and $h(i) = h(i^*)$ is at most $\frac{\Pr[i \in \mathcal{B}] \cdot 2^{-2000(I^A+1)}}{\Pr[i^* \text{ is defined}]}$. Thus,

$$\begin{aligned}
\frac{\Pr[i \in \mathcal{B}] \cdot 2^{-2000(I^A+1)}}{\Pr[i^* \text{ is defined}]} &= \frac{\sum_{z \in U} q_A(z) \cdot 2^{-1000(I^B+1)} q_B(z) 2^{-2000(I^A+1)}}{|U| \cdot \Pr[i^* \text{ is defined}]} \\
&\leq \frac{2^{-2000(I^A+1)-1000(I^B+1)}}{|U| \cdot \Pr[i^* \text{ is defined}]}.
\end{aligned}$$

Thus, by the union bound, the probability than any such i is accepted by Bob is at most $\frac{L \cdot 2^{-2000(I^A+1)-1000(I^B+1)}}{|U| \Pr[i^* \text{ is defined}]}$

(b) We know that

$$\begin{aligned}\Pr[i^* \in \mathcal{B} | i^* \text{ is defined}] &= \sum_{z \in |U|} \Pr[z_{i^*} = z, i^* \in \mathcal{B} | i^* \text{ is defined}] \\ &\geq \sum_{z \in \mathcal{G}} \mu(z)/2^{1000(I^B+1)} = \mu(\mathcal{G}) \cdot 2^{-1000(I^B+1)},\end{aligned}$$

where the first inequality is using Lemma 3.6 and the last equality is using Lemma 3.5.

□

We are finally ready to prove the second item of the theorem.

Lemma 3.8. $\Pr[\textit{Both parties accept}] \geq \frac{7}{10} \cdot 2^{-1000(I^B+1)}$.

Proof. We know that

$$\begin{aligned}\Pr[\text{Both parties accept}] &= \Pr[i^* \text{ is defined}] \Pr[\text{Both parties accept} | i^* \text{ is defined}] \\ &\geq \frac{8(1-e^{-10})}{10 \cdot 2^{1000(I^B+1)}} > \frac{7}{10} \cdot 2^{-1000(I^B+1)},\end{aligned}$$

which follows from Lemma 3.7 and Lemma 3.5. □

We now proceed to prove the third item of the theorem (see Lemma 3.11).

Lemma 3.9. $|\pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z)| \leq 2(1 - \mu(\mathcal{G}))/\mu(\mathcal{G})$.

Proof. By Lemma 3.6,

$$\Pr[i^* \in \mathcal{B} | i^* \text{ is defined}] \cdot \Pr[z_{i^*} = z | i^* \in \mathcal{B}] \leq \mu(z) \cdot 2^{-1000(I^B+1)}.$$

Combining the above inequality and Claim 3.3(b), $\Pr[z_{i^*} = z | i^* \in \mathcal{B}] \leq \mu(z)/\mu(\mathcal{G})$.

For any set $T \subseteq U$,

$$\sum_{z \in T} \pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z) \leq \sum_{z \in T} \mu(z)/\mu(\mathcal{G}) - \mu(z) \leq (1 - \mu(\mathcal{G}))/\mu(\mathcal{G}),$$

where the last inequality follows from μ being a distribution. Therefore,

$$|\pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z)| = 2 \max_T \left(\sum_{z \in T} \pi(z_{i^*} = z | i^* \in \mathcal{B}) - \mu(z) \right) \leq 2(1 - \mu(\mathcal{G}))/\mu(\mathcal{G}). \quad \square$$

Lemma 3.10.

$$\Pr[E^c|i^* \in \mathcal{B}] < 0.00025 \qquad \textit{and} \qquad \Pr[E^c|\textit{Both parties accept}] < 0.00025.$$

Proof. We have,

$$\begin{aligned}
Pr[E^c|i^* \in \mathcal{B}] &= \Pr[E^c|i^* \in \mathcal{B}, i^* \text{ is defined}] \\
&= \frac{\Pr[E^c \ \& \ i^* \in \mathcal{B}|i^* \text{ is defined}]}{\Pr[i^* \in \mathcal{B}|i^* \text{ is defined}]} \\
&\leq \frac{\Pr[E^c|i^* \text{ is defined}]}{\Pr[i^* \in \mathcal{B}|i^* \text{ is defined}]} \\
&\leq \frac{10}{\mu(\mathcal{G})(1-e^{-10})} \cdot 2^{-1000(I^A+1)} \\
&< 0.00025.
\end{aligned}$$

The penultimate inequality follows from Claim 3.3. Simultaneously,

$$\begin{aligned}
\Pr[E^c|\text{Both parties accept}] &= \Pr[E^c|\text{Both parties accept}, i^* \text{ is defined}] \\
&= \frac{\Pr[E^c \ \& \ \text{Both parties accept}|i^* \text{ is defined}]}{\Pr[\text{Both parties accept}|i^* \text{ is defined}]} \\
&\leq \frac{\Pr[E^c|i^* \text{ is defined}]}{\Pr[\text{Both parties accept}|i^* \text{ is defined}]} \\
&\leq \frac{100}{8(1-e^{-10})} \cdot 2^{-1000(I^A+1)} \\
&< 0.00025.
\end{aligned}$$

The penultimate inequality follows from Claim 3.3(a) and Lemma 3.7. □

Lemma 3.11. $|\mu(z) - \pi(z|\textit{Both parties accept})| < 0.01$

Proof. Applying Proposition 2.5 twice gives,

$$\begin{aligned}
&|\mu(z) - \pi(z|\text{Both parties accept})| \\
&\leq |\pi(z_{i^*} = z|i^* \in \mathcal{B}, E) - \mu(z)| + 4\Pr[E^c|\text{Both parties accept}] \\
&\leq |\pi(z_{i^*} = z|i^* \in \mathcal{B}) - \mu(z)| + 4\Pr[E^c|i^* \in \mathcal{B}] + 4\Pr[E^c|\text{Both parties accept}].
\end{aligned}$$

Lemmas 3.9 and 3.10 give,

$$|\mu(z) - \pi(z|\text{Both parties accept})| < 2(1-\mu(\mathcal{G}))/\mu(\mathcal{G}) + 0.001 + 0.001 < 0.01,$$

as required. □

The above lemma shows the third property of the theorem, completing its proof. □

3.2.1 Proof of Theorem 3.2

Define U to be the set of all messages of protocol π. For every $m \in U$, define $\pi_x(m) = \pi(m|x)$, $\pi_y(m) = \pi(m|y)$, $\pi_{xy}(m) = \pi(m|xy)$. We have,

$$I_\pi(X; M|YR) = \mathbb{E}_{xyr}\left[\mathbb{D}\left(\frac{\pi_{xy}(m|r)}{\pi_y(m|r)}\right)\right].$$

Define

$$\Gamma = \left\{(x,y,r) \;\middle|\; \mathbb{D}\left(\frac{\pi_{xy}(m|r)}{\pi_y(m|r)}\right) \le 50 \cdot I_\pi^A, \mathbb{D}\left(\frac{\pi_{xy}(m|r)}{\pi_x(m|r)}\right) \le 50 \cdot I_\pi^B\right\}.$$

By a union bound and Markov's inequality, we get that

$$\Pr[(x,y,r) \in \Gamma] \ge \frac{24}{25} \tag{3.6}$$

Theorem 3.6, with $\mu(m) = \pi_{xy}(m|r)$, $p(m) = \pi_x(m|r)$, $q(m) = \pi_y(m|r)$, implies that for every $(x,y,r) \in \Gamma$, there exists a constant c and a randomized protocol τ with communication $O(I_\pi^A)$ that samples a message m such that

$$|\pi(m) - \tau(m|\text{Both parties accept})| < 0.01 \text{ and } \Pr[\text{Both parties accept in } \tau] \ge 2^{-c(I_\pi^B+1)}. \tag{3.7}$$

The simulation of π with the desired guarantees of the theorem is given by Σ, which is described in Figure 3.3, with parameter $t = 10 \cdot 2^{c(I_\pi^B+1)}$. We have,

$$\|\Sigma\| \le t \cdot \|\tau\| = O\left(I_\pi^A 2^{c(I_\pi^B+1)}\right).$$

Input: x, y, the inputs to π. A parameter t.

Public Randomness: Sequence of strings $L_1, \cdots, L_t, r$. A random transcript m'.

```
1 i=1;
2 while i ≤ t do
3     Run protocol τ with L_i as the public random tape;
4     if τ Accepts then return the output of τ;
5     else i=i+1;
6 end
7 return m';
```

Figure 3.3: Protocol Σ simulating π

This bounds the communication complexity of Σ, and we are only left with showing that the ℓ_1 distance between Σ and τ is bounded. The following lemma upper bounds the ℓ_1 distance, whose proof will complete the proof of Theorem 3.2:

Lemma 3.12. $|\Sigma - \pi| \leq 0.14$

Before proving the above lemma, we need to establish a property about τ and another property about Σ.

Lemma 3.13. $\mathbb{E}_{\pi(xyr)}|\pi(m|xyr) - \tau(m|xyr, \textit{Both parties accept})| < \frac{1}{25} + 0.01$

Let J be the value of i at the time of termination of Σ.

Lemma 3.14. $\Pr[J = t + 1] \leq 1/25 + e^{-10}$

The proofs of these lemmas are deferred.

Proof of Lemma 3.12. Conditioned on $J \leq t$, we know that Σ's output corresponds to the output of τ. Therefore,

$$|\pi(m) - \Sigma(m|J \leq t)| = \mathbb{E}_{\pi(xyr)}|\pi(m|xyr) - \tau(m|xyr, \text{Both parties accept})|$$

By Proposition 2.4 and Lemma 3.14,

$$\begin{aligned}|\Sigma - \pi| &\leq 2\Pr[J = t+1] + |\pi(z) - \Sigma(z|J \leq t)| \\ &= 2\Pr[J = t+1] + \mathbb{E}_{\pi(xyr)}|\pi(m|xyr) - \tau(m|xyr, \text{Both parties accept}) \\ &< 2/25 + 2e^{-10} + 1/25 + 0.01 \leq 0.14,\end{aligned}$$

where the penultimate inequality follows from Lemma 3.14 and Lemma 3.13. □

We are now left with the task of proving Lemmas 3.13 and 3.14

Proof of Lemma 3.13. τ guarantees that for every $(x, y, r) \in \Gamma$,

$$|\pi(m|xyr) - \tau(m|xyr, \text{Both parties accept})| < 0.01.$$

Therefore,

$$\begin{aligned}&\mathbb{E}_{\pi(xyr)}|\pi(m|xyr) - \tau(m|xyr, \text{Both parties accept})| \\ &\leq \Pr[(x,y,r) \in \Gamma] \cdot \max_{(x,y,r)\in\Gamma} |\pi(z|xyr) - \tau(z|xyr, \text{Both parties accept})| + \Pr[(x,y,r) \notin \Gamma] \\ &< 0.01 + \frac{1}{25},\end{aligned}$$

where the last inequality follows from Eq. (3.6). □

Proof of Lemma 3.14. Conditioned on $(x, y) \in \Gamma$, the probability that τ does not accept in iteration i equals $1 - \Pr[\text{Both parties accepts}]$. Therefore,

$$\begin{aligned}\Pr[J = t+1|(x,y,r) \in \Gamma] &= (1 - \Pr[\text{Both party accepts in } \tau])^t \\ &\leq \left(1 - 2^{-c(I^B+1)}\right)^t \\ &\leq e^{-10}, \qquad (3.8)\end{aligned}$$

where the first inequality follows from Eq. (3.7) and the last inequality follows from the fact that for every $x \geq 0$, $(1-x)^n \leq e^{-xn}$. Now,

$$\begin{aligned}\Pr[J = t+1] &= \Pr[(x,y,r) \in \Gamma] \cdot \Pr[J = t+1|(x,y,r) \in \Gamma] \\ &\quad + \Pr[(x,y,r) \notin \Gamma] \cdot \Pr[J = t+1|(x,y,r) \notin \Gamma] \\ &\leq e^{-10} + 1/25,\end{aligned}$$

where the last inequality follows from Eqs. (3.6) and (3.8). □

3.3 A Rectangle Lower Bound

In this section, we show a corollary to Theorem 3.2. Particularly, the simulation of Theorem 3.6 shows the existence of almost monochromatic rectangles of the right dimension.

Corollary 3.3 (Restatement of Corollary 3.1)**.** *Given any randomized protocol π with internal information (I^A, I^B), there exists sets S, T and $z \in \{0,1\}$ such that*

$$\mu(x \in S) \geq 2^{-O(I^A)}, \mu(y \in T | x \in S) \geq 2^{-O(I^B)} \text{ and } \mu(f(x,y) \neq z | S \times T) \leq 3\epsilon + 0.06,$$

where ϵ is the error incurred by π under μ in the computation of f.

Proof. First we wish to fix the public randomness of π, such that the internal information and the error bound are within limit. We have,

$$I_A = \mathop{\mathbb{E}}_r \left[I(M; X | Yr) \right], I_B = \mathop{\mathbb{E}}_r \left[I(M; Y | Xr) \right] \text{ and } \mathop{\mathbb{E}}_r \left[\mu(\pi(x,y) \neq f(x,y) | r) \leq \epsilon \right].$$

By Markov's inequality, $\Pr_r[I(M; X|Yr) > 3I_A] < 1/3$, $\Pr_r[I(M; Y|Xr) > 3I_B] < 1/3$ and $\Pr_r[\mu(\pi(x,y) \neq f(x,y)|r) > 3\epsilon] < 1/3$. Therefore, by a union bound, there exits an r such that

$$I(M; X|Yr) \leq 3I_A, I(M; Y|Xr) \leq 3I_B \text{ and } \mu(\pi(x,y) \neq f(x,y)|r) \leq 3\epsilon.$$

After fixing r, we now have a protocol with internal information at most $(3I_A, 3I_B)$ and no public randomness. In what follows, for the sake of simplicity, we will refer to this protocol as π.

The following equation will become useful.

$$\begin{aligned} \pi(m|xy) &= \left(\prod_{i \,:\, m_i \text{ sent by Alice}} \pi(m_i | xym_{<i}) \right) \cdot \left(\prod_{i \,:\, m_i \text{ sent by Bob}} \pi(m_i | xym_{<i}) \right) \\ &= \left(\prod_{i \,:\, m_i \text{ sent by Alice}} \pi(m_i | xm_{<i}) \right) \cdot \left(\prod_{i \,:\, m_i \text{ sent by Bob}} \pi(m_i | ym_{<i}) \right), \end{aligned}$$

where the last inequality follows from the fact that Alice's messages depend only on x, the public randomness and the previous messages, and Bob's messages depend only on y, the public randomness and the previous messages. Define $\pi_A, \pi_B, \pi'_A, \pi'_B$ satisfying,

$$\pi_A(m|x) = \prod_{i\,:\,m_i \text{ sent by Alice}} \pi(m_i|xm_{<i}), \qquad \pi_B(m|y) = \prod_{i\,:\,m_i \text{ sent by Bob}} \pi(m_i|ym_{<i}),$$
$$\pi'_A(m|x) = \prod_{i\,:\,m_i \text{ sent by Bob}} \pi(m_i|ym_{<i}) \text{ and } \pi'_B(m|y) = \prod_{i\,:\,m_i \text{ sent by Alice}} \pi(m_i|ym_{<i}). \quad (3.9)$$

Let

$$\Gamma = \left\{ (x,y) \;\middle|\; \mathbb{D}\left(\frac{\pi(m|xy)}{\pi(m|y)}\right) \leq 600 \cdot I^A, \mathbb{D}\left(\frac{\pi(m|xy)}{\pi(m|x)}\right) \leq 600 \cdot I^B_\pi \right\}.$$

Observe that $\Pr[(x,y) \in \Gamma] \geq 99/100$, which follows from Markov's inequality.

Simulation

Public randomness: A sequence of $L = 10 \cdot |U| \cdot 2^{1000(600I^A+1)}$ tuples $(z_i, a_i, b_i) \in \mathcal{M} \times \left[0, 2^{1000(600I^A+1)}\right] \times \left[0, 2^{1000(600I^B+1)}\right]$, for $i = 1, 2, \ldots, L$, r and a random function $h : [L] \to \left[2^{2000(600I^A+1)}\right]$, where $\mathcal{M}$ is the set of all transcripts of π.

1. Alice computes the set $\mathcal{A} = \left\{ i \;\middle|\; a_i \leq \pi_A(z_i|xr), b_i \leq \pi'_A(z_i|xr) \cdot 2^{1000(600I^B+1)} \right\}$, and Bob computes the set $\mathcal{B} = \left\{ i \;\middle|\; a_i \leq \pi'_B(z_i|yr) \cdot 2^{1000(600I^A+1)}, b_i \leq \pi_B(z_i|yr) \right\}$.

2. Alice computes i^*, the smallest element of $\mathcal{A}$.

3. Alice accepts if $h(i^*) = 0^{2000(600I^A+1)}$.

4. If there is a unique $i \in \mathcal{B}$ such that $h(i) = 0^{2000(600I^A+1)}$, Bob accepts and assumes that the outcome of the protocol is z_i. Otherwise Bob aborts.

Figure 3.4: The sampling procedure

Consider the simulation τ in Figure 3.4. Let $\mathcal{L}$ denote the sequence of L tuples in the public tape. In τ, since Alice accepts only if $h(i^*) = 0^{2000(600I^A+1)}$ and h is a random hash function, we can say that $\mathbb{E}_{\mathcal{L},h}\left[\mu(\text{Alice Accepts})\right] \geq 2^{-O(I_A)}$. Note that τ is the same as the simulation in Figure 3.2 invoked with $p_A = \pi_A, p_B = \pi'_A, q_A = \pi'_B$ and $q_B = \pi_B$, except the step in τ where Alice accepts only if the hash matches the all-zeros string. In other words, conditioned on Alice accepting, τ is the same as the simulation in Figure 3.2, whose guarantees are given in Theorem 3.6. Therefore, for every $(x, y) \in \Gamma$, τ satisfies

- $\mathbb{E}_{\mathcal{L},h}\left[\mu(\text{Bob Accepts}|\text{Alice Accepts})\right] \geq 2^{-O(I_B)}$
- Given both parties accept, the distribution of the samples is $0.01-$close in ℓ_1 distance to the distribution where both parties sample the same from $\pi(x, y)$. Alternatively, $\mathbb{E}_{\mathcal{L},h}\left[|\pi(m|xy) - \tau(m|xy, \text{Both parties accept})|\right] \leq 0.01$.

We now have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathcal{L},h}\left[|\pi(m|xy) - \tau(m|xy,\ \text{Both parties accept})|\right] \\
&\leq \Pr[(x,y) \in \Gamma] \cdot \max_{(x,y)\in\Gamma} \left(\mathop{\mathbb{E}}_{\mathcal{L},h}\left[|\pi(m|xy) - \tau(m|xy,\ \text{Both parties accept}), (x,y) \in \Gamma|\right] \right) \\
&\qquad + \Pr[(x,y) \notin \Gamma] \\
&\leq 0.01 \cdot (99/100 + 1) \leq 0.02.
\end{aligned}$$

Again applying Markov's inequality and a union bound, we can say that there exists a fixing of $\mathcal{L}, h$ satisfying

$$\mu(\text{Alice Accepts}) \geq 2^{-O(I_A)}, \qquad \mu(\text{Bob Accepts}|\text{Alice Accepts}) \geq 2^{-O(I_B)} \text{ and}$$
$$|\pi(m|xy) - \tau(m|xy, \text{Both parties accept})| \leq 0.06.$$

Note that we have fixed the randomness in τ and obtained a deterministic protocol. Any transcript in a deterministic protocol corresponds to a rectangle. Therefore, the state of both parties accepting, corresponds to a rectangle $S \times T$ with

$$\mu(x \in S) \geq 2^{-O(I^A)} \qquad \text{and} \qquad \mu(y \in T | x \in S) \geq 2^{-O(I^B)}.$$

In addition, conditioning on the event that both parties accept, the output $z \in \{0,1\}$ of the protocol (the value of the function that both parties agree on) has the property that $\mu(f(x,y) \neq z) \leq 3\epsilon + 0.06$, where ϵ is the error incurred by the protocol π under μ. □

3.3.1 Application - Lower Bounds for Lopsided Set Disjointness

In this subsection, we use the rectangle lower bound to reprove the well known lower bound for *lopsided* set disjointness by Pătraşcu in [80]. The problem of lopsided set disjointness is defined as follows,

Definition 3.1. *The set disjointness function on sets $x, y \subseteq [NN']$ is*

$$\mathsf{SD}(x,y) = \begin{cases} 1 & \text{if } x \cap y = \emptyset, \\ 0 & \text{otherwise.} \end{cases}$$

Lopsided set disjointness(LSD) is a restricted version of the problem, in which we are promised that $x, y \subseteq [NN']$ and $|x| = N$. The following bound proved by Patrascu [80] has found many applications to proving data structure lower bounds. The bound was shown to be tight by Sağlam [91].

Theorem 3.7 ([80])**.** *For any protocol computing* LSD *with error probability ϵ, if $N' = \Omega(1)$, then one of the following holds,*

- *Alice communicates at least $\gamma N \log N'$ bits.*
- *Bob communicates at least $N(N')^{1-c\gamma}$ bits.*

where $c = c_1 + 1 + \frac{(\log 2c_1 - \log(1-\mathsf{h}(12\epsilon+0.06)))}{\gamma \log N'}$ for some constant c_1.

Here we give a slightly different proof of Theorem 3.7, using Corollary 3.3. The universe is taken to be $\left(2^{[N']}\right)^N$, the cartesian product of N power sets of $[N']$. $(x,y) \in \left(2^{[N']}, 2^{[N']}\right)$ is restricted tuples with $|x| = 1$ and y takes exactly one element from the pair $(2k, 2k+1)$, for all $2k, 2k+1 \in [N']$. We define two distributions ψ and μ on (x,y) as follows,

- ψ is a uniform distribution on all such pairs (x, y), with $\mathsf{LSD}(x, y) = 1$.

- μ is a uniform distribution on all such pairs (x, y).

The hard distribution for disjointness μ_h is one where $i \in [N]$ chosen at random, with (x_i, y_i) drawn from distribution μ and rest of the coordinates (x_j, y_j), for $j \in [N] \setminus \{i\}$ is drawn independently and identically from ψ. Having described the hard distribution, we are set to describe the proof of Theorem 3.7.

Proof of Theorem 3.7. We assume the contrary that there exists a protocol π computing $\mathsf{LSD}(x, y)$ on the distribution μ_h with Alice communicating $a < \gamma N \log N'$ bits and Bob communicating $b < N(N')^{1-c\gamma}$ bits.

We now use protocol π to compute LSD on a single block. Consider the case when the inputs (x, y) is drawn according to the distribution ψ. We know that protocol π computes LSD on inputs drawn independently and identically in ψ with information $\left(I^A, I^B\right) < (\gamma N \log N', N(N')^{1-c\gamma})$. By Theorem 3.4, there exist a protocol τ computing $\mathsf{LSD}(x, y)$ on ψ with information $\left(I^A_\tau, I^B_\tau\right) \leq (\gamma \log N', (N')^{1-c\gamma})$.

Define $\mathcal{M} = \{m \mid \mu(\mathsf{LSD}(x, y) \neq \tau(x, y)|m) \leq 4\epsilon\}$, a subset of all transcripts of τ. Note that $\mu(\mathcal{M}) \geq 1 - \frac{1}{4} = \frac{3}{4}$, using the fact that $\mu(\mathsf{LSD}(x, y) \neq \tau(x, y)) \leq \epsilon$. Therefore, $\psi(\mathcal{M}) \geq 1 - \frac{2}{4} = \frac{1}{2}$, since the density of $\mathsf{Supp}(\psi)$ under μ is one half.

First observe that Corollary 3.3 holds even when the transcripts are restricted to the set $\mathcal{M}$. So, Corollary 3.3 shows the existence of a constant c_1 and sets S, T such that

$$\psi(x \in S) \geq 2^{-c_1(I^A)} \geq 2^{-c_1(\gamma \log N')} \text{ and } \psi(y \in T | x \in S) \geq 2^{-c_1(I^B)} \geq 2^{-c_1((N')^{1-4\gamma})},$$

where we used the upper bounds on information (I^A, I^B) under ψ. Since restricted to $m \in \mathcal{M}$, $\mu(\mathsf{LSD}(x, y) \neq 1 | S \times T) \leq 3 \cdot 4\epsilon + 0.06 = 12\epsilon + 0.06$.

The marginal distribution in x being the same on μ and ψ implies

$$\mu(x \in S) \geq 2^{-c_1(\gamma \log N')}$$

Also, $\mu(y) \geq \frac{1}{2} \cdot \psi(y)$ since $\psi(x, y) = \mu(x, y | x \cap y = \emptyset)$ and $\mu(x \cap y = \emptyset) = \frac{1}{2}$. Therefore,

$$\mu(y \in T) \geq \frac{1}{2} \cdot \psi(y \in T) \geq 2^{-c_1(\gamma \log N' + (N')^{1-c\gamma}) - 1}.$$

From here on, we work only with the distribution μ. The bounds on probabilities imply the following bounds on the corresponding entropy,

$$\mathsf{H}(X|S) \geq (1 - c_1\gamma) \log N' \tag{3.10}$$

$$\mathsf{H}(Y|T) \geq \frac{N'}{2} - c_1\gamma \log N' - c_1(N')^{1-c\gamma} - 1. \tag{3.11}$$

The error bound implies,

$$\forall x \in S, \mu(Y_x = 1|T) \leq \mu(\pi(x, y) \neq 1 | S \times T) \leq 12\epsilon + 0.06. \tag{3.12}$$

Note that Y_x is the projection of the vector Y onto the coordinate indexed by x. Hence, for every $x \in S$, $H(Y_x|T) \leq \mathsf{h}(12\epsilon + 0.06)$.

Using Proposition 2.8, we upper bound $\mathsf{H}(Y|T)$ by $\mathsf{H}(Y_S|T) + \mathsf{H}(Y_{S^c}|T)$, where S^c is the complement of S. Y_S, Y_{S^c} are projections of the vector Y onto coordinates indexed by elements of sets S and S^c, respectively. The first term in the expression can be simplified to $\mathsf{H}(Y_S|T) \leq \mathsf{h}\left(12\epsilon + 0.06\right) \cdot |S|$, where the inequality follows from Proposition 2.8 and Eq. (3.12). By an application of Proposition 2.8 and the fact that the binary entropy is at most 1, the second term yields $\mathsf{H}(Y_{S^c}|T) \leq \frac{N'}{2} - |S|$. Eq. (3.10) implies $|S| \geq (N')^{1-c_1\gamma}$. Therefore

$$\mathsf{H}(Y|T) \leq \frac{N'}{2} - \left(1 - \mathsf{h}(12\epsilon + 0.06)\right)(N')^{1-c_1\gamma}.$$

Eq. (3.11) implies that

$$\mathsf{H}(Y|T) \geq \frac{N'}{2} - c_1\gamma \log N' - c_1(N')^{1-c\gamma} - 1.$$

Owing to $c > c_1 + \frac{1}{\gamma \log N'} \left(\log 2c_1 - \log\left(1 - \mathsf{h}(12\epsilon + 0.06)\right)\right)$ and $N' = \Omega(1)$, we can conclude that

$$\frac{N'}{2} - \left(1 - \mathsf{h}(12\epsilon + 0.06)\right)(N')^{1-c_1\gamma} < \frac{N'}{2} - c_1\gamma \log N' - c_1(N')^{1-c\gamma} - 1,$$

which is a contradiction. □

Chapter 4

STATIC DATA STRUCTURE LOWER BOUNDS

In this chapter, we prove a new static data structure lower bound for the *Vector-Matrix-Vector* problem. We will present all known static lower bound techniques by applying them to the Vector-Matrix-Vector problem and its close variant called the *Vector-Vector* problem. We demonstrate the communication complexity reduction [70], communication complexity reduction combined with making multiple queries [83] and cell sampling [77, 66].

A static data structures in Yao's [106] cell probe model is defined as follows.

Definition 4.1 (Static Data Structures)**.** *Let Q be an arbitrary set of queries and $\mathcal{X}$ be the set of all possible input data. Let $f : \mathcal{D} \times \mathcal{Q} \to \Sigma$ be the query function for some output alphabet Σ. A static data structure on input $x \in \mathcal{X}$ with query set $\mathcal{Q}$ is one that*

1. *stores x in s cells in memory, where each cell is w bits long.*

2. *provides a* query *algorithm, which on query $q \in \mathcal{Q}$ outputs $f(x, q)$ by accessing a subset of the stored cells.*

The three parameters of interest are the space s, word length w and query time t, where the query time is the maximum number of accesses on any input x and query q.

Randomization and Error. The query algorithm can have access to randomness and can also make errors in its computation. We say that a randomized data structure computes a query function f correctly with probability p to mean that the query algorithm computes f correctly with probability p on every input data and query. Unless we explicitly mention the use of randomness or that the query algorithm errs, it is assumed that the query algorithm is deterministic and makes no errors.

We now define the Vector-Vector Problem and recall the Vector-Matrix-Vector problem.

- **Vector-Vector Problem:** Let n be a natural number. The input data is given by $x \in \mathbb{F}_2^n$, the query set $\mathcal{Q}$ is $\mathbb{F}_2^n$, and the query function is $\langle q, x \rangle$ for $q \in Q$. The data structure must store x and provide a query algorithm to compute the inner product $\langle q, x \rangle$.

- **Vector-Matrix-Vector Problem.** Let n be a square natural number. The input data is given by a matrix $M \in \mathbb{F}_2^{\sqrt{n} \times \sqrt{n}}$, the query set $\mathcal{Q}$ consists of pairs $(u, v) \in \mathbb{F}_2^{\sqrt{n}} \times \mathbb{F}_2^{\sqrt{n}}$, and the query function is $u^\intercal M v$. The data structure must store the matrix M and provide a query algorithm to compute $u^\intercal M v$.

The main theorem of this chapter is a lower bound on the query time of data structures computing $u^\intercal M v$, which is joint work with Rashtchian [76].

Theorem 4.1. *Let $M \in \mathbb{F}_2^{\sqrt{n} \times \sqrt{n}}$ be a matrix. If a randomized data structure with space s, word size w, and query time t correctly computes queries for the Vector-Matrix-Vector problem with probability at least $\frac{1}{2} + \frac{1}{2^{\sqrt{n}/64}}$, then*

$$t \geq \min \left\{ \frac{c\sqrt{n}}{\log\left(\frac{s\alpha}{n}\right)}, \ \frac{cn}{\alpha} \right\},$$

where $0 < c \leq 1/36$ is a universal constant and $\alpha = 2\left(w + \log\left(\frac{sw}{n}\right)\right)$.

Prior work by Chattopadhyay, Koucký, Loff, and Mukhopadhyay showed that the query time must be at least $\min\left\{ \frac{c\sqrt{n}}{\log \frac{sw}{\sqrt{n}}}, \ \frac{cn}{w} \right\}$ by utilizing a general lifting result for two-way communication complexity from parity decision trees [32]. To obtain the improved bound, we use a variant of the cell sampling technique [66, 77] combined with a reduction to a new lower bound on one-way communication (via discrepancy). The modifications over standard techniques are needed to handle the high error regime for a binary output problem. We note that a recent result of Larsen, Weinstein and Yu also uses one-way communication to prove lower bounds for binary output problems for dynamic data structures [67]. However, their method seems limited to only handling zero error query algorithms.

The proof of Theorem 4.1 uses the Yao's min-max principle [108] that lets us work with deterministic data structures with an underlying distribution on the input data and queries to prove a lower bound against randomized data structures. We state below the version that is sufficient for our proofs.

Theorem 4.2. *Let $0 < \epsilon < 1$. Let Q be the query set, $\mathcal{X}$ be the set of all possible input data and f be the query function of a data structure. If there exists a distribution on Q and $\mathcal{X}$ under which any deterministic query algorithm computing f correctly with probability at least $1 - \epsilon$ has query time at least t, then any randomized data structure computing f correctly with probability at least $1 - \epsilon$ must have query time at least t.*

Before presenting the proof of Theorem 4.1 in Section 4.3, we discuss the communication complexity reduction in Section 4.1 and the cell sampling technique in Section 4.2. The ideas and results in Sections 4.1 and 4.2 are yet to be published. In some proofs that follow, we need an upper bound on the number of bits to encode the contents and locations of a subset of the cells, which is given by the following proposition.

Proposition 4.1. *Let S be a subset of the cells of a data structure with word length w and size s. Then, the contents and locations of S can be encoded in $|S| \cdot w + |S| \cdot \log \frac{es}{|S|}$ bits.*

Proof. Since each cell stores w bits, the number of bits to encode the contents is $|S| \cdot w$. Since the total number of cells is s, the locations can be encoded in $\log \binom{s}{|S|} \leq |S| \cdot \log \frac{es}{|S|}$ bits, where the inequality followed from Proposition 2.1(a). □

4.1 Lower Bounds from Communication Complexity

We first prove a lower bound for the Vector-Vector problem using communication complexity via a reduction demonstrated in [70, 72].

Theorem 4.3. *Consider any randomized data structure for the Vector-Vector problem with space s, word length w and query time t. If the query algorithm is correct with probability at least $\frac{1}{2}(1 + 2^{-n/8})$, then $t \geq \min\left\{\frac{3n}{16 \log s}, \frac{3n}{16w}\right\}$.*

To understand the implications of the above theorem, let us consider a few values of s and w:

(a) If $s = \mathsf{poly}(n)$ and $w = O(\log n)$, then the query time must be at least $\Omega(n/\log n)$.

(b) If $s = O(n)$ and $w = O(1)$, then the query time lower bound is $\Omega(n/\log n)$. In a later section, we will improve this lower bound to $\Omega(n)$.

It is important to note that the above theorem is tight for data structures with polynomial space. The data structure of [15] discussed in the introduction also applies to the Vector-Vector problem; it has space $\mathsf{poly}(n)$, word length $O(\log n)$ and query time $O(n/\log n)$. To prove Theorem 4.3, we need the following well known lower bound on the communication complexity of the inner product function.

Theorem 4.4. *Let $0 < \epsilon \leq 1$, and let Alice and Bob receive $x \in \mathbb{F}_2^n$ and $y \in \mathbb{F}_2^n$ as inputs, respectively. Then, any randomized communication protocol that correctly computes $\langle x, y\rangle$ with probability at least $\frac{1}{2}(1+\epsilon)$ must have communication complexity at least $n/2 - \log(1/\epsilon)$.*

See [89, Chapter 5, Theorem 5.6] for a proof of the above theorem, which follows from an upper bound on the *discrepancy* of the inner product function under the uniform distribution. We now proceed to prove Theorem 4.3.

Proof of Theorem 4.3. Let $x, y \in \mathbb{F}_2^n$. We first show how a data structure for the Vector-Vector problem can be used to obtain a communication protocol that computes $\langle x, y\rangle$. Bob on input y builds the data structure with y as the input data. Alice treats her input x as the query to the data structure. Alice and Bob now simulate the query algorithm in rounds: Alice communicates the location of the cell the query algorithm wants to access and Bob responds with the contents. Since the space of the data structure is s and the word length is w, observe that the location and contents of each cell in memory can be described using $\log s$ and w bits, respectively. As the query time is t, Alice and Bob communicate $t \log s$ and tw bits in total, respectively. Moreover, they compute $\langle x, y\rangle$ with probability at least

$\frac{1}{2}(1+2^{-n/8})$, as guaranteed by the data structure. By Theorem 4.4, we know that Alice must communicate at least $3n/16$ bits or Bob must communicate at least $3n/16$ bits. Hence, $t \geq \min\left\{\frac{3n}{16\log s}, \frac{3n}{16w}\right\}$. □

The lower bound obtained for the Vector-Matrix-Vector problem via communication complexity is stated below.

Theorem 4.5. *Consider any randomized data structure for the Vector-Matrix-Vector problem with space s, word length w and query time t. If the query algorithm is correct with probability at least $\frac{1}{2}(1+2^{-\sqrt{n}/8})$, then $t \geq \min\left\{\frac{3\sqrt{n}}{64\log\left(\frac{se}{\sqrt{n}}\right)}, \frac{3\sqrt{n}}{16w}\right\}$.*

Proof of Theorem 4.5. Let $x, y \in \mathbb{F}_2^n$. Define $x_1, \ldots, x_{\sqrt{n}} \in \mathbb{F}_2^{\sqrt{n}}$ to be the projections of x on to coordinates indexed by the sets

$$\left\{1, 2, \ldots, \sqrt{n}\right\}, \left\{\sqrt{n}+1, \sqrt{n}+2, \ldots, 2\sqrt{n}\right\}, \ldots, \left\{n-\sqrt{n}+1, n-\sqrt{n}+2, \ldots, n\right\}.$$

$y_1, \ldots, y_{\sqrt{n}}$ are defined analogously with respect to y, and let Y be the $\sqrt{n} \times \sqrt{n}$ matrix in which the i-th row is y_i. Additionally, let $e_1, \ldots, e_{\sqrt{n}}$ be the standard basis vectors of $\mathbb{F}_2^{\sqrt{n}}$.

We will now use the data structure for the Vector-Matrix-Vector problem to design a communication protocol that computes $\langle x, y\rangle$. We have that

$$\sum_{i=1}^{\sqrt{n}} e_i^\intercal Y x_i = \sum_{i=1}^{\sqrt{n}} \langle x_i, y_i\rangle = \langle x, y\rangle .$$

Bob on input y will build the data structure on Y as the input data. Alice on input x will construct the queries $(e_1, x_1), \ldots, (e_{\sqrt{n}}, x_{\sqrt{n}})$. Alice with the help of Bob will simulate the queries $(e_1, x_1), \ldots, (e_{\sqrt{n}}, x_{\sqrt{n}})$ in parallel, ensuring that the randomness used by the query algorithm on each simulation is independent of each other - altogether, they will simulate $\sqrt{n}$ queries[1]. In every round of Alice, she will communicate the locations of the $\sqrt{n}$ cells, one for each simulation, and Bob will respond with the contents. This amounts to $\log\binom{s}{\sqrt{n}} \leq 4\sqrt{n}\log\left(\frac{se}{\sqrt{n}}\right)$ and $\sqrt{n}w$ bits of communication by Alice and Bob, respectively.

[1]This idea of simulating multiple queries was first used in [83].

Since the query time is t, Alice and Bob communicate $4t\sqrt{n}\log\left(\frac{se}{\sqrt{n}}\right)$ and $t\sqrt{n}w$ bits in total, respectively. Each each $e_i^\intercal Y x_i = \langle x_i, y_i\rangle$ is correctly computed with probability at least $\frac{1}{2}(1+2^{-\sqrt{n}/8})$, we can use Proposition 2.6 to infer that $\sum_{i=1}^{\sqrt{n}} \langle x_i, y_i\rangle = \langle x, y\rangle$ is correctly computed with probability at least $\frac{1}{2}(1+2^{-n/8})$. By Theorem 4.4, we know that either Alice or Bob must communicate at least $3n/16$ bits. Therefore, $t \geq \min\left\{\frac{3\sqrt{n}}{64\log\left(\frac{se}{\sqrt{n}}\right)}, \frac{3\sqrt{n}}{16w}\right\}$. □

4.2 Lower Bounds from Cell Sampling

Recall that Theorem 4.3 gives a query time lower bound of $\Omega(n/\log n)$ when the space is $O(n)$ and word length is $O(1)$ for the Vector-Vector problem. In this section, we will use cell sampling [77, 66] to improve this lower bound to $\Omega(n)$ for zero-error data structures. In the next section, we will show how to prove the same lower bound even when the query algorithm makes an error with high probability. We prove

Theorem 4.6. *Consider any deterministic data structure for the Vector-Vector problem with space s, word length w and query time t. If the query algorithm does not err, then $t \geq \min\left\{\frac{n}{256\log\left(\frac{s\alpha}{n}\right)}, \frac{n}{256\alpha}\right\}$, where $\alpha = 2\left(w + \log\left(\frac{sw}{n}\right)\right)$ and c is a constant.*

Proof. First note that $\alpha \leq \frac{n}{256}$ and $n \geq 32$; otherwise the lower bound on t is vacuous. Assume by contradiction that $t \leq \min\left\{\frac{n}{256\log\left(\frac{s\alpha}{n}\right)}, \frac{n}{256\alpha}\right\}$. We now state the cell sampling lemma, which guarantees the existence of a small subset of cells S such that a large number of queries Q' can be computed by accessing cells only in S.

Lemma 4.1 ([66])**.** *If $t \leq \min\left\{\frac{n}{256\log\left(\frac{s\alpha}{n}\right)}, \frac{n}{256\alpha}\right\}$, then there exist $Q' \subseteq \mathbb{F}_2^n$ and a subset of cells S such that*

1. $|S| = \left\lceil \frac{n}{128\alpha} \right\rceil$,
2. $\mathbb{E}\left[|Q'|\right] \geq 2^{-n/16}$,
3. *Q' is the set of all queries computed by accessing cells only in S.*

We now complete the proof of Theorem 4.6 and then prove Lemma 4.1. Let $X \in \mathbb{F}_2^n$ be the random variable that denotes a uniformly chosen input. We know that $\mathsf{H}(X) = n$; however, will show that $\mathsf{H}(X) < n$, which is the desired contradiction.

Every $x \in \mathbb{F}_2^n$ is encoded as follows. Treating x as the input data to the data structure, apply Lemma 4.1 to obtain a subset of cells S and queries Q' such that all queries in Q' can be answered by accessing cells only in S. Let $b_1, \ldots, b_r$ for a basis for the span of Q'. Let $a_1, \ldots, a_{n-r}$ to be such that $b_1, \ldots, b_r, a_1, \ldots, a_{n-r}$ is a basis for $\mathbb{F}_2^n$. We claim that x can be recovered from S and $\langle a_1, x\rangle, \ldots, \langle a_{n-r}, x\rangle$. Indeed, using S, we can construct Q' by simulating the query algorithm on all queries in $\mathbb{F}_2^n$ and discarding the ones that access cells not in S. Once we know Q', we have $b_1, \ldots, b_r$ and hence $\langle b_1, x\rangle, \ldots, \langle b_r, x\rangle$. These inner products together with $\langle a_1, x\rangle, \ldots, \langle a_{n-r}, x\rangle$ recover x.

The total length of this encoding is at most the sum of the number of bits to encode S and $n - r$. We claim that S can be encoded in at most $n/10$ bits, whose proof is deferred to later.

Claim 4.1. *S can be encoded using at most $n/10$ bits.*

Since $\mathbb{E}[|Q'|] \geq 2^{-n/16}$, we have that $r \geq n - n/16 = 15n/16$. Thus, the number of bits to encode x is at most

$$\frac{n}{10} + n - r \leq \frac{n}{10} + \frac{n}{16} < n,$$

implying that $\mathsf{H}(X) < n$. □

Proof of Lemma 4.1. Let S be a uniformly random subset of the cells of size $\left\lceil \frac{n}{128\alpha} \right\rceil$. Define $D(q, S) = 1$ if the query algorithm on $q \in \mathbb{F}_2^n$ only accesses cells in S to compute $\langle q, x\rangle$, where x is the input data; otherwise $D(q, S) = 0$. For every $q \in Q$, we have

$$\begin{aligned}\mathop{\mathbb{E}}_{S}[D(q,S)] = \frac{\binom{s-t}{|S|-t}}{\binom{s}{|S|}} &= \frac{|S| \cdot (|S|-1) \cdots (|S|-t+1)}{s \cdot (s-1) \cdots (s-t+1)} \\ &\geq \left(\frac{|S|-t}{s}\right)^t.\end{aligned}$$

Recall that $|S| \geq \frac{n}{128\alpha}$ and $t \leq \frac{n}{256\alpha}$. Moreover, $\alpha = 2\left(w + \log\left(\frac{sw}{n}\right)\right) \geq 2$. This implies that

$$\left(\frac{|S| - t}{s}\right)^t \geq 2^{-t \cdot \log \frac{256s\alpha}{n}} \geq 2^{-16t \cdot \log \frac{s\alpha}{n}}.$$

So we get

$$\mathop{\mathbb{E}}_{q,S}\left[D(q,S)\right] = \mathop{\mathbb{E}}_{q}\left[\mathop{\mathbb{E}}_{S}\left[D(q,s)\right]\right] \geq \mathop{\mathbb{E}}_{q}\left[2^{-16 \cdot t \cdot \log \frac{s\beta}{n}}\right] \geq 2^{-16 \cdot t \cdot \log \frac{s\beta}{n}}.$$

Therefore, there exists an S such that

$$\mathop{\mathbb{E}}_{q}\left[D(q,S)\right] \geq 2^{-16 \cdot t \cdot \log \frac{s\alpha}{n}} \geq 2^{-n/16},$$

where the last inequality follows from the fact that $16 \cdot t \cdot \log \frac{s\alpha}{n} \leq \frac{n}{16}$. To complete the proof of the lemma, set $Q' = \{q \in \mathbb{F}_2^n \mid D(q,S) = 1\}$. □

Proof of Claim 4.1. Let c be the number of bits to encode S. Using a guarantee of Lemma 4.1 along with Proposition 4.1, we get that

$$\begin{aligned} c &\leq \left\lceil \frac{n}{128\alpha} \right\rceil \cdot w + \left\lceil \frac{n}{128\alpha} \right\rceil \cdot \log \frac{128e \cdot s\alpha}{n} \\ &= \left\lceil \frac{n}{128\alpha} \right\rceil \cdot \left(w + \log \frac{s\alpha}{n}\right) + \left\lceil \frac{n}{128\alpha} \right\rceil \cdot \log 128e. \end{aligned}$$

Since $\alpha \geq 2w$ and $\alpha \geq 2 \log \frac{s}{n}$, we get that $w + \log \frac{s\alpha}{n} \leq 2\alpha$. Moreover, using the fact that $\left\lceil \frac{n}{128\alpha} \right\rceil \leq \frac{n}{128\alpha} + 1$, $\alpha \geq 2$ and $\alpha \leq n/256$, we can say that

$$\begin{aligned} c &\leq 1 + \frac{2n}{128} + 2\alpha + \frac{n \log 128e}{128\beta} + \log 128e \\ &\leq 1 + \frac{2n}{128} + \frac{4.5n}{128(\alpha/2)} + \frac{n}{128} + \log 128e \leq 10 + \frac{7.5n}{128} < \frac{n}{10}, \end{aligned}$$

where the last inequality follows from $n \geq 36$. □

4.3 Lower Bounds for the Vector-Matrix-Vector Problem

Theorem 4.1's proof is presented in this section and we begin with a proof outline. By Yao's min-max principle (Theorem 4.2), it suffices to prove a lower bound on deterministic data structures. The hard distribution on the input data M and query (u, v) we use is given

by sampling $M, (u, v)$ uniformly and independently at random. We prove the theorem by contradiction, and we start by assuming that the query time is small. The proof is carried out in three steps. First, modify the data structure so that for every M, the fraction of queries correctly computed is at least $1/2$. This modification only increases the query time and space by 1, and it can only increase the overall probability of the query algorithm being correct. Second, for a given M, we use a variant of cell sampling (see Lemma 4.3) to obtain a small subset of cells S and a large subset of queries Q' such that all queries in Q' can be computed by only accessing cells in S. Moreover,

$$\begin{aligned}&\Pr\left[\text{query algorithm correctly computes } u^\intercal Mv \mid (u, v) \in Q'\right] \\ &\approx \Pr\left[\text{query algorithm correctly computes } u^\intercal Mv\right].\end{aligned}$$

Third, we show that S can be used to design an efficient protocol for the following communication game: Alice's input is M and Bob's input is (u, v), and the goal is for Bob to correctly compute $u^\intercal Mv$ on a sufficiently good fraction of the inputs after receiving a message from Alice.

We now describe the protocol (see Figure 4.1). Alice sends the locations and contents of S. This ensures that Bob correctly computes $u^\intercal Mv$ on a large fraction of queries in Q'. Alice also communicates the majority value of $u^\intercal Mv$ for $(u, v) \notin Q'$ so that Bob is correct on half of his possible inputs that are not in Q'. Overall, Bob's output is correct on a sufficiently good fraction of all $M, (u, v)$. Since we have assumed that the query time is small, we are able to show that Alice's communication is small. This contradicts a lower bound on the communication complexity of this game. More precisely, we prove the following lower bound.

Theorem 4.7. *Suppose that Alice gets a uniformly random matrix $M \in \mathbb{F}_2^{\sqrt{n}\times\sqrt{n}}$ as input and Bob receives a uniform pair $(u, v) \in \mathbb{F}_2^{\sqrt{n}} \times \mathbb{F}_2^{\sqrt{n}}$ as input. If Alice sends a deterministic message to Bob and Bob computes $u^\intercal Mv$ such that*

$$\Pr_{M,u,v}[\textit{Bob computes } u^\intercal Mv \textit{ correctly}] \geq \frac{1}{2} + \frac{1}{2^{\sqrt{n}/8}},$$

then Alice must communicate at least $n/10$ bits.

Previously, in the *randomized* two-way communication setting, Chattopadhyay, Koucký, Loff, and Mukhopadhyay [32] proved a lower bound for the game given in Theorem 4.7. Their lower bound implies the lower bound in Theorem 4.7 against randomized protocols. We need a lower bound against *deterministic* protocols under the *uniform distribution* on the inputs, and we cannot use their theorem as a black-box. We provide a straightforward proof of Theorem 4.7 in Section 4.3.1 by using the *discrepancy method* on a related communication game (resembling a direct sum, where Bob receives multiple inputs).

We now state the theorem that relates discrepancy to communication complexity. For sets $\mathcal{A}$ and $\mathcal{B}$, $R \subseteq \mathcal{A} \times \mathcal{B}$ is defined to be a rectangle if there exist indicator functions $A_R : \mathcal{A} \to \{0,1\}$ and $B_R : \mathcal{B} \to \{0,1\}$ satisfying $(a,b) \in R$ if and only if $A_R(a) = 1$ and $B_R(b) = 1$.

Theorem 4.8. *Let $0 < \epsilon \leq 1/2$ and $f : \mathcal{A} \times \mathcal{B} \to \{0,1\}$. Let $a \in \mathcal{A}$ and $b \in \mathcal{B}$ be chosen independently and uniformly at random, and suppose that Alice and Bob receive a and b as inputs, respectively. Any deterministic protocol between Alice and Bob computing f correctly with probability at least $\frac{1}{2} + \epsilon$ must have at least $\log\left(\frac{2\epsilon}{D}\right)$ bits of communication, where*

$$D = \max_{\text{rectangle } R \subseteq \mathcal{A} \times \mathcal{B}} \left| \mathop{\mathbb{E}}_{a,b} \left[A_R(a) B_R(b) (-1)^{f(a,b)} \right] \right|.$$

D in the above theorem is referred to as the discrepancy of f with respect to the uniform distribution. See [89, Chapter 5, Theorem 5.2] for a proof of Theorem 4.8.

4.3.1 Proof of Theorem 4.7

We start by discussing a slightly related problem, whose solution will lead to the proof strategy used here. Let $M \in \mathbb{F}_2^{\sqrt{n} \times \sqrt{n}}$, $v \in \mathbb{F}_2^{\sqrt{n}}$, and $e_1, \ldots, e_{\sqrt{n}}$ be the standard basis vectors in $\mathbb{F}_2^{\sqrt{n}}$. Consider the communication game in which Alice gets as input a uniform random M and Bob gets as input a uniform random pair (e_i, v). Bob's goal is to compute $e_i^\intercal M v$ after receiving a message from Alice. To understand how much Alice has to communicate, it is natural to look at the problem where Bob computes $\sum_{i=1}^{\sqrt{n}} e_i^\intercal M v_i$, where $v_1, \ldots, v_{\sqrt{n}} \in F_2^{\sqrt{n}}$.

Now observe that this sum is the same as the trace of $\left(\sum_{i=1}^{\sqrt{n}} e_i v^\intercal\right) M$, which in turn is the inner product between two n-bit vectors. The communication complexity of the inner product between two n-bit vectors is very well understood. Therefore, the lower bound on the amount of communication to compute the inner product between two n-bit vectors translates to a lower bound to the problem of computing $e_i^\intercal M v$. This strategy applied to our setting gives us the following lower bound, which will be used to prove Theorem 4.7. Our presentation closely follows [89, Chapter 5].

Lemma 4.2. *Let $0 < \epsilon \leq 1/2$ and let k be an integer. Alice gets a uniformly random $M \in \mathbb{F}_2^{\sqrt{n}\times\sqrt{n}}$ as input and Bob receives k uniform pairs $(u_1, v_1), \ldots, (u_k, v_k) \in \mathbb{F}_2^{\sqrt{n}} \times \mathbb{F}_2^{\sqrt{n}}$ as input. Assume that Alice communicates a deterministic message to Bob, and Bob computes $\sum_{i=1}^k u_i^\intercal M v_i$ with*

$$\Pr_{M,u_1,v_1,\ldots,u_k,v_k}\left[\textit{Bob computes } \sum_{i=1}^{k} u_i^\intercal M v_i \textit{ correctly}\right] \geq \frac{1}{2} + \epsilon.$$

If $k \leq \sqrt{n}$, then Alice must communicate at least $9k\sqrt{n}/40 - \log(1/\epsilon)$ bits.

Proof. We use the discrepancy method to prove the communication lower bound. For ease of notation, let $\mathcal{U} = u_1, \ldots, u_k$ and $\mathcal{V} = v_1 \ldots, v_k$. Let R be a rectangle of the communication matrix, which is defined by indicator functions A_R and B_R such that $(M, (\mathcal{U}, \mathcal{V}))$ is in the rectangle R if and only if $A_R(M) = 1$ and $B_R((\mathcal{U}, \mathcal{V})) = 1$. We upper bound the discrepancy under the uniform distribution on $M, \mathcal{U}, \mathcal{V}$. In other words, we claim that for every rectangle R,

$$\left|\mathop{\mathbb{E}}_{M,(\mathcal{U},\mathcal{V})}\left[A_R(M) B_R((\mathcal{U}, \mathcal{V}))\,(-1)^{\sum_{i=1}^k u_i^\intercal M v_i}\right]\right| \leq 2 \cdot 2^{-9k\sqrt{n}/40}. \tag{4.1}$$

Theorem 4.8 and Eq. (4.1) imply that Alice must communicate at least $9k\sqrt{n}/40 - \log(1/\epsilon)$

bits. We are left with the proof of Eq. (4.1).

$$\begin{aligned}
&\left(\mathop{\mathbb{E}}_{(\mathcal{U},\mathcal{V})}\left[B_R\left((\mathcal{U},\mathcal{V})\right)\mathop{\mathbb{E}}_{M}\left[A_R(M)(-1)^{\sum_{i=1}^k u_i^\intercal M v_i}\right]\right]\right)^2\\
&\leq \mathop{\mathbb{E}}_{(\mathcal{U},\mathcal{V})}\left[B_R\left((\mathcal{U},\mathcal{V})\right)^2\left(\mathop{\mathbb{E}}_{M}\left[A_R(M)(-1)^{\sum_{i=1}^k u_i^\intercal M v_i}\right]\right)^2\right]\\
&\leq \mathop{\mathbb{E}}_{(\mathcal{U},\mathcal{V})}\left[\left(\mathop{\mathbb{E}}_{M}\left[A_R(M)(-1)^{\sum_{i=1}^k u_i^\intercal M v_i}\right]\right)^2\right].
\end{aligned}$$

where the first inequality follows from convexity and the second one follows from the fact that $B_R\left((\mathcal{U},\mathcal{V})\right)\leq 1$. Now

$$\begin{aligned}
\mathop{\mathbb{E}}_{(\mathcal{U},\mathcal{V})}\left[\left(\mathop{\mathbb{E}}_{M}\left[A_R(M)(-1)^{\sum_{i=1}^k u_i^\intercal M v_i}\right]\right)^2\right] &\leq \mathop{\mathbb{E}}_{(\mathcal{U},\mathcal{V}),M,M'}\left[A_R(M)A_R(M')(-1)^{\sum_{i=1}^k (u_i^\intercal M v_i + u_i^\intercal M' v_i)}\right]\\
&\leq \mathop{\mathbb{E}}_{M,M'}\left[\left|\mathop{\mathbb{E}}_{(\mathcal{U},\mathcal{V})}\left[(-1)^{\sum_{i=1}^k u_i^\intercal (M+M') v_i}\right]\right|\right]\\
&= \mathop{\mathbb{E}}_{M}\left[\left|\mathop{\mathbb{E}}_{(u,v)}\left[(-1)^{u^\intercal M v}\right]\right|^k\right],
\end{aligned}$$

where the last equality follows from the fact that $(u_1,v_1),\ldots,(u_k,v_k)$ are chosen independent of each other and $M+M'$ is uniformly distributed as M and M' are chosen uniformly and independently at random. We are left with upper bounding $\mathbb{E}_M\left[\left|\mathbb{E}_{(u,v)}\left[(-1)^{u^\intercal M v}\right]\right|^k\right]$. First note that if M has rank r, then $\mathbb{E}_{u,v}\left[(-1)^{u^\intercal M v}\right]=2^{-r}$. This is because,

$$\mathop{\mathbb{E}}_{u,v}\left[(-1)^{u^\intercal M v}\right]=\frac{1}{2\sqrt{n}}\cdot\left(\sum_{v:Mv=0}1\right)+\frac{1}{2\sqrt{n}}\cdot\left(\sum_{v:Mv\neq 0}\mathop{\mathbb{E}}_{u}\left[(-1)^{u^\intercal M v}\right]\right)=\frac{2^{\sqrt{n}-r}}{2\sqrt{n}}+0=2^{-r}.$$

In addition, $\Pr_M\left[\text{rank of } M\leq 9\sqrt{n}/20\right]\leq 2^{-9n/10}$. Indeed, the number of matrices in $\mathbb{F}_2^{\sqrt{n}\times\sqrt{n}}$ of rank at most k is at most

$$\binom{2^{\sqrt{n}}}{k}\cdot\left(2^k\right)^{\sqrt{n}}\leq 2^{2k\sqrt{n}}.$$

Therefore, using the law of total expectation, we have that

$$\mathop{\mathbb{E}}_{M}\left[\left|\mathop{\mathbb{E}}_{(u,v)}\left[(-1)^{u^\intercal M v}\right]\right|^k\right]\leq\Pr_{M}\left[\text{rank of } M\leq 9\sqrt{n}/20\right]+2^{-9k\sqrt{n}/20}\leq 2\cdot 2^{-9k\sqrt{n}/20},$$

where the last inequality followed from the fact that $k\leq\sqrt{n}$. □

Proof of Theorem 4.7. Let c be the number of bits communicated by Alice. We show that $c > n/10$. Define $Z_M(u,v) = 1$ if Bob correctly computes $u^\intercal Mv$ and $Z_M(u,v) = -1$ otherwise. By the definition of $Z_M(u,v)$ and the lower bound on the probability of Bob's computation being correct, we have that $\mathbb{E}_{M,u,v}\left[Z_M(u,v)\right] \geq 2 \cdot 2^{-\sqrt{n}/8}$.

We note that it is without loss of generality that $\mathbb{E}_{u,v}\left[Z_M(u,v)\right] \geq 0$ for every $M \in \mathbb{F}_2^{\sqrt{n}\times\sqrt{n}}$. This is because Alice on input M can send an extra bit indicating whether $\mathbb{E}_{u,v}\left[Z_M(u,v)\right] < 0$ and Bob will flip his output accordingly.

We now use the given protocol to design a protocol for a new communication game: Suppose that Alice gets a uniformly random $M \in \mathbb{F}_2^{\sqrt{n}\times\sqrt{n}}$ as input and Bob receives $\sqrt{n}$ uniform pairs $(u_1,v_1),\ldots,(u_{\sqrt{n}},v_{\sqrt{n}}) \in \mathbb{F}_2^{\sqrt{n}} \times \mathbb{F}_2^{\sqrt{n}}$ as input. We will use Lemma 4.2 with $k = \sqrt{n}$ to obtain the desired lower bound on c.

We claim that there is a communication protocol in which Alice communicates c bits and Bob computes $\sum_{i=1}^{\sqrt{n}} u_i^\intercal M v_i$ such that

$$\Pr_{M,u_1,v_1,\ldots,u_{\sqrt{n}},v_{\sqrt{n}}}\left[\text{Bob computes } \sum_{i=1}^{\sqrt{n}} u_i^\intercal M v_i \text{ correctly}\right] \geq \frac{1}{2} + \frac{2^{\sqrt{n}-1}}{2^{n/8}}. \tag{4.2}$$

Alice's message is same as before, and Bob computes each of $u_i^\intercal M v_i$ separately and outputs the sum modulo 2. We now prove Eq. (4.2). For a fixed M, the probability that Bob correctly computes $\sum_{i=1}^{\sqrt{n}} u_i^\intercal M v_i$ is $\frac{1}{2}\left(1 + \left(\mathbb{E}_{u,v}\left[Z_M(u,v)\right]\right)^{\sqrt{n}}\right)$. Therefore the overall probability that Bob correctly computes $\sum_{i=1}^{\sqrt{n}} u_i^\intercal M v_i$ is at least

$$\frac{1}{2}\left(1 + \frac{\sum_M \left(\mathbb{E}_{u,v}\left[Z_M(u,v)\right]\right)^{\sqrt{n}}}{2^n}\right) \geq \frac{1}{2}\left(1 + \left(\mathop{\mathbb{E}}_{M,u,v}\left[Z_M(u,v)\right]\right)^{\sqrt{n}}\right) \geq \frac{1}{2} + \frac{2^{\sqrt{n}-1}}{2^{n/8}},$$

where the first inequality follows from convexity of the function $f(x) = x^k$ with $k = \sqrt{n}$. Applying Lemma 4.2 with $k = \sqrt{n}$ implies that $c > n/10$, which completes the proof of the theorem. □

4.3.2 Proof of Theorem 4.1

If $n < 36$, the theorem is vacuously true as $c \leq 1/36$. For the rest of the argument we will assume that $n \geq 36$. We prove a lower bound on the query time t against deterministic data

structures with space s and word size w. Suppose that the input data M and query (u, v) is given by choosing M, u, v uniformly and independently at random, and the query algorithm is guaranteed to satisfy

$$\Pr_{M,u,v}[\text{query algorithm computes } u^\intercal Mv \text{ correctly}] \geq \frac{1}{2} + 2^{-\sqrt{n}/16}.$$

By Yao's minmax principle, this will imply a lower bound on randomized data structures.

We first modify the given data structure to ensure that for every $M \in \mathbb{F}_2^{\sqrt{n}\times\sqrt{n}}$, the probability that $u^\intercal Mv$ is correctly computed is at least $1/2$. Assume that we have a data structure with query time t', space s' and word size w. The modified data structure stores an extra bit indicating whether the $\Pr_{u,v}[\text{query algorithm computes } u^\intercal Mv \text{ correctly}]$ is less than $1/2$ or not for a given M. The query algorithm is the same as before, but accesses this extra bit to flip the output if it is set to 1. Clearly, the new data structure has query time $t = t' + 1$, space $s = s' + 1$ and word size w. Moreover, under this modification, we have

- $\Pr_{M,u,v}[\text{query algorithm computes } u^\intercal Mv \text{ correctly}] \geq 1/2 + 2^{-\sqrt{n}/16}$.
- $\Pr_{u,v}[\text{query algorithm computes } u^\intercal Mv \text{ correctly}] \geq 1/2$ for every M.

In the rest of the proof, we work with this modification to show that $t \geq \Omega\left(\min\left\{\frac{n}{\beta}, \frac{\sqrt{n}}{\log\frac{s\beta}{n}}\right\}\right)$, where $\beta = 2(w + \log sw/n)$. Observe that $\beta \leq n/256$; otherwise the lower bound is vacuous.

Assume by contradiction that $t \leq \min\left\{\frac{n}{256\beta}, \frac{\sqrt{n}}{256\log\frac{s\beta}{n}}\right\}$. Define $Z_M(u, v) = 1$ if the query algorithm correctly computes $u^\intercal Mv$, and -1 otherwise. We have

$$\mathop{\mathbb{E}}_{M,u,v}[Z_M(u, v)] = 2 \cdot \Pr_{M,u,v}[\text{query algorithm computes } u^\intercal Mv \text{ correctly}] - 1 \geq 2 \cdot 2^{-\sqrt{n}/16}. \tag{4.3}$$

Note that $\mathbb{E}_{M,u,v}[Z_M(u, v)]$ captures the *advantage* or *bias* of the data structure - it is much more convenient to work with the advantage than the probability of the query algorithm being correct.

The following lemma, a variant of cell sampling, guarantees the existence of a small subset S of cells such that a large number of queries Q' can be computed by only accessing S, and $\mathbb{E}_{u,v}[Z_M(u, v) \mid (u, v) \in Q'] \approx \mathbb{E}_{u,v}[Z_M(u, v)]$.

Input: Alice's input is M and Bob's input is (u, v)

Output: Alice communicates a deterministic message and Bob computes $u^\intercal Mv$.

1 Let $Q_1 = \{(u, v) \mid Z_M(u, v) = 1\}$ and $Q_2 = \{(u, v) \mid Z_M(u, v) = -1\}$;

2 Apply Lemma 4.3 with Q_1, Q_2 to obtain a subset of cells S and subsets $Q_1' \subseteq Q_1$ and $Q_2' \subseteq Q_2$;

3 Let $b \in \{0, 1\}$ be such that $\Pr_{u,v}[u^\intercal Mv = b \mid (u, v) \notin Q'] \geq \Pr_{u,v}[u^\intercal Mv = 1 - b \mid (u, v) \notin Q']$, where $Q' = Q_1' \cup Q_2'$;

4 Alice communicates b followed by locations and contents of S;

5 **if** $(u, v) \in Q'$ **then** Bob uses the query algorithm to compute $u^\intercal Mv$;

6 **else** Bob outputs b;

Figure 4.1: One-way protocol on inputs $M, (u, v)$ computing $u^\intercal Mv$.

Lemma 4.3. *Let $M \in \mathbb{F}_2^{\sqrt{n} \times \sqrt{n}}$. Define*

$$Q_1 = \{(u, v) \mid Z_M(u, v) = 1\} \qquad \text{and} \qquad Q_2 = \{(u, v) \mid Z_M(u, v) = -1\}.$$

If $t \leq \min\left\{\frac{n}{256\beta}, \frac{\sqrt{n}}{256 \log \frac{s\beta}{n}}\right\}$, then there exist a subset of cells S, and subsets $Q_1' \subseteq Q_1$ and $Q_2' \subseteq Q_2$ such that

1. $|S| = \left\lceil \frac{n}{128\beta} \right\rceil$,
2. $\Pr_{u,v}[(u, v) \in Q_1'] - \Pr_{u,v}[(u, v) \in Q_2'] \geq \mathbb{E}_{u,v}[Z_M(u, v)] \cdot 2^{-\sqrt{n}/16}$,
3. *$Q_1' \cup Q_2'$ is the set of all queries computed by accessing cells only in S.*

We move on to the final step of the proof of Theorem 4.1. What is left is to design a one-way protocol using the sets guaranteed by Lemma 4.3. The protocol is described in Figure 4.1. We will show the validity of this protocol by showing that both Alice and Bob

know the subset Q' of queries. Since Alice's input is M, she knows the contents of all the cells, which gives S. With regard to knowing Q', the locations and contents of cells in S suffice. This is because the query algorithm can be simulated on all queries to check if any cell outside of S is being accessed. We are proving Theorem 4.1 by contradicting Theorem 4.7, which is achieved by the following.

Lemma 4.4. *The protocol in Figure 4.1 has the following guarantees (a) Alice communicates fewer than $n/10$ bits, and (b) $\Pr_{M,u,v}[$Bob computes $u^\intercal Mv$ correctly$] \geq 1/2 + 1/2^{\sqrt{n}/8}$.*

Now, we need to prove Lemmas 4.3 and 4.4 to complete the proof of Theorem 4.1.

Proof of Lemma 4.3. Let S be a uniformly random subset of the cells of size $|S| = \left\lceil \frac{n}{128\beta} \right\rceil$. Define $D(u, v, S) = Z_M(u, v)$ if the query algorithm only accesses cells in S to compute $u^\intercal Mv$; otherwise $D(u, v, S) = 0$. By linearity of expectation,

$$\begin{aligned}\mathop{\mathbb{E}}_{u,v,S}[D(u,v,S)] = \mathop{\mathbb{E}}_{u,v}[Z_M(u,v)] \cdot \frac{\binom{s-t}{|S|-t}}{\binom{s}{|S|}} &= \mathop{\mathbb{E}}_{u,v}[Z_M(u,v)] \cdot \frac{|S| \cdot (|S|-1) \cdots (|S|-t+1)}{s \cdot (s-1) \cdots (s-t+1)} \\ &\geq \mathop{\mathbb{E}}_{u,v}[Z_M(u,v)] \cdot \left(\frac{|S|-t}{s}\right)^t.\end{aligned}$$

Recall that $|S| \geq \frac{n}{128\beta}$ and $t \leq \frac{n}{256\beta}$. Moreover, $\beta = 2(w + \log sw/n) \geq 2$. This implies that

$$\left(\frac{|S|-t}{s}\right)^t \geq 2^{-t \cdot \log \frac{256s\beta}{n}} \geq 2^{-16t \cdot \log \frac{s\beta}{n}}.$$

So we get $\mathbb{E}_{u,v,S}[D(u,v,S)] \geq \mathbb{E}_{u,v}[Z_M(u,v)] \cdot 2^{-16 \cdot t \cdot \log \frac{s\beta}{n}}$. Therefore, there exists an S such that

$$\mathop{\mathbb{E}}_{u,v}[D(u,v,S)] \geq \mathop{\mathbb{E}}_{u,v}[Z_M(u,v)] \cdot 2^{-16 \cdot t \cdot \log \frac{s\beta}{n}} \geq \mathop{\mathbb{E}}_{u,v}[Z_M(u,v)] \cdot 2^{-\sqrt{n}/16},$$

where the last inequality follows from the fact that $16 \cdot t \cdot \log \frac{s\beta}{n} \leq \sqrt{n}/16$. Setting

$$Q'_1 = \{(u,v) \in Q_1 \mid D(u,v,S) = 1\} \text{ and } Q'_2 = \{(u,v) \in Q_2 \mid D(u,v,S) = -1\}$$

completes the proof of the lemma. □

Proof of Lemma 4.4. We first prove part (a). Recall that $\beta = 2\left(w + \log \frac{sw}{n}\right)$. Let c be the number of bits communicated by Alice. By Proposition 4.1 and the definition of β,

$$\begin{aligned} c &\leq 1 + \left\lceil \frac{n}{128\beta} \right\rceil \cdot w + \left\lceil \frac{n}{128\beta} \right\rceil \cdot \log \frac{128e \cdot s\beta}{n} \\ &= 1 + \left\lceil \frac{n}{128\beta} \right\rceil \cdot \left(w + \log \frac{s\beta}{n} \right) + \left\lceil \frac{n}{128\beta} \right\rceil \cdot \log 128e. \end{aligned}$$

Since $\beta \geq 2w$, $\beta \geq 2 \log \frac{s}{n}$ and $\beta \geq \log \beta$, we get that $w + \log \frac{s\beta}{n} \leq 2\beta$. Moreover, using the fact that $\left\lceil \frac{n}{128\beta} \right\rceil \leq \frac{n}{128\beta} + 1$, $\beta \geq 2$ and $\beta \leq n/256$, we can say that

$$\begin{aligned} c &\leq 1 + \frac{2n}{128} + 2\beta + \frac{n \log 128e}{128\beta} + \log 128e \\ &\leq 1 + \frac{2n}{128} + \frac{4.5n}{128(\beta/2)} + \frac{n}{128} + \log 128e \leq 10 + \frac{7.5n}{128} < \frac{n}{10}, \end{aligned}$$

where the last inequality follows from $n \geq 36$.

We now prove part (b) of the claim. Define $Z'_M(u, v) = 1$ if the Bob correctly computes $u^\intercal M v$ and $Z'_M(u, v) = -1$ otherwise. The probability with which Bob correctly computes $u^\intercal M v$ is given by $\left(1 + \mathbb{E}_{M,u,v}\left[Z'_M(u, v)\right]\right)/2$. We will show that $\mathbb{E}_{M,u,v}\left[Z'_M(u, v)\right] \geq 2 \cdot 2^{-\sqrt{n}/8}$, which will imply that the probability of being correct is at least $1/2 + 2^{-\sqrt{n}/8}$.

Let Q_1, Q_2, Q'_1, Q'_2, and Q' be as defined in the protocol in Figure 4.1. We first establish some properties about these sets. We know that

$$\Pr_{u,v}[(u, v) \in Q_1] - \Pr_{u,v}[(u, v) \in Q_2] = \mathop{\mathbb{E}}_{u,v}\left[Z_M(u, v)\right].$$

Moreover, the application of Lemma 4.3 in the protocol is valid since $t \leq \frac{n}{256\alpha}$, and hence

$$\Pr_{u,v}\left[(u, v) \in Q'_1\right] - \Pr_{u,v}\left[(u, v) \in Q'_2\right] \geq \mathop{\mathbb{E}}_{u,v}\left[Z_M(u, v)\right] \cdot 2^{-\sqrt{n}/16}. \tag{4.4}$$

Since Bob can simulate the query algorithm on Q' by accessing only S, which is guaranteed

by Lemma 4.3, we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{u,v}\left[Z'_M(u,v)\right] \\
&= \Pr_{u,v}\left[(u,v)\in Q'\right]\cdot\left(\Pr_{u,v}\left[(u,v)\in Q'_1 \mid (u,v)\in Q'\right]-\Pr_{u,v}\left[(u,v)\in Q'_2 \mid (u,v)\in Q'\right]\right) \\
&\qquad + \Pr_{u,v}\left[(u,v)\notin Q'\right]\left(\Pr_{u,v}\left[u^\intercal Mv = b \mid (u,v)\notin Q'\right]-\Pr_{u,v}\left[u^\intercal Mv = 1-b \mid (u,v)\notin Q'\right]\right) \\
&\geq \left(\Pr_{u,v}\left[(u,v)\in Q'_1\right]-\Pr_{u,v}\left[(u,v)\in Q'_2\right]\right) \geq \mathop{\mathbb{E}}_{u,v}\left[Z_M(u,v)\right]\cdot 2^{-\sqrt{n}/16},
\end{aligned}$$

where the first inequality follows from the choice of b and the second inequality used Eq. (4.4).

To conclude,

$$\begin{aligned}
\mathop{\mathbb{E}}_{M,u,v}\left[Z'_M(u,v)\right] = \mathop{\mathbb{E}}_{M}\left[\mathop{\mathbb{E}}_{u,v}\left[Z'_M(u,v)\right]\right] &\geq \mathop{\mathbb{E}}_{M}\left[\mathop{\mathbb{E}}_{u,v}\left[Z_M(u,v)\right]\cdot 2^{-\sqrt{n}/16}\right] \\
&= \mathop{\mathbb{E}}_{M}\left[\mathop{\mathbb{E}}_{u,v}\left[Z_M(u,v)\right]\right]\cdot 2^{-\sqrt{n}/16} \\
&= \mathop{\mathbb{E}}_{M,u,v}\left[Z_M(u,v)\right]\cdot 2^{-\sqrt{n}/16} \geq 2\cdot 2^{-\sqrt{n}/8},
\end{aligned}$$

where the last inequality follows from Eq. (4.3). □

Chapter 5

SYSTEMATIC LINEAR DATA STRUCTURES AND RIGIDITY

This chapter concerns results related to the connections between systematic linear data structures, linear data structures and rigidity. Recall from the introduction that if a query set Q is close to an r-dimensional subspace, in that all points in Q are at a Hamming distance of at most t from that subspace, then there is a data structure for Q with redundancy r and query time t in the systematic linear model. Here, we will show that these two notions are indeed equivalent. We will then use this equivalence to prove a new lower bound against systematic linear data structures for the Vector-Matrix-Vector problem by proving a new rigidity lower bound for the set of vectors corresponding to rank one matrices. Moreover, we will also establish a relationship between the linear model and the systematic linear model, which will then help us prove that strong query time lower bounds in the linear model will imply new rigidity lower bounds. Before formally stating our results, we start with some basic definitions.

Let n be a natural number. For the rest of this chapter, let $m = m(n)$ and $t = t(n)$ and $r = r(n)$ denote positive integers, with $m \geq n \geq t, r$. Alon, Panigrahy and Yekhanin defined the following notion of a rigid set [11].

Definition 5.1 (Rigid Set)**.** *A set $Q \subseteq \mathbb{F}_2^n$ is (r,t)-rigid if for every subspace $U \subseteq \mathbb{F}_2^n$ with dimension at most r, some vector $q \in Q$ has Hamming distance at least t from all vectors in U, that is, $d_H(q, U) \geq t$.*

We define (r', t')-rigid for non-integral r', t' to mean $(\lfloor r' \rfloor, \lceil t' \rceil)$-rigid. It will be convenient to equate a set Q with a matrix M_Q by arranging vectors in Q as rows in M_Q in any order. If Q is (r,t)-rigid and $|Q| = m$, then the corresponding matrix $M_Q \in \mathbb{F}_2^{m \times n}$ is rigid in the usual sense: for any rank r matrix A, some row in $(M_Q - A)$ contains at least t nonzero

entries. Hence, we may refer to rigid sets and rigid rectangular matrices interchangeably. A matrix in $\mathbb{F}_2^{m\times n}$ (or a set of n-dimensional vectors) is *explicit* if every entry can be computed in $\mathsf{poly}(n)$ time.

A random $m \times n$ matrix with $m = \text{poly}(n)$ will be $(\epsilon n, \delta n/\log n)$-rigid with high probability for some constants $\epsilon, \delta \in (0, 1)$. The key challenge here is to construct explicit rigid matrices, because they provide circuit lower bounds for functions that can be described in polynomial time [97]. Alon, Panigrahy and Yekhanin [11] followed by Alon and Cohen [8] exhibit multiple examples of explicit $m \times n$ matrices that are (r, t)-rigid with

$$t \geq \min\left\{\frac{cn}{r}\log\frac{m}{r},\ n\right\} \tag{5.1}$$

where $m \geq n$ and c is a constant. Note that when $r = \epsilon n$, the current best bound is $t = \Omega\left(\log\frac{m}{n}\right)$. For $m = \mathsf{poly}(n)$, this amounts to $t = \Omega(\log n)$, exponentially far from the ideal bounds (i.e., matching random constructions). It is an important open problem to improve the dependence on m in Eq. (5.1) and to find other candidate sets that may be rigid with better parameters.

Our connection between rigidity and data structures arises via the inner product problem. The task is to preprocess a vector $v \in \mathbb{F}_2^n$ to compute inner products. The queries are specified by $Q \subseteq \mathbb{F}_2^n$, which is called the *query set*. The data structure must compute the inner product of v and any $q \in Q$, that is, $\langle q, v\rangle = \sum_{i=1}^{n} q[i] \cdot v[i] \mod 2$, where $q[i]$ denotes the i-th coordinate of q.

Consider the following model for solving this problem, known as a systematic linear data structure. During preprocessing, the data structure stores x along with the evaluations of r linear functions $\langle a_1, x\rangle, \ldots, \langle a_r, x\rangle$, where these inner products are single bits, and $a_1, \ldots, a_r$ denote vectors in $\mathbb{F}_2^n$. To compute the answer on query q, the data structure accesses these r bits in addition to any t entries of x. That is, the r linear functions are fixed, and the t bits from x may depend on q and the linear functions. Finally, the query algorithm must output a linear function of these r bits and the t entries of x. In this fashion it must be able to correctly compute $\langle q, x\rangle$ for all queries $q \in Q$. We note that a result of Jukna

and Schnitger [59] shows that the $\{a_1, \ldots, a_r\}$ vectors do not depend on x without loss of generality. Letting $T(Q, r)$ denote the minimum value t of the best data structure for this problem (over worst-case x), we formalize the model as follows.

Definition 5.2 (Systematic Linear Model)**.** *Let $Q \subseteq \mathbb{F}_2^n$ be a set. Define $T(Q, r)$ to be the maximum over all $x \in \mathbb{F}_2^n$ of the minimum t sufficient to compute the inner product $\langle q, x \rangle$ for every $q \in Q$ when only allowed to output a linear function of r precomputed linear functions of x along with any t bits of x.*

Note that the model does not charge the query time for accessing the r precomputed bits, even if $t \ll r$. This coincides with the systematic model studied by Chakraborty, Kamma and Larsen [30].

All results in this chapter are joint work with Rashtchian [76]. Our first result is that the rigidity of a set Q corresponds to the time complexity $T(Q, r)$ in the systematic linear data structure model. Some aspects of this result are implicit in prior work [59, 87], but no previous work seems to show this exact correspondence.

Theorem 5.1. *A set $Q \subseteq \mathbb{F}_2^n$ is (r, t)-rigid if and only if $T(Q, r) \geq t$.*

Equivalences all the way down. We note that the systematic data structure model is identical to the common bits model defined by Valiant [99]. Corrigan-Gibbs and Kogan [34] demonstrate a relationship between the common bits model and a variant of the systematic model defined by Gal and Miltersen [47]. The common bits model is nothing but a certain depth two circuit, and the systematic linear model is simply the common bits model with the restriction that the common bits and output gates are linear functions [87]. Hence, in language of data structures, the linearization conjecture of Jukna and Schnitger posits that the systematic linear model is asymptotically as powerful as the systematic model for answering linear queries [59].

We use the equivalence in Theorem 5.1 to obtain new lower bounds for the Vector-Matrix-Vector problem in systematic linear model. It will be convenient to consider a $\sqrt{n} \times \sqrt{n}$ matrix

as an n-bit vector $\mathsf{vec}(M)$ by concatenating consecutive rows. More formally, let $x = \mathsf{vec}(M)$, and for $i \in \{1, 2, \ldots, n\}$, set $x[i] = M[a, b]$, where a and b satisfy $i = (a-1)\sqrt{n} + b$ and $a, b \in \{1, 2, \ldots, \sqrt{n}\}$. Then, $u^\intercal M v = \langle \mathsf{vec}(uv^\intercal), \mathsf{vec}(M) \rangle$. In this way we consider the Vector-Matrix-Vector problem as a special case of the inner product problem. The query set is the collection of rank one binary matrices. Let $\Upsilon \subseteq \mathbb{F}_2^n$ denote the set of vectors obtained from rank one binary matrices via $M \mapsto \mathsf{vec}(M)$, that is,

$$\Upsilon = \left\{ \mathsf{vec}(uv^\intercal) \;\middle|\; u, v \in \mathbb{F}_2^{\sqrt{n} \times \sqrt{n}} \right\} \subseteq \mathbb{F}_2^n. \tag{5.2}$$

Note that $|\Upsilon| = 2^{2\sqrt{n}} - 2 \cdot 2^{\sqrt{n}} + 1$. Our second result is a lower bound the rigidity of Υ, defined in Eq. (5.2). This also implies a lower bound in the systematic linear model. The proof is inspired by a result of Alon, Panigrahy, and Yekahnin [11].

Theorem 5.2. *Let $n \geq 1024$. The set $\Upsilon \subseteq \mathbb{F}_2^n$ of rank one matrices is (r, t)-rigid with $t \geq \frac{n^{3/2}}{128 \cdot \max\{\sqrt{n}, r\}} \geq \Omega\left(\frac{n}{r} \cdot \log |\Upsilon|\right)$.*

We improve the prior bound due to Chakraborty, Kamma and Larsen [30] by an $\Omega(\log n)$ factor. For example, when $r \leq \sqrt{n}$, then $t = \Omega(n)$, and when $r = n/2$, then $t = \Omega\left(n^{1/2}\right)$. Theorem 5.2 matches Eq. (5.1), the current best bound for explicit rigid sets. We do not know whether there is a subspace U of linear dimension such that all elements of Υ are at distance $o(n)$ from U (unlike for some set rigidity results, where the bounds are tight). As a corollary of Theorem 5.1, we immediately get that

$$T(\Upsilon, r) \geq \frac{n^{3/2}}{128 \cdot \max\{\sqrt{n}, r\}}.$$

In other words, we prove a lower bound for the Vector-Matrix-Vector problem in the systematic linear model that improves the prior bound by an $\Omega(\log n)$ factor. The proof of Theorem 5.2 appears in Section 5.2.

As our third result, we relate systematic linear data structures and linear data structures. Consequently, this gives rise to a new connection between the latter and rigidity. This connection along with a detailed comparison with previous work [36] is discussed in Section 5.3.

In Section 5.4, we present an elementary proof of the main theorem of [36]. We now prove Theorem 5.1

5.1 *Proof of Theorem 5.1*

We first prove that $T(Q, r) \geq t$ implies that Q is (r, t)-rigid. Assume for contradiction that there is an r-dimensional subspace U such that $d_H(q, U) < t$ for all $q \in Q$. Let $x \in \mathbb{F}_2^n$ be the input data. Store x along with the r bits $\langle b_1, x\rangle, \ldots, \langle b_r, x\rangle$, where $b_1, \ldots, b_r$ form a basis for U. For every $q \in Q$, there exists $u_q \in U$ such that $q - u_q$ has Hamming weight less than t. Using the r redundant bits, the algorithm on query q can compute $\langle u_q, x\rangle$ by writing u_q in terms of the stored basis vectors. Then, it computes $\langle q - u_q, x\rangle$ by accessing fewer than t coordinates of x. Since $\langle q, x\rangle = \langle u_q, x\rangle + \langle q - u_q, x\rangle$, we have that $T(Q, r) < t$, which is a contradiction.

We now prove that if Q is (r, t)-rigid, then $T(Q, r) \geq t$. Let $e_1, \ldots, e_n$ denote the standard basis, and let $k = T(Q, r)$ be the query time. We show that $k \geq t$. Consider a systematic linear data structure whose redundant bits are given by $\langle a_1, x\rangle, \ldots, \langle a_r, x\rangle$. Let U denote the span of $\{a_1, \ldots, a_r\}$. As Q is (r, t)-rigid, there exists $q^* \in Q$ with $d_H(q^*, U) \geq t$. When q^* is the query, assume that the query algorithm accesses the bits $x_{i_1}, \ldots, x_{i_k}$ for indices $i_1, \ldots, i_k$ to compute $\langle q^*, x\rangle$. Now, define U' to be the span of $\{a_1, \ldots, a_r, e_{i_1}, \ldots, e_{i_k}\}$. Observe that all points in U' are at distance at most k from U. Thus, $d_H(q^*, U) \leq d_H(q^*, U') + k$. We will show that $d_H(q^*, U') = 0$, which implies that $k \geq t$. We claim that if $d_H(q^*, U') \geq 1$, then the query algorithm makes an error. Since $d_H(q^*, U') \geq 1$, there exists a vector y with $\langle y, q^*\rangle = 1$. Moreover, this vector can be chosen to satisfy $\langle y, u\rangle = 0$ for every $u \in U'$. In other words, for every $u \in U'$ we have $\langle y + x, u\rangle = \langle y, u\rangle + \langle x, u\rangle = \langle x, u\rangle$. Hence, the query algorithm sees the same values on input data $y + x$ and x because it only accesses the input via vectors in U', and we have $u \in U'$. Thus, the algorithm on query q^* must err either on input $y + x$ or x because $\langle q^*, y + x\rangle \neq \langle q^*, x\rangle$.

5.2 Rigidity Lower Bound for Rank One Matrices

In this section we prove Theorem 5.2. Before presenting the proof, we first establish a useful property about the distance of a point from a subspace.

Lemma 5.1. *Let $V \subseteq \mathbb{F}_2^\ell$ be a subspace. For $u_1, u_2 \in \mathbb{F}_2^\ell$, $d_H(u_1 + u_2, V) \leq d_H(u_1, V) + d_H(u_2, V)$.*

Proof. Let $u_1', u_2' \in V$ be the points in V closest to u_1 and u_2 respectively. Since $u_1' + u_2' \in V$, we have

$$d_H(u_1 + u_2, V) \leq d_H(u_1 + u_2, u_1' + u_2') = d_H(u_1 + u_2, u_1' + u_2').$$

Note that $d_H(u_1 + u_2, u_1' + u_2')$ is the number of ones in $u_1 + u_2 + u_1' + u_2'$, which is at most the sum of the number of ones in $u_1 + u_1'$ and $u_2 + u_2'$. Therefore,

$$d_H(u_1 + u_2, u_1' + u_2') \leq d_H(u_1, u_1') + d_H(u_2, u_2') = d_H(u_1, V) + d_H(u_2, V). \qquad \square$$

A simple counting argument shows the existence of a point that is far away in Hamming distance from a collection of large sized sets.

Lemma 5.2. *Let $V_1, \ldots, V_k$ be subsets of $\mathbb{F}_2^\ell$, each of size at most $2^{\ell/2}$. If $k < 2^{\ell/4}$, then there is a vector $v \in \mathbb{F}_2^\ell$ such that the Hamming distance of v from each V_i is at least $\ell/16$.*

Proof. For every $i \in [k]$, define $\mathsf{B}(V_i, \ell/16) = \left|\left\{v \in \mathbb{F}_2^\ell \;\middle|\; d_H(v, V_i) < \ell/16\right\}\right|$. For any $u \in V_i$, the number of vectors in $\mathbb{F}_2^\ell$ at a distance less than $\ell/16$ from u is at most $\sum_{j=0}^{\ell/16} \binom{\ell}{j} \leq 2^{\ell/4}$, where the inequality follows from Proposition 2.1(b). Hence $\mathsf{B}(V_i, \ell/16) \leq |V_i| \cdot 2^{\ell/4} = 2^{3\ell/4}$. Since

$$\sum_{i=1}^{k} \mathsf{B}(V_i, \ell/16) \leq k \cdot 2^{3\ell/4} < 2^\ell,$$

there is a $v \in \mathbb{F}_2^\ell$ such that $d_H(v, V_i) \geq \ell/16$ for every $i \in [k]$. $\square$

5.2.1 *Proof of Theorem 5.2*

Let V be any r'-dimensional subspace of $\mathbb{F}_2^n$, where $r' \geq r$ is the smallest positive integer divisible by $\sqrt{n}$. We first define the inverse of $\mathsf{vec}(\cdot)$. For every $v \in \mathbb{F}_2^n$, define $\mathsf{M}(v)$ to be the matrix obtained by splitting v into $\sqrt{n}$ length consecutive blocks and stacking each of these blocks to form a $\sqrt{n} \times \sqrt{n}$ matrix. Formally, $\mathsf{M}(v) \in \mathbb{F}_2^{\sqrt{n} \times \sqrt{n}}$ is such that $\mathsf{M}(v)[a, b] = v[(a-1)\sqrt{n} + j]$ for every $a, b \in [\sqrt{n}]$. Note that $\mathsf{vec}(\mathsf{M}(v)) = v$.

We provide a brief outline of the proof of Theorem 5.2. The first step of the proof is to produce a vector in v that is at a distance of $\Omega(n)$ from V and $\mathsf{M}(v)$ is low rank. The rank being low is helpful as we can express $\mathsf{M}(v)$ as the sum of a small number of rank one matrices. Lemma 5.1 will then imply the existence of a rank one matrix that is far away from V. If we only cared about the existence of a vector that is far away from V, Lemma 5.2 would suffice. To ensure that simultaneously the rank is small, we first project V on to $n/2r'$ coordinates indexed by consecutive blocks each of length $2r'$. Then we find a vector $v' \in \mathbb{F}_2^{2r'}$ that is far away from all the projections, which is still guaranteed by Lemma 5.2. Concatenating v' with itself $2r'$ times has the property that its corresponding matrix is low rank.

Let $k = \max\left\{\left\lfloor \frac{n}{2r'} \right\rfloor, 1\right\}$. The goal is to find a $v \in \mathbb{F}_2^n$ such that $d_H(v, V) \geq k \cdot r'/8$ and the rank of $\mathsf{M}(v)$ is at most $2r'/\sqrt{n}$. If $\left\lfloor \frac{n}{2r'} \right\rfloor \geq 1$, then define $S_1, \ldots, S_k$ such that

$$S_i = \{(i-1) \cdot 2r' + 1, \ldots, i \cdot 2r'\}$$

for $i \in [k]$; otherwise, define $S_1 = [n]$. By definition, the dimension of V_{S_i} is at most $r' = |S_i|/2$, for every $i \in [k]$. Since $r' \geq \sqrt{n}$ and $n \geq 1024$, we can infer that $k \leq 2r'$ and $2r' < 2^{r'/2}$. Lemma 5.2 implies the existence of a $v' \in \mathbb{F}_2^{2r'}$ with the property that $d_H(v', V_{S_i}) \geq r'/8$ for every $i \in [k]$. Now define $v \in \mathbb{F}_2^n$ by

$$v[i] = \begin{cases} v'[i \bmod 2r'] & \text{if } i \leq k \cdot 2r' \text{ and } i \bmod 2r' \neq 0, \\ v'[2r'] & \text{if } i \leq k \cdot 2r' \text{ and } i \bmod 2r' = 0, \\ 0 & \text{if } i > 2kr', \end{cases}$$

for all $i \in [n]$. In words, v is the length n vector that is the concatenation of k copies of v' along with the vector of zeros of length $n - 2kr'$. By the choice of v, we get that,

$$d_H(v, V) \geq \sum_{i=1}^{k} d_H(v, V_{S_i}) \geq k \cdot r'/8.$$

Moreover, the rank of $\mathsf{M}(v)$ is at most $\frac{2r'}{\sqrt{n}}$. Therefore we can express

$$\mathsf{M}(v) = \sum_{i=1}^{2r'/\sqrt{n}} a_i b_i^{\intercal},$$

for some $a_1, b_1, \ldots, a_{\frac{2r'}{\sqrt{n}}}, b_{\frac{2r'}{\sqrt{n}}} \in \mathbb{F}_2^{\sqrt{n}}$. By Lemma 5.1, we know that

$$d_H(v, V) \leq \sum_{i=1}^{2r'/\sqrt{n}} d_H(\mathsf{vec}(a_i b_i^{\intercal}), V).$$

Hence there exists an $i \in \left[\frac{2r'}{\sqrt{n}}\right]$ such that $d_H(\mathsf{vec}(a_i b_i^{\intercal}), V) \geq \frac{\sqrt{n} \cdot k}{16} \geq \frac{n^{3/2}}{64r'}$. The observation that $r' \leq 2\max\{\sqrt{n}, r\}$ completes the proof of the theorem.

Remark (Extension to strong rigidity)**.** Alon and Cohen [8] defined the notion of *strong* rigidity; a set $Q \subseteq \mathbb{F}_2^n$ is (r,t)-strongly rigid if for every subspace of $\mathbb{F}_2^n$ of dimension at most r, the average distance of all the points to the subspace is at least t. For strong rigidity, the best lower bounds known for explicit sets are also of the form given in Eq. (5.1). We can show that Υ is (r,t)-strongly rigid with $t \geq \Omega\left(\frac{n^{3/2}}{\max\{\sqrt{n}, r\}}\right)$, matching the best strong rigidity bounds known for explicit sets. We sketch the proof here. We know that

$$u^{\intercal} M v + (u + e_i)^{\intercal} M v + u^{\intercal} M (v + e_j) + (u + e_i)^{\intercal} M (v + e_j) = b_i^{\intercal} M b_j,$$

where $u, v \in \mathbb{F}_2^{\sqrt{n}}$ and $e_1, \ldots, e_{\sqrt{n}}$ are standard basis vectors in $\mathbb{F}_2^{\sqrt{n}}$. This fact can be used to prove that the matrix M_Υ corresponding to the set Υ is a generator matrix of a 4-query locally decodable code that tolerates a constant fraction of errors. A result of [36, Theorem 6] shows that Theorem 5.2 and the locally decodable code property of M_Υ imply the strong rigidity of Υ.

5.3 Linear Data Structures and Rigidity

In this section, we relate linear data structures and rigidity. As linear data structures are a special case of the cell probe model, we may obtain rigidity lower bounds from strong enough static data structure lower bounds (when the queries are linear). We also compare with Dvir, Golovnev, and Weinstein, who exhibit a similar connection [36]. We first provide some notation.

Definition 5.3. *Let $Q \subseteq \mathbb{F}_2^n$ be a set. Define $\mathsf{LT}(Q, s)$ to be the maximum over all $x \in \mathbb{F}_2^n$ of the minimum t sufficient to compute the inner product $\langle q, x\rangle$ for every $q \in Q$ when the query algorithm's output is a linear function of t bits chosen from the s precomputed linear functions of x.*

Table 5.1 provides a glimpse of our results on linear data structures along with a comparison to [36]. Recall that a set $Q \subseteq \mathbb{F}_2^n$ is explicit if each coordinate of an arbitrary element of the set can be computed in $\mathsf{poly}(n)$ time. The prior work shows that sufficiently strong lower bounds against linear data structures will imply *semi-explicit* rigid sets. A bit more formally, consider a data structure query set $Q \subseteq \mathbb{F}_2^n$ of size m for the inner product problem. They show the following: If $\mathsf{LT}(\mathsf{Q}, \mathsf{c} \cdot \mathsf{n}) \geq t$ for some constant c, then there is a $(n'/2, t/\log n)$-rigid set Q' of size at most m contained in $\mathbb{F}_2^{n'}$, where $n' \geq t$. However, the set Q' is only semi-explicit in that it is in P^{NP} – every element can be computed by a $\mathsf{poly}(m)$ time algorithm with access to an NP oracle.

We now summarize a few differences between our work and [36]. Our result proves that polynomial lower bounds on the query time imply the existence of an explicit rigid set, which is in contrast to semi-explicit sets obtained by [36]. On the other hand, explicitness comes with a cost; when $m = \mathsf{poly}(n)$, we need much stronger data structure lower bounds to produce explicit rigid sets. When $m \gg \mathsf{poly}(\mathsf{n})$, the algorithm of [36] takes $\mathsf{poly}(m)$ time with access to an NP oracle to compute an element of the semi-explicit rigid set. For problems such as the Vector-Matrix-Vector problem, this is super polynomial time. The rest of this section concerns proving the following theorem, which implies all of our results in Table 5.1.

Theorem 5.3. *Let $k = \mathsf{LT}(Q, 3n/2)$ and let $Q \subseteq \mathbb{F}_2^n$ of size m be an explicit query set. There exists a set $Q' \subseteq \mathbb{F}_2^k$ with size at most $m \cdot \lceil \frac{n}{k} \rceil$, whose elements can be computed in $\mathsf{poly}(n)$ time. Moreover, if $k \geq 2\sqrt{n}$, then Q' is explicit and $\left(\frac{k}{2}, \frac{k^2}{4n}\right)$-rigid.*

Note that for every $s \geq 3n/2$, we have that $\mathsf{LT}(Q, 3n/2) \geq \mathsf{LT}(Q, s)$. Hence, a sufficiently strong lower bound on $\mathsf{LT}(Q, s)$ for any $s \geq 3n/2$ will imply a rigidity lower bound. The following corollary shows the consequence of Theorem 5.3 for specific values of k.

Corollary 5.1. *Let $k = \mathsf{LT}(Q, 3n/2)$ and let $Q \subseteq \mathbb{F}_2^n$ of size m be an explicit query set. There exists a set $Q' \subseteq \mathbb{F}_2^k$ with size at most $m \cdot \lceil \frac{n}{k} \rceil$, whose elements can be computed in $\mathsf{poly}(n)$ time. Moreover,*

(a) If $k = \omega\left(\sqrt{n \log m}\right)$, then Q' is explicit and $(k/2, \omega(\log m))$-rigid.

(b) If $k = \Omega\left(n^{(1+\delta)/2}\right)$ for some $\delta > 0$, then Q' is explicit and $\left(k/2, \Omega\left(n^{\delta}\right)\right)$-rigid.

Corollary 5.1(a) explains the first and last rows in Table 5.1, and Corollary 5.1(b) explains the middle row. Using Corollary 5.1(a) applied to Υ with $m = 2^{2\sqrt{n}} - 2^{\sqrt{n}+1} + 1$, we obtain that a lower bound of $\mathsf{LT}(\Upsilon, 3n/2) \geq \omega(n^{3/4})$ would imply the existence of an explicit set $Q' \subseteq \mathbb{F}_2^k$ of size $2^{O(\sqrt{n})}$ that is $(k/2, \omega(\sqrt{n}))$-rigid. We note that it is an open question to prove $\mathsf{LT}(\Upsilon, 3n/2) \geq \omega(\sqrt{n})$.

5.3.1 Proof of Theorem 5.3

We already know the equivalence between systematic linear data structures and rigidity (from Theorem 5.1). Therefore, it is sufficient to design a linear data structure from a systematic linear data structure to relate the former with rigidity.

Proposition 5.1. *Let $Q \subseteq \mathbb{F}_2^n$ be a query set. If $T(Q, r) \leq t$, then $\mathsf{LT}(Q, n + r) \leq t + r$.*

Proof. Let $x \in \mathbb{F}_2^n$ be the input data, and let $\langle a_1, x\rangle, \ldots, \langle a_r, x\rangle$ be the r redundant bits stored by the systematic linear data structure. We now describe a linear data structure for

m vs n	$k = \mathsf{LT}(Q, 3n/2)$	Rigidity Bounds	Explicitness	Reference
$m = n^c$	$k = \omega\left(\sqrt{n \log n}\right)$	$(k/2, \omega(\log n))$-rigid	$\mathsf{poly}(n)$ time	This work
	$k = \omega\left(\log^2 n\right)$	$(k/2, \omega(\log n))$-rigid	$\mathsf{poly}(n)$ time + NP oracle calls	[36]
$m = n^c$	$k = \Omega\left(n^{(1+\delta)/2}\right)$	$\left(k/2, \Omega\left(n^{\delta}\right)\right)$-rigid	$\mathsf{poly}(n)$ time	This work
	$k = \Omega\left(n^{\delta} \log n\right)$	$\left(k/2, \Omega\left(n^{\delta}\right)\right)$-rigid	$\mathsf{poly}(n)$ time + NP oracle calls	[36]
$m = 2^{c\sqrt{n}}$	$k = \omega\left(n^{3/4}\right)$	$(k/2, \omega(\sqrt{n}))$-rigid	$\mathsf{poly}(n)$ time	This work
	$k = \omega(\sqrt{n} \cdot \log n)$	$(k/2, \omega(\sqrt{n}))$-rigid	$\mathsf{poly}\left(2^{\sqrt{n}}\right)$ time + NP oracle calls	[36]

Table 5.1: Comparison with [36, Theorem 7.1]: Let $Q \subseteq \mathbb{F}_2^n$ of size m be a query set, $c \geq 1$ and $\delta > 0$ be constants, and let $k = \mathsf{LT}(Q, 3n/2)$. The second column states the lower bound on $\mathsf{LT}(Q, 3n/2)$ that implies existence of rigid sets whose parameters are given in the third column. All rigid sets have size at most $\mathsf{poly}(m)$ and are contained in $\mathbb{F}_2^k$.

Q with space $n+r$ and query time $t+r$. $\langle a_1, x\rangle, \ldots, \langle a_r, x\rangle, \langle e_1, x\rangle, \ldots, \langle e_n, x\rangle$ are the stored bits, where $e_1, \ldots, e_n$ denote the standard basis vectors. The query algorithm on $q \in Q$ first accesses $\langle a_1, x\rangle, \ldots, \langle a_r, x\rangle$ and then simulates the query algorithm of the systematic linear data structure on q. Since the systematic linear data structure accesses at most t bits from $\langle e_1, x\rangle, \ldots, \langle e_n, x\rangle$, we can conclude that the query time is at most $t + r$. □

We prove that if a set contained in a n-dimensional space is (r, t)-rigid, then there is another $(r, tr/n)$-rigid set which is contained in a $2r$-dimensional space.

Lemma 5.3. *Let r, n be positive integers. If $S \subseteq \mathbb{F}_2^n$ is (r, t)-rigid of size m, then there is a set $S' \subseteq \mathbb{F}_2^{2r}$ of size at most $m \cdot \lceil \frac{n}{2r} \rceil$ that is $(r, tr/n)$-rigid. Moreover, if S is explicit, then each element of S' can be computed in $\mathsf{poly}(n)$ time.*

Proof. Let $k = \lfloor \frac{n}{2r} \rfloor$ and define $S_1, \ldots, S_k \subseteq \mathbb{F}_2^{2r}$ by

$$S_i = \{(s[2r \cdot (i-1) + 1], \ldots, s[2r \cdot i]) \mid s \in S\}$$

for each $i \in \{1, 2, \ldots, k\}$. Additionally, if $n/2r$ is not an integer, then define

$$S_{k+1} = \{(s[2r \cdot k + 1], \ldots, s[n], 0, \ldots, 0) \mid s \in S\} \subseteq \mathbb{F}_2^{2r};$$

otherwise set $S_{k+1} = \emptyset$. Define $S' = \bigcup_{i=1}^{k+1} S_i$. We claim that S' is $(r, tr/n)$-rigid. Indeed, for the sake of contradiction assume that there is a subspace V in $\mathbb{F}_2^{2r}$ of dimension r such that all points in S' are at a distance less than tr/n from V. Consider the subspace $\{(v, v, \ldots, v) \mid v \in V\} \subseteq \mathbb{F}_2^{2r \cdot (k+1)}$ and project it to the first n coordinates. Call this subspace V', which has dimension r. Now, the distance of each point in S from V' is less than $\frac{tr}{n} \cdot \lceil \frac{n}{2r} \rceil < t$, which is a contradiction.

Regarding the explicitness of S', it is clear that all coordinates of an element of S' correspond to some coordinate of a specific element of S. Since S is explicit, we can infer that each element of S' can be computed in $\mathsf{poly}(n)$. □

Proof of Theorem 5.3. Since $\mathsf{LT}(Q, 3n/2) = k$ and $k \leq n$, Proposition 5.1 implies that $T(Q, k/2) \geq k/2$. Therefore by Theorem 5.1, we can conclude that Q is $(k/2, k/2)$-rigid.

Lemma 5.3 implies that there exists a set Q' that is $\left(\frac{k}{2}, \frac{k^2}{4n}\right)$-rigid and the size of Q' is at most $m \cdot \lceil \frac{n}{k} \rceil$. Moreover, every element of Q' can be computed in time $\mathsf{poly}(n)$. Since $k/2 \geq \sqrt{n}$, we can conclude that Q' is explicit. □

5.4 *An Elementary Proof of the Main Result of [36]*

In this section, we present an elementary proof of the theorem of Dvir, Golovnev and Weinstein [36] that shows how lower bounds on linear data structures imply *semi-explicit* rigid matrices. We remark that Dvir, Golovnev and Weinstein prove a more general theorem by showing an algorithmic relationship between the inner dimension and the outer dimension of sets, notions due to Paturi and Pudlák [85]. Here, we take a more straightforward approach. For any $A \subseteq \mathbb{F}_2^n$, let $\mathsf{dim}(A)$ denote the dimension of the subspace spanned by A.

Theorem 5.4 ([36])**.** *Let $Q \subseteq \mathbb{F}_2^n$. If $T(Q, 2n) \geq t$, then there is a $Q' \subseteq \mathbb{F}_2^n$ such that $\mathsf{dim}(Q) = n'$ and Q' is $\left(\frac{n'}{2}, \frac{t}{\log n}\right)$-rigid. Moreover, Q' can be computed by an algorithm with runtime $\mathsf{poly}(|Q|, n)$ and access to an NP oracle.*

We need the following key claim.

Lemma 5.4. *If Q is not $\left(\frac{n}{2}, \frac{t}{\log n}\right)$-rigid, then there exists a $Q' \subseteq \mathbb{F}_2^n$ such that*

1. *$\mathsf{dim}(Q') \leq n/2$.*
2. *$T(Q, 2n) < T(Q', n) + t/\log n$.*
3. *Each point in Q' can be computed by an algorithm with runtime $\mathsf{poly}(n)$ and access to an NP oracle.*

Proof. Since Q is not $(n/2, t/\log n)$-rigid, there exists an $n/2$-dimensional subspace such that for every $q \in Q$, $d_H(q, V) < t/\log n$. In other words, every $q \in Q$ can be expressed as $q = v_q + u_q$, where $v_q \in V$ and the Hamming weight of u_q is less than $t/\log n$.

Define $Q' = \{v_q \mid q \in Q\} \subseteq V$, and hence $\mathsf{dim}(Q') \leq n/2$. Consider a data structure for Q' on input data $x \in \mathbb{F}_2^n$, space n and query time $T(Q', n)$. Let $e_1, \ldots, e_n$ denote the

standard basis for $\mathbb{F}_2^n$. Now consider a new data structure for Q on input data x that stores $\langle x, e_1 \rangle, \ldots, \langle x, e_n \rangle$ along with the bits stored by the data structure for Q'. For $q \in Q$, this data structure's computation is done using the identity $\langle q, x \rangle = \langle v_q, x \rangle + \langle u_q, x \rangle$. $\langle v_q, x \rangle$ can be computed in time $T(Q', n)$, and $\langle u_q, x \rangle$ requires less than $t/\log n$ accesses to $\langle x, e_1 \rangle, \ldots, \langle x, e_n \rangle$, which follows from the upper bound on the Hamming weight of u_q. Therefore, we showed the existence of a data structure for Q with space $2n$ and query time less than $T(Q', n) + t/\log n$. This completes the proof for the first two items of the lemma.

We now proceed to prove the third item of the lemma. To compute Q', we first compute V, and then v_q for every $q \in Q$. V can be computed by incrementally finding its basis, where each element in the basis is found by making NP oracle calls. Concretely, to find the first basis vector, we make an oracle call to decide whether the first bit of the vector is a 0 or a 1. Similarly, other coordinates are computed. The same strategy extends to computing all the basis vectors of V. To compute v_q for $q \in Q$, a similar procedure can be used to identify each coordinate of v_q by making NP oracle calls. Once we know v_q, u_q is computed using the identity $u_q = q + v_q$ in linear time. This completes the proof of the lemma. □

Proof of Theorem 5.4. The above lemma now easily implies the theorem. For the sake of contradiction assume that no such Q' exists. Repeated application of Lemma 5.4 will imply that $T(Q, 2n) < t$, which is a contradiction. Q' being computed by an algorithm with $\mathsf{poly}(|Q|, n)$ runtime and access to an NP oracle follows from the third item of Lemma 5.4 and the fact that an NP oracle call is sufficient to decide whether a set of points in $\mathbb{F}_2^n$ is rigid. □

Chapter 6

NON-ADAPTIVE DATA STRUCTURES FOR MAINTAINING SETS OF NUMBERS

As discussed in the introduction, maintaining a set of numbers to be able to compute the minimum, median and predecessors, while supporting the insertion of new numbers is a basic data structure problem. The goal of this chapter is to show how to use the sunflower lemma of Erdős and Rado [40] from combinatorics to prove lower bounds against data structures for these tasks, in which some operations are non-adaptive. Before formally stating our results, we discuss some related work.

Lower bounds on data structures for computing single statistics like the median or minimum have been particularly elusive. Computing statistics like the median and the minimum are very fundamental in algorithm design. The best known upper bounds require $O(\log \log n)$ time for insertions, and median and minimum computations. It is surprising that no previous lower bounds were known in the cell-probe model. We prove the first lower bounds on the performance of data structures computing the median and minimum. Brodal, Chaudhuri and Radhakrishnan [26] showed that if the data structure is only allowed to compare the contents of cells and perform no other computation with the cells, then we must have $t_{\mathsf{min}} \geq \Omega\left(n/4^{t_{\mathsf{ins}}}\right)$, where t_{min} is the number of comparisons used to compute the minimum, and t_{ins} is the number of comparisons used to insert numbers into the set. Moreover, [26] gave a data structure matching these bounds. The same bounds apply for computing the median as well. It remains an interesting open problem to prove a lower bound of $t_{\mathsf{ins}} + t_{\mathsf{med}} \geq \Omega(\log \log n)$ in the cell-probe model when $w = O(\log n)$, where t_{ins} is the time for insertions and t_{med} is the time to compute the median. We note here that there is a long sequence of works proving lower bounds on computing the median in the context of branching programs [31, 73, 20, 28].

Past work had found more success with understanding the complexity of the predecessor search problem. A long sequence of works has proved lower bounds here [2, 70, 72, 19, 93, 82]. In particular, [19, 93] showed that some operation must take time $\Omega(\log\log n/\log\log\log n)$, when $w = \log n$, and this was improved to $\Omega(\log\log n)$ by [82]. Still, it remains open to understand the full trade-off between the time complexity of inserting elements and the time complexity of computing predecessors[1].

Several past works have proved lower bounds on various computational models under the assumption of non-adaptivity (see for example [60]). In the context of data structures, Brody and Larsen [27] showed polynomial lower bounds for various dynamic problems in the non-adaptive setting. Among other results, they showed that any data structure for reachability in directed graphs that non-adaptively checks for reachability between pairs of vertices must take time $\Omega(n/w)$, where n is the size of the underlying graph. [9, 50] proved non-adaptive lower bounds on static data structures for the dictionary problem in the bit probe model.

We prove new lower bounds against non-adaptive data structures maintaining a subset of $\{1, 2, \ldots, n\}$ and supporting insertions, deletions, minimum computation, median computation and predecessor computation. Our results are obtained via an application a variant of the famous sunflower lemma of Erdős and Rado [40]. The sunflower lemma was used in the past to prove lower bounds on dynamic data structures by Frandsen and Milterson [41] and then again for static data structures by Gal and Milterson [47], and our use of it is similar. However, in the setting of non-adaptive data structures, we can leverage the lemma to get results even when the word size is large.

All results in this chapter are joint work with Rao [75]. Our first result proves a lower bound when both deletions and minimum computations are non-adaptive[2]. Similar results hold for computing the median and predecessors as well, but they are subsumed by the

[1]We thank Mikkel Thorup for bringing this question to our attention.

[2]The analogous result for computing the maximum also holds. Its proof is nearly identical to the proof for theorem about the minimum.

theorems to follow.

Theorem 6.1. *Any data structure that computes the minimum of a subset of* $\{1, 2, \ldots, n\}$ *while supporting non-adaptive delete operations and non-adaptive minimum computations must take time* $\Omega\left(\frac{\log n}{\log\log n + \log w}\right)$ *for some operation, where* w *is the word size of the cells.*

Our second result concerns non-adaptive data structures for computing the median. Here the lower bound holds even if the median computation is adaptive and the insertion operation is non-adaptive:

Theorem 6.2. *Any data structure that computes the median of a subset of* $\{1, 2, \ldots, n\}$ *while supporting non-adaptive insert operations must satisfy*

$$t_{\mathsf{med}} \geq \Omega\left(\frac{n^{\frac{1}{t_{\mathsf{ins}}+1}}}{w^2 \cdot t_{\mathsf{ins}}^2}\right),$$

where t_{med} *is the time required to compute the median,* t_{ins} *is the time required to insert elements, and* w *is the word size of the cells.*

Our last result concerns the predecessor search problem. Here the lower bound holds even if the insertion operation is adaptive, as long as the predecessor computations are non-adaptive:

Theorem 6.3. *Any data structure that maintains a subset of* $\{1, 2, \ldots, n\}$ *while supporting non-adaptive predecessor operations must satisfy*

$$t_{\mathsf{pred}} \geq \Omega\left(\frac{\log n}{\log\log n + \log w}\right) \qquad \text{or} \qquad t_{\mathsf{ins}} \geq \Omega\left(n^{\frac{1}{2(t_{\mathsf{pred}}+1)}}\right),$$

where t_{ins} *is the time required for inserts,* t_{pred} *is the time required for computing predecessors and* w *is the word-size of the cells.*

Our theorems are complemented by the observation that a variant of binary search trees gives a data structure that can insert and delete elements non-adaptively, compute predecessors non-adaptively, and perform all operations in time $O(\log n)$, with $w = \log n$. Theorem

6.2 and Theorem 6.3 show that there is a gap between adaptive and non-adaptive data structures computing the median and predecessors, since we know that the van Emde Boas data structure can compute both in time $O(\log \log n)$ with $w = \log n$. Very recently, Boninger, Brody and Kephart [21] independently obtained some lower bounds on non-adaptive data structures computing predecessors. Among other results, they showed that any data structure with non-adaptive insertions and non-adaptive predecessor computations must have[3] $t_{\mathsf{ins}} \geq \Omega(\log n)$, or $t_{\mathsf{pred}} \geq \frac{\log n}{\log w + \log t_{\mathsf{ins}}}$. Our bounds do not require non-adaptivity for the insertion operations and are quantitatively better when $t_{\mathsf{pred}} = o(\log n / \log \log n)$. We also note that the cell sampling technique ([77, 66]) does not give any meaningful lower bounds for these problems.

The binary search tree based data structure is described in Section 6.1. We then discuss our variant of sunflowers, which we call *flowers*, and present the proof of the flower lemma in Section 6.2. In Section 6.3, we demonstrate the application of the flower lemma by assuming all the operations to be non-adaptive. Specifically, Theorem 6.1 is proved in the same section. Theorems 6.2 and 6.3 are proved in Sections 6.4 and 6.5, respectively.

6.1 A Data Structure based on Binary Search Trees

Before showing the lower bounds, we describe a data structure based on binary search tree that maintains a subset of $\{1, \ldots, n\}$ allowing non-adaptive inserts, non-adaptive predecessor computations and adaptive median computations. The data structure builds on the well known binary search tree on $\{1, \ldots, n\}$ and is very close to the *x-fast trie* (see [103]). This data structure matches many of the lower bounds in our proofs.

Theorem 6.4. *There is a data structure that maintains a subset of* $\{1, 2, \ldots, n\}$ *and supports insertions, deletions and computing the median, minimum, and predecessors. All operations take time* $O(\log n)$*, the word size is* $\log n$*, and all operations except for the median operation*

[3] [21] consider the tradeoff with the size of the set being added, which allows them to prove lower bounds even when the data structure is only required to maintain small sets. The bound stated here is what they obtain when the size of the set is allowed to be arbitrary.

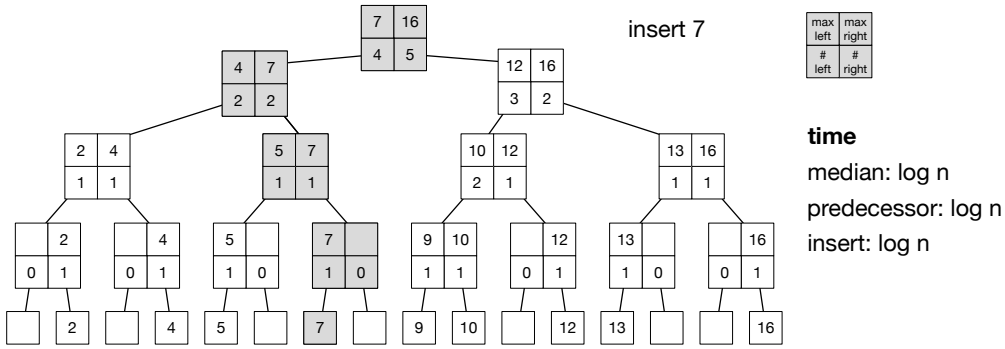

Figure 6.1: A data structure based on binary search trees for $\{2, 4, 5, 7, 9, 10, 12, 13, 16\}$.

are non-adaptive.

Proof. Without loss of generality, we may assume that n is a power of 2. We maintain a balanced binary tree of height $\log n$. Every leaf is assigned an element from the universe.

There is a memory cell associated with every leaf and four memory cells associated with every internal node of the tree. The cells corresponding to each internal node store the number of elements in the left subtree rooted at that node, the number of elements stored in the right subtree, the maximum element of the left subtree and the maximum element of the right subtree. Figure 6.1 shows an example of the data structure.

To insert an element into the set, we only need to access the cells associated with each node on the path from the root to the corresponding leaf. These are the only cells that need to be modified to make the data structure consistent with the new set. Deletions can be performed in the same way. The time required for these operations is $O(\log n)$, and they are non-adaptive.

To compute the median or minimum, we read the cells associated with the root to determine if the desired value belongs to the left or the right sub tree. Accordingly, we read the cells associated with either the left or the right child and recurse to find the median or minimum. The time required for this operation is $O(\log n)$, but it is adaptive.

To compute the predecessor of an element, we only need to access the cells associated with

every node on the path from the root to the corresponding leaf in the tree. The predecessor is the maximum of last non-empty left-subtree seen on this path. Again, we see that this operation takes $O(\log n)$ time and is non-adaptive. □

6.2 Flowers

The theorems in this chapter are proved using a variant of the sunflower lemma of Erdős and Rado [40] from combinatorics, which we call the *flower lemma*. Using the sunflower lemma would give us the same asymptotic bounds, but the flower lemma gives cleaner bounds. The lemma we need is almost identical to a lemma proved by Alon and Boppana [7], and we use their ideas to prove it.

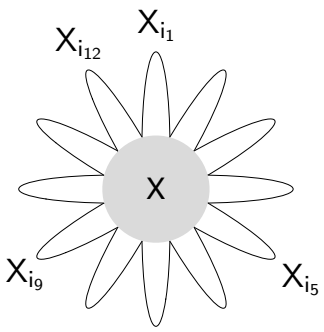

Figure 6.2: A flower with 12 petals. X denotes the core of the Flower.

Definition 6.1. *A sequence of sets $X_1, \cdots, X_p$ is called a* t-flower *with p petals if each set in the sequence is of size t, and there is a set X of size at most t such that for every i, j, $X_i \cap X_j \subseteq X$. X is called the* core *of the flower.*

See Figure 6.2 for an illustration of a flower. Next, following [7], we show that a long enough sequence of sets must contain a flower.

Lemma 6.1 (Flower Lemma)**.** *Let $X_1, \cdots, X_n$ be a sequence of sets each of size t. If $n > (p-1)^{t+1}$, then there is a subsequence that is a t-flower with p petals.*

Proof. We prove the bound by induction on t, p. When $t = 1$, if $n > (p-1)^2$, either there are p sets that are the same or p sets that are distinct. Either way, we obtain a 1-flower with p petals. When $p = 1$ the statement is trivially true.

Suppose that $t \geq 2$, and the sequence does not contain a t-flower with p petals. For each set $X \subseteq X_1$, we get a subsequence by restricting our attention to the sets X_i such that $X_i \cap X_1 = X$ and $i > 1$. By induction, the length of this subsequence can be at most $(p-2)^{t+1-|X|}$ since all of these sets have X in common, and any $(t-|X|)$-flower with $p-1$ petals yields a t-flower with p petals in our original sequence, by adding X_1 to the list of petals. Thus we get,

$$\begin{aligned} n &\leq 1 + \sum_{X \subseteq X_1} (p-2)^{t+1-|X|} \\ &= 1 + (p-2) \cdot \sum_{X \subseteq X_1} (p-2)^{t-|X|} \\ &\leq 1 + (p-2) \cdot (p-2+1)^t \leq (p-1)^{t+1}, \end{aligned}$$

as desired. □

6.3 Lower Bounds When All Operations are Non-Adaptive

As a warm up, we prove some loose lower bounds when all operations in the data structure are non-adaptive. In the next section, we prove our final theorems where we only assume that some of the operations are non-adaptive.

We start by proving Theorem 6.1, which gives a lower bound on the time for any data structure that computes minimum and deletions non-adaptively.

Proof of Theorem 6.1. Consider the sequence of sets $\mathcal{X} = X_1, \cdots, X_n$ where

$$X_i = \{j \mid \text{cell } j \text{ is accessed while deleting } i, \text{ or when computing the minimum}\}.$$

If t is the time required for the operations of the data structure, then each set X_i is of size at most $2t$. Without loss of generality, we can assume that each X_i is of size *exactly* $2t$. The key observation is that there cannot be a large $2t$-flower in $\mathcal{X}$:

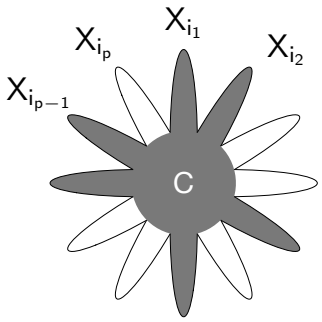

Figure 6.3: C denotes the core of the Flower, and the shaded cells are the only cells accessed when deleting $\{i_1, i_2, \cdots, i_p\} \setminus S$.

Claim 6.1. *If $\mathcal{X}$ has a $2t$-flower with p petals, then $p \leq 2wt$.*

Proof. Suppose for the sake of contradiction that the sequence $X_{i_1}, \cdots, X_{i_p}$ is a $2t$-flower with $i_1 < i_2 < \cdots < i_p$, and $p = 2wt + 1$. Then let S be any subset of $\{i_1, i_2, \cdots, i_p\}$ and C denote the contents of the core of the $2t$-flower after inserting the set $\{i_1, \ldots, i_p\}$ and then deleting the elements of $\{i_1, i_2, \ldots, i_p\} \setminus S$.

We show that C serves as an encoding of S. This is because C is all we need to reconstruct the execution of the following sequence of deletion and minimum operations: compute the minimum, delete the minimum, compute the minimum, delete the minimum, and so on. The answers to these computations determine the elements in S. The answer to the first minimum computation can be reconstructed from C, since C contains all cells used in this computation. If we attempt to delete i_j, then the only cells of X_{i_j} that were modified by a previous deletion operation are contained in C. Thus, every such deletion operation can be simulated with access to C (see Figure 6.3).

C can be described using at most $2t \cdot w$ bits, yet C encodes an arbitrary subset of p elements. This proves the claim. □

By the flower lemma (Lemma 6.1), the sequence $\mathcal{X}$ has a $2t$-flower with $n^{\frac{1}{2t+1}}$ petals. So,

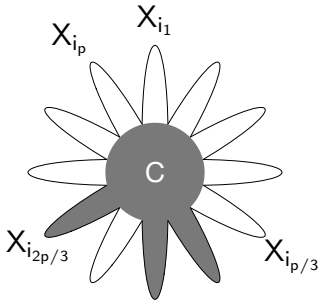

Figure 6.4: C denotes the core of the Flower, and the shaded cells are the only cells accessed when inserting S.

we get

$$t \geq \frac{p}{2w} \geq \frac{n^{\frac{1}{2t+1}}}{2w},$$

where the last inequality follows from the choice of p. After rearranging, we get

$$t \cdot \log wt \geq \Omega\left(\log n\right).$$

Proposition 2.3 implies the desired bound on t. □

Next we prove a similar result for computing the median.

Theorem 6.5. *Any data structure with non-adaptive insertions and median computations must take time* $\Omega\left(\frac{\log n}{\log\log n + \log w}\right)$ *for some operation.*

Proof. Consider the sequence of sets $\mathcal{X} = X_1, \cdots, X_n$ where

$$X_i = \{j \mid \text{cell } j \text{ is accessed while inserting } i, \text{ or when computing the median}\}.$$

If t is the time required for the operations of the data structure, then each set X_i is of size at most $2t$. Without loss of generality, we can assume that each X_i is of size *exactly* $2t$. The key observation is that there cannot be a large $2t$-flower in $\mathcal{X}$:

Claim 6.2. *If* $\mathcal{X}$ *has a* $2t$*-flower with* p *petals, then* $p \leq 6wt + 2$.

Proof. Suppose for the sake of contradiction that the sequence $X_{i_1}, \cdots, X_{i_p}$ is a $2t$-flower with $i_1 < i_2 < \cdots < i_p$, and $p = 6wt+3$. Then let S be any subset of $\left\{i_{p/3+1}, i_{p/3+2}, \cdots, i_{2p/3}\right\}$ and C denote the contents of the core of the $2t$-flower after inserting elements of S into the data structure (see Figure 6.4).

We show that C serves as an encoding of S. This is because C is all we need to reconstruct the execution of the following sequence of insert and median operations: insert i_1, compute the median, insert i_2, compute the median,$\cdots$, insert $i_{p/3}$, compute the median. These operations determine the elements in S between its smallest element and median. By the definition of the flower, the only cells of $X_{i_1}, \ldots, X_{i_{p/3}}$ that were accessed when S was inserted are contained in C. Therefore, the sequence of operations can be simulated using C (see Figure 6.4). Similarly, executing the following operations helps retrieve elements in S between its median and largest element: insert $i_{2p/3+1}$, compute the median, insert $i_{2p/3+2}$, compute the median,$\cdots$, insert i_p, compute the median.

C can be described using at most $2t \cdot w$ bits, yet C encodes a subset of $p/3 = (2tw + 1)$ elements. This proves the claim. □

By the Flower-Lemma (Lemma 6.1), the sequence $\mathcal{X}$ has a $2t$-flower with $n^{\frac{1}{2t+1}}$ petals. Then we get

$$t \geq \frac{p-2}{6w} \geq \frac{n^{\frac{1}{2t+1}} - 2}{6w},$$

where the last inequality follows from the choice of p. After rearranging, we get

$$t \cdot \log wt \geq \Omega\left(\log n\right).$$

Proposition 2.3 implies the desired bound on t. □

Next we prove a lower bound for the predecessor search problem.

Theorem 6.6. *Any data structure for the predecessor problem with non-adaptive insert operations and non-adaptive predecessor operations must have time* $\Omega\left(\frac{\log n}{\log\log n + \log w}\right)$.

Proof. Let $\mathcal{X} = X_1, \cdots, X_n$, where

$$X_i = \{j \mid \text{cell } j \text{ is accessed while inserting } i \text{ or computing the predecessor of } i\}.$$

It t is the time required for the operations of the data structure, then each set X_i is of size at most $2t$. Without loss of generality, we can assume that each X_i is of size *exactly* $2t$. We first show that the time complexity can be lower bounded in terms of the number of petals in a $2t$-flower belonging to $\mathcal{X}$.

Claim 6.3. *If $\mathcal{X}$ has a $2t$-flower with p petals, then $p \leq 4tw + 1$.*

Proof. Supposed for the sake of contradiction that the sequence $X_{i_1}, \cdots, X_{i_p}$ is a $2t$-flower and $i_1 < i_2 < \cdots < i_p$, and $p = 4tw + 2$. Let S be any subset of $\{i_1, i_3, \cdots, i_{p-1}\}$ and C denote the contents of the cells in the core after inserting the elements of S.

We show that C serves as an encoding of S. To reconstruct S, it suffices to compute the predecessors of the following elements: $i_2, i_4, \cdots, i_p$. By the definition of the $2t$-flower, the only cells accessed in $X_{i_2}, X_{i_4}, \ldots, X_{i_p}$ during the insertion operations are contained in the core of the $2t$-flower. Therefore, the sequence of predecessor operations can be simulated by access only to the cells in the core.

Hence C encodes S. Since there are 2^{2tw+1} possible sets S, and C can be described using $2tw$ bits, we must have $2tw \geq p/2$. This proves the claim. □

By the flower lemma 6.1, the sequence $\mathcal{X}$ has a $2t$-flower with $n^{\frac{1}{2t+1}}$ petals. So $t \geq \frac{p-1}{4w} \geq \frac{n^{\frac{1}{2t+1}}-1}{4w}$, which follows from the choice of p. After rearranging, we get

$$t \cdot \log wt \geq \Omega(\log n).$$

Proposition 2.3 implies the desired bound on t. □

6.4 Lower Bounds for Median when Insertions are Non-Adaptive

In this section, we prove Theorem 6.2. We start by giving an outline of the proof. As before, we first associate every element in $\{1, 2, \cdots, n\}$ with the set of cells that are accessed while

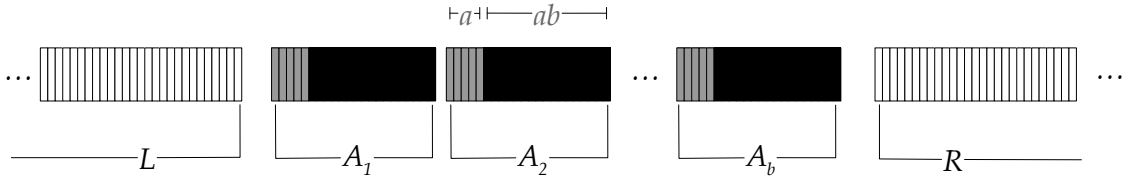

Figure 6.5: The elements corresponding to petals are partitioned into disjoint intervals $L, A_1, \ldots, A_q, R$. T is the set of black elements. S_i is a random subset of the i-th gray elements from each interval A_j.

inserting the element. We then identify a flower among these sets. Proving a lower bound on the time to compute the median is challenging as the computation is adaptive. We shall have to use the flower found above in a subtle way. We come up with a carefully chosen sequence of insertions, followed by a median computation that recovers the k-th smallest element of the set. The sequence of insertions are performed in batches, and every cell that is not in the core of the flower is associated with the batch number of the insertion operation that last accessed it. Ignoring the cells that belong to the core of the flower, we show that at least one cell associated with every batch is accessed with constant probability. Since these cells are disjoint, this will prove that the time to compute the median is at least a constant fraction of the number of batches. To make the above argument work, we use Shannon entropy to quantify the amount of information that the median computation must recover from the cells associated with each batch of insertions.

We now proceed with the formal proof. Define the sequence of sets $X = X_1, \ldots, X_n$, where

$$X_i = \{j \mid \text{cell } j \text{ is accessed while inserting } i\}\,.$$

By the flower lemma (Lemma 6.1), this sequence of sets must contain a t_{ins}-flower with $p = n^{1/(t_{\mathsf{ins}}+1)}$ petals, and without loss of generality, we assume that the petals are $X_1, \ldots, X_p$. Let C denote the core of the t_{ins}-flower.

To carry out the proof, we need to carefully define a sequence of operations[4] that insert a subset of the elements $\{1, 2, \ldots, p\}$. For parameters a, b, let $L, A_1, \ldots, A_b, R \subseteq \{1, 2, \ldots, p\}$ be consecutive disjoint intervals in ascending order, such that L is of size $p/3$, R is of size $p/3$ and for each i, A_i is of size $a + ab$, and $b(a + ab) \leq p/3$. See Figure 6.5. Let $S_1, \ldots, S_a$ be independently sampled sets, such that S_i is a uniformly random subset of $\{j : j \text{ is the } i\text{-th element of } A_r \text{ for some } r\}$. So each S_i is a subset of the gray elements in Figure 6.5. Finally, let T be the set

$$T = \{j : \text{for some } i \in [b],\ j \in A_i \text{ and } j \text{ is not one of the first } a \text{ elements of } A_i\},$$

so T is the set of black elements in Figure 6.5. Let k be a uniformly random element of $\{a, a + (a + ab), a + 2(a + ab), \ldots, a + (b - 1)(a + ab)\}$.

Consider the following sequence of operations with the data structure:

1. Phase 1:
   (a) Insert the elements of T.
   (b) Insert the elements of S_1, then the elements of S_2, and so on, until S_a has been inserted.
2. Phase 2:
   (a) Insert an appropriate number of elements into L or R so that the median of all the elements inserted is the k-th smallest element of $T \cup S_1 \cup S_2 \ldots \cup S_a$.
   (b) Compute the median of the inserted set.

We shall prove that the expected number of cells accessed to compute the median must be close to a. In order to prove this, we use ideas inspired by the chronogram approach

[4]This sequence of operations is inspired by an argument in [84]

of Fredman and Saks [43]. Consider the cells accessed during Phase 1. We say that a cell *belongs to* S_i if it is in the set

$$\bigcup_{j:j \text{ is the } i\text{-th element of } A_r \text{ for some } r} X_j \setminus C$$

So, every cell of the data structure can belong to at most one of the sets $S_1, \ldots, S_a$. Moreover, every cell that is accessed when inserting S_i either belongs to S_i or is in the core of the t_{ins}-flower.

Define

$$E_i = \begin{cases} 1 & \text{if a cell that belongs to } S_i \text{ is accessed in Phase 2,} \\ 0 & \text{otherwise.} \end{cases}$$

Observe that the insertions in Phase 2(a) never access a cell that belongs to S_i for any i. Since $E_i = 1$ whenever a cell that belongs to S_i is accessed, all such accesses must come from the median computation in Phase 2. Thus, $t_{\mathsf{med}} \geq \sum_{i=1}^{a} \mathbb{E}[E_i]$. We now proceed to lower bound $\sum_{i=1}^{a} \mathbb{E}[E_i]$.

Let C_i denote the contents of the core immediately after S_i was inserted. Let S_i^j denote the set $S_i \cap A_j$ and $S_i^{<j}$ denote the set $S_i^1 \cup S_i^2 \cup \ldots \cup S_i^{j-1}$. Recall that S_{-i} denotes $S_1, \ldots, S_{i-1}, S_{i+1}, \ldots, S_a$.

Claim 6.4. *The variables S_{-i}, C_i determine the contents, after Phase 1, of all cells that do not belong to S_i.*

Proof. If a cell does not belong to S_i, then there are three possibilities. If it belongs to a set $S_{i'}$ for $i' < i$, then its value can be reconstructed from $S_1, \ldots, S_{i'}$. If it belongs to $S_{i'}$ for $i' > i$, its value can be reconstructed from C_i and $S_{i+1}, \ldots, S_a$. If it does not belong to any set, then if it is in the core, it is determined by C_i and $S_{i+1}, \ldots, S_a$, and if it is not in the core, its value is fixed. □

Let $k = a + (j-1)(a+ab)$, so j is a uniformly random number from the set $\{1, 2, \ldots, b\}$.

Claim 6.5. *The k-th smallest element of $T \cup S_1 \cup S_2 \ldots \cup S_a$ computed in Phase 2 and S_{-i} together determine $\sum_{\ell=1}^{j} \left|S_i^\ell\right|$.*

Proof. The k-th smallest element of $T \cup S_1 \cup S_2 \ldots \cup S_a$ is e if and only if the number of elements in $A_1 \cup A_2 \ldots \cup A_b$ that are less than e and missing in $T \cup S_1 \cup S_2 \ldots \cup S_a$ is exactly $e - k$. In other words, the k-th smallest element of $T \cup S_1 \cup S_2 \ldots \cup S_a$ is e if and only if

$$\left|\{j : j < e, j \in (A_1 \cup A_2 \ldots \cup A_b) \setminus (T \cup S_1 \cup S_2 \ldots \cup S_a)\}\right| = e - k.$$

Let α be the number of elements missing from the intervals $A_1, A_2, \cdots, A_b$, and e be the k-th smallest element of $T \cup S_1 \cup S_2 \ldots \cup S_a$. We know that $0 \leq \alpha \leq ab$, and hence $k \leq e \leq k + ab$. Therefore, the k-th smallest element must be the a-th smallest element in A_j or belong to $T \cap A_j$, and must determine the total number of elements missing before this point. This proves the claim. □

Claim 6.6. $\mathbb{E}\left[E_i\right] \geq \mathbb{E}_j\left[\mathsf{H}\left(S_i^j | S_i^{<j}, S_{-i}, C_i, |S_i|\right)\right]$.

Proof. The intuition behind the proof is that in Phase 2, the algorithm starts out knowing only the size of the sets, and learns the k-th smallest element of the sets after computing the median. The contents of all cells needed to insert elements in Phase 2 are determined by S_{-i}, C_i, since these variables determine the cells in the core. By Claim 6.4, after fixing $S_i^{<j}, S_{-i}, C_i, |S_i|$, all the cells that do not belong to S_i are determined. Thus, after fixing $S_i^{<j}, S_{-i}, C_i, |S_i|$, the value of E_i is determined. Now if $E_i = 0$, then the k-th smallest element is determined, which means that $\mathsf{H}\left(S_i^j | S_i^{<j}, S_{-i}, C_i, |S_i|\right) = 0$. If $E_i = 1$, the inequality holds trivially. □

Recall that $t_{\mathsf{med}} \geq \sum_{i=1}^{a} \mathbb{E}\left[E_i\right]$. Then by the above claim, linearity of expectation and

the chain rule for entropy, we have:

$$t_{\mathsf{med}} \geq \sum_{i=1}^{a} \mathbb{E}\left[E_i\right] \geq \sum_{i=1}^{a} \mathop{\mathbb{E}}_{j}\left[\mathsf{H}(S_i^j | S_i^{<j}, C_i, S_{-i}, |S_i|)\right] = (1/b)\sum_{i=1}^{a} \mathsf{H}(S_i | C_i, S_{-i}, |S_i|)$$

$$\geq (1/b)\sum_{i=1}^{a} \mathsf{H}(S_i | S_{-i}) - \mathsf{H}(C_i, |S_i|)$$

$$\geq a \cdot \left(1 - \frac{t_{\mathsf{ins}} w + \log b}{b}\right), \tag{6.1}$$

where the last inequality follows from the facts that

$$\mathsf{H}(S_i | S_{-i}) = \mathsf{H}(S_i) = b, \text{ and } \mathsf{H}(C_i, |S_i|) \leq \mathsf{H}(C_i) + \mathsf{H}(|S_i|) \leq w t_{\mathsf{ins}} + \log b.$$

Set $b = 4wt_{\mathsf{ins}}$ and a to be the largest integer such that $a \leq \frac{p}{3b(b+1)}$. Since $b \geq 4$, $\frac{\log b}{b} \leq \frac{1}{2}$. Now, (6.1) implies that

$$t_{\mathsf{med}} \geq a/4 \geq \Omega\left(\frac{n^{1/(t_{\mathsf{ins}}+1)}}{w^2 \cdot t_{\mathsf{ins}}^2}\right),$$

where the last inequality follows from the fact that $a \geq \frac{p}{3b(b+1)} - 1$.

6.5 *Lower Bounds when Predecessors Computations are Non-Adaptive*

In this section we prove Theorem 6.3. Consider the sequence $\mathcal{X} = X_1, \cdots, X_n$, where

$$X_i = \{j \mid \text{cell } j \text{ is accessed while computing the predecessor of } i\}.$$

By the flower lemma (Lemma 6.1), $\mathcal{X}$ contains a t_{pred}-flower with $n^{\frac{1}{t_{\mathsf{pred}}+1}}$ petals. Let a be the largest even integer such that $a(a+1) \leq n^{\frac{1}{t_{\mathsf{pred}}+1}}$. Note that $a \geq \frac{n^{\frac{1}{2(t_{\mathsf{pred}}+1)}}}{2}$. For ease of notation, we assume that $X_1, X_2 \cdots, X_{a(a+1)}$ are the promised t_{pred}-flower.

Let S be any subset of $\{i \mid i = (j-1)(a+1)+1 \text{ for some } j \in [a]\}$. Insert all elements of S. For $j \in [a(a+1)]$, let $\mathsf{Pred}'(j)$ be the value obtained by simulating the predecessor computation assuming that the cells outside the core were never accessed when S was inserted. Note that $\mathsf{Pred}'(j)$ can be computed from the cells in the core. Let $\mathsf{Pred}(j)$ be the predecessor of j. For every $i \in [a]$, define

$$Z_i = \begin{cases} 1, & |\{j \in \{i(a+1)-a+1, \cdots, i(a+1)\} \mid \mathsf{Pred}(j) \neq \mathsf{Pred}'(j)\}| > a/2 \\ 0, & \text{otherwise.} \end{cases}$$

(a) $\sum_{i=1}^{3} Z_i = 2$. (b) $\sum_{i=1}^{3} Z_i = 0$.

Figure 6.6: $S \subseteq \{1, 5, 9\}$. Cells in petal X_i are shaded black when $\mathsf{Pred}'(i) \neq \mathsf{Pred}(i)$.

Figure 6.6 shows an example with $a = 3$. Since $|S| \leq a$ and the total number of cells accessed while inserting S is at least $\frac{a}{2} \cdot \sum_{i=1}^{a} Z_i$,

$$\sum_{i=1}^{a} Z_i \cdot (a/2) \leq t_{\mathsf{ins}} \cdot a. \tag{6.2}$$

Let C denote the contents of the core after inserting elements of S, the names of the elements i with $Z_i = 1$, and whether or not $i \in S$ for every element with $Z_i = 1$. In other words, C encodes the core, the set $\{i : Z_i = 1\}$ and the set $S \cap \{i : Z_i = 1\}$.

Lemma 6.2. *C encodes S.*

Proof. It suffices to come up with a decoding procedure that given C recovers S. The decoding algorithm first recovers elements of S in $\{i|Z_i = 1\}$ from the description of C. By definition, if $i \in S$ and $i \notin \{i|Z_i = 1\}$, then $\mathsf{Pred}'(j) = i$ for the majority values of $j \in \{i(a+1) - a + 1, \cdots, i(a+1)\}$. If $i \notin \{i|Z_i = 1\}$, then the decoding algorithm computes $\mathsf{Pred}'(j)$ for every $j \in \{i(a+1) - a + 1, \cdots, i(a+1)\}$. If the majority of the answers equal i, then the decoding algorithm infers that $i \in S$. Otherwise, it infers that $i \notin S$. This determines whether or not $i \in S$. □

We now analyze the length of the encoding of C. The contents of the core can be described with wt_{pred} bits. It takes at most $2 \log a$ bits to encode $|\{i|Z_i = 1\}|$ and $|S \cap \{i|Z_i = 1\}|$.

Given their sizes, it takes $\log \binom{a}{\sum_{i=1}^{a} Z_i}$ bits to encode $\{i|Z_i = 1\}$, and $\log \binom{a}{|S\cap\{i|Z_i=1\}|}$ bits to encode $S \cap \{i|Z_i = 1\}$. Therefore, the length of the encoding is at most

$$wt_{\mathsf{pred}} + \log \binom{a}{\sum_{i=1}^{a} Z_i} + \log \binom{a}{|S \cap \{i|Z_i = 1\}|} + 2\log a.$$

Since there are 2^a possible sets S, we must have

$$a \leq wt_{\mathsf{pred}} + \log \binom{a}{\sum_{i=1}^{a} Z_i} + \log \binom{a}{|S \cap \{i|Z_i = 1\}|} + 2\log a. \tag{6.3}$$

Observe that either $t_{\mathsf{ins}} \geq \frac{a}{64}$ or not. In the former case, since $a \geq \frac{n^{\frac{1}{2(t_{\mathsf{pred}}+1)}}}{2}$, we can conclude that $t_{\mathsf{ins}} \geq \Omega\left(n^{\frac{1}{2(t_{\mathsf{pred}}+1)}}\right)$. In the latter case, Equation 6.2 implies that $\sum_{i=1}^{a} Z_i \leq a/32$. Note that $\binom{a}{|S\cap\{i|Z_i=1\}|} \leq \binom{a}{\sum_{i=1}^{a} Z_i}$ when $\sum_{i=1}^{a} Z_i \leq a/32$. Using Proposition 2.1(a), we get

$$\log \binom{a}{\sum_{i=1}^{a} Z_i} + \log \binom{a}{|S \cap \{i|Z_i = 1\}|} \leq 2\left(\sum_{i=1}^{a} Z_i\right) \cdot \log \frac{ea}{\sum_{i=1}^{a} Z_i} \leq \frac{a}{2},$$

where the last inequality follows from the fact that $\sum_{i=1}^{a} Z_i \leq a/32$. After rearranging (6.3), the previous inequality implies that $t_{\mathsf{pred}} \geq \frac{a}{2w} - \frac{2\log a}{w}$. Since $a \geq \frac{n^{\frac{1}{2(t_{\mathsf{pred}}+1)}}}{2}$, we can conclude that $t_{\mathsf{pred}} \cdot \log(wt_{\mathsf{pred}}) \geq \Omega(\log n)$. Using Proposition 2.3, we obtain the desired lower bound of $t_{\mathsf{pred}} \geq \Omega\left(\frac{\log n}{\log\log n + \log w}\right)$.

Chapter 7

BALANCING SETS AND DEPTH-2 THRESHOLD CIRCUITS

Balancing sets are interesting families of sets from a finite universe with discrepancy-type properties, and depth-2 majority and threshold circuits are an important subclass of Boolean circuits. Recall from the introduction that the notions of balancing sets and depth-2 majority circuits are closely related, in that a balancing family with k sets can be used to design a depth-2 majority circuit computing the majority of n-bits with top fan-in $2k + 2$. The goal of this chapter is to prove new lower bounds on the size of balancing set families and the fan-in of depth-2 threshold circuits computing the majority of n-bits. All our lower bounds are obtained via the polynomial method; the results in this chapter are joint work with Hrubeš, Rao and Yehudayoff [56].

Before presenting our results, we first state the key polynomial lemma used in our lower bound proofs that shows a lower bound on the degree of a special class of polynomials.

Lemma 7.1. *Let p be prime, and let $f(x[1], \ldots, x[2p])$ be a polynomial over $\mathbb{F}_p$, where $\mathbb{F}_p$ is the field with p elements. Let f be such that for every input $x \in \{0,1\}^{2p}$ with exactly p ones, we have $f(x) = 0$, and $f(x)$ is non-zero when $x[1] = x[2] = \ldots = x[2p] = 0$. Then, the degree of f is at least p.*

Hegedűs [53] used a similar lemma to prove lower bounds for balancing sets (in his lemma there are $4p$ variables, and the focus is on inputs with $3p$ ones). Hegedűs's proof uses Gröbner basis methods and linear algebra. Srinivasan found a simpler proof of Hegedűs's lemma that is based on Fermat's little theorem and linear algebra. Alon [5] gave an alternate proof of Hegedűs's lemma using the Combinatorial Nullstellensatz. The above lemma is inspired by Srinivasan's proof of Hegedűs's lemma [96, 10]. Our simple proof is presented in Section 7.2.

Balancing Sets

Various notions of balancing set families have been considered in the past [42, 38, 6, 53, 10] with various terminologies. We use the following definition here.

Definition 7.1. *Let k be a positive integer and n be a positive even integer. We say that proper non-empty subsets $S_1, \ldots, S_k \subset [n]$ are a* balancing set family *if for every $X \subset [n]$ of size $n/2$ there is an $i \in [k]$ such that $|S_i \cap X| = |S_i|/2$.*

Given any even n, let $\mathsf{B}(n)$ denote the minimum k for which a balancing set family of size k exists. Our first result gives tight bounds on $\mathsf{B}(n)$:

Theorem 7.1. *If $n = 2p$ for a prime p, then $\mathsf{B}(n) = n/2 = p$.*

Moreover, if n is divisible by 4, we give an example of a balancing set family establishing that $\mathsf{B}(n) \leq n/2 - 1$. If n is divisible by 2, we show that $\mathsf{B}(n) \leq n/2$ by constructing a balancing set family of size $n/2$, in which each set is of size 2. We also show that this is tight when each set in the family is of size 2 (see Theorem 7.9 in Section 7.3). Previously, for arbitrary values of n, Alon, Kumar and Volk [10] showed that $\mathsf{B}(n) \geq \Omega(n)$. We show

Theorem 7.2. *If n is an even integer, then $\mathsf{B}(n) \geq n/2 - O\left(n^{0.98}\right)$.*

Our lower bounds on $\mathsf{B}(n)$ are the most interesting and they are proved using Lemma 7.1. See Section 7.3 and Section 7.4 for a full exposition of the proofs. We also apply our techniques to other questions about balancing sets in the literature and improve some of the previous bounds. We now briefly discuss two such notions from the literature.

(a) Galvin's question [42, 38, 53] asks for the smallest balancing family, denoted by $\mathsf{G}(n)$, where each set in the family is of size $n/2$, and n is a positive integer that is a multiple of 4.

(b) Jansen [57] and Alon, Kumar, and Volk [10] studied a variant where the size of each set in the family must satisfy $2\tau \leq |S_i| \leq n - 2\tau$ for a positive integer τ, and for every $X \subset [n]$ of size $n/2$, there is a set in the family such that $|S_i|/2 - \tau < |S_i \cap X| < |S_i|/2 + \tau$. Denote by $\mathsf{J}(n, \tau)$ to be the family of smallest size satisfying the above conditions.

We defer the discussion of previous known bounds on the quantities $\mathsf{G}(n)$ and $\mathsf{J}(n,\tau)$ to Section 7.1. We prove the following lower bounds on $\mathsf{G}(n)$ and $\mathsf{J}(n,\tau)$.

Theorem 7.3. *If n is divisible by 4, then $\mathsf{G}(n) \geq n/2 - O\left(n^{0.53}\right)$.*

Theorem 7.4.

1. *If $n = 2p$ for a prime p then $\mathsf{J}(n,\tau) \geq \frac{n}{4\tau-2}$.*
2. $\mathsf{J}(n,\tau) \geq \frac{n-O\left(n^{0.98}\right)}{7\tau}$.

We proceed to define the notion of unbalancing set families.

Definition 7.2. *Let n be a positive even integer, and $k \geq 0, 0 \leq t \leq n/2$ be integers. We say that subsets $S_1, \ldots, S_k \subset [n]$ are an* unbalancing set family *if for every $X \subset [n]$ of size $n/2 - t$, there is an $i \in [k]$ such that $|S_i \cap X| > |S_i|/2$.*

Given any even n, let $\mathsf{U}(n,t)$ denote the minimum k for which an unbalancing set family of size k exists. For unbalancing set families, we determine $\mathsf{U}(n,t)$ exactly:

Theorem 7.5. $\mathsf{U}(n,t) = 2t + 2$.

Again, the lower bound here is more interesting than the upper bound. It is proved by showing a connection between $\mathsf{U}(n,t)$ and the chromatic number of an appropriately defined Kneser graph [69].

Threshold Circuits

We now discuss our results on depth-2 majority and threshold circuits. The majority function, $\mathsf{MAJ}(x)$ for $x \in \{0,1\}^n$, is defined as

$$\mathsf{MAJ}(x[1], \ldots, x[n]) = \begin{cases} 1 & \sum_{i=1}^{n} x[i] \geq n/2, \\ 0 & \text{otherwise.} \end{cases}$$

The unweighted threshold function, $T_t(x)$ for $x \in \{0,1\}^n$, is defined as

$$T_t(x[1],\ldots,x[n]) = \begin{cases} 1 & \sum_{i=1}^{n} x[i] \geq t, \\ 0 & \text{otherwise,} \end{cases}$$

for some non-negative integer t. In what follows, unless stated otherwise, we refer to threshold functions when we mean unweighted threshold functions.

A depth-2 circuit is defined by Boolean functions $h, g_1, \ldots, g_k$, for some integer k, and the depth-2 circuit is said to compute a function f on input $x \in \{0,1\}^n$ if

$$f(x) = h(g_1(x),\ldots,g_k(x)).$$

Here $h, g_1, \ldots, g_k$ are called the *gates* of the circuit. h is referred to as the *top* gate, and $g_1, \ldots, g_k$ are referred to as the *bottom* gates of the circuit. Our lower bounds often hold even when h is allowed to be an arbitrary Boolean function. The *fan-in* of a gate in the circuit measures the number of variables that need to be read for the gate to carry out its computation. The fan-in of the top gate in the circuit is defined to be k. The fan-in of each of the gates g_i is r_i if g_i depends on r_i of the input variables. We sometimes refer to the top fan-in when we mean k and the bottom fan-in when we mean the maximum of $r_1, \ldots, r_k$. We say that the fan-in of the circuit is r, if r is the maximum of the top fan-in and bottom fan-in.

When functions $g_1, \ldots, g_k, h$ each compute majority, the circuit is called a majority circuit. Similarly, if all gates compute thresholds, then the circuit is called a threshold circuit. Kulikov and Podolskii [63] asked the following question: What is the minimum fan-in required to compute majority using a depth-2 majority circuit? As mentioned earlier, balancing set families are closely related to depth-2 majority circuits computing majority. Precisely, when n is even, there is a depth-2 majority circuit computing the majority of n bits with top fan-in at most $2 \cdot \mathsf{B}(n) + 2$.

To obtain a lower bound on the fan-in of such circuits, a potential approach is to show that every depth-2 majority or threshold circuit corresponds to a balancing set family. We

are able to leverage the ideas that are used to prove Theorem 7.1 to obtain lower bounds on the fan-in of these circuits. Moreover, our lower bounds are sharp up to a constant factor.

Let $n = 2p$ for a prime p. Note that the threshold function defined by the inequality $\sum_{i=1}^{n} x[i] \geq p$ is the majority function on n bits, and yields a circuit with top fan-in 1. We prove a lower bound on the top fan-in of a depth-2 threshold circuit when the bottom gates do not have the threshold p:

Theorem 7.6. *Suppose that $n = 2p$ for a prime p. Then in any depth-2 circuit computing the majority of n bits, if the bottom gates compute unweighted thresholds and read no constants, either the top fan-in is at least $n/2 = p$, or some gate at the bottom computes a threshold T_t with $t = p$.*

In fact, Theorem 7.6 implies a similar lower bound on the top fan-in when the bottom threshold gates read constants - see Section 7.5. Observe that in Theorem 7.6 we do not assume that the top gate h computes a threshold function. The lower bound holds with no restrictions on h.

Theorem 7.6 also gives tight lower bounds for the fan-in of threshold circuits computing majority. Firstly, any non-constant threshold function T_t reading at most r inputs must have $t \leq r$. Secondly, any bottom gate that computes a threshold function T_t by reading constants is equivalent to computing a threshold function $T_{t'}$ on the same input variables, for some $t' \leq t$, and $T_{t'}$ reads no constants. Here, $t' = t - \alpha$ where α is the number of ones read by T_t. Consequently, we get:

Corollary 7.1. *Suppose that $n = 2p$ for a prime p. Then in any depth-2 circuit computing the majority of n bits, if the bottom gates compute unweighted thresholds, the fan-in of the circuit must be at least $n/2 = p$.*

Since majority is a special case of the threshold function, the above corollary implies the same lower bound on the fan-in of majority circuits that compute the majority. However, by directly invoking Theorem 7.6, we obtain a slightly stronger lower bound for majority circuits computing the majority:

Corollary 7.2. *Suppose that $n = 2p$ for a prime p. Then in any depth-2 majority circuit computing the majority of n bits, either the bottom fan-in is more than $2p - 2 = n - 2$ or the top-fan in is at least $p = n/2$.*

This is because when the bottom fan-in of the majority circuit is at most $2p - 2$, the threshold of bottom gates are at most $p - 1$ and Theorem 7.6 applies.

Theorem 7.6, Corollary 7.1 and Corollary 7.2 discuss the case when $n = 2p$ for a prime p. For arbitrary values of n, we can generalize Theorem 7.6 to show that either the top fan-in is at least $n/2 - o(n)$ or some gate at the bottom computes a threshold T_t with $t \geq p$, where p is the largest prime such that $p \leq n/2$ (see Section 7.5 for the proof). Naturally, this lower bound translates to Corollary 7.1 and Corollary 7.2. In particular, we get that any depth-2 majority circuit computing the majority of n bits must have that either the bottom fan-in at least $n - o(n)$ or the top fan-in at least $n/2 - o(n)$. This nearly matches Amano's [12] construction of a depth-2 majority circuit with bottom fan-in $n - 2$ and top fan-in $n/2 + 2$.

Another kind of result that we investigate is whether *weighted* threshold functions can be computed using unweighted thresholds of low fan-in. To that end, let $n = (3p - 1)/2$ for an odd prime p, and consider the weighted threshold function

$$T(x) = \begin{cases} 1 & \text{if } \sum_{i \leq p-1} x[i] + 2 \sum_{i > p-1} x[i] \geq p, \\ 0 & \text{otherwise.} \end{cases}$$

$T(x)$ is a weighted threshold function with weights 1 and 2.

Theorem 7.7. *Any depth-2 circuit computing $T(x)$ where the bottom gates compute unweighted thresholds must have top fan-in at least $(p - 1)/2 = (n - 1)/3$.*

Observe that in Theorem 7.7 we do not assume an upper bound on the fan-in of the bottom gates. Our bounds are much stronger and significantly simpler than past lower bounds ([63, 37]) on such circuits. Our proofs of Theorem 7.1 and Theorem 7.6 are based on proving lower bounds on the degree of specific polynomials, using Lemma 7.1, that are constructed using the balancing set families and depth-2 threshold circuits, respectively.

All our results are summarized in Table 7.1 and Table 7.2.

Balancing Sets	$\mathsf{B}(n) = n/2$ when $n = 2p$	Theorem 7.1
	$\mathsf{B}(n) \geq n/2 - o(n)$	Theorem 7.2
	$\mathsf{G}(n) \geq n/2 - o(n)$	Theorem 7.3
	$\mathsf{J}(n, \tau) \geq n/(4\tau - 2)$ when $n = 2p$	Theorem 7.4
	$\mathsf{J}(n, \tau) \geq n(1 - o(1))/7\tau$	Theorem 7.4
Unbalancing Sets	$\mathsf{U}(n, t) = 2t + 2$	Theorem 7.5

Table 7.1: Summary of results on balancing and unbalancing families. p is a prime.

Function	**Bottom Gates**	**Result**	Reference
Majority	thresholds and reads no constants	$k \geq n/2$ or threshold $= p$ when $n = 2p$	Theorem 7.6
Majority	thresholds	$\max\{k, r\} \geq n/2$ when $n = 2p$	Corollary 7.1
Majority	majority	$k \geq n/2$ or $r > n - 2$ when $n = 2p$	Corollary 7.2
Majority	thresholds	$k \geq n/2 - o(n)$ or threshold $\geq \mu(n/2)$	Theorem 7.10
Majority	thresholds	$\max\{k, r\} \geq n/2 - o(n)$	Corollary 7.3
$T(x)$	unbounded fan-in thresholds	$k \geq (n - 1)/3$	Theorem 7.7

Table 7.2: Summary of results on depth-2 circuits. n is the number of input bits and p is a prime. k is the top fan-in and r is the maximum fan-in of the bottom gates. $\mu(n)$ denotes the largest prime that is no more than n.

Notation. We use the following notation in the rest of this chapter. For a positive integer n, $\mu(n)$ denotes the largest prime p so that $p \leq n$. For $x \in \{0,1\}^n$, when $x[1] = x[2] = \ldots = x[n] = 0$, we refer to x as the all-zeros vector or the all-zeros input. The all-ones vector or all-ones input is defined similarly.

Bounds on $\mu(n)$

Generalizations of Theorems 7.1 and 7.6 to the case when $n \neq 2p$ for a prime p are obtained by using a known lower bound on $\mu(n)$. Baker, Harman and Pintz [16] showed that the largest gap between consecutive primes is bounded by $O(n^{0.53})$. As a consequence, we can conclude that

Theorem 7.8. *[16]* $\mu(n) \geq n - O(n^{0.53})$.

The rest of this chapter is organized as follows. We discuss related work in Section 7.1. We prove Lemma 7.1 in Section 7.2. Theorem 7.1 is proved in Section 7.3, and the application of our techniques to generalizations of balancing set families are discussed in Section 7.4. In particular, Section 7.4 contains the proofs of Theorem 7.2, 7.3 and 7.4. Theorems 7.6 and 7.7 are proved in Sections 7.5 and 7.6 respectively. Theorem 7.5 is proved in Section 7.7.

7.1 Related Work

7.1.1 Balancing Families

Various notions of balancing set families have been studied. We first describe the question posed by Galvin [42, 38, 53].

Definition 7.3. *Let n be a positive integer that is divisible by 4. A family of proper subsets $S_1, \ldots, S_k \subset [n]$ is* exactly balancing *if each S_i is of size $n/2$ and for every $X \subset [n]$ of size $n/2$ there is an $i \in [k]$ such that $|X \cap S_i| = |S_i|/2$.*

When n is divisible by 4, let $\mathsf{G}(n)$ denote the minimum k for which an exactly balancing set family of size k exists. Clearly, the family of all subsets of $[n]$ of size $n/2$ is exactly

balancing, and any family with only one set is not exactly balancing. Therefore finding the minimum number of sets in any exactly balancing set family is interesting.

Galvin [42] observed that $\mathsf{G}(n) \leq n/2$; take $n/2$ consecutive intervals of length $n/2$. Frankl and Rödl [42] proved that $\mathsf{G}(n) \geq \Omega(n)$ if $n/4$ is odd, and later Enomote, Frankl, Ito and Nomura [38] proved that if $n/4$ is odd, then $\mathsf{G}(n) \geq n/2$. Proofs in [42, 38] are based on techniques from linear algebra and extremal set theory. Recently, Hegedűs [53] used algebraic techniques to prove that if $n/4$ is prime, then $\mathsf{G}(n) \geq n/4$. For arbitrary values of n, Alon, Kumar and Volk [10] proved that $\mathsf{G}(n) \geq \Omega(n)$. Theorem 7.3 improves the bound of Alon, Kumar and Volk.

Several natural variants of Galvin's problem have been studied. One such variant was studied by Jansen [57], and Alon, Kumar and Volk [10]:

Definition 7.4. *Let n be an even integer, and let τ be a positive integer. Let $S_1, \ldots, S_k \subset [n]$ with $2\tau \leq |S_i| \leq n - 2\tau$. We say that $S_1, \ldots, S_k$ is a* τ-balancing set family *if for every $X \subset [n]$ of size $n/2$ there is an $i \in [k]$ such that*

$$|S_i|/2 - \tau < |X \cap S_i| < |S_i|/2 + \tau.$$

When n is even and τ is positive, let $\mathsf{J}(n, \tau)$ denote the minimum k for which such a family of size k exists. This variant allows the family to have sets with different sizes and the intersection sizes to take more than just one value. Alon, Kumar and Volk proved that $\mathsf{J}(n, \tau) \geq \frac{1}{10^5} \cdot (n/\tau)$. This lower bound is sharp up to a constant factor. Theorem 7.4 improves their bound to $\frac{n - o(n)}{7\tau}$.

Our techniques yield a quantitatively stronger lower bound on balancing set families. The improvement stems from the fact that the ratio of the degree of the polynomial to the number of variables of the polynomial increases from 1/4 to 1/2. Moreover, the application of Lemma 7.1 eliminates an additional argument using the probabilistic method employed in the work of Alon, Kumar and Volk.

There are many applications of balancing set families. Alon, Bergmann, Coppersmith and Odlyzko [6] studied a different version of balancing sets that has applications to optical

data communication. Jansen [57] and Alon, Kumar, and Volk [10] showed applications to proving lower bounds for syntactic multilinear algebraic circuits (also see [90]).

7.1.2 Threshold Circuits

A depth-d majority circuit can be defined in analogy to depth-2 majority circuits. Let $\mathsf{M}_d(n)$ denote the minimum fan-in of a depth-d majority circuit that computes the majority of n bits. A long line of work has addressed the question of computing the majority function using majority circuits. Ajtai, Komlós and Szemerédi [3] showed that $\mathsf{M}_{c\cdot\log n}(n) = O(1)$, for some constant c. Using probabilistic arguments, Valiant [98] showed the existence of depth $O(\log n)$ majority circuit that computes the majority, where each gate has constant fan-in. Allender and Koucky [4] showed that $\mathsf{M}_c(n) = O(n^{\epsilon(c)})$, where c is a constant and $\epsilon(c)$ is a function of c. Kulikov and Podolskii proved that $\mathsf{M}_3(n) \leq \tilde{O}\left(n^{2/3}\right)$[1]. See [63, 37, 39] and references within for a detailed treatment.

We now discuss previous bounds on $\mathsf{M}_2(n)$. Kulikov and Podolskii [63] used probabilistic arguments to show that $\mathsf{M}_2(n) \geq \tilde{\Omega}\left(n^{7/10}\right)$. They also proved that $\mathsf{M}_2(n) \geq \tilde{\Omega}\left(n^{13/19}\right)$ when the gates are not required to read distinct variables. Amano and Yoshida [13] showed that for every odd $n \geq 7$, $\mathsf{M}_2(n) \leq n - 2$, where they allowed some of the gates to read variables multiple times. Later, Engles, Garg, Makino and Rao [37] used ideas from discrepancy theory to prove that $\mathsf{M}_2(n) \geq \Omega(n^{4/5})$ when the gates do not read constants. Posobin [86] showed that majority can be computed by a depth-2 majority circuit of fan-in at most $2n/3 + 4$ (this was also proved independently by Bauwens [86]). Very recently, Amano [12] gave a construction of a depth-2 majority circuit computing majority with bottom fan-in $n-2$ and top fan-in $n/2 + 2$.

Kulikov and Podolskii [63] studied and proved lower bounds on other variants of depth-2 majority circuits. In particular, they consider circuits in which each majority gate can read a variable multiple times. Let W be the maximum over the number of times a variable is read.

---

[1] In the rest of the chapter, $\tilde{O}(a)$ and $\tilde{\Omega}(a)$ mean that $\mathsf{polylog}(a)$ factors are ignored.

They prove that $\mathsf{M}_2(n) \geq \min\left\{\tilde{\Omega}\left(n^{13/19}\right), \tilde{\Omega}\left(\frac{n^{7/10}}{W^{3/10}}\right)\right\}$. In this case, our techniques yield a lower bound of $\mathsf{M}_2(n) \geq \Omega\left(\frac{n}{W}\right)$. Essentially, their lower bound is stronger when $W \geq n^{6/19}$ and our bound is stronger when $W \leq n^{6/19}$.

The question of computing weighted thresholds using a depth-2 threshold function is connected to the study of exact threshold circuits initiated by Hansen and Podolskii [51]. It may also be useful in studying the expressibility of general functions using threshold or *ReLu* gates; see the work of Williams [104].

We would like to emphasize that the lower bounds in Theorems 7.6 and 7.7 are tight and only off by constant factors. In addition, most functions considered in past work on majority and threshold circuit lower bounds do not admit depth-2 majority or threshold circuits with linear fan-in on the gates. In fact, one can prove exponential lower bounds on the size of circuits computing these functions (see [51]).

7.2 Proof of the Polynomial Lemma (Lemma 7.1)

Let f be as in the assumption of Lemma 7.1. Consider the polynomial

$$g(x[1], \ldots, x[2p]) = (1 - x[1]) \cdot \prod_{i=1}^{p-1} \left(i - \sum_{i=1}^{2p} x[i]\right),$$

which has degree p. For $x \in \{0,1\}^{2p}$, observe that $g(x) = 0$ if the number of ones in x is not a multiple of p or x is the all-ones input, and $g(x) \neq 0$ if x is the all-zeros input. Therefore, $f \cdot g$ is non-zero on the all-zeros input and 0 elsewhere in $\{0,1\}^{2p}$.

We will now show that the degree of $f \cdot g$ is at least $2p$. Consider the polynomial h that is obtained by multilinearizing $f \cdot g$. In other words, replace every power x_i^k with x_i in $f \cdot g$, for $k \geq 1$, Observe that the degree of h is at most the degree of f. Define $\alpha = h(0, \ldots, 0)$. Recall that there is a one-to-one correspondence between multilinear polynomials over $\mathbb{F}_p$ on $2p$ variables and the set of all functions from $\{0,1\}^{2p} \to \mathbb{F}_p$. Since h is the same as the function that is α on the all-zeros input and 0 elsewhere in $\{0,1\}^{2p}$, we can use this

correspondence to conclude that

$$h(x[1],\ldots,x[2p]) = \alpha \cdot \prod_{i=1}^{2p}(1-x[i]).$$

Therefore the degree of h is $2p$.

Hence the degree of $f \cdot g$ is at least $2p$, implying that the degree of f is at least p.

7.3 Upper and Lower Bounds on $\mathsf{B}(n)$

In this section, we describe some explicit balancing set families.

Lemma 7.2.

1. *If n is divisible by 4 and $n \neq 4$, then $\mathsf{B}(n) \leq n/2 - 1$.*

2. *If n is divisible by 2 and $n \neq 2$, then $\mathsf{B}(n) \leq n/2$.*

Proof. When 4 divides n, there is a family of $k = \frac{n}{2} - 1$ sets that are balancing: take any k sets, each of size 4, satisfying $S_i \cap S_j = \{1,2\}$ for all $i \neq j$. This family has the property that for any subset $X \subset [n]$ of size $n/2$, there is an $i \in [k]$ such that $|X \cap S_i| = 2$.

When 2 divides n, there is a family of $k = n/2$ sets that are balancing: take any k sets, each of size 2, satisfying $S_i \cap S_j = \{1\}$ for all $i \neq j$. This family has the property that for any subset $X \subset [n]$ of size $n/2$, there is an $i \in [k]$ such that $|X \cap S_i| = 1$. □

As implied by Theorem 7.1, when $n = 2p$ for a prime p, there is no construction with $k = \frac{n}{2} - 1$ sets; the minimum possible k in this case is $\frac{n}{2}$. We now prove Theorem 7.1.

Proof of Theorem 7.1. Lemma 7.2 implies that $\mathsf{B}(n) \leq p = n/2$. We now proceed to show that $\mathsf{B}(n) \geq p = n/2$. Let $S_1,\ldots,S_k$ be the balancing set family. Without loss of generality each $|S_i|$ is even, and therefore $1 \leq |S_i|/2 \leq p-1$ for all $i \in [k]$. We will now construct a polynomial that is non-zero on the all-zeros input and vanishes on all $x \in \{0,1\}^{2p}$ with p ones. Define the polynomial

$$f(x[1],\ldots,x[2p]) = \prod_{i=1}^{k}\left(|S_i|/2 - \sum_{j\in S_i} x[j]\right),$$

over $\mathbb{F}_p$ that has degree k. Since $1 \leq |S_i|/2 \leq p-1$ for all $i \in [k]$, $f(0) \neq 0$. We will show that $f(x) = 0$, for $x \in \{0,1\}^{2p}$, when x exactly has p ones. This is because the input x to f with exactly p ones corresponds to a set $X \subset [2p]$ of size p. The fact that there is an $i \in [k]$ such that $|S_i \cap X| = |S_i|/2$, implies that $|S_i|/2 - \sum_{j \in S_i} x[j] = 0$. By applying Lemma 7.1, we can conclude that $k \geq p$. □

Remark. In Definition 7.1, since $|S_i \cap X| = |S_i|/2$, it is no loss of generality to assume that each S_i is even sized. The definition can be relaxed by having $|S_i \cap X| = \lceil |S_i|/2 \rceil$. In this relaxed definition, the family $\{1\}, \{2, \ldots, 2p\}$ is balancing and the size of the family is 2. However, if we impose an extra condition that each $|S_i| \geq 2$, then we can prove that the size of any such family is at least p.

When n is even and all sets in the balancing set family are of size 2, we use a graph theoretic argument to prove that the size of the family is at least $n/2$. This shows that the construction in the proof of part 2 of Lemma 7.2 is tight.

Theorem 7.9. *Let n be a positive even integer. Let $S_1, \ldots, S_k \subset [n]$ be a balancing set family. If $|S_i| = 2$ for all $i \in [k]$, then, $k \geq n/2$.*

We need the following lemma about integers to prove the above theorem.

Lemma 7.3. *Let $a_1, \ldots, a_k$ be positive integers such that $\sum_{i=1}^{k} a_i = n$. If $k > n/2$, then for every non-negative integer $s \leq n$, there is a $S \subseteq [k]$ such that $\sum_{i \in S} a_i = s$.*

Proof. We prove the lemma by induction on n. The base case is when $n = 1$. In this case $k = 1$, and the possible values for s is 0 and 1. Hence the statement is true for $n = 1$. Induction hypothesis assumes that the statement is true for all positive integers that are at most $n-1$, and we prove it for n. Without loss of generality, we can assume that a_1 is the largest among $a_1, \ldots, a_k$. If $a_1 = 1$, then the statement is true because this implies that $k = n$, and for every $s \leq n$, $\sum_{i=1}^{s} a_i = s$. For the rest of the proof, we assume that $a_1 \geq 2$. We will first show that $a_1 \leq \lceil n/2 \rceil$. Assume for contradiction that $a_1 > \lceil n/2 \rceil$.

Since each $a_i \geq 1$, we can conclude that $\sum_{i=2}^{k} a_i \geq k-1 \geq \lfloor n/2 \rfloor$. Therefore, we get that $\sum_{i=1}^{k} a_i > \lfloor n/2 \rfloor + \lceil n/2 \rceil = n$, which is a contradiction.

We claim that $k-1 > \frac{n-a_1}{2}$. Indeed, since $k > n/2$, we have

$$k-1 > \frac{n-2}{2} \geq \frac{n-a_1}{2},$$

where the last inequality follows from the fact that $a_1 \geq 2$. As $k-1 > \frac{n-a_1}{2}$, we can apply the induction hypothesis on $a_2, \ldots, a_k$. By the induction hypothesis, for every $s \leq n-a_1$, there is a $S \subseteq \{2,3,\ldots,k\}$ such that $\sum_{i \in S} a_i = s$. For every $s > n-a_1$, let s' be such that $s = a_1 + s'$. We have

$$n - a_1 \geq s' = s - a_1 > n - 2a_1 \geq -1$$

where the last inequality follows from $a_1 \leq \lceil n/2 \rceil$. This implies that $0 \leq s' \leq n-a_1$, and the induction hypothesis gives a set $S \subseteq \{2,3,\ldots,k\}$ such that $\sum_{i \in S} a_i = s'$. Hence $\sum_{i \in S \cup \{1\}} a_i = s$. This completes the proof. □

Proof of Theorem 7.9. Consider a graph defined on the vertex set $[n]$. $(u,v) \in [n] \times [n]$, for $u \neq v$, is an edge in the graph iff $u, v \in S_i$ for some $i \in [k]$. The number of edges in this graph is k. Let k' be the number of connected components, and let $a_1, \ldots, a_{k'}$ be such that a_i is the size of the i-th connected component. Observe that $\sum_{i=1}^{k'} a_i = n$. Since the number of edges in the i-th connected component is at least $a_i - 1$, we get that $k' \geq n-k$. We will proceed to show that $k \geq n/2$. Assume for contradiction that $k < n/2$. We then know that $k' > n/2$. Lemma 7.3 implies there is a $S \subset [k']$ such that $\sum_{i \in S} a_i = n/2$. Define $X \subset [n]$ to be the set of all vertices in the connected components indexed by S. $|X| = n/2$ and there is no edge from X to $[n] \setminus X$. Therefore, for every $i \in [n]$, $|S_i \cap X| \neq 1$ and this is a contradiction. Hence, $k' \leq n/2$ and $k \geq n/2$. □

7.4 Balancing Families: Generalizations and Improvements

In this section we prove Theorems 7.2, 7.3 and 7.4. The following lemma is crucial in the proofs these theorems.

Lemma 7.4. *Let n be an even integer. Let $S_1, \ldots, S_k \subset [n]$ and $T_1, \ldots, T_k \subseteq [\mu(n/2) - 1]$. Suppose that there is a set $R \subseteq [n]$ of size $n - 2\mu(n/2)$ such that for every $i \in [k]$ and $t \in T_i$, $|S_i \cap R| < t$, and for every $X \subset [n]$ of size $n/2$ there is an $i \in [k]$ such that $|X \cap S_i| \in T_i$. Then $\sum_{i=1}^{k} |T_i| \geq \mu(n/2)$.*

Proof. Define the polynomial

$$F(x[1], \ldots, x[n]) = \prod_{i=1}^{k} \prod_{t \in T_i} \left(t - \sum_{j \in S_i} x[j] \right).$$

Let $p = \mu(n/2)$. Define the polynomial $f(x[1], x[2], \ldots, x[2p])$ over $\mathbb{F}_p$ by setting in F half of the variables indexed by R to 0 and the other half to 1. The degree of f is at most $\sum_{i=1}^{k} |T_i|$. We claim that f takes the value 0 on all inputs with exactly p ones and f is non-zero on the all-zeros input. This is sufficient to prove the theorem as Lemma 7.1 implies that $\sum_{i=1}^{k} |T_i| \geq p$.

The former part of the claim is true because the input x to f with exactly p ones along with the variables in R that are set to 1 correspond to a set $X \subset [n]$ of size $n/2$. The fact that there is an $i \in [k]$ and $t \in T_i$ with $|S_i \cap X| = t$, implies $t - \sum_{j \in S_i} x[j] = 0$.

We now proceed to show that f is non-zero on the all-zeros input. On the all-zeros input for f, we know that all variables indexed by $[n] \setminus R$ are set to 0 and we do not have any control on the assignment to the variables in R. However, since for every $i \in [k]$ and $t \in T_i$, $0 < t < p$ and $|S_i \cap R| < t$, f is non-zero on the all-zeros input. □

Implications of Lemma 7.4

We now discuss the implications of Lemma 7.4 to the questions about balancing set families discussed earlier. The choice of R in Lemma 7.4 depends on the context. We obtain an asymptotically sharp lower bound for Galvin's problem and an improvement over the lower bound of Alon, Kumar and Volk.

$\mathbf{B}(n)$: We prove Theorem 7.2 using the following claim, which is proved in Section 7.4.1.

Claim 7.1. *Let n be a positive integer and $S_1, \ldots, S_k \subset [n]$ be a balancing set family. If n is large enough and $k < n/2 - 2n^{0.98}$, then there exists a $R \subset [n]$ of size $n - 2\mu(n/2)$ such that for every $i \in [k]$, $|S_i \cap R| < |S_i|/2$.*

Proof of Theorem 7.2. Assume for contradiction that $\mathsf{B}(n) < n/2 - 2n^{0.98}$. Let R be the set given by Claim 7.1. By invoking Lemma 7.4 with R and each $T_i = \{|S_i|/2\}$, we get $\mathsf{B}(n) \geq \mu(n/2) \geq n/2 - O\left(n^{0.53}\right)$, where the last inequality follows from Theorem 7.8. This contradicts the assumption for large values of n. □

$\mathsf{J}(n, \tau)$: We prove Theorem 7.4. We have that each

$$T_i = \{|S_i|/2 - \tau + 1, \ldots, |S_i|/2, \ldots, |S_i|/2 + \tau - 1\}.$$

When $n = 2p$ for a prime p, $R = \emptyset$. Observing that each T_i is of size $2\tau - 1$, Lemma 7.4 implies Part 1 of Theorem 7.4.

We now proceed to prove Part 2 of Theorem 7.4. We need the following claim, and this claim is proved in Section 7.4.1.

Claim 7.2. *Let n be a positive integer, τ be a positive integer, and $S_1, \ldots, S_k \subset [n]$ be τ-balancing set family. If n is large enough and $k < n/(7\tau) - n^{0.98}/(7\tau)$, then there exists a $R \subset [n]$ of size $n - 2\mu(n/2)$ such that for every $i \in [k]$, $|S_i \cap R| \leq |S_i|/2 - \tau$.*

Proof of Part 2 of Theorem 7.4. Assume for contradiction that

$$\mathsf{J}(n, \tau) < n/(7\tau) - n^{0.98}/(7\tau).$$

Let R be the set given by Claim 7.2. By invoking Lemma 7.4 with R and each

$$T_i = \{|S_i|/2 - \tau + 1, \ldots, |S_i|/2, \ldots, |S_i|/2 + \tau - 1\},$$

we get $\mathsf{J}(n, \tau) \geq \frac{\mu(n/2)}{2\tau - 1} \geq \frac{n - O\left(n^{0.53}\right)}{4\tau - 2}$, where the last inequality follows from Theorem 7.8. This contradicts the assumption for large values of n. □

$\mathsf{G}(n)$: We prove Theorem 7.3. For Galvin's problem, n is divisible by 4, each S_i is of size $n/2$ and each $T_i = \{n/4\}$. R can be chosen to be any arbitrary set of size $n - 2\mu(n/2)$. For Lemma 7.4 to apply, we need that for each $i \in [k]$ and $t \in T_i$, $T_i \subseteq [\mu(n/2) - 1]$ and $|S_i \cap R| < t$. This translates into the condition that $\mu(n/2) > 3n/8$. Lemma 7.4 in conjunction with Theorem 7.8 implies Theorem 7.3.

Specifically Theorem 7.3 shows that our lower bound is sharp up to an additive $o(n)$ term as $\mathsf{G}(n) \leq n/2$. It is worth noting that $\mathsf{G}(n) < n/2$ for $n \in \{8, 16\}$, so a general $n/2$ lower bound is false (see [10]).

7.4.1 Proofs of Claim 7.1 and Claim 7.2

We need the following claim to prove Claim 7.1 and Claim 7.2.

Claim 7.3. *Let n be a positive integer, $\emptyset \subset S \subset [n]$, and $0 \leq t \leq n$ be an integer. Let T be a random subset of $[n]$ of size r. Then*

$$\Pr[|S \cap T| \geq t] \leq 2^{|S|} \cdot \left(\frac{r}{n}\right)^t.$$

Proof. We have

$$\Pr[|S \cap T| \geq t] \leq \binom{|S|}{t} \cdot \frac{\binom{n-t}{r-t}}{\binom{n}{r}} \leq 2^{|S|} \cdot \left(\frac{r}{n}\right)^t. \qquad \square$$

Proof of Claim 7.1. Let $p = \mu(n/2)$ and r be such that $r = n - 2p$. By Theorem 7.8, for large enough n, $r < n^{0.54}$. In addition, we can assume without loss of generality that for all $i \in [k]$, $|S_i|$ is even.

Define

$$A = \{j \mid j \in S_i \text{ for some } i \in [k] \text{ and } |S_i| = 2\}$$

and

$$B = \{j \mid j \in S_i \text{ and } j \in S_{i'} \text{ for } i, i' \in [k], i \neq i', \text{ and } |S_i| = |S_{i'}| = 4\}.$$

In words, A is the set of elements which are in sets of size 2 and B is the set of elements that belong to at least 2 sets of size 4. We claim that $|A \cup B| \leq n - 4n^{0.98}$. Indeed, we have that

$$|A| \leq 2 \cdot (\#\text{ sets of size } 2)$$

and

$$|B| \leq (4 \cdot (\#\text{ sets of size } 4))/2 = 2 \cdot (\#\text{ sets of size } 4).$$

Since $k \leq n/2 - 2n^{0.98}$, we get that $|A \cup B| \leq n - 4n^{0.98}$.

By the definition of B, we have the property that for every $j \in [n] \setminus B$, it is the case that j appears in at most one set S_i of size 4. Let $X \subseteq [n] \setminus (A \cup B)$ be the largest set such that for every $i \in [k]$ if $|S_i| = 4$ and $|S_i \cap (A \cup B)| < 4$, then $|S_i \cap X| = 1$. The definition of X implies that $|X| \geq |[n] \setminus (A \cup B)|/4 \geq n^{0.98}$ for large enough n.

We will pick a random subset R of size r from X, and show that

$$\Pr[\exists i \in [k], |S_i \cap R| \geq |S_i|/2] < 1/n^{0.2}. \tag{7.1}$$

This is sufficient as this implies that there exists a set $R \subset [n]$ of size r such that for every $i \in [k]$, $|S_i \cap R| < |S_i|/2$.

We now proceed to prove Eq. (7.1). For every $i \in [k]$, by the definition of X, if $|S_i| = 2$, then $|S_i \cap R| = 0$. Similarly, if $|S_i| = 4$, then $|S_i \cap R| \leq 1$. We now consider the case when $|S_i| \geq 6$ for $i \in [k]$. Using $r < n^{0.54}$ and $|X| \geq n^{0.98}$, we get from Claim 7.3 that for every $i \in [k]$,

$$\begin{aligned}\Pr[|S_i \cap R| \geq |S_i|/2] &\leq 2^{|S_i|} \cdot \left(\frac{1}{n^{0.44}}\right)^{|S_i|/2} \\ &\leq 2^{|S_i| - 0.22 \cdot |S_i| \cdot \log n}.\end{aligned}$$

For large enough n we have,

$$\Pr[|S_i \cap R| \geq |S_i|/2] \leq 2^{-0.21 \cdot |S_i| \cdot \log n} \leq n^{-1.2},$$

where the last inequality follows from the fact that $|S_i| \geq 6$. Therefore by a union bound over $i \in [k]$, we can conclude that

$$\Pr[\exists i \in [k], |S_i \cap R| \geq |S_i|/2] < 1/n^{0.2}. \qquad \square$$

Proof of Claim 7.2. Note that $\tau \geq 1$. Let $p = \mu(n/2)$ and r be such that $r = n - 2p$. By Theorem 7.8, for large enough n, $r < n^{0.54}$.

Consider the following iterative process: If there is a set S_i with at most 7τ elements, remove the set and its elements from other sets to which they belong. Repeat this process until all remaining sets have size more than 7τ. Let A denote the set of elements from $[n]$ that did not belong a S_i that was removed, and let $T_1, \ldots, T_{k'} \subseteq A$ ($k' \leq k$) be the sets that remain. Observe that since $k < n/(7\tau) - n^{0.98}/(7\tau)$, $|A| \geq n^{0.98}$.

If $k' = 0$, then let R be any subset of A of size r. We now consider the case when $k' \geq 1$. We will pick a random subset R of size r from A, and show that

$$\Pr\left[\exists i \in [k'], |T_i \cap R| \geq |T_i|/2 - \tau\right] < 1/n^{0.01}. \tag{7.2}$$

This is sufficient because it shows the existence of a R of size r such that $|S_i \cap R| < |S_i|/2 - \tau$ for every $i \in [k]$. Indeed for each S_i that was removed, we have that $|S_i \cap R| = 0$. For each S_i that was not removed, we have that $|S_i \cap R| < |T_i|/2 - \tau \leq |S_i|/2 - \tau$, where T_i is the set corresponding to S_i that remained.

We now proceed to show Eq. (7.2). Using $r < n^{0.54}$ and $|A| \geq n^{0.98}$, we get from Claim 7.3 that for every $i \in [k']$,

$$\begin{aligned}
\Pr\left[|T_i \cap R| \geq |T_i|/2 - \tau\right] &\leq 2^{|T_i|} \cdot \left(\frac{1}{n^{0.44}}\right)^{|T_i|/2-\tau} \\
&= 2^{|T_i|(1-0.22\log n)+0.44\tau\log n} \\
&\leq 2^{(0.44\tau - 0.21|T_i|)\cdot\log n},
\end{aligned}$$

where the last inequality is true for large enough n. Since, $|T_i| \geq 7\tau$, we have that $0.44\tau - 0.21|T_i| < -1.01\tau$. Therefore, $\Pr[|T_i \cap R| \geq |T_i|/2 - \tau] \leq n^{-1.01}$, as $\tau \geq 1$. By a union bound over $i \in [k']$, we can conclude that

$$\Pr\left[\exists i \in [k'], |T_i \cap R| \geq |T_i|/2 - \tau\right] < 1/n^{0.01}. \qquad \square$$

7.5 Computing Majority using Depth-2 Threshold Circuits

We first prove Theorem 7.6.

Proof of Theorem 7.6. Let k be the top fan-in of the circuit, and let $g_1, \ldots, g_k$ be the threshold functions given by the bottom gates of the circuit. We know that g_i is defined by an inequality of the form $L_i(x) \geq t_i$ for a linear function L_i. Assume towards a contradiction that $k < p$ and each $t_i \neq p$.

Define the polynomial

$$f(x) = \prod_{i \in \{j \mid 0 < t_j < 2p\}} (L_i(x) - t_i)$$

over $\mathbb{F}_p$ that has degree at most k. By definition, $f(0)$ is non-zero. We claim that $f(x) = 0$ on every $x \in \{0,1\}^{2p}$ with p ones. Indeed, for such a x we have that $\mathsf{MAJ}(x) = 1$, but for x' that is obtained from x by flipping a coordinate with value 1 to 0, we have that $\mathsf{MAJ}(x') = 0$. Observe that each L_i is a linear function with coefficients in $\{0,1\}$. Since x and x' only differ in one coordinate, we have $L_i(x) - L_i(x') \in \{0,1\}$ for every $i \in [k]$. $\mathsf{MAJ}(x) = 1$ and $\mathsf{MAJ}(x') = 0$ implies that there is an $i \in [k]$ such that $g_i(x) = 1$ and $g_i(x') = 0$. This means that $L_i(x) = t_i$, but $L_i(x') = t_i - 1$. Moreover, this implies that $0 < t_i < 2p$. Hence, for every $x \in \{0,1\}^{2p}$ with p ones, there is an $i \in [k]$ such that $L_i(x) = t_i$ and $0 < t_i < 2p$, which makes $f(x) = 0$. Therefore Lemma 7.1 implies that the degree of f is at least p, which is a contradiction. □

The following theorem for arbitrary values of n is proved using Theorem 7.6.

Theorem 7.10. *In any depth-2 circuit computing the majority of n bits, if the bottom gates compute unweighted thresholds, either the top fan-in is at least $\mu(n/2)$, or some gate at the bottom computes a threshold T_t with $t \geq \mu(n/2)$.*

Proof. Let k be the top fan-in of the circuit, and let $p = \mu(n/2)$. If there exists a bottom gate with threshold at least p, then we are done. So assume that all bottom gates have threshold less than p. Set half the variables in $x[2p+1], \ldots, x[n]$ to 0 and the other half to 1. We get a new depth-2 circuit computing the majority of $x[1], \ldots, x[2p]$. Any bottom threshold gate computing T_t that reads constants is equivalent to a threshold gate computing $T_{t'}$ on the

same input variables with $t' \leq t < p$, and $T_{t'}$ reads no constants. Here, $t' = t - \alpha$, where α is the number of ones read by T_t. Replacing each bottom gate that reads constants with its equivalent gate that reads no constants, we obtain a depth-2 circuit in which each bottom gate computes a threshold function with threshold less than p and does not read constants. By applying Theorem 7.6, we can conclude that $k \geq p$. □

Using Theorem 7.8 we get a corollary to Theorem 7.10.

Corollary 7.3. *In any depth-2 circuit computing the majority of n bits, if the bottom gates compute unweighted thresholds, then the fan-in is at least $n/2 - O(n^{0.53})$.*

7.6 Computing Weighted Thresholds

We now give the proof of Theorem 7.7 Let $g_1, \ldots, g_k$ be the threshold functions given by the bottom gates of the circuit. Let

$$L(x) = \sum_{i \leq p-1} x[i] + 2 \sum_{i > p-1} x[i].$$

Note that L is a polynomial on $\frac{3p-1}{2}$ variables. For $i \in [k]$, we know that g_i is defined by an inequality of the form $L_i(x) \geq t_i$ for a linear function L_i with coefficients in $\{0, 1\}$.

Consider the polynomial

$$f(x) = \prod_{i \in \{j \ \mid \ 0 < t_j < p\}} (L_i(x) - t_i)$$

over $\mathbb{F}_p$ that has degree at most k. By definition, f is non-zero on the all-zeros input. We will show that $f(x) = 0$ on $x \in \{0,1\}^{\frac{3p-1}{2}}$ such that $L(x) = p$.

Let $x \in \{0,1\}^{\frac{3p-1}{2}}$ be such that $L(x) = p$. Note that for every such x, the number of ones in it is at most $p - 1$ and at least 1. For every $x' \in \{0,1\}^{\frac{3p-1}{2}}$ that is obtained by flipping one of the coordinates of x with value 1 to 0, we have $T(x') = 0$. For such x, x', there must be an $i \in [k]$ such that $g_i(x) = 1$ and $g_i(x') = 0$. Moreover, L_i being a linear function with coefficients in $\{0, 1\}$ implies that $L_i(x) - L_i(x') \in \{0, 1\}$. Since $g_i(x) \neq g_i(x')$,

we have $L_i(x) = t_i$. In addition, since the number ones in x is at most $p-1$ and at least 1, we get that $0 < t_i < p$. Hence we can conclude that $f(x) = 0$.

We now find a polynomial g that is 0 everywhere in $\{0,1\}^{\frac{3p-1}{2}}$, except on the all-zeros input and x such that $L(x) = p$. Define

$$g(x) = (1 - x[1]) \cdot \prod_{i=1}^{p-1} (i - L(x)) .$$

The degree of g is p, and $f \cdot g$ is non-zero on the all-zeros input and 0 elsewhere in $\{0,1\}^{\frac{3p-1}{2}}$. We will show that the degree of $f \cdot g$ is at least $(3p-1)/2$. As in the proof of Lemma 7.1, let h be the multilinearization of $f \cdot g$. Then h is non-zero on the all-zeros input and 0 elsewhere in $\{0,1\}^{\frac{3p-1}{2}}$. Therefore the degree of h is at least $(3p-1)/2$. Since the degree of h is at most the degree of $f \cdot g$, the degree of f is at least $(p-1)/2$.

7.7 *Upper and Lower Bounds on* $\mathsf{U}(n,t)$

Theorem 7.5 is proved in this section. We first recall the definition of a Kneser graph. The Kneser graph $K_{n,\alpha}$ is a graph whose vertices are identified with the subsets of $[n]$ of size α, and there is an edge between two vertices if and only if the corresponding subsets are disjoint. We need the following theorem bounding the chromatic number of Kneser graphs.

Theorem 7.11 ([69])**.** *Consider the Kneser graphs in which the vertex set is given by subsets of* $[n]$ *of size* α*. Then the chromatic number of this graph is* $\max\{1, n - 2\alpha + 2\}$.

Proof of Theorem 7.5. We first prove the upper bound. The following $2t+2$ sets form an unbalancing family:

$$\{1\}, \{2\}, \ldots, \{2t+1\}, \{2t+2, 2t+3, \ldots, n\}.$$

The above family has the property that for a given $X \subseteq [n]$ of size $n/2 - t$, either $X \subseteq \{2t+2, 2t+3, \ldots, n\}$ or not. In the former case,

$$|X \cap \{2t+2, 2t+3, \ldots, n\}| = n/2 - t > \frac{n-2t-1}{2}.$$

In the latter case, there will be an $i \in [2t+1]$ such that $i \in X$. Therefore, $|X \cap \{i\}| = 1 > \frac{1}{2}$.

We now prove the lower bound. Consider the Kneser graph in which the vertex set is given by subsets of $[n]$ of size $n/2 - t$. We claim that the chromatic number of this graph is at most k. The coloring is as follows: For every $X \subseteq [n]$ of size $n/2 - t$, we know that there is an $i \in [k]$ such that $|S_i \cap X| > |S_i|/2$. The vertex associated with X is given the color i. This is a proper coloring because for every $X, Y \subseteq [n]$, each of size $n/2 - t$ that are disjoint, it cannot be the case that $|X \cap S_i| > |S_i|/2$ and $|Y \cap S_i| > |S_i|/2$. Therefore by Theorem 7.11, we can conclude that $k \geq 2t + 2$. □

Chapter 8

CONCLUSION AND OPEN PROBLEMS

In this thesis we used techniques from information theory, algebra and combinatorics to answer lower bound questions in communication complexity, data structures and depth-2 threshold circuits. Specifically, we showed (a) how to compress two party protocols when the information revealed by the parties is asymmetric, (b) how to combine a modification of cell sampling with the discrepancy method in communication complexity to obtain new lower bounds against static data structures for the Vector-Matrix-Vector problem, (c) how systematic linear data structures are equivalent to rigidity, (d) how to use the sunflower lemma for lower bounds against non-adaptive data structures that compute the median, and (e) how to use polynomials over finite fields to prove lower bounds on the fan-in of depth-2 threshold circuits computing the majority of n bits.

We list some open problems related to compressing two party communication protocols, data structure lower bounds and balancing sets.

Protocol Compression. Given a protocol with communication C and internal information cost I, can it be simulated with communication at most $O(I \cdot \mathsf{poly}(\log C))$? Though we know near optimal simulations when the inputs to the parties are independent [62, 95], the best known upper bound in the general setting is $\min\left\{O\left(\sqrt{IC}\cdot\log C\right), 2^{O(I)}\right\}$ (see [18, 23]). From the perspective of lower bounds, Ganor, Kol and Raz [48, 49] showed that there is no simulation with communication $\alpha I \log C/\log\log C$, for some constant α. Any progress on either the upper bound or the lower bound would be interesting.

Data Structure Lower Bounds. As discussed earlier, there are many static data structure problems for which we can show a query time lower bound of $\Omega(\log|Q|)$ when the space

is linear in the number of bits required to store the data, where Q denotes the query set. However, there is a big gap to the best know upper bounds for some of these problems. The open question is to exhibit an explicit query set Q such that the query time is least $\omega(\log |Q|)$, when the space is linear in the number of bits required to store the data . This question is also open for linear data structures; studying this question for linear data structures has many advantages in that it is mathematically cleaner and simpler, and techniques from linear algebra and algebra come in handy.

Balancing Set Families. We proved that for arbitrary n, $n/2 - o(1) \leq \mathsf{B}(n) \leq n/2 - 1$. It is an intriguing open question to bridge the gap between the upper bound and lower bound.

Other Open Questions. We end this thesis by discussing three open questions related to linear data structure lower bounds, some of which are small steps towards resolving the major open question mentioned earlier (the first two questions are joint work with Rao).

(a) The first open question has a geometric flavor. For sets $S, T \subset \mathbb{R}^2$, we say that T *line spans* S if all points in S lie on the lines generated by every pair of points in T. The goal is to understand the relationship between $|S|$ and $|T|$ when T line spans S. As an example, if all points in S lie on a circle, then it is not hard to show that $|T| > \sqrt{|S|}$. The question is to find an explicit finite set S such that for any T that line spans S, it must be the case that $|T| \geq \omega\left(|S|^{1/2}\right)$. This question amounts to proving a lower bound on the space of linear data structures when the query time is 2. Hrubeš [55] proved the existence of such sets using algebraically independent numbers, but it is not clear if his construction can be made explicit. Dvir [35] showed that such sets exist using a dimension counting argument, but the set is not explicit.

(b) The second question concerns distances in Cayley graphs. Let $Q \subseteq \mathbb{F}_2^n$. Consider a Calyey graph over $\mathbb{F}_2^n$ with generator set $S \subset \mathbb{F}_2^n$: the vertex set is $\mathbb{F}_2^n$ and there is an edge between two vertices u and u' if $u + v = u'$ for some $v \in S$. To see how

Cayley graphs can help us understand linear data structures, consider a data structure for Q with space s and query time t, which stores $\langle v_1, x\rangle, \ldots, \langle v_s, x\rangle$ for input data x. Then, the Cayley graph with the generator set $S = \{v_1, \ldots, v_s\}$ has the property that $\max_{q\in Q} d_S(q) \leq t$, where $d_S(q)$ is the shortest path distance of q from the all zeros vector $(0, 0, \ldots, 0)$. Indeed, the data structure promises that for every $q \in Q$, there are at most t indices $i_1, \ldots, i_k$ such that $q = v_{i_1} + \ldots + v_{i_k}$. This linear combination corresponds to a path in the Cayley graph of length k, which is at most t. The linear data structure lower bound question translates to finding an explicit set $Q \subseteq \mathbb{F}_2^n$ such that for every $S \subseteq \mathbb{F}_2^n$ of size $O(n)$, the Cayley graph with generator set S must have $\max_{q\in Q} d_S(q) \geq \omega(\log |Q|)$. This opens up a new and interesting line of attack via spectral techniques, especially because the eigenvectors are given by the Fourier basis.

(c) The final question is on the *Matrix-Vector* problem in the static setting. In the Matrix-Vector problem, the data structure must store a matrix $M \in \mathbb{F}_2^{\sqrt{n}\times\sqrt{n}}$ using s bits that are linear functions of M and provide a query algorithm to compute Mv on the query $v \in \mathbb{F}_2^{\sqrt{n}}$. If t is the query time of any such data structure and $s = O(n)$, we then know that $t \geq \Omega(\sqrt{n}) = \Omega(\log |Q|)$, where Q denotes the query set; this easily follows from the fact that the output of the query algorithm is a $\sqrt{n}$ length vector. The Matrix-Vector problem is a good candidate for proving a query time lower bound of $\omega(\log |Q|)$ because the output Mv resembles a direct sum and any new lower bound for the Vector-Matrix-Vector problem also implies a new lower bound for the Matrix-Vector problem.

BIBLIOGRAPHY

[1] Farid Ablayev. Lower Bounds for One-Way Probabilistic Communication Complexity. In Andrzej Lingas, Rolf Karlsson, and Svante Carlsson, editors, *ICALP*, volume 700 of *LNCS*, pages 241–252. Springer-Verlag, 1993.

[2] Miklós Ajtai. A Lower Bound for Finding Predecessors in Yao's Cell Probe Model. *Combinatorica*, 8(3), 1988.

[3] Miklós Ajtai, János Komlós, and Endre Szemerédi. Sorting in c logn Parallel Steps. *Combinatorica*, 3(1):1–19, Mar 1983.

[4] Eric Allender and Michal Koucký. Amplifying Lower Bounds by Means of Self-Reducibility. *Journal of the ACM*, 57(3):1–36, Mar 2010.

[5] Noga Alon. Personal communication, 2019.

[6] Noga Alon, Ernest E. Bergmann, Don Coppersmith, and Andrew M. Odlyzko. Balancing Sets of Vectors. *IEEE Transactions on Information Theory*, 34(1):128–130, 1988.

[7] Noga Alon and Ravi B. Boppana. The Monotone Circuit Complexity of Boolean Functions. *Combinatorica*, 7(1):1–22, 1987.

[8] Noga Alon and Gil Cohen. On Rigid Matrices and U-Polynomials. *Computational Complexity*, 24(4):851–879, Dec 2015.

[9] Noga Alon and Uriel Feige. On The Power of Two, Three and Four Probes. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009*. SIAM, 2009.

[10] Noga Alon, Mrinal Kumar, and Ben Lee Volk. Unbalancing Sets and an Almost Quadratic Lower Bound for Syntactically Multilinear Arithmetic Circuits. In Rocco A. Servedio, editor, *CCC*, volume 102 of *LIPIcs*, pages 11:1–11:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[11] Noga Alon, Rina Panigrahy, and Sergey Yekhanin. Deterministic Approximation Algorithms for the Nearest Codeword Problem. In *APPROX '09 / RANDOM '09*, pages 339–351, 2009.

[12] Kazuyuki Amano. Depth Two Majority Circuits for Majority and List Expanders. In Igor Potapov, Paul Spirakis, and James Worrell, editors, *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS 2018)*, volume 117, pages 81:1–81:13, 2018.

[13] Kazuyuki Amano and Masafumi Yoshida. Depth Two (n-2)-Majority Circuits for n-Majority. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E101.A(9):1543–1545, 2018.

[14] Alexandr Andoni, Piotr Indyk, and Mihai Pătraşcu. On the Optimality of the Dimensionality Reduction Method. In *Proc. 47th IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 449–458, 2006.

[15] Vladimir L. Arlazarov, E. A. Dinic, M. A. Kronrod, and I. A. Faradzev. On Economical Construction of the Transitive Closure of a Directed Graph. *Soviet Math. Dokl.*, 11:1209–1210, 1970.

[16] Roger C. Baker, Glyn Harman, and János Pintz. The Difference Between Consecutive Primes, II. *Proceedings of the London Mathematical Society*, 83(3):532–562, 2001.

[17] Bar-Yehuda, Chor, Kushilevitz, and Orlitsky. Privacy, Additional Information, and Communication. *IEEETIT: IEEE Transactions on Information Theory*, 39, 1993.

[18] Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to Compress Interactive Communication. In *Proceedings of the 2010 ACM International Symposium on Theory of Computing*, pages 67–76, 2010.

[19] Paul Beame and Faith E. Fich. Optimal Bounds for the Predecessor Problem and Related Problems. *JCSS: Journal of Computer and System Sciences*, 65, 2002.

[20] Paul Beame, Vincent Liew, and Mihai Pătraşcu. Finding the Median (Obliviously) with Bounded Space. In *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, pages 103–115, 2015.

[21] Joseph Boninger, Joshua Brody, and Owen Kephart. Non-Adaptive Data Structure Bounds for Dynamic Predecessor. In *37th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2017, December 11-15, 2017, Kanpur, India*, pages 20:1–20:12, 2017.

[22] Allan Borodin, Rafail Ostrovsky, and Yuval Rabani. Lower Bounds for High Dimensional Nearest Neighbor Search and Related Problems. In Jeffrey Scott Vitter,

Lawrence L. Larmore, and Frank Thomson Leighton, editors, *STOC*, pages 312–321. ACM, 1999.

[23] Mark Braverman. Interactive information complexity. In Howard J. Karloff and Toniann Pitassi, editors, *STOC*, pages 505–524. ACM, 2012.

[24] Mark Braverman and Anup Rao. Information Equals Amortized Communication. In Rafail Ostrovsky, editor, *FOCS*, pages 748–757. IEEE, 2011.

[25] Mark Braverman and Omri Weinstein. A Discrepancy Lower Bound for Information Complexity. In Anupam Gupta, Klaus Jansen, José D. P. Rolim, and Rocco A. Servedio, editors, *APPROX-RANDOM*, volume 7408 of *Lecture Notes in Computer Science*, pages 459–470. Springer, 2012.

[26] Gerth Stølting Brodal, Shiva Chaudhuri, and Jaikumar Radhakrishnan. The Randomized Complexity of Maintaining the Minimum. In *SWAT: Scandinavian Workshop on Algorithm Theory*, 1996.

[27] Joshua Brody and Kasper Green Larsen. Adapt or Die: Polynomial Lower Bounds for Non-Adaptive Dynamic Data Structures. *Theory of Computing*, 11(19):471–489, 2015.

[28] Amit Chakrabarti, T. S. Jayram, and Mihai Pătraşcu. Tight Lower Bounds for Selection in Randomly Ordered Streams. In *Proc. 19th Symp. on Discrete Algorithms (SODA)*, pages 720–729. ACM/SIAM, 2008.

[29] Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Yao. Informational Complexity and the Direct Sum Problem for Simultaneous Message Complexity. In *FOCS*, pages 270–278, 2001.

[30] Diptarka Chakraborty, Lior Kamma, and Kasper Green Larsen. Tight Cell Probe Bounds for Succinct Boolean Matrix-Vector Multiplication. In *Proc. 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1297–1306, 2018.

[31] Timothy M. Chan. Comparison-Based Time-Space Lower Bounds for Selection. *ACM Trans. Algorithms*, 6(2), 2010.

[32] Arkadev Chattopadhyay, Michal Koucký, Bruno Loff, and Sagnik Mukhopadhyay. Simulation Beats Richness: New Data-Structure Lower Bounds. In *Proc. 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1013–1020, 2018.

[33] Raphaël Clifford, Allan Grønlund, and Kasper Green Larsen. New Unconditional Hardness Results for Dynamic and Online Problems. In *IEEE 56th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1089–1107, 2015.

[34] Henry Corrigan-Gibbs and Dmitry Kogan. The Function-Inversion Problem: Barriers and Opportunities. In Dennis Hofheinz and Alon Rosen, editors, *Theory of Cryptography*, pages 393–421, Cham, 2019. Springer International Publishing.

[35] Zeev Dvir. Personal communication, 2017.

[36] Zeev Dvir, Alexander Golovnev, and Omri Weinstein. Static Data Structure Lower Bounds Imply Rigidity. In *Proc. 51st Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 967–978, 2019. see https://arxiv.org/abs/1811.02725v3.

[37] Christian Engels, Mohit Garg, Kazuhisa Makino, and Anup Rao. On Expressing Majority as a Majority of Majorities. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 24, page 174, 2017.

[38] Hikoe Enomoto, Peter Frankl, Noboru Ito, and Kazumasa Nomura. Codes with Given Distances. *Graphs and Combinatorics*, 3(1):25–38, 1987.

[39] David Eppstein and Daniel S. Hirschberg. From Discrepancy to Majority. *Algorithmica*, 80(4):1278–1297, 2018.

[40] Paul Erdős and Richard Rado. Intersection Theorems for Systems of Sets. *Journal of London Mathematical Society*, 35:85–90, 1960.

[41] Gudmund Skovbjerg Frandsen, Peter Bro Miltersen, and Sven Skyum. Dynamic Word Problems. *J. ACM*, 44(2):257–271, 1997.

[42] Peter Frankl and Vojtěch Rödl. Forbidden Intersections. *Transactions of the American Mathematical Society*, 300(1):259–286, 1987.

[43] Michael Fredman and Michael Saks. The Cell Probe Complexity of Dynamic Data Structures. In *ACM Symposium on Theory of Computing (STOC)*, 1989.

[44] Michael L. Fredman. A Lower Bound on the Complexity of Orthogonal Range Queries. *J. ACM*, 28(4):696–705, 1981.

[45] Michael L. Fredman. Lower Bounds on the Complexity of Some Optimal Data Structures. *SIAM Journal on Computing*, 10(1):1–10, February 1981.

[46] Michael L. Fredman and Dan E. Willard. Surpassing the Information Theoretic Bound with Fusion Trees. *JCSS: Journal of Computer and System Sciences*, 47, 1993.

[47] Anna Gál and Peter Bro Miltersen. The Cell Probe Complexity of Succinct Data Structures. *Theoretical Computer Science*, 379(3):405–417, 2007.

[48] Anat Ganor, Gillat Kol, and Ran Raz. Exponential Separation of Information and Communication. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 176–185, 2014.

[49] Anat Ganor, Gillat Kol, and Ran Raz. Exponential Separation of Information and Communication for Boolean Functions. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 557–566, 2015.

[50] Mohit Garg and Jaikumar Radhakrishnan. Set Membership with Non-Adaptive Bit Probes. In *34th Symposium on Theoretical Aspects of Computer Science, STACS 2017, March 8-11, 2017, Hannover, Germany*, pages 38:1–38:13, 2017.

[51] Kristoffer Arnsfelt Hansen and Vladimir V. Podolskii. Exact Threshold Circuits. In *2010 25th Annual IEEE Conference on Computational Complexity*, pages 270–279. IEEE, 2010.

[52] Johan Håstad, Guillaume Lagarde, and Joseph Swernofsky. d-Galvin Families. *arxiv:1901.02652*, 2019.

[53] Gábor Hegedűs. Balancing Sets of Vectors. *Studia Scientiarum Mathematicarum Hungarica*, 47(3):333–349, 2009.

[54] Monika Henzinger, Sebastian Krinninger, Danupon Nanongkai, and Thatchaphol Saranurak. Unifying and Strengthening Hardness for Dynamic Problems via the Online Matrix-Vector Multiplication Conjecture. In *Proc. 47th Annual ACM Symposium on Theory of Computing (STOC)*, pages 21–30, 2015.

[55] Pavel Hrubeš. Personal communication, 2017.

[56] Pavel Hrubeš, Sivaramakrishnan Natarajan Ramamoorthy, Anup Rao, and Amir Yehudayoff. Lower Bounds on Balancing Sets and Depth-2 Threshold Circuits. In *ICALP*, volume 132. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2019.

[57] Maurice J. Jansen. Lower Bounds for Syntactically Multilinear Algebraic Branching Programs. In *International Symposium on Mathematical Foundations of Computer Science*, pages 407–418, 2008.

[58] T. S. Jayram, Subhash Khot, Ravi Kumar, and Yuval Rabani. Cell-Probe Lower Bounds for the Partial Match Problem. *J. Comput. Syst. Sci*, 69(3), 2004.

[59] Stasys Jukna and Georg Schnitger. Min-Rank Conjecture for Log-Depth Circuits. *Journal of Computer and System Sciences*, 77(6):1023–1038, 2011.

[60] Jonathan Katz and Luca Trevisan. On the Efficiency of Local Decoding Procedures for Error-Correcting Codes. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2000.

[61] Iordanis Kerenidis, Sophie Laplante, Virginie Lerays, Jérémie Roland, and David Xiao. Lower Bounds on Information Complexity via Zero-Communication Protocols and Applications. In *FOCS*, pages 500–509. IEEE Computer Society, 2012.

[62] Gillat Kol. Interactive Compression for Product Distributions. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 987–998. ACM, 2016.

[63] Alexander S. Kulikov and Vladimir V. Podolskii. Computing Majority by Constant Depth Majority Circuits with Low Fan-In Gates. *arXiv:1610.02686*, 2016.

[64] Eyal Kushilevitz and Noam Nisan. *Communication Complexity*. Cambridge University Press, Cambridge, 1997.

[65] Kasper Green Larsen. Higher Cell Probe Lower Bounds for Evaluating Polynomials. In *FOCS*, pages 293–301. IEEE Computer Society, 2012.

[66] Kasper Green Larsen. The Cell Probe Complexity of Dynamic Range Counting. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 85–94, 2012.

[67] Kasper Green Larsen, Omri Weinstein, and Huacheng Yu. Crossing the Logarithmic Barrier for Dynamic Boolean Data Structure Lower Bounds. In *Proc. 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 978–989, 2018.

[68] Kasper Green Larsen and Ryan Williams. Faster Online Matrix-Vector Multiplication. In *Proc. 28th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2182–2189, 2017.

[69] László Lovász. Kneser's Conjecture, Chromatic Number, and Homotopy. *Journal of Combinatorial Theory, Series A*, 25(3):319 – 324, 1978.

[70] Peter Bro Miltersen. Lower Bounds for Union-Split-Find Related Problems on Random Access Machines. In *Proceedings of the 26th Annual Symposium on the Theory of Computing*, pages 625–634, New York, May 1994. ACM Press.

[71] Peter Bro Miltersen. Cell Probe Complexity — A Survey. In V. Raman, C. Pandu Rangan, and R. Ramanujam, editors, *Foundations of Software Technology and Theoretical Computer Science: 19th Conference, Chennai, India, December 13–15, 1999: Proceedings*, volume 1738 of *Lecture Notes in Computer Science*, pub-SV:adr, 1999. Springer-Verlag.

[72] Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On Data Structures and Asymmetric Communication Complexity. *Journal of Computer and System Sciences*, 57(1):37–49, 1998.

[73] J. Ian Munro and Venkatesh Raman. Selection from Read-Only Memory and Sorting with Minimum Data Movement. *TCS: Theoretical Computer Science*, 165, 1996.

[74] Sivaramakrishnan Natarajan Ramamoorthy and Anup Rao. How to Compress Asymmetric Communication. In *Conference on Computational Complexity*, volume 33. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015.

[75] Sivaramakrishnan Natarajan Ramamoorthy and Anup Rao. Lower Bounds on Non-Adaptive Data Structures Maintaining Sets of Numbers, from Sunflowers. In *Conference on Computational Complexity*. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[76] Sivaramakrishnan Natarajan Ramamoorthy and Cyrus Rashtchian. Equivalence of Systematic Linear Data Structures and Matrix Rigidity. In Thomas Vidick, editor, *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*, volume 151 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 35:1–35:20, Dagstuhl, Germany, 2020. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[77] Rina Panigrahy, Kunal Talwar, and Udi Wieder. Lower Bounds on Near Neighbor Search via Metric Expansion. In *Proc. 51st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 805–814, 2010.

[78] Mihai Pătraşcu. Lower Bounds for 2-Dimensional Range Counting. In *STOC: ACM Symposium on Theory of Computing (STOC)*. ACM, 2007.

[79] Mihai Pătraşcu. Towards Polynomial Lower Bounds for Dynamic Problems. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 603–610. ACM, 2010.

[80] Mihai Pătraşcu. Unifying the Landscape of Cell-Probe Lower Bounds. *SIAM Journal on Computing*, 40(3):827–847, 2011. See also FOCS'08, arXiv:1010.3783.

[81] Mihai Pătraşcu and Erik D. Demaine. Logarithmic Lower Bounds in the Cell-Probe Model. *SIAM Journal on Computing*, 35(4):932–963, 2006. See also STOC'04, SODA'04.

[82] Mihai Pătraşcu and Mikkel Thorup. Time-Space Trade-Offs for Predecessor Search. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 232–240, 2006.

[83] Mihai Pătraşcu and Mikkel Thorup. Higher Lower Bounds for Near-Neighbor and Further Rich Problems. *SIAM Journal on Computing*, 39(2):730–741, 2010. See also FOCS'06.

[84] Mihai Pătraşcu and Mikkel Thorup. Dynamic Integer Sets with Optimal Rank, Select, and Predecessor Search. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 166–175, 2014.

[85] Ramamohan Paturi and Pavel Pudlák. Circuit Lower Bounds and Linear Codes. *Journal of Mathematical Sciences*, 134(5):2425–2434, 2006.

[86] Gleb Posobin. Computing Majority with Low-Fan-In Majority Queries. *arXiv:1711.10176*, 2017.

[87] Pavel Pudlák, Vojtech Rödl, and Jirí Sgall. Boolean Circuits, Tensor Ranks, and Communication Complexity. *SIAM Journal on Computing*, 26(3):605–633, 1997.

[88] Anup Rao and Makrand Sinha. Simplified Separation of Information and Communication. *Theory of Computing*, 14(1):1–29, 2018.

[89] Anup Rao and Amir Yehudayoff. *Communication Complexity and Applications*. Cambridge University Press, 2020.

[90] Ran Raz, Amir Shpilka, and Amir Yehudayoff. A Lower Bound for the Size of Syntactically Multilinear Arithmetic Circuits. *SIAM Journal on Computing*, 38(4):1624–1647, 2008.

[91] Mert Sağlam. Private communication, 2014.

[92] Michael E. Saks and Xiaodong Sun. Space Lower Bounds for Distance Approximation in the Data Stream Model. In *STOC*, pages 360–369. ACM, 2002.

[93] Pranab Sen and Srinivasan Venkatesh. Lower Bounds for Predecessor Searching in the Cell Probe Model. *J. Comput. Syst. Sci*, 74(3):364–385, 2008.

[94] Claude E. Shannon. A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[95] Alexander A. Sherstov. Compressing Interactive Communication Under Product Distributions. *SIAM J. Comput*, 47(2):367–419, 2018.

[96] Srikanth Srinivasan. Personal communication, 2018.

[97] Leslie G. Valiant. Graph-Theoretic Arguments in Low-Level Complexity. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science*, pages 162–176, 1977.

[98] Leslie G. Valiant. Short Monotone Formulae for the Majority Function. *Journal of Algorithms*, 5(3):363 – 366, 1984.

[99] Leslie G. Valiant. Why Is Boolean Complexity Theory Difficult? In *Poceedings of the London Mathematical Society Symposium on Boolean Function Complexity*, pages 84–94, New York, NY, USA, 1992. Cambridge University Press.

[100] Peter van Emde Boas. Preserving Order in a Forest in less than Logarithmic Time and Linear Space. *Information Processing Letters*, 6(3):80–82, June 1977.

[101] Emanuele Viola. Lower Bounds for Data Structures with Space Close to Maximum Imply Circuit Lower Bounds. *Electronic Colloquium on Computational Complexity (ECCC)*, 25:186, 2018.

[102] Omri Weinstein and Huacheng Yu. Amortized Dynamic Cell-Probe Lower Bounds from Four-Party Communication. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 305–314, 2016.

[103] Dan E. Willard. Log-Logarithmic Worst-Case Range Queries are Possible in Space $\Theta(N)$. *Information Processing Letters*, pages 81–84, 1983.

[104] Richard Ryan Williams. Limits on Representing Boolean Functions by Linear Combinations of Simple Functions: Thresholds, ReLUs, and Low-Degree Polynomials. In Rocco A. Servedio, editor, *CCC*, volume 102 of *LIPIcs*, pages 6:1–6:24. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2018.

[105] Andrew Yao. Some Complexity Questions related to Distributive Computing. In *STOC*, pages 209–213, 1979.

[106] Andrew Yao. Should Tables be Sorted? *JACM: Journal of the ACM*, 28, 1981.

[107] Andrew C. Yao. On the Complexity of Maintaining Partial Sums. *SIAM Journal on Computing*, 14(2):277–288, May 1985.

[108] Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity. *18th Annual Symposium on Foundations of Computer Science (sfcs 1977)*, pages 222–227, 1977.

[109] Huacheng Yu. Cell-Probe Lower Bounds for Dynamic Problems via a New Communication Model. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 362–374. ACM, 2016.