The Pennsylvania State University
The Graduate School

EFFICIENT COMBINATORIAL METHODS IN SPARSIFICATION, SUMMARIZATION AND TESTING OF LARGE DATASETS

A Dissertation in
Computer Science and Engineering
by
Grigory Yaroslavtsev

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2014

The dissertation of Grigory Yaroslavtsev was reviewed and approved* by the following:

Sofya Raskhodnikova
Associate Professor of Computer Science and Engineering
Chair of Committee and Dissertation Advisor

Piotr Berman
Associate Professor of Computer Science and Engineering

Jason Morton
Assistant Professor of Mathematics and Statistics

Adam D. Smith
Associate Professor of Computer Science and Engineering

Raj Acharya
Head of Department of Computer Science and Engineering

*Signatures are on file in the Graduate School.

Abstract

Increasingly large amounts of structured data are being collected by personal computers, mobile devices, personal gadgets, sensors, etc., and stored in data centers operated by the government and private companies. Processing of such data to extract key information is one of the main challenges faced by computer scientists. Developing methods for constructing compact representations of large data is a natural way to approach this challenge.

This thesis is focused on rigorous mathematical and algorithmic solutions for sparsification and summarization of large amounts of information using discrete combinatorial methods. Areas of mathematics most closely related to it are graph theory, information theory and analysis of real-valued functions over discrete domains. These areas, somewhat surprisingly, turn out to be related when viewed through the computational lens. In this thesis we illustrate the power and limitations of methods for constructing small representations of large data sets, such as graphs and databases, using a variety methods drawn from these areas.

The primary goal of sparsification, summarization and sketching methods discussed here is to remove redundancy in distributed systems, reduce storage space and save other resources by compressing the representation, while preserving the key structural properties of the original data or system. For example, the size of the network can be reduced by removing redundant nodes and links, while (approximately) preserving the distances or connectivities between the nodes.

This thesis serves as an overview of results in [31, 30, 152, 38, 171, 33]. It also contains an extension of results in [152] obtained jointly with David Woodruff.

Table of Contents

List of Figures

List of Tables

Acknowledgments

First and foremost, I would like to thank my advisor Sofya Raskhodnikova for the three exciting years of joint collaboration. During these years Sofya has been both an invaluable mentor and a helpful colleague, while also granting me an unlimited freedom to pursue my research interests and continuous support at different stages of my PhD. For all this I am forever grateful. I would also like to thank Piotr Berman for our fruitful collaboration and his availability to discuss all sorts of algorithmic questions any time. I am grateful to Adam Smith for research advice and our collaboration on privacy projects not included in this thesis. I am also very thankful to Sofya, Piotr and Adam for bringing many great parts of the research culture from MIT to the Theory group at Penn State, which made my time there extremely enjoyable and productive.

I would like to thank my co-authors on the papers included into this thesis – Piotr Berman, Arnab Bhattacharyya, Eric Blais, Elena Grigorescu, Konstantin Makarychev, Marco Molinaro, Sofya Raskhodnikova and David Woodruff. During my PhD I was lucky to work with many other co-authors, whom I would like to thank for their hard work – Alexandr Andoni, Joshua Brody, Amit Chakrabarti, Ranganath Kondapally, Aleksandar Nikolov, Krzysztof Onak, Cecilia M. Procopiuc, Adam Smith, Divesh Srivastava and Anthony Wirth.

Roughly a half of my PhD was spent in research labs and I am particularly grateful to my mentors during the time there – Graham Cormode, Howard Karloff, Cecilia M. Procopiuc and Divesh Srivastava (AT&T Labs – Research, Summer 2011), David Woodruff (IBM Research, Almaden, Summer 2012), Alex Andoni (Microsoft Research, Silicon Valley, Fall 2012) and Konstantin Makarychev (Microsoft Research, Redmond, Summer 2013). These internships have been tremendously helpful in extending my expertise in the design and analysis of algorithms. In particular, the strong direct sum results in Chapter 7 were obtained at IBM, Almaden (jointly with Marco Molinaro and David Woodruff) and the bounded-round protocols for the set interstion problem in Chapter 8 were obtained there as well (jointly with David Woodruff). These research internships also resulted in lasting collaborations with members of the labs, which I maintain until the present day.

Discussions with multiple people have contributed to some parts of this thesis – Paul Beame, Vitaly Feldman, Nick Harvey, Kevin Matulef, Ryan O'Donnell, Lev Reyzin, Rocco Servedio, C. Seshadhri and Jan Vondrak.

I would like to thank the faculty members of the Theory group at Penn State for multiple useful discussions – Martin Furer, Sean Hallgren and Paul Medvedev. I would also like to thank my office mates at the Theory lab for the fun times there – Ishan Behoora, Kashyap Dixit, Abhradeep Guha Thakurta, Megan Heysham, Madhav Jha, Ngai Lam Ho, Meiram Mirzabulatov, Sandeep Narayanaswami, Fang Song, Youngtae Youn, Huiwen Yu and Ye Zhang.

During all three years of working on this thesis I was supported by the College of Engineering Fellowship and partially by University Graduate Fellowship at Penn State. I am deeply grateful for this support which made it possible for me to concentrate fully on my research pursuits.

Finally, I would like to thank the committee members Adam Smith, Piotr Berman and Jason Morton for their comments on this manuscript.

Research included into this thesis wouldn't be possible without the generous support provided by the NSF/CCF CAREER award 0845701.

To my mother.

Chapter 1

Introduction

1.1 Organization

This thesis consists of three parts, covering different aspects of our study of compact representations.

Sparsification. In Part I we discuss design of efficient algorithms for structure-preserving network design. Sparsification is used to speed up algorithms and reduce communication overheads. It allows one to cut costs and optimize performance in networks. We focus on algorithms for sparsification of networks by removing extra features such as redundant links and nodes while (approximately) preserving their connectivity properties and distances.

This part consists of two chapters. In Chapter 2 we discuss approximate sparsification of directed graphs. We present approximation algorithms for constructing graph spanners in directed graphs and consider several related problems. This part is based on [31]. In Chapter 3 we discuss approximate sparsification of planar graphs. This part is based on [33].

Property Testing. In Part II we discuss algorithmic techniques for testing structural properties of the data. Property testing can be informally described as an extremely efficient algorithmic way to do a sanity check. It allows one to test whether data satisfies a certain structural property by performing an analysis of a carefully chosen small sample. Very efficient algorithms for selection and analysis of such a sample are the main subject of studies in this area. Because of their efficiency such algorithms are natural candidates for applications in fast decision making and for selection of further algorithmic solution for data analysis. For example, by looking at a very small portion of the data it is possible to make an approximate decision about whether it is sorted and analyze other related properties.

This part consists of two chapters. In Chapter 5 we discuss relationship between property testing algorithms and graph spanners introduced in Chapter 2. We show almost optimal lower bounds on a certain type of spanners for the hypergrid domains. In particular, these lower bounds rule out a specific approach to property testing of functions over hypergrids. This part is based on [30]. In Chapter 4 we discuss sparsification of submodular functions with discrete ranges. This part is based on [171].

Communication Compelxity. In Part III we discuss the rigorous theoretical framework of communication complexity. It allows to reason about efficiency of bandwidth utilization by algorithms, operating in a distributed setting. It is also one of the main techniques for proving lower bounds in the analysis of discrete algorithms in many diverse areas of computer science, ranging from data structures to circuit lower bounds. By giving reductions to their communication counterparts one can show information-theoretic lower bounds for the problems of interest.

In this thesis we use communication complexity as a tool for lower bounds in applications to sketching and property testing. These applications motivate advancements in the field of communication complexity in new directions. In Chapter 6 we show reductions from communication complexity to a number of property testing problems, involving functions over hypergrids. The properties we consider are monotonicity, the Lipschitz property and convexity. This chapter is based on [38]. In Chapter 7 we give a new theorem in communication complexity, which falls into a class of direct sum results. Informally, our results show that for a certain class of communication problems solving multiple instances simultaneously is as hard as solving every individual instance with high probability. We also explain how our direct sum result implies optimum lower bounds for a large number of sketching techniques. This chapter is based on [152]. Finally, in Chapter 8 we give almost optimal protocols for the problem of finding an intersection between two distributed databases in bounded number of rounds of communication. This chapter is closely related to Chapter 7 because our lower bound for the set intersection problem follows from the results developed there. This chapter is based on joint work with David Woodruff, which was done while the author was an intern at IBM Research, Almaden.

1.2 Overview

In this section we give an overview of the technical results presented in this thesis and place them into the context of the previous work.

Part I. Approximate Sparse Network Design.

Sparsification of undirected graphs is one of the most beautiful parts of theoretical computer science, which ties together deep mathematics with applications to the analysis of large graphs. Branches of mathematics, such as spectral graph theory and metric embeddings, give us powerful tools to address multiple problems, such as graph partitioning, sparsest cut and others. For multiple specific variants of sparsification strongly positive results are known (spanners, cut and vertex sparsifiers, mimicking networks, distance-preserving minors, etc.).

For directed graphs the situation is much more challenging: there is no consistent spectral graph theory for directed graphs, metric embeddings give only limited results. Under most of the specific definitions mentioned above no (non-trivial) sparsifiers exist (or are known) for directed graphs. My research is focused on methods in graph sparsification, which are applicable to directed graphs by design. Using the examples below I will illustrate that the state of the art approximation algorithms for many problems in sparse network design in directed graphs (as compared to undirected graphs) still have a very combinatorial flavor.

A k-spanner of a graph is a subset of edges which preserves distances in the original graph up to a factor of k. A Steiner forest of a graph is a subset of edges which preserves connectivity between designated source-sink pairs of terminals. In Chapter 2 we present approximation algorithms for minimizing the size of spanners and Steiner forests in directed graphs. Following our work, an efficient implementation of our techniques was shown to give promising experimental results on real instances of the iBGP overlay design problem by Dinitz and Wilfong [72]. Our worst-case approximation guarantee was complemented by hardness results shown by Dinitz, Kortsarz and Raz [70]. Although our algorithms are designed for directed graphs, the algorithm for 3-spanners improves over the classic result of Althöfer, Das, Dobkin, Joseph and Soares [11], who give an approximation algorithm for undirected graphs.

In Chapter 3 we study sparsification of planar graphs, giving approximation algorithms for a set of basic node-weighted network design problems, including Feedback Vertex Set, Bipartization, Subset Feedback Vertex Set, Directed Feedback Vertex Set and Node-Weighted Steiner Forest. We give analysis of the approximation ratio achieved by classic primal-dual algorithms of Goemans and Williamson [102] and justify their successful experimental performance in applications to VLSI design observed by Kahng, Vaya and Zelikovsky [130]. We also show how to obtain better approximation by using more sophisticated oracles with the primal-dual framework of [102].

Part II. Concise Representations of Real Functions in Property Testing.

In Chapter 5 we discuss sparsification of posets (transitively closed directed acyclic graphs). Such sparsification can be used for construction of efficient access control hierarchies. It is known [12, 182] that posets corresponding to such hierarchies can be represented by a small number of attributes and thus can be embedded into low-dimensional hypergrids in an order-preserving manner. This chapter also serves as a transition point from network design applications of sparsification to applications to property testing. Existence of a sparse spanner for the poset domain implies an efficient property testing algorithm for monotonicity [95]. In general, even an optimal spanner for a poset might be as dense as the original graph. We show that efficient and explicit sparsification by spanners is possible if one allows to introduce Steiner nodes. We also show that the dependence on the size of the poset and the dimension of the hosting hypergrid is almost optimal in our constructions.

In network design the role of sparsification is rather straightforward and can be seen as optimization of the design by removal of redundant elements. In the analysis of functions the role of sparsification is sometimes more implicit — existence of a concise representation implies certain structural properties, which can be used in the algorithm design without constructing the representation explicitly.

In Chapter 4 we focus on understanding concise representations of real-valued submodular functions with applications to learning theory and property testing. We study integer submodular functions $f\colon 2^X \to \{0,\dots,k\}$ (discrete analogs of convex functions over the Boolean hypercube). We show that such functions can be represented by formulas of bounded width. This lets us argue that they have non-trivial concentration in the Fourier spectrum and thus Kushilevitz-Mansour learning algorithm can be used to PAC-learn such functions with query complexity and running time polynomial in $|X|$.

Previous work on learning submodular functions gives mostly negative results [100, 111, 21, 55, 18] in the regime when the size of the domain $|X|$ is large. Our results can be seen as positive because our learning algorithms are exact (as compared to additive/multiplicative approximation) and much more efficient. Positive results obtained in the previous only apply to restricted subclasses of submodular functions (such as coverage functions) or under assumptions of discrete continuity (the Lipschitz condition). Mostly negative results about property testing submodular functions were obtained by Seshadhri and Vondrak [185] and positive results are only known for coverage functions [47]. Our work illustrates that the barriers established in the previous work can be overcome for submodular functions with discrete range.

Part III. Communication Complexity Methods in Summarization.

Communication complexity is a theoretical framework used to reason about efficiency of bandwidth utilization by algorithms operating in a distributed setting (see [143]). It is also one of the main techniques for proving lower bounds in the analysis of discrete algorithms in many diverse areas of computer science, ranging from data structures to circuit lower bounds. By giving reductions to their communication counterparts one can show information-theoretic lower bounds for the problems of interest. In Part III we use communication complexity as a tool for lower bounds in applications to sketching and property testing. These applications motivate advancements in the field of communication complexity in new directions.

In Chapter 6 we introduce strong, and in many cases optimal, lower bounds for the number of queries required to nonadaptively test three fundamental properties of functions $f\colon [n]^d \to \mathbb{R}$ on the hypergrid: monotonicity, convexity, and the Lipschitz property. Our lower bounds also apply to the more restricted setting of functions $f\colon [n] \to \mathbb{R}$ on the line (i.e., to hypergrids with $d = 1$), where they give optimal lower bounds for all three properties. The lower bound for testing convexity is the first lower bound for that property, and the lower bound for the Lipschitz property is new for tests with 2-sided error. The lower bounds are obtained via the connection to communication complexity established in [37]. These results are the first to apply this method to functions with non-hypercube domains. A key ingredient in this generalization is the set of Walsh functions, an orthonormal basis of the set of functions $f\colon [n]^d \to \mathbb{R}$.

In Chapter 7 we study the communication complexity counterpart of the problem of constructing a small sketch of multiple objects (vectors, matrices, databases). It is known in communication complexity as a direct sum theorem. It gives a lower bound on the complexity of solving multiple copies of the same communication problem simultaneously. We give an optimal such theorem for one-way communication for a natural class of communication problems (e.g., Equality and Augmented Indexing). Using the connection mentioned above, this implies that several sketching techniques (Johnson-Lindenstrauss transform, sketching matrix products, generation of mergeable summaries of databases) are optimal for sketching multiple objects simultaneously.

In Chapter 8 we study the communication complexity of computing the intersection of two datasets stored distributedly. The motivation for this problem comes from applications to computing joins in databases, Jaccard similarity index, rarity and related statistics. When the sizes of both sets are bounded (at most k) we show that the set intersection problem requires a different approach as compared to its well-studied decision version ("small-set-disjointness"). We present a family of protocols which achieves an almost optimal tradeoff between the total communication and the number of rounds. The optimality guarantee is shown by using an extension of our techniques developed in Chapter 7.

Part I

Approximate Sparse Network Design

Chapter 2

Approximate Sparsification of Directed Graphs

2.1 Introduction

A spanner of a graph is a sparse subgraph that approximately preserves distances in the original graph. This notion was first used by Awerbuch [15] and explicitly introduced by Peleg and Schäffer [165].

Definition 2.1.1 (k-spanner, [15, 165])**.** *Given a graph $G = (V, E)$ with nonnegative edge lengths $d : E \to \mathbb{R}^{\geq 0}$ and a real number $k \geq 1$, a subgraph $H = (V, E_H)$ is a k-**spanner** of G if for all edges $(s, t) \in E$, the graph H contains a path from s to t of length at most $k \cdot d(s, t)$. The parameter k is called the **stretch**.*

Spanners have numerous applications, such as efficient routing [62, 63, 167, 175, 190], simulating synchronized protocols in unsynchronized networks [166], parallel, distributed and streaming algorithms for approximating shortest paths [59, 60, 82, 92], algorithms for distance oracles [26, 191], property testing, property reconstruction and key management in access control hierarchies (see [35, 34, 128], the survey in [170] and references therein).

We study the computational problem of finding the sparsest k-spanner of a given *directed* graph G, that is, a k-spanner of G with the smallest number of edges. We refer to this problem as Directed k-Spanner and distinguish between the case of unit edge lengths (i.e., $d(e) = 1$ for all $e \in E$) and arbitrary edge lengths. The Undirected k-Spanner problem refers to the task of finding the sparsest k-spanner of a given undirected graph. The natural reduction from Undirected k-Spanner to Directed k-Spanner preserves the approximation ratio.

Our main results are an algorithm with approximation ratio $O(\sqrt{n} \log n)$ for Directed k-Spanner with arbitrary edge lengths and an algorithm with approximation ratio $O(\sqrt[3]{n} \log n)$ for Directed 3-Spanner with unit edge lengths, where n is the number of nodes in the input graph G. Our approximation guarantee for Directed 3-Spanner almost matches the integrality gap of $\Omega(n^{1/3-\epsilon})$ by Dinitz and Krauthgamer [71] for a natural linear programming relaxation of the problem. Our result also directly implies the same approximation ratio for the Undirected 3-Spanner problem with unit edge lengths.

Our techniques also apply to the Directed Steiner Forest problem. Our result for this problem is discussed in Section 2.1.3.

2.1.1 Relation to Previous Work

Directed k-Spanner with *unit edge lengths* has been extensively studied. Note that in this case, we can assume that k is a positive integer. For $k = 2$, the problem has been completely resolved: Kortsarz and Peleg [139] and Elkin and Peleg [84] gave an $O(\log n)$-approximation, and Kortsarz [138] proved that this approximation ratio cannot be improved unless P=NP. Elkin and Peleg [85] gave an $\tilde{O}(|E|^{1/3})$-approximation for Directed 3-Spanner, which is an $\tilde{O}(n^{2/3})$-approximation for dense graphs with $\Theta(n^2)$ edges. For general $k \geq 3$, Bhattacharyya *et al.* [35] presented an $\tilde{O}(n^{1-1/k})$-approximation; then Berman, Raskhodnikova and Ruan [32] improved it to $\tilde{O}(n^{1-1/\lceil k/2 \rceil})$, and recently Dinitz and Krauthgamer [71] gave $\tilde{O}(n^{2/3})$-approximation, presenting the first algorithm with approximation ratio independent of k. For the special cases of $k = 3$ and $k = 4$, Berman, Raskhodnikova and Ruan's algorithm gives an $\tilde{O}(\sqrt{n})$-approximation. Dinitz and Krauthgamer also gave an $\tilde{O}(\sqrt{n})$-approximation for the case $k = 3$, using different techniques than in [32]. Thus, our algorithms improve on [32] for all $k \geq 3$, where $k \neq 4$, and on [71] for all $k \geq 3$.

Dinitz and Krauthgamer's algorithms also work for Directed k-Spanner with arbitrary edge lengths. For this case, one can no longer assume that k is an integer. Dinitz and Krauthgamer achieved an $\tilde{O}(n^{2/3})$-approximation for all $k > 1$ and $\tilde{O}(\sqrt{n})$ for $k = 3$ for arbitrary edge lengths. We improve this approximation ratio to $\tilde{O}(\sqrt{n})$ for all $k > 1$.

In contrast to the directed case, a simple approximation algorithm for Undirected k-Spanner was known for decades. For all integer $k \geq 3$ and for all undirected graphs G with arbitrary edge lengths, a k-spanner can be constructed in polynomial time by a greedy algorithm proposed by Althofer, Das, Dobkin, Joseph and Soares [11]. It follows from the Moore bound for irregular graphs by Alon, Hoory and Linial [8] that the graph constructed by this greedy algorithm has $O(n^{1+\frac{1}{\lceil k/2 \rceil}})$ edges. Since a k-spanner of a connected graph must have at least $n - 1$ edges, an approximation ratio $O(n^{\frac{1}{\lceil k/2 \rceil}})$ follows. Our result improves the ratio for Undirected 3-Spanner from $O(\sqrt{n})$ to $\tilde{O}(n^{1/3})$ in the case of unit-length edges.

Elkin and Peleg [83, 86], improving on [138], showed that it is quasi-NP-hard to approximate Directed k-Spanner, even when restricted to unit edge lengths, with ratio better than $2^{\log^{1-\epsilon} n}$ for $k \in (3, n^{1-\delta})$ and all $\delta, \epsilon \in (0, 1)$. For Undirected k-Spanner with unit-length edges, such a strong hardness result does not hold since the problem is $O(1)$-approximable when $k = \Omega(\log n)$.

2.1.2 Our Techniques

Our algorithms operate by combining two graphs: the first obtained from randomized rounding of a fractional solution to a linear programming relaxation of the problem and the second obtained by growing shortest-path trees from randomly selected vertices. The idea of combining a linear programming approach with sampling of shortest-path trees to solve Directed k-Spanner first appeared in [35]. Dinitz and Krauthgamer [71] used the same approach in their main algorithm (for arbitrary stretch k), but with a novel, flow-based linear program (LP). In this paper, we propose alternative randomized LP rounding schemes that lead to better approximation ratios. Sampling and randomized rounding has been previously used by Kortsarz and Peleg [140] to construct undirected low-degree 2-spanners. In that work, the sampling step selects uniformly random edges, and the LP is different from ours.

We also give new LP relaxations of DIRECTED k-SPANNER, slightly simpler than that in [71], although they describe the same polytope. Our LP relaxation for the general case is stated in terms of *antispanners*, a graph object "dual" to spanners. An antispanner for an edge (s,t) is a set of edges whose removal from the graph destroys all paths of stretch at most k from s to t. Like in [71], our LP has a polynomial number of variables and an exponential number of constraints. We use the ellipsoid algorithm with a randomized separation oracle to solve it. In the case of unit edge lengths, we present a different LP that has an extra advantage: it has a polynomial number of constraints and thus can be solved quickly without using the ellipsoid algorithm. We apply two different rounding schemes to the fractional solution of this LP: one for general stretch, another for stretch $k = 3$.

We note, however, that our method would yield the same approximation ratios with the LP of Dinitz and Krauthgamer [71] and, in the case of 3-spanners for graphs with unit edge lengths, with their rounding method as well. Dinitz and Krauthgamer gave a separate algorithm for DIRECTED 3-SPANNER that uses randomized rounding, but does not combine it with sampling. By combining with sampling, we obtain an algorithm with better approximation ratio for the case of unit lengths. Our rounding method allows for simpler analysis.

2.1.3 Directed Steiner Forest

Finally, we apply our techniques to the DIRECTED STEINER FOREST (DSF) problem, a fundamental network design problem on directed graphs. In this problem, the input is a directed graph $G = (V,E)$ with edge costs and a collection $D \subseteq V \times V$ of vertex pairs. The goal is to find a minimum-cost subgraph of G that contains a path from s to t for every pair $(s,t) \in D$. DSF is an NP-hard problem and is known [76] to be quasi-NP-hard to approximate with ratio better than $2^{\log^{1-\epsilon} n}$ for all $\epsilon \in (0,1)$. DSF is also known [93] to be as hard as MAX-REP, a basic problem used for hardness reductions, for which the current best approximation ratio is $O(n^{1/3})$ [52].

Previous to this work, the best known approximation ratio for DSF, independent of the size of D, was $O(n^{\epsilon} \cdot \min(n^{4/5}, m^{2/3}))$ due to Feldman, Kortsarz and Nutov [93]. Their algorithm has the same structure as the algorithms for DIRECTED k-SPANNER in [35, 71]: it combines two graphs obtained, respectively, by sampling and solving an LP. In addition, the LP relaxation they formulate is closely related to that developed by Dinitz and Krauthgamer, with edge costs replaced by edge lengths. Our technique for the spanner problem also applies to the DSF problem, yielding an improved approximation ratio of $O(n^{2/3+\epsilon})$ for any fixed $\epsilon > 0$.

2.1.4 Organization

In Section 2.2, we explain the general outline of our algorithms, introduce antispanners and show how to find an $\tilde{O}(n^{1/2})$-approximate solution to Directed k-Spanner in polynomial time. In Section 2.3, we present a more efficient algorithm for the special case when all the edges of the graph are of unit length. In Section 2.4, we show the $\tilde{O}(n^{1/3})$-approximation for Directed 3-Spanner with unit-length edges. Finally, Section 2.5 describes the $O(n^{2/3+\epsilon})$-approximation for Directed Steiner Forest. In Section 2.6 we give a conclusion and directions for future work.

2.2 An $\tilde{O}(\sqrt{n})$-Approximation for Directed k-Spanner

Our first result is stated in the following theorem.

Theorem 2.2.1. *There is a polynomial time randomized algorithm for* Directed k-Spanner *with expected approximation ratio $O(\sqrt{n}\log n)$.*

All algorithms in this paper have the same structure. They break the problem into two parts and obtain separate solutions to each part: one by random sampling and the other by randomized rounding of a solution to a linear program. We start by explaining how we break Directed k-Spanner into two parts. In Section 2.2.1, we describe how to obtain a solution to the first part using random sampling. Section 2.2.2 describes our randomized rounding scheme for Directed k-Spanner. In Section 2.2.3, we introduce antispanners, a graph object used to formulate and analyze our linear programming relaxations. In Section 2.2.4, we formulate our linear programming relaxation and separation oracle, and finish the description and analysis of the algorithm, completing the proof of Theorem 2.2.1.

Let $G = (V, E)$ be a directed graph with edge lengths $d : E \to \mathbb{R}^{\geq 0}$, given as an input to our algorithm, and OPT be the size of its sparsest k-spanner. We assume that G is weakly connected. Otherwise, our algorithm should be executed for each weakly connected component separately.

Definition 2.2.1 (Local graph $G^{s,t}$)**.** *For an edge $(s,t) \in E$, let $G^{s,t} = (V^{s,t}, E^{s,t})$ be the subgraph of G induced by the vertices that belong to paths from s to t of length at most $k \cdot d(s,t)$.*

We classify edges according to the sizes of their local graphs.

Definition 2.2.2 (Thick & thin edges)**.** *Let β be a parameter in $[1, n]$. If $|V^{s,t}| \geq n/\beta$, the corresponding edge (s,t) is* thick*, and otherwise, it is* thin*. The set of all thin edges is denoted by $\mathcal{E}$. In Sections 2.2.1–2.3, we set $\beta = \sqrt{n}$ and in Section 2.4, $\beta = n^{1/3}$.*

Definition 2.2.3. *A set $E' \subseteq E$* settles *an edge $(s,t) \in E$ if (V, E') satisfies the k-spanner property for this edge, i.e., it contains a path of length at most $k \cdot d(s,t)$ from s to t.*

Our algorithm must find a small subset of edges that settles all edges in E. To accomplish this, it finds two subsets of edges, E' and E'', such that E' settles all thick edges and E'' settles all thin edges. The output of the algorithm is $(V, E' \cup E'')$.

2.2.1 Sampling

The following procedure uses random sampling to construct an edge set E' that settles all thick edges. Recall that an *in-arborescence is a directed rooted tree where all edges are oriented towards the root; an* out-arborescence is defined similarly.

Algorithm 2.1 Sample(β)

```
1: $E' \leftarrow \varnothing$, $S \leftarrow \varnothing$;
2: for $i = 1$ to $\beta \ln n$ do
3:     $v \leftarrow$ a uniformly random element of $V$;
4:     $T_v^{in} \leftarrow$ a shortest path in-arborescence rooted at $v$;
5:     $T_v^{out} \leftarrow$ a shortest path out-arborescence rooted at $v$;
6:     $E' \leftarrow E' \cup T_v^{in} \cup T_v^{out}$, $S \leftarrow S \cup \{v\}$;  //Set $S$ is used only in the analysis.
7: end for
8: Add all unsettled thick edges to $E'$;
9: return $E'$.
```

Lemma 2.2.2. *Algorithm 2.1, in polynomial time, computes a set E' that settles all thick edges and has expected size at most $3\beta \ln n \cdot OPT$.*

Proof. After the execution of the **for**-loop in Algorithm 2.1, $|E'| \leq 2(n-1)\beta \ln n \leq 2\beta \ln n \cdot OPT$. The last inequality holds because $OPT \geq n-1$ for weakly connected graphs G.

If some vertex v from a set $V^{s,t}$ appears in the set S of vertices selected by Sample, then T_v^{in} and T_v^{out} contain shortest paths from s to v and from v to t, respectively. Thus, both paths are contained in E'. Since $v \in V^{s,t}$, the sum of lengths of these two paths is at most $k \cdot d(s,t)$. Therefore, if $S \cap V^{s,t} \neq \varnothing$, then the edge (s,t) is settled. For a thick edge (s,t), the set $S \cap V^{s,t}$ is empty with probability at most $(1 - 1/\beta)^{\beta \ln n} \leq e^{-\ln n} = 1/n$. Thus, the expected number of unsettled thick edges added to E' in Step 8 of Sample is at most $|E|/n \leq n-1 \leq OPT$.

Step 8 ensures that the set E', returned by the algorithm, settles all thick edges. Computing shortest path in- and out-arborescences and determining whether an edge is thick can be done in polynomial time. □

2.2.2 Randomized Rounding

To obtain a set E'' that settles all thin edges, each of our algorithms solves a linear program and rounds the resulting fractional solution. The LP is a relaxation of Directed k-Spannerfor the set of all thin edges. It has a variable x_e and a constraint $x_e \geq 0$ for each edge $e \in E$. The variable x_e in the corresponding optimal {0,1}-solution indicates whether the edge e is present in the smallest spanner for all thin edges. The following randomized rounding procedure is used in our algorithms for Directed k-Spanner, both for arbitrary and for unit lengths. As an input it gets a fractional vector $\{\hat{x}_e\}$ with nonnegative entries.

Algorithm 2.2 RandomizedSelection($\hat{x}_e$)

```
1: $E'' \leftarrow \varnothing$;
2: **for** each edge $e \in E$ **do**
3:     Add $e$ to $E''$ with probability $\min(\sqrt{n}\ln n \cdot \hat{x}_e, 1)$;
4: **end for**
5: **return** $E''$.
```

The following proposition shows that if the sum of values assigned by $\{\hat{x}_e\}$ to edges in some $A \subseteq E$ is at least 1 then E'' intersects A with high probability.

Claim 2.2.3. *Let $A \subseteq E$. If Algorithm 2.2 receives a fractional vector $\{\hat{x}_e\}$ with nonnegative entries satisfying $\sum_{e\in A} \hat{x}_e \geq 1$, the probability that it outputs a set E'' disjoint from A is at most* $\exp(-\sqrt{n}\ln n)$.

Proof. If A contains an edge e, such that $\hat{x}_e \geq (\sqrt{n}\ln n)^{-1}$, then $e \in E''$ with probability 1. That is, E'' is never disjoint from A.

Otherwise, for all edges $e \in A$, the probability that $e \in E''$ is exactly $\sqrt{n}\ln n \cdot \hat{x}_e$. The probability that no edges of A are in E'' is, therefore,

$$\prod_{e\in A}(1-\sqrt{n}\ln n\cdot \hat{x}_e) \leq \exp\left(-\sum_{e\in A}\sqrt{n}\ln n \cdot \hat{x}_e\right) \leq \exp(-\sqrt{n}\ln n).$$

The first inequality above follows from the fact that $1-x \leq \exp(-x)$ for $x \geq 0$. The second one holds because $\sum_{e\in A}\hat{x}_e \geq 1$. □

2.2.3 Antispanners

In this section, we introduce antispanners, a graph object used in the description of our LP for Directed k-Spanner and crucial in the analysis of the parts of our algorithms that settle thin edges. After giving the definition, we show how to construct minimal antispanners (in Claim 2.2.4) and give an upper bound on their number (in Claim 2.2.5.)

For a given edge (s,t), we define an antispanner to be a subset of edges of G, such that if we remove this subset of edges from G, the length of the shortest path from s to t becomes larger than $k \cdot d(s,t)$.

Definition 2.2.4 (Antispanner)**.** *A set $A \subseteq E$ is an antispanner for an edge $(s,t) \in E$ if $(V, E \setminus A)$ contains no path from s to t of length at most $k \cdot d(s,t)$. If no proper subset of an antispanner A for (s,t) is an antispanner for (s,t) then A is minimal. The set of all* minimal *antispanners for all* thin *edges is denoted by $\mathcal{A}$.*

The edge set of a k-spanner of G must intersect all antispanners for all edges of G. In other words, it has to be a hitting set for all minimal antispanners. Specifically, a set E'' that settles all thin edges must be a hitting set for all minimal antispanners in $\mathcal{A}$. We now prove that if a set E'' does not settle some thin edge, then we can efficiently find a minimal antispanner $A \in \mathcal{A}$ disjoint from E''.

Claim 2.2.4. *There exists a polynomial time algorithm that, given a set of edges $E'' \subset E$ that does not settle some thin edge, outputs a minimal antispanner $A \in \mathcal{A}$ for some thin edge, such that $A \subseteq E \setminus E''$.*

Proof. The algorithm first finds a thin edge (s,t) with no directed path from s to t of length at most $k \cdot d(s,t)$ in E''. Recall that all paths from s to t of length at most $k \cdot d(s,t)$ in G lie in the local graph $G^{s,t} = (V^{s,t}, E^{s,t})$. (See Definition 2.2.1.) Therefore, $E^{s,t} \setminus E''$ is an antispanner for (s,t). The algorithm sets $A = E^{s,t} \setminus E''$ and then greedily deletes edges e from A while $A \setminus \{e\}$ is an antispanner, that is, while $(V^{s,t}, E^{s,t} \setminus A)$ contains no paths of length at most $k \cdot d(s,t)$ from s to t. When no more such edges can be deleted, the algorithm returns A. □

Next, we give an upper bound on the number of minimal antispanners for thin edges.

Claim 2.2.5. $|\mathcal{A}| \le |E| \cdot (n/\beta)^{n/\beta}$. *In particular, if $\beta = \sqrt{n}$, then $|\mathcal{A}| \le \sqrt{n}^{\sqrt{n}+4}$.*

Proof. Fix a thin edge (s,t) and a minimal antispanner A for (s,t). Let T_A be an out-arborescence (shortest path tree) rooted at s in the graph $(V^{s,t}, E^{s,t} \setminus A)$. Denote by $d_{T_A}(u)$ the distance from s to u in the tree T_A. If T_A contains no directed path from s to u, let $d_{T_A}(u) = \infty$. We show that $A = \{(u,v) \in E^{s,t} : d_{T_A}(u) + d(u,v) < d_{T_A}(v)\}$, and thus T_A uniquely determines A for a given thin edge (s,t).

Consider an edge $(u,v) \in A$, and let A^- denote $A \setminus \{(u,v)\}$. Since the antispanner A is minimal, the graph $(V, E \setminus A^-)$ contains a path from s to t of length at most $k \cdot d(s,t)$. This path must lie in $(V^{s,t}, E^{s,t} \setminus A^-)$ and must contain the edge (u,v). Thus, the distance from s to t in the graph $(V^{s,t}, E^{s,t} \setminus A^-)$ is at most $k \cdot d(s,t)$ and is strictly less than $d_{T_A}(t)$. Hence, T_A is not a shortest path tree in the graph $(V^{s,t}, E^{s,t} \setminus A^-)$. Therefore, $d_{T_A}(u) + d(u,v) < d_{T_A}(v)$.

If $(u,v) \in E^{s,t}$ satisfies the condition $d_{T_A}(u) + d(u,v) < d_{T_A}(v)$, then $(u,v) \notin E^{s,t} \setminus A$; otherwise, T_A would not be a shortest path tree. Hence, $(u,v) \in A$.

We now count the number of out-arborescences rooted at s in $(V^{s,t}, E^{s,t} \setminus A)$. For every vertex $u \in V^{s,t}$, we may choose the parent vertex in at most $|V^{s,t}|$ possible ways (if a vertex is a not reachable from s, we choose it as its own parent). Thus, the total number of trees is at most $|V^{s,t}|^{|V^{s,t}|} \le (n/\beta)^{n/\beta}$.

Since there are at most $|E|$ thin edges, the claim follows. □

2.2.4 LP, Separation Oracle and Overall Algorithm

In this section, we describe a randomized algorithm for constructing a small subset of edges $E'' \subseteq E$ that settles all thin edges. First, we formulate an LP relaxation of this problem. Then we describe how to solve it using the ellipsoid method with a separation oracle (Section 2.2.4.1). Finally, in Section 2.2.4.2, we summarize the resulting algorithm for Directed k-Spanner and complete the proof of Theorem 2.2.1.

A set E'' that settles must intersect all minimal antispanners for all thin edges. This condition can be expressed using linear program LP-A (see Fig. 2.1). LP-A has a variable x_e for each edge $e \in E$ and a constraint (2.2) for each minimal antispanner A for every thin edge. Recall that $\mathcal{A}$ is the set of all minimal antispanners for thin edges. In the integral solution $\{x_e^{int}\}$ corresponding to a k-spanner with an edge set $E'' \subseteq E$, we set $x_e^{int} = 1$ if

$$\text{Minimize} \sum_{e \in E} x_e \text{ subject to:} \tag{2.1}$$

$$\sum_{e \in A} x_e \geq 1 \qquad \forall A \in \mathcal{A} \tag{2.2}$$

$$x_e \geq 0 \qquad \forall e \in E \tag{2.3}$$

Figure 2.1: Linear program for the arbitrary-length case, LP-A. $\mathcal{A}$ is the set of all minimal antispanners for thin edges.

$e \in E''$ and $x_e^{int} = 0$ otherwise. All constraints in (2.2) are satisfied for $\{x_e^{int}\}$ since E'' intersects every antispanner. The value of the objective function $\sum_e x_e^{int}$ is equal to the size of E''. Hence, LP-A is a relaxation of Directed k-Spanner.

For ease of presentation, we assume that we have guessed OPT, the size of the sparsest spanner. (We can try all values in $\{n-1, \ldots, n^2\}$ for OPT and output the sparsest spanner found in all iterations). We replace the objective function (2.1) with

$$\sum_{e \in E} x_e \leq OPT, \tag{2.4}$$

and call the resulting linear program LP-A$'$.

2.2.4.1 Separation Oracle

LP-A$'$ has a polynomial number of variables and, by Claim 2.2.5, an exponential in $\tilde{O}(\sqrt{n})$ number of constraints. We solve it using the ellipsoid algorithm with a separation oracle. Our separation oracle receives a fractional vector $\{\hat{x}_e\}$, satisfying (2.3) and (2.4). If $\{\hat{x}_e\}$ is a feasible solution to LP-A$'$, then the separation oracle outputs a set E'' of size at most $2OPT \cdot \sqrt{n} \ln n$, which settles thin edges. Otherwise, it outputs either a set E'' with the same guarantee or a violated constraint from (2.2) for some antispanner A. The separation oracle can also fail with small probability. If it happens during an execution of the ellipsoid algorithm, we output the input graph with all its edges as a k-spanner.

Algorithm 2.3 SeparationOracle($\hat{x}_e$)

```
1: //Sample a random set of edges $E''$, picking each $e \in E$
   //with probability $\min(\hat{x}_e\sqrt{n}\ln n, 1)$ (see Algorithm 2.2).
   $E'' \leftarrow$ RandomizedSelection($\hat{x}_e$)
2: if $E''$ settles all thin edges then
3:     if $|E''| \leq 2OPT \cdot \sqrt{n}\ln n$ then return $E''$;
4:     else fail;
5: else
6:     Find an antispanner $A \subseteq E \setminus E''$ from $\mathcal{A}$ using the algorithm from Claim 2.2.4.
7:     if $\sum_{e\in A} x_e < 1$ then return violated constraint $\sum_{e\in A} x_e \geq 1$;
8:     else fail.
9: end if
```

The separation oracle is described in Algorithm 2.3. Next we analyze the probability that the separation oracle fails.

Lemma 2.2.6. *The probability that the separation oracle fails during an execution of the ellipsoid algorithm is exponentially small in n.*

Proof. The separation oracle can fail for two reasons:

1. The size of the sampled set E'' is too large.
2. The minimal antispanner A found by the oracle does not correspond to a violated constraint.

To analyze the probability of the first event, note that the expected size of E'' is at most $\sqrt{n}\ln n\sum_{e\in E} x_e \leq OPT \cdot \sqrt{n}\ln n$. By the Chernoff bound,

$$\Pr(|E''| > 2OPT \cdot \sqrt{n}\ln n) \leq \exp(-c \cdot OPT \cdot \sqrt{n}\ln n) = \exp(-\Omega(n\sqrt{n}\ln n)).$$

Thus, the probability that the separation oracle fails because $|E''| > 2OPT \cdot \sqrt{n}\ln n$ is exponentially small in n.

To analyze the probability of the second event, consider one call to the separation oracle. Fix a minimal antispanner A satisfying $\sum_{e\in A}\hat{x}_e \geq 1$. Claim 2.2.3 shows that the probability that E'' is disjoint from A is at most $\exp(-\sqrt{n}\ln n)$. Claim 2.2.5 demonstrates that $|\mathcal{A}| \leq \sqrt{n}^{\sqrt{n}+4}$. Therefore, by a union bound, the probability that there is a minimal antispanner $A \in \mathcal{A}$ satisfying $\sum_{e\in A}\hat{x}_e \geq 1$ and also disjoint from E'' is at most $\sqrt{n}^{\sqrt{n}+4} \cdot \exp(-\sqrt{n}\ln n) = \exp(-\frac{1}{2}\sqrt{n}\ln n + 2\ln n)$. Thus, the probability that the separation oracle fails during one call because $\sum_{e\in A}\hat{x}_e \geq 1$ is exponentially small in n. Since the number of iterations of the ellipsoid algorithm is polynomial in n, a union bound over all iterations gives that the overall probability that the separation oracle fails during an execution of the ellipsoid algorithm is exponentially small in n. □

Lemma 2.2.6 implies, in particular, that when the separation oracle is given a feasible solution to LP-A′, it fails to output a set E'' with exponentially small probability. Since E'' is obtained by running Algorithm 2.2, we obtain the following corollary that will be used in Section 2.3.

Corollary 2.2.7. *Given a feasible solution to LP-A$'$, Algorithm 2.2 with all but exponentially small probability produces a set E'' that settles thin edges and has size at most $2OPT \cdot \sqrt{n} \ln n$.*

2.2.4.2 Overall Algorithm for Directed k-Spanner

Proof of Theorem 2.2.1. We settle thick edges by running SAMPLE($\sqrt{n}$), according to Lemma 2.2.2. We settle thin edges by running the ellipsoid algorithm as described in Sections 2.2.4 and 2.2.4.1. If the separation oracle fails, which, by Lemma 2.2.6, happens with exponentially small probability, we output a spanner containing all edges E. Thus, the expected size of the set E'' is at most $2OPT \cdot \sqrt{n} \ln n + o(1)$, and the resulting approximation ratio of the algorithm is $O(\sqrt{n} \ln n)$. The ellipsoid algorithm terminates in polynomial time, so the overall running time is polynomial. □

2.3 LP and Rounding for Graphs with Unit-Length Edges

In this section, we describe how to settle all thin edges, and thus prove Theorem 2.2.1, for the case of unit-length edges. Our motivation for presenting this special case is two-fold. First, we show that for the unit-length case, one can directly formulate a polynomial-sized LP relaxation, and this makes the approximation algorithm more efficient. Second, we also use the LP from this section to present a better algorithm for 3-spanners in Section 2.4.

Our LP for the case of unit lengths, LP-U, is stated in terms of *local layered graphs* which we introduce next.

Definition 2.3.1 (Layered expansion)**.** *Given a directed graph $G = (V, E)$, its* layered expansion *is a directed graph $\bar{G} = (\bar{V}, \bar{E})$, satisfying the following:*

1. *Let $\bar{V} = \{v_i : \ v \in V$ and $i \in \mathbb{Z}^{\geq 0}\}$, where v_i denotes the i-th copy of v. The set of all the i-th copies of nodes in V is the i-th layer of $\bar{V}$.*
2. *Let $L = \{(u, u) : \ u \in V\}$ be the set of loops. Define the i-th copy of an edge $e = (u, v)$ to be $e_i = (u_i, v_{i+1})$, and the i-th copy of a loop $e = (u, u)$ to be $e_i = (u_i, u_{i+1})$. Let $\bar{E} = \{e_i : \ e \in E \cup L$ and $i \in \mathbb{Z}^{\geq 0}\}$.*

Layered expansion $\bar{G}$ contains a path from s_0 to t_k if and only if G contains a path from s to t of length at most k. A *local layered graph* for a thin edge (s, t) is defined next. It consists of all paths in the layered expansion $\bar{G}$ that correspond to paths from s to t of length at most k in the original graph G or, in other words, to paths in the local graph $G^{s,t}$, defined in Definition 2.2.1.

Definition 2.3.2 (Local layered graph)**.** *For a thin edge (s, t) and $k \geq 1$, the* local layered graph *is a subgraph $\bar{G}^{s,t} = (\bar{V}^{s,t}, \bar{E}^{s,t})$ of $\bar{G}$ with a source $\bar{s} = s_0$ and a sink $\bar{t} = t_k$, such that $\bar{G}^{s,t}$ contains all nodes and edges on paths from $\bar{s}$ to $\bar{t}$.*

Our algorithm solves the linear program LP-U defined in Figure 2.2. Recall that $\mathcal{E}$ denotes the set of thin edges. LP-U has variables of two types: x_e, where $e \in E$, and $f^{s,t}_{e_i}$, where $(s, t) \in \mathcal{E}$ and $e_i \in \bar{E}^{s,t}$. A variable x_e represents whether the edge e is included in

Minimize $\sum_{e \in E} x_e$ subject to:

$$\begin{array}{lrl}
\text{Flow requirement} & \displaystyle\sum_{e_0 \in Out(s_0)} f^{s,t}_{e_0} \geq 1 & \forall (s,t) \in \mathcal{E} \\
\text{Flow conservation} & \displaystyle\sum_{e_{i-1} \in In(v_i)} f^{s,t}_{e_{i-1}} - \sum_{e_i \in Out(v_i)} f^{s,t}_{e_i} = 0 & \forall (s,t) \in \mathcal{E}, \forall v_i \in \bar{V}^{s,t} \setminus \{\bar{s}, \bar{t}\} \\
\text{Capacity constraints} & \displaystyle x_e - \sum_{i=0}^{k-1} f^{s,t}_{e_i} \geq 0 & \forall (s,t) \in \mathcal{E}, \forall e \in E \\
 & x_e \geq 0 & \forall e \in E \\
 & f^{s,t}_{e_i} \geq 0 & \forall (s,t) \in \mathcal{E}, \forall e_i \in \bar{E}^{s,t}
\end{array}$$

Figure 2.2: Linear program for the unit-length case, LP-U.

the k-spanner. We think of a path from s to t of length at most k in G as a unit flow from $\bar{s}$ to $\bar{t}$ in $\bar{G}^{s,t}$. A variable $f^{s,t}_{e_i}$ represents flow along the edge e_i in $\bar{G}^{s,t}$. We denote the sets of incoming and outgoing edges for a vertex $v_i \in \bar{G}^{s,t}$ by $In(v_i)$ and $Out(v_i)$, respectively.

Given $\hat{x}_e$, a fractional solution of LP-U, we construct the set E'' by first running Algorithm 2.2 and then adding all unsettled thin edges.

Lemma 2.3.1. *The algorithm described above, in polynomial time, computes a set E'' that settles all thin edges and has expected size at most $2\sqrt{n}\ln n \cdot OPT + o(1)$.*

Proof. We prove, in Claim 2.3.2, that in a fractional optimal solution $\{\hat{x}_e\} \cup \{\hat{f}^{s,t}_{e_i}\}$ to LP-U, the vector $\{\hat{x}_e\}$ is a fractional solution to LP-A$'$. Then we apply Corollary 2.2.7 to get the desired bound on the expected size of E''. At the end, we argue that the algorithm runs in polynomial time.

Claim 2.3.2. *In a fractional optimal solution $\{\hat{x}_e\} \cup \{\hat{f}^{s,t}_{e_i}\}$ to LP-U, the vector $\{\hat{x}_e\}$ is a fractional solution to LP-A$'$.*

Proof. First, we argue that LP-U is a relaxation of Directed k-Spanner for the unit-length case or, in other words, that an optimal solution to this program has value at most OPT. Let H be a sparsest k-spanner of G. Assign $x_e = 1$ if e is in H and $x_e = 0$ otherwise. For each thin edge (s,t), consider a simple path from s to t in H of length ℓ, where $\ell \leq k$. Set $f^{s,t}_{e_i}$ to 1 if either e is the ith edge on that path or $i \in \{\ell+1, \ldots, k\}$ and $e_i = (t_{i-1}, t_i)$; otherwise, set it to 0. Since the resulting assignment is a feasible solution to LP-U, the optimal solution to this program has value $\sum_{e \in E} \hat{x}_e \leq OPT$.

Next, we argue that if $\{\hat{x}_e\} \cup \{\hat{f}^{s,t}_{e_i}\}$ is a feasible solution to LP-U then $\{\hat{x}_e\}$ satisfies the antispanner constraints for LP-A$'$, given in (2.2). Consider a thin edge (s,t) and a minimal antispanner $A \in \mathcal{A}$ for (s,t). Let $\bar{A} = \{e_i : e \in A \text{ and } e_i \in \bar{E}^{s,t}\}$ be the set of copies of the edges in A in the local layered graph. Let $\bar{S} \subseteq \bar{V}^{s,t}$ be the set of nodes that can be reached from $\bar{s}$ in $(\bar{V}^{s,t}, \bar{E}^{s,t} \setminus \bar{A})$ and $\bar{T} = \bar{V}^{s,t} \setminus \bar{S}$ be the set of the remaining nodes. Since A is an antispanner for (s,t), node $\bar{t}$ is in $\bar{T}$, and thus $(\bar{S},\bar{T})$ is an $(\bar{s},\bar{t})$ cut in $\bar{G}^{s,t}$. Note that only edges from $\bar{A}$ can cross the cut because for an edge $(u_i, v_{i+1}) \notin \bar{A}$ if u_i is reachable from $\bar{s}$ then so is v_{i+1}.

For a fractional solution $\{\hat{x}_e\} \cup \{\hat{f}^{s,t}_{e_i}\}$ to LP-U,

$$\sum_{e\in A} \hat{x}_e \geq \sum_{e\in A}\sum_{i=0}^{k-1} \hat{f}^{s,t}_{e_i} = \sum_{e_i\in \bar{A}} \hat{f}^{s,t}_{e_i} \geq \sum_{e_i \in \, cut\,(\bar{S},\bar{T})} \hat{f}^{s,t}_{e_i} = \sum_{e_0\in Out(s_0)} \hat{f}^{s,t}_{e_0} \geq 1. \tag{2.5}$$

The first inequality above follows from the capacity constraints in LP-U, the following equality holds by definition of $\bar{A}$, the second inequality holds because $\bar{A}$ contains the edges in the cut $(\bar{S},\bar{T})$, the last equality follows from the flow conservation, and the last inequality is the flow requirement.

We proved that in a fractional optimal solution $\{\hat{x}_e\} \cup \{\hat{f}^{s,t}_{e_i}\}$ to LP-U, the vector $\{\hat{x}_e\}$ satisfies constraints (2.2) and (2.4) of LP-A$'$. Since constraints (2.3) are also in LP-U, vector $\{\hat{x}_e\}$ is a fractional solution to LP-A$'$. □

By, Claim 2.3.2, vector $\{\hat{x}_e\}$ is a fractional solution to LP-A$'$. Corollary 2.2.7 says that, given such a solution, Algorithm 2.2 with all but exponentially small probability produces a set E'' that settles thin edges and has size at most $2OPT\cdot\sqrt{n}\ln n$. After we add all unsettled thin edges, the expected size of the resulting set E'' is at most $2OPT\cdot\sqrt{n}\ln n + o(1)$.

It remains to argue that the described algorithm takes polynomial time. To write down LP-U, we only need to know V, E, k and the set of thin edges, $\mathcal{E}$. The first three are inputs to the algorithm, and $\mathcal{E}$ can be computed in polynomial time. LP-U can be written down and solved in polynomial time because it has $O(|E|^2\cdot k) = O(n^5)$ variables and constraints. □

Proof of Theorem 2.2.1 for the case of unit-lengths. We run Algorithm 2.1 to get E'. We construct E'' by running Algorithm 2.2 and adding all unsettled thin edges. Let the edge set of our k-spanner be $E'\cup E''$. By Lemmas 2.2.2 and 2.3.1, E' settles all thick edges, E'' settles all thin edges, the expected size of $E'\cup E''$ is $O(\sqrt{n}\ln n\cdot OPT)$, and the resulting algorithm runs in polynomial time, as required. □

2.4 An $\tilde{O}(n^{1/3})$-Approximation for Directed 3-Spanner with Unit-Length Edges

In this section, we show an improved approximation for the special case of Directed 3-Spanner with unit length edges. The algorithm follows the general strategy explained in Section 2.2. The LP rounding scheme here is different from that presented in Section 2.2.2 and used in the two algorithms for Directed k-Spanner in Sections 2.2 and 2.3. We note that Algorithm 2 from [71] with $\rho = \tilde{\Theta}(n^{1/3})$ could also be used to prove our result. The rounding scheme we present is simpler and allows for simpler analysis.

As in [71], we use random variables for vertices instead of edges to guide edge selection process. Intuitively, this allows us to introduce positive correlations in selection of edges adjacent to the same vertex. Because the correlations are local, the improvement in approximation deteriorates for larger values of k. To simplify analysis, instead of threshold rounding (as in the previous sections) we use Poisson random variables.

Theorem 2.4.1. *There is a polynomial time randomized algorithm for* Directed 3-Spanner *for graphs with unit edge lengths with expected approximation ratio* $O(n^{1/3}\log n)$.

Proof. We define thick and thin edges as in Definition 2.2.2, with $\beta = n^{1/3}$, and run Sample$(n^{1/3})$. By Lemma 2.2.2, the resulting edge set E' settles all thick edges and has expected size at most $3n^{1/3}\ln n \cdot OPT$. Then we obtain an optimal solution $\{\mathring{x}_e\} \cup \{\mathring{f}^{s,t}_{e_i}\}$ of the linear program LP-U from Fig. 2.2. Our rounding scheme is stated in Algorithm 2.4. It consists of two stages: first, we round $\{\mathring{x}_e\}$ to obtain a new solution $\{\hat{x}_e\}$, where every assignment $\hat{x}_e$ is an integer multiple of $n^{-2/3}$; second, we round $\{\hat{x}_e\}$ to obtain an edge set E'' that settles all thin edges with high probability.

In the first step we sample a random variable from Poisson distribution for every edge. Recall, that a Poisson random variable X with mean λ is supported over non-negative integers and has a probability density function:

$$\Pr[X = k] = \frac{\lambda^k e^-\lambda}{k!}, \forall k \in \mathbb{Z}^{\geq 0}.$$

The only properties of the Poissson distribution that we use in the analyisis are concentration bound stated in the Appendix, integrality of the support and the fact that the sum of Poisson random variables is again a Poisson random variable.

Algorithm 2.4 RandomizedSpannerSelection$(\mathring{x}_e)$

```
 1: $E'' \leftarrow \varnothing$;
    //Obtain a new solution $\{\hat{x}_e\}$, where each coordinate $\hat{x}_e$ is a multiple of $n^{-2/3}$ :
 2: for each edge $e \in E$ do
 3:     $P_e \leftarrow$ sample from the Poisson distribution with mean $\lambda_e = 6n^{2/3}\mathring{x}_e$;
 4:     $\hat{x}_e \leftarrow P_e n^{-2/3}$;
 5: end for
    //Round $\{\hat{x}_e\}$ to get $E''$:
 6: for each vertex $u \in V$ do
 7:     $r_u \leftarrow$ uniform sample from $(0, 1)$;
 8: end for
 9: for each edge $e = (u, v) \in E$ do
10:     if $\min(r_u, r_v) \leq \hat{x}_e \alpha\, n^{1/3} \ln n$ then add $e$ to $E''$;
11:                                          //$\alpha > 1$ is an absolute constant
12: end for
13: return $E''$.
```

Lemma 2.4.2 below analyzes the first stage. Then Lemmas 2.4.3 and 2.4.4 analyze the set E'' produced by the second stage. Lemma 2.4.3 bounds the expected size of E'' by $O(OPT n^{1/3}\ln n)$. Lemma 2.4.4 shows that E'' settles a given thin edge with probability at least $1 - 1/n$. Consequently, the expected number of unsettled thin edges is at most

$|E|/n \le n-1 \le OPT$, and they can be added to the solution without affecting the approximation ratio. This completes the proof of Theorem 2.4.1. □

It remains to prove the lemmas that were used in the proof of Theorem 2.4.1.

Recall that $\bar{s}$ and $\bar{t}$ are used to denote the source and the sink the the local layered graph of an edge (s,t) as in Definition 2.3.2.

Lemma 2.4.2. *Given a feasible solution $\{\mathring{x}_e\} \cup \{\mathring{f}^{s,t}_{e_i}\}$ of LP-U of cost LP, Algorithm 2.4 on lines 2–5 computes a vector $\{\hat{x}_e\}$ of cost at most $20LP$ (i.e., satisfying $\sum_e \hat{x}_e \le 20LP$) such that all $\hat{x}_e$ are integer multiples of $n^{-2/3}$. Moreover, for every thin edge (s,t) and cut $(\bar{S},\bar{T})$ in the local layered graph $\bar{G}^{s,t}$ with $\bar{s}, s_1 \in \bar{S}$ and $t_2, \bar{t} \in \bar{T}$, vector $\{\hat{x}_e\}$ satisfies*

$$\sum_{(u,v)\in E^{s,t}:(u_i,v_{i+1})\in \bar{S}\times\bar{T}} \hat{x}_{(u,v)} \ge 1. \tag{2.6}$$

This stage of the algorithm succeeds with probability $1-\exp(-cn^{2/3})$ for some constant $c>0$.

Proof. For every edge e, we independently sample a Poisson random variable P_e with mean $\lambda_e = 6n^{2/3}\mathring{x}_e$, and set $\hat{x}_e = P_e n^{-2/3}$. Since the support of the Poisson distribution is on nonnegative integers, all $\hat{x}_e$ are integer multiples of $n^{-2/3}$. We need to verify that $\hat{x}_e$ satisfies (2.6) and that its cost is bounded by $20LP$.

Fix a thin edge (s,t) and a cut $(\bar{S},\bar{T})$ in $\bar{G}^{s,t}$ with $\bar{s}, s_1 \in S$ and $t_2, \bar{t} \in T$. Let $A = \{(u,v) \in E^{s,t} : (u_i, v_{i+1}) \in \bar{S}\times\bar{T}\}$. We will show that it is an antispanner for (s,t). For every path $p = s \to u \to v \to t$ of length 3 in $G^{s,t}$, one of the edges on the path $\bar{s} \to u_1 \to v_2 \to \bar{t}$ crosses the cut $(\bar{S},\bar{T})$ and, consequently, one of the edges of p belongs to A. Similarly, for every path $p = s \to u \to t$ of length 2 (respectively, path $p = s \to t$ of length 1) one of the edges on the path $\bar{s} \to u_1 \to t_2 \to \bar{t}$ (respectively, path $\bar{s} \to s_1 \to t_2 \to \bar{t}$) crosses the cut $(\bar{S},\bar{T})$, and one of the edges of p belongs to A. Therefore, A is an antispanner for (s,t). By Claim 2.3.2, if $\{\mathring{x}_e\} \cup \{\mathring{f}^{s,t}_{e_i}\}$ is a feasible solution to LP-U then $\{\mathring{x}_e\}$ satisfies the antispanner constraints for LP-A′, given in (2.2). That is, $\sum_{e\in A} \mathring{x}_e \ge 1$.

Next, we bound $\sum_{e\in A} \hat{x}_e = n^{-2/3}\sum_{e\in A} P_e$. The sum $\sum_{e\in A} P_e$ is distributed as a Poisson random variable with mean $\lambda_A = \sum_{e\in A}\lambda_e \ge 6n^{2/3}$. By Lemma A.1 in the Appendix,

$$\Pr[\sum_{e\in A}\hat{x}_e < 1] = \Pr[\sum_{e\in A} P_e < n^{2/3}] \le \Pr[\sum_{e\in A} P_e \le 6/e \cdot n^{2/3}] \le \exp(-6n^{2/3}/4).$$

Since (s,t) is a thin edge, $|\bar{V}^{s,t} \setminus \{\bar{s},\bar{t}\}| \le 2n^{2/3}$, and the number of cuts $(\bar{S},\bar{T})$ in $\bar{V}^{s,t}$ separating $\bar{s}$ and $\bar{t}$ is at most $2^{2n^{2/3}} = \exp(\ln 4 \cdot n^{2/3})$. Hence, by a union bound, (2.6) holds for all such cuts simultaneously with probability at least $1 - e^{-cn^{2/3}}$, where $c = 6/4 - \ln 4 > 0$. By a union bound, the previous sentence is true for all thin edges (s,t) simultaneously with a constant if c is set to $c/2 = \frac{1}{2}(6/4 - \ln 4)$.

Finally, observe that the cost of $\{\hat{x}_e\}$ is $n^{-2/3} \times \sum_{e\in E} P_e$. The sum $\sum_{e\in E} P_e$ is a Poisson random variable with mean $6n^{2/3}\sum \mathring{x}_e = 6n^{2/3}LP$. By Lemma A.1,

$$\Pr[\sum_{e\in E} P_e \geq 20n^{2/3}LP] \leq \Pr[\sum_{e\in E} P_e \geq 6\cdot e \cdot n^{2/3}LP] \leq \exp(-6n^{2/3}LP) \leq \exp(-6n^{2/3}).$$

Thus, the probability that the cost of $\{\hat{x}_e\}$ exceeds $20LP$ is exponentially small. □

Lemma 2.4.3 (Analog of Lemma 4.1 in [71]). $\mathbb{E}[|E''|] = O(OPTn^{1/3}\ln n)$.

Proof. By a union bound, the probability that an edge e belongs to E'' is at most $2\hat{x}_e\alpha n^{1/3}\ln n$. Therefore, since α is a constant,

$$\mathbb{E}[|E''|] \leq \sum_{e\in E} 2\hat{x}_e\alpha\, n^{1/3}\ln n = O(OPTn^{1/3}\ln n).$$

□

Lemma 2.4.4 (Analog of Lemma 4.2 in [71]). *If (s,t) is a thin edge for which condition (2.6) holds, then E'' settles (s,t) with probability at least $1-1/n$.*

Proof. Fix a thin edge (s,t). Let

$$\bar{E}'' = \{e_i :\ e\in E'' \text{ and } e_i \in \bar{E}^{s,t}\} \cup \{(\bar{s},s_1),(t_2,\bar{t})\}$$

be the set of copies of the edges in E'' in the local layered graph $\bar{G}^{s,t}$. We show that, with probability at least $1-1/n$, there is a path from $\bar{s}$ to $\bar{t}$ in $(\bar{V}^{s,t},\bar{E}'')$ and, consequently, the edge (s,t) is settled.

Let $\{\hat{f}^{s,t}_{e_i}\}$ be the maximum flow from $\bar{s}$ to $\bar{t}$ in the graph $\bar{G}^{s,t}$ with capacities $\hat{x}_{e_i}$ set to $\hat{x}_{(u,v)}$ on edges $e_i = (u_i, v_{i+1})$ for $u \neq v$, infinite capacities ($\hat{x}_{e_i} = \infty$) on edges $e_i \in \{(\bar{s},s_1),(t_2,\bar{t})\}$ and zero capacities ($\hat{x}_{e_i} = 0$) on edges $e_i = (u_i, u_{i+1})$ for $e_i \notin \{(\bar{s},s_1),(t_2,\bar{t})\}$. Note that this flow may be different from the flow $\{\mathring{f}^{s,t}_{e_i}\}$ obtained by LP-U. By (2.6), the capacity of the minimum cut between $\bar{s}$ and $\bar{t}$ is at least 1. Thus, the value of the flow $\{\hat{f}^{s,t}_{e_i}\}$ is at least 1.

In the simplest case, the flow is routed along $n^{2/3}$ disjoint paths of capacity $n^{-2/3}$ each. The probability that a given path $\bar{s}\to u_1\to v_2\to\bar{t}$ belongs to $(\bar{V}^{s,t},\bar{E}'')$ is at least $0r_u \leq \alpha\, n^{-1/3}\ln n$ and $r_v \leq \alpha\, n^{-1/3}\ln n \geq (\alpha\, n^{-1/3}\ln n)^2$. The probability that at least one path belongs to $\bar{E}''$ is $1-(1-\alpha^2\, n^{-2/3}\ln^2 n)^{n^{2/3}} > 1-1/n$. In the general case, however, we need a more involved analysis.

To analyze the general case, we partition the set $V^{s,t}$ into two disjoint sets S and T such that at least 1/4 units of flow $\{\hat{f}^{s,t}_{e_i}\}$ are routed along the paths $\bar{s}\to u_1\to v_2\to\bar{t}$, where $u\in S$ and $v\in T$. To see that such a partition exists, randomly add every vertex in $V^{s,t}\setminus\{s,t\}$ to S or T with probability 1/2. Add s to S and t to T. Then for every path $\bar{s}\to u_1\to v_2\to\bar{t}$ (where $u\neq v$), $0u\in S$ and $v\in T \geq 1/4$, so the expected contribution of every path to the new flow is at least 1/4 of the original flow over the path. Because the total new flow from $\bar{s}$ to $\bar{t}$ can be represented as a sum of flows over such paths, the expected flow routed from $\bar{s}$ to $\bar{t}$ through S and T (as described above) is at least 1/4. That is, for at least one partition (S,T) the flow is at least 1/4. Fix this partition.

Let $\{f_{e_i}\}$ be the maximum flow in $\bar{G}^{s,t}$ (with the same capacities as above) routed from $\bar{s}$ to $\bar{t}$ through S and T, such that all f_{e_i} are multiples of $n^{-2/3}$. Such a flow exists because all capacities are multiples of $n^{-2/3}$. Observe that $(u,v) \in \bar{E}''$ if $\min(r_u, r_v) \leq \hat{x}_{(u,v)}\alpha\, n^{1/3} \ln n$ and, consequently, also if $\min(r_u, r_v) \leq f_{(u,v)}\alpha\, n^{1/3} \ln n$, since $f_{(u,v)} \leq \hat{x}_{(u,v)}$.

Consider the following two cases:

1. $f_{(\bar{s},u_1)} \geq n^{-1/3}$ for some vertex $u \in S$.

2. $f_{(\bar{s},u_1)} < n^{-1/3}$ for all vertices $u \in S$.

Case 1. Fix a vertex $u \in S$ for which $f_{(\bar{s},u_1)} \geq n^{-1/3}$. We will show that with probability at least $1 - 1/n$ there is a path from $\bar{s}$ to $\bar{t}$ via u_1 in $\bar{G}^{s,t}$. The edge $(\bar{s}, u_1)$ always belongs to $\bar{E}''$ because $\alpha\hat{x}_{(\bar{s},u_1)} n^{1/3} \ln n \geq \alpha f_{(\bar{s},u_1)} n^{1/3} \ln n > 1 \geq r_u$. Consider an arbitrary path $u_1 \to v_2 \to \bar{t}$. Note that $f_{(v_2,\bar{t})} \geq f_{(u_1,v_2)}$, since all flow from u_1 to v_2 must be routed to $\bar{t}$ along the edge $(v_2, \bar{t})$. Thus, if $r_v \leq \alpha f_{(u_1,v_2)}\, n^{1/3} \ln n$, then $(u_1, v_2) \in \bar{E}''$ and $(v_2, \bar{t}) \in \bar{E}''$. Therefore, if there is no path from u_1 to $\bar{t}$ in $\bar{E}''$, then $r_v > \alpha f_{(u_1,v_2)}\, n^{1/3} \ln n$ for all $v \in T$. This happens with probability at most

$$\begin{aligned}
\prod_{v\in T} \min(1 - \alpha f_{(u_1,v_2)}\, n^{1/3} \ln n, 0) &\leq \prod_{v \in T} \exp\left(-\alpha f_{(u_1,v_2)}\, n^{1/3} \ln n\right) \\
&= \exp\left(-\left(\sum_{v\in T} f_{(u_1,v_2)}\right) \alpha n^{1/3} \ln n\right) \\
&\leq \exp\left(-f_{(\bar{s},u_1)}\, \alpha n^{1/3} \ln n\right) \\
&\leq \exp(-\ln n) = \frac{1}{n}.
\end{aligned}$$

Therefore, with probability at least $1 - 1/n$, there is a path from $\bar{s}$ to $\bar{t}$ in $\bar{G}^{s,t}$ and, consequently, the edge (s,t) is settled.

Case 2. For every $u \in S$, define a random variable F_{u_1}:

$$F_{u_1} = \sum_{v \in T : r_u \leq \alpha f_{(u_1,v_2)} n^{1/3} \ln n} f_{(u_1,v_2)}.$$

This random variable gives a lower bound on the amount of flow that can be routed along the edges $\bar{E}''$ from the source $\bar{s}$ to the set of copies of nodes in T through the vertex u_1. (Recall that $\bar{E}''$ is a random set.)

Claim 2.4.5. $\Pr_{r_u \,:\, u \in S}\left[\sum_{u\in S} F_{u_1} \geq \frac{\alpha n^{-1/3} \ln n}{8}\right] \geq 1 - \frac{1}{2n}.$

Proof. The value of F_{u_1} depends only on r_u, and hence all random variables F_{u_1} are independent. If $f_{(u_1,v_2)} > 0$ then $f_{(u_1,v_2)} \geq n^{-2/3}$ because $f_{(u_1,v_2)}$ is a multiple of $n^{-2/3}$. Therefore, for all nodes $u \in S$ and $v \in T$ with positive flow $f_{(u_1,v_2)}$,

$$\Pr_{r_u}\left[r_u \leq \alpha f_{(u_1,v_2)} n^{1/3} \ln n\right] = \min(\alpha f_{(u_1,v_2)} n^{1/3} \ln n, 1) \geq \alpha n^{-2/3} n^{1/3} \ln n \geq \alpha n^{-1/3} \ln n.$$

This implies that for all nodes $u \in S$ and $v \in T$,

$$f_{(u_1,v_2)} \cdot \Pr_{r_u}\left[r_u \le \alpha f_{(u_1,v_2)} n^{1/3} \ln n\right] \ge f_{(u_1,v_2)} \cdot \alpha n^{-1/3} \ln n.$$

Therefore,

$$\begin{aligned}
\mathbb{E}[\sum_{u\in S} F_{u_1}] &= \sum_{u\in S}\left(\sum_{v\in T} f_{(u_1,v_2)} \Pr_{r_u}\left[r_u \le \alpha f_{(u_1,v_2)} n^{1/3}\ln n\right]\right) \\
&\ge \sum_{u\in S}\Big(\sum_{v\in T} f_{(u_1,v_2)}\Big)\alpha n^{-1/3}\ln n = \Big(\sum_{u\in S} f_{(\bar{s},u_1)}\Big)\alpha n^{-1/3}\ln n \ge \frac{\alpha}{4} n^{-1/3}\ln n.
\end{aligned}$$

Now we use the assumption that $f_{(\bar{s},u_1)} \le n^{-1/3}$ for all $u \in S$. By flow conservation, it implies that all F_{u_1} are bounded from above by $n^{-1/3}$. By the Hoeffding inequality [1] applied with $\epsilon = 1/2$ and $c = n^{-1/3}$,

$$\begin{aligned}
\Pr_{r_u}\left[\sum_{u\in S} F_{u_1} \ge \frac{\alpha n^{-1/3}\ln n}{8}\right] &= 1 - \Pr_{r_u}\left[\sum_{u\in S} F_{u_1} < \frac{1}{2}\mathbb{E}[\sum_{u\in S} F_{u_1}]\right] \\
&\ge 1 - \exp\left(-\frac{\alpha \ln n}{32}\right) \ge 1 - \frac{1}{2n}.
\end{aligned}$$

□

Next, we condition on the event that $\sum_{u\in S} F_{u_1} \ge \alpha n^{-1/3}\ln n/8$, and bound the conditional probability that there exists a path from $\bar{s}$ to $\bar{t}$.

Claim 2.4.6. *For any fixed $\{r_u\}_{u\in S}$, such that $\sum_{u\in S} F_{u_1} \ge \alpha n^{-1/3}\ln n/8$, we have*

$$\Pr_{r_v\,:\,v\in T}\left[\textit{there is no path } \bar{s} \to u_1 \to v_2 \to \bar{t} \textit{ in } E''\right] \le \frac{1}{2n}.$$

Proof. For every $v \in T$, let

$$F_{v_2} = \sum_{u_1 : r_u \le \alpha f_{(u_1,v_2)} n^{1/3}\ln n} f_{(u_1,v_2)}.$$

If for some $\tilde{v} \in T$ we have $F_{\tilde{v}_2} > 0$, then for some $\tilde{u} \in S$, $r_{\tilde{u}} \le \alpha f_{(\tilde{u}_1,\tilde{v}_2)} n^{1/3}\ln n$, $r_{\tilde{u}} \le \alpha f_{(\bar{s},\tilde{u}_1)} n^{1/3}\ln n$ and, hence, the path $\bar{s} \to \tilde{u}_1 \to \tilde{v}_2$ belongs to E''. Also,

$$f_{(\tilde{v}_2,\bar{t})} = \sum_{u\in S} f_{(u_1,\tilde{v}_2)} \ge F_{v_2}.$$

[1] Here we use the following variant of the Hoeffding's inequality. Let $X_1,\ldots,X_n$ be independent random variables taking values in $[0,c]$. Let $S_n = \sum X_i$, let $\mu = \mathbb{E}[S_n]$. Then, for every positive ε,

$$0S_n \le (1-\varepsilon)\mu \le e^{-\frac{1}{2}\varepsilon^2\mu/c}.$$

For reference see, e.g., [112] Theorem 2.3(c) (page 200).

Now for a fixed $\{r_u\}_{u \in S}$ and a vertex $\tilde{v} \in T$, we bound the probability that $(\tilde{v}_2, \bar{t}) \in E''$ from below by

$$\Pr_{r_{\tilde{v}}}\left[r_{\tilde{v}} \le \alpha f_{(\tilde{v}_2,\bar{t})} n^{1/3} \ln n\right] \ge \min(\alpha f_{(\tilde{v}_2,\bar{t})} n^{1/3} \ln n, 1) \ge \min(\alpha F_{\tilde{v}_2} n^{1/3} \ln n, 1).$$

Note that we have a lower bound on the sum of F_{v_2}'s:

$$\sum_{v \in T} F_{v_2} = \sum_{u \in S} F_{u_1} \ge \frac{\alpha n^{-1/3} \ln n}{8}.$$

Thus, we can use the same argument as in Claim 2.2.3 to get a lower bound on the overall probability:

$$\begin{aligned} \Pr_{r_v}\left[(v_2, \bar{t}) \notin E'', \text{ for all } v \in T \text{ with } F_{v_2} > 0\right] &\le \exp\bigl(-\sum_{v \in T} \alpha F_{v_2} n^{1/3} \ln n\bigr) \\ &\le \exp(-\alpha^2 \ln^2 n/8) < \frac{1}{2n}. \end{aligned}$$

□

By Claims 2.4.5 and 2.4.6, the probability that there exists a path $\bar{s} \to u_1 \to v_2 \to \bar{t}$ is at least $(1 - 1/(2n))^2 > 1 - 1/n$. □

2.5 An $O(n^{2/3+\epsilon})$-Approximation for Directed Steiner Forest

Let us first recall the Directed Steiner Forest (DSF) problem. Given a directed graph $G = (V, E)$, a cost function $c : E \to \mathbb{R}^+$ and a set $D \subseteq V \times V$ of ordered pairs, the goal is to find a min-cost subgraph H of G that contains a path from s to t for every $(s, t) \in D$. In contrast to spanners, there is no restriction on the paths used to connect pairs, but the objective to be optimized depends on arbitrary edge costs.

Theorem 2.5.1. *For any fixed $\epsilon > 0$, there is a polynomial time randomized algorithm for* Directed Steiner Forest *with expected approximation ratio $O(n^{2/3+\epsilon})$.*

Our algorithm for DSF builds on the algorithm of Feldman, Kortsarz and Nutov [93] for the problem. We describe their algorithm and most of their analysis, using notation compatible with previous sections of this paper, and show where we make our improvement. As mentioned in [93], one can assume without loss of generality that $D \subseteq S \times T$ for two *disjoint* subsets S and T of V and that the costs are metric.

Let τ denote our guess for the optimal value of OPT. We start from $\tau = 1$ and repeatedly double our guess each time we find it is too small. Thus, it suffices to give the approximation guarantee for the iteration when $OPT \le \tau \le 2 \cdot OPT$. The algorithm has two parameters: β and ℓ. We set $\beta = n^{1/3}$ and $\ell = \tau/n^{2/3}$ below.

Let us adapt some terminology from the previous sections to this new setting.

Definition 2.5.1 (Thick & thin pairs)**.** *For a pair $(s, t) \in D$, let $G^{s,t} = (V^{s,t}, E^{s,t})$ be the subgraph of G induced by the vertices on paths from s to t of cost at most ℓ. A pair $(s, t) \in D$ is* thick *if $|V^{s,t}| \ge n/\beta$ and it is* thin *otherwise.*

Definition 2.5.2. *A set $E' \subseteq E$* settles *a pair $(s,t) \in D$ if the subgraph (V, E') contains a path from s to t.*

The high-level structure of the algorithm is the same as for the spanner problem. We will describe how to find in polynomial time two sets $E', E'' \subseteq E$,such that E' settles all the thick pairs and E'' settles all the thin pairs.

The thick pairs can be settled by random sampling, just as in Section 2.2.1. For $p = O((\log n)/(n/\beta))$, if each vertex is selected with probability p to lie in a set R, then for every $(s,t) \in D$, $R \cap V^{s,t} \neq \emptyset$ with high probability. Let the set E' be constructed by adding, for each $u \in R, s \in S, t \in T$, the edges of a path from s to u of cost at most ℓ if one exists and the edges of a path from u to t of cost at most ℓ if one exists. The expected number of thick pairs still not settled is at most $|D|/n^2 \leq 1$. Thus, we can add the edges of a minimum-cost path from s to t for any unsettled thick pair (s,t) and still have that the expected cost of E' be $O(n \cdot pn \cdot \ell + \tau) = \tilde{O}(n\ell\beta + \tau) = \tilde{O}(n^{2/3}\tau)$, where we use τ as an upper bound on the cost of a minimum-cost (s,t)-path.

We remove the settled thick pairs from D, so that it only consists of the unsettled thin pairs. Next, we construct an edge set E'' that settles all the thin pairs. Define the *density* of a subset of E to be the ratio between the total cost of the subset and the number of pairs in D settled by it. We show how to efficiently construct a subset K with expected density $O(n^{2/3+\epsilon}) \cdot \tau/|D|$. This allows us to compute the set E'': starting from $|D|$ unsettled thin pairs and $E'' = \emptyset$, find K of expected density $O(n^{2/3+\epsilon}) \cdot \tau/|D|$, add the edges in K to E'', remove the settled pairs from D, and repeat. As shown in Theorem 2.1 of [93], this greedy procedure produces a subset E'' of expected cost $O(n^{2/3+\epsilon}) \cdot \tau$ that settles all the thin pairs, completing the proof of Theorem 2.5.1.

The edge set K is produced by constructing two sets K_1 and K_2 and letting K be the set of smaller density. We guarantee that one of K_1 and K_2 has expected density $O(n^{2/3+\epsilon}) \cdot \tau/|D|$. Whether the guarantee is provided for K_1 or K_2 depends upon which one of the two cases below holds. Suppose H is an optimal solution with cost τ (we ignore the factor of 2 for simplicity). Let C be the set of pairs $(s,t) \in D$ for which the minimum cost of an (s,t)-path in H is at least ℓ; that is, these are the costly pairs to settle. The two cases are: $|C| \geq |D|/2$ and $|C| < |D|/2$.

Case 1: $|C| \geq |D|/2$. This case relies on a result of Chekuri, Even, Gupta and Segev [54]. Define a *junction tree* to be the union of an ingoing tree and an outgoing tree (not necessarily disjoint) rooted at the same vertex. [54] shows an $O(n^\epsilon)$-approximation for the minimum density junction tree of a graph. Fortunately, there exists a junction tree of density at most $\tau^2/(|C|\ell)$. To see why, take the paths in H connecting the pairs in C. The sum of the costs of all such paths is at least $|C|\ell$. If we denote the maximum number of these paths that any edge belongs to as μ, then the sum of the costs of the paths is at most $\mu \cdot \tau$ and thus there exists an edge, which belongs to $\mu \geq |C|\ell/\tau$ paths. Therefore, there must be a junction tree K_1 which contains this edge and connects at least $|C|\ell/\tau$ pairs in D. K_1 has density at most $\tau/(|C|\ell/\tau) = \tau^2/(|C|\ell)$. Thus, when $|C| \geq |D|/2$, the algorithm of [54] (deterministically) returns a junction tree of density $O(n^\epsilon \cdot \tau/\ell \cdot \tau/|D|) = O(n^{2/3+\epsilon}) \cdot \tau/|D|$.

Case 2: $|D - C| > |D|/2$. In this case, we attempt to find a subgraph that connects many pairs of D using low-cost edges. Consider the problem of connecting at least $|D|/2$ pairs from D using paths of cost at most ℓ while minimizing the total cost of the edges. For $(s,t) \in D$, let $\Pi(s,t)$ be the set of (s,t)-paths of cost at most ℓ, and let $\Pi = \bigcup_{(s,t)\in D} \Pi(s,t)$.

$$\text{Minimize} \sum_{e\in E} c(e)\cdot x_e \text{ subject to:} \tag{2.7}$$

$$\sum_{(s,t)\in D} y_{s,t} \geq |D|/2$$

$$\sum_{\Pi(s,t)\ni P\ni e} f_P \leq x_e \qquad \forall (s,t)\in D, e\in E$$

$$\sum_{P\in\Pi(s,t)} f_P = y_{s,t} \qquad \forall (s,t)\in D$$

$$0 \leq y_{s,t}, f_P, x_e \leq 1 \qquad \forall (s,t)\in D, P\in\Pi, e\in E$$

Figure 2.3: Linear program LP-DSF for the case $|D - C| > |D|/2$

We can formulate an LP-relaxation for this problem, LP-DSF, shown in Figure 2.3, which closely resembles the LP used by [71] for Directed k-Spanner. Each edge e has a capacity x_e, each path $P \in \Pi$ carries f_P units of flow, and $y_{s,t}$ is the total flow through all paths from s to t. Also, the total flow through all paths in Π should be at least $|D|/2$. It is clear that LP-DSF is a relaxation of the problem of connecting at least $|D|/2$ pairs in D while minimizing the cost of the edges. [93] shows that in polynomial time, we can find a solution $\{\hat{x}_e\} \cup \{\hat{y}_{s,t}\}$ such that $\sum_{e\in E} c(e)\cdot \hat{x}_e$ is within $(1+\epsilon)$ factor of OPT, the optimal solution to LP-DSF, for any fixed $\epsilon > 0$.

Our improvement comes in the analysis of the rounding algorithm for LP-DSF. Suppose $\{\hat{x}_e\} \cup \{\hat{y}_{s,t}\}$ is a feasible solution to LP-DSF. Let K_2 be the edge set obtained by selecting each edge in E with probability $\min((8n\ln n)/\beta \cdot x_e, 1)$.

Lemma 2.5.2. *With probability $\geq 1 - 1/n^2$, set K_2 settles every thin pair (s,t) with $\hat{y}_{s,t} \geq 1/4$.*

Proof. We reinterpret Definition 2.2.4 in terms of edge costs instead of lengths. More precisely, define a set $A \subseteq E$ to be an antispanner for a pair $(s,t) \in D$ if $(V, E\setminus A)$ contains no path from s to t of cost at most ℓ. By exactly the same argument as in Claim 2.2.5, the set of all minimal antispanners for thin pairs is of size at most $n^2(n/\beta)^{n/\beta}$.

For every thin pair $(s,t) \in D$ with $\hat{y}_{s,t} \geq 1/4$, if A is an antispanner for (s,t), then $\sum_{e\in A} \hat{x}_e \geq \sum_{P\in\Pi(s,t)} \hat{f}_P \geq 1/4$, where $\hat{f}_P$ is the value of the variable f_P in LP-DSF that corresponds to the solution $\{\hat{x}_e\}\cup\{\hat{y}_{s,t}\}$. So, the probability that K_2 is disjoint from A is at most $\exp(-(n\ln n)/\beta)$, by the same argument as in Claim 2.2.3. Thus, by the bound on the total number of antispanners of thin pairs from above, the union bound, and Claim 2.2.4, it follows that with high probability, K_2 settles every thin pair (s,t) with $\hat{y}_{s,t} \geq 1/4$. □

We add to K_2 a minimum cost path between any pair (s,t) with $\hat{y}_{s,t} \geq 1/4$ that is still not settled. In expectation, the number of such pairs is $|D|/n^2 \leq 1$, so that the total expected cost[2] of K_2 is at most $(16n\ln n)/\beta\cdot\tau$. A simple argument shows that the number

[2]This is where we save over [93]. The cost of their comparable K_2 is $O(n^2/\beta^2\cdot\tau)$.

of pairs (s,t) in D for which $\hat{y}_{s,t} < 1/4$ is at most $2|D|/3$; assuming the opposite makes the total amount of flow between all pairs strictly less than $|D|/2$. So, the expected density of K_2 is at most:

$$\left(\frac{16n\ln n}{\beta}\cdot\tau\right) / \left(|D| - 2|D|/3\right) = \frac{48n\ln n}{\beta}\cdot\frac{\tau}{|D|} = \tilde{O}(n^{2/3})\cdot\tau/|D|.$$

As we said earlier, the set K is taken to be either K_1 or K_2, depending upon which has the smaller density. By the above, expected density of K is at most $O(n^{2/3+\epsilon})\cdot\tau/|D|$. This concludes the proof of Theorem 2.5.1.

2.6 Conclusion

For general Directed k-Spanner, we obtained an approximation ratio of $\tilde{O}(\sqrt{n})$ and for Directed 3-Spanner with unit edge lengths we obtained an approximation ratio of $\tilde{O}(n^{1/3})$. The second bound almost matches the LP integrality gap of Dinitz and Krauthgamer [71]. It remains an interesting open question whether one can get an approximation ratio of $\tilde{O}(n^{1/3})$ for the general case.

All our algorithms are randomized and have an expected approximation factor. Our algorithms consist of multiple stages and the analysis of concentration of the cost of the solution can be done using standard concentration bounds for each stage separately in the same way as in the previous work (e.g. [35, 71, 93]), so we omit it to simplify presentation. It remains open whether these algorithms can be derandomized.

Chapter 3

Sparsification of Node-Weighted Planar Graphs

In feedback vertex set problems input is a graph $G = (V, E)$, a family of cycles $\mathcal{C}$ in G and a non-negative weight function $w \colon V \to \mathbb{R}^{\geq 0}$ on the set of vertices of G. The goal is to find a set of vertices $H \subset V$ which contains a node in every cycle in $\mathcal{C}$ such that the total weight of vertices in H is minimized. This is a special case of a general hitting set problem, when sets correspond to cycles in the graph.

FEEDBACK VERTEX SET (FVS) problem in a graph is the problem of finding a hitting set for all cycles. We consider the problem of hitting sets for cycles that satisfy some special properties. There are four natural examples.

- Odd cycles. If $H \subset V$ is a hitting set for all odd-length cycles then the subgaph of G, induced by the vertex set $V \setminus H$ is bipartite and removal of H is called *graph bipartization*. The corresponding hitting set problem is denoted as BIPARTIZATION (BIP).
- The set of all cycles which contain at least one node from a given set of nodes. The corresponding hitting set problem is known as SUBSET FEEDBACK VERTEX SET (S-FVS).
- The set of all directed cycles of a given directed graph. The corresponding problem is called DIRECTED FEEDBACK VERTEX SET (D-FVS).
- In NODE-WEIGHTED STEINER FOREST problem we are given a weighted undirected graph and a set of terminal pairs (s_i, t_i). The goal is to select a subset of vertices of the graph of minimum weight such that in the subgraph induced by these vertices all pairs of terminals are connected. In Section 3.1.1 we show that NODE-WEIGHTED STEINER FOREST (NWSF) belongs to a class of problems which can be expressed as hitting set problems for some collection of cycles.

While in general graphs FEEDBACK VERTEX SET can be approximated within factor of 2 for all graphs, as shown by Becker and Geiger [29] and Bafna, Berman and Fujito [19], hitting a restricted family of cycles can be much harder. For example, the best known approximation ratio for graph bipartization in general graphs is $O(\log n)$ by Garg, Vazirani and Yannakakis [99]. For D-FVS the best known approximation is $O(\log n \log \log n)$, as

Problem	Approximation	Hardness of approximation
FVS	2 [29] 2 [19]	MAX-SNP, [144]
BIP	$O(\log n)$ [99]	1.3606, ($P \neq NP$ [73])
D-FVS	$\tilde{O}(\log n)$ [88]	$2-\epsilon$, (UGC, [134])
S-FVS	8 [89]	
NWSF	$O(\log k)$ [137]	$(1-o(1))\log k$ [91], if $NP \not\subseteq ZTIME(2^{n^\eta})$ for some $\eta > 0$

Table 3.1: General graphs

Problem	Previous work (our analysis)[1]	Our work
FVS	10 [22] 3(18/7) [102]	2.4
BIP, D-FVS, S-FVS	3(18/7) [102]	2.4
NWSF	6 [67] 3(18/7) [151]	2.4

Table 3.2: Planar graphs

shown by Even, Naor, Schieber and Sudan [88]. These and other results for general graphs are summarized in Table 3.1.

Yannakakis [195] has given an NP-hardness proof for many vertex deletion problems restricted to planar graphs which applies to all problems that we consider. Also, it is known that D-FVS is NP-hard even if both indegree and outdegree of every vertex are at most 3 (Garey and Johnson, [98, p. 191]). For planar graphs, the unweighted Feedback Vertex Set problem admits a PTAS, as shown by Demaine and Hajiaghayi [68] using a bidimensionality technique. Goemans and Williamson [102] created a framework for primal-dual algorithms that for planar instances of all above problems provide approximation algorithms with constant approximation factors. More specifically, they showed 9/4-approximations for FVS, S-FVS, D-FVS and BIP. For Node-Weighted Steiner Forest it was shown by Demaine, Hajiaghayi and Klein [67] that the generic framework of Goemans and Williamson gives a 6-approximation which was improved to 9/4-approximation by Moldenhauer [151]. However, the original paper by Goemans and Williamson [102] contains a mistake in the analysis. Similar mistake was repeated in [151]. We exhibit the mistake on an example and prove that no worse example exists. More precisely, primal-dual approximation algorithms of Goemans and Williamson for all problems described above give approximation factor 18/7 rather than 9/4. We also give an improved version of the violation oracle which can be used within the primal-dual framework of Goemans and Williamson and guarantees approximation factor 2.4. Results for planar graphs are summarized in Table 3.2.

The edge counterparts of the given problems, i.e. finding a minimum-weight subset of

[1]See discussion in the text.

edges which intersects with every cycle in a given collection, are also well-studied. They reduce to vertex-weighted versions by adding a new vertex on each edge and assigning its weight to be equal to the weight of the edge. These problems are also significantly simpler, especially in planar graphs. FEEDBACK EDGE SET problem is a complement of the maximum spanning tree problem. Minimum-weight graph bipartization by edge removals is complementary to the maximum-weight cut problem which in planar graphs can be solved in polynomial time (Hadlock [113], Dorfman and Orlova [77]). DIRECTED FEEDBACK EDGE SET problem in planar graphs reduces to finding a minimum-weight dijoin in the dual graph which can be solved in polynomial time (see, e.g. Grötschel, Lovász and Schrijver [109, p.253-254]). Edge-weighted STEINER FOREST problem in planar graphs is NP-hard [98], but admits a PTAS, as shown recently by Bateni, Hajiaghayi and Marx [27].

Applications and ramifications Node-weighted Steiner problems have been studied theoretically in many different settings, see e.g. [137, 154, 174, 145]. Applications of such problems range from maintenance of electric power networks [110] to computational sustainability [69]. Experimental evaluation of primal-dual algorithms for feedback vertex set problems in planar graphs in applications to VLSI design was shown by Kahng, Vaya and Zelikovsky [130].

As observed by Goemans and Williamson primal-dual algorithms for feedback vertex set problems in planar graphs have close connections to conjectures of Akiyama and Watanabe and Gallai and Younger about the size of minimum feedback vertex set in planar graphs. See [102] for more details.

Organization We give basic definitions and preliminary observations in Section 3.1. In Section 3.1.1 we show that a wide class of node-weighted network design problems in planar graphs, introduced by Demaine, Hajiaghayi and Klein [67], can be equivalently defined as a class of hitting set problems for appropriately defined collections of cycles satisfying *uncrossing property*, as introduced by Goemans and Williamson [102]. In Section 3.2 we introduce local-ratio analog of primal-dual framework of Goemans and Williamson for such problems and give examples of violation oracles which can be used within this framework.

In Section 3.3 we give a corrected version of the analysis of the approximation factor achieved by the generic primal-dual algorithm with a violation oracle, presented by Goemans and Williamson in [102]. In A.3 we present analysis of primal-dual algorithms with a new violation oracle which gives approximation factor 2.4. In Section 3.3.4 we show examples, on which these approximation factors are achieved.

3.1 Preliminaries

A *simple* cycle of length k is a sequence of vertices $v_1, \ldots, v_{k+1}$, where $v_{k+1} \equiv v_1$, all vertices $v_1, \ldots v_k$ are distinct, $(v_i, v_{i+1}) \in E$ for all $1 \le i \le k$ and all these edges are distinct. When working with simple graphs, the edge set above is uniquely defined. Note that in undirected simple graphs a simple cycle has length at least three. For a cycle C, the edge set of C is denoted as $E(C)$, although to simplify presentation we will abuse the notation slightly and sometimes refer to it as just C.

Every planar graph has a combinatorial embedding that for each vertex specifies a cyclic ordering of edges that are adjacent to it. A subset $U \subset V$ defines $G[U]$, the *induced* *subgraph* of G, with node set U and edges $\{(u,v) \in E : u,v \in U\}$. An embedding of a planar graph naturally defines embeddings of all its induced subgraphs. We denote the set of faces of a planar graph as F (for a standard definition of the set of faces via a combinatorial embedding, see e.g. [136]). The *planar dual* of a graph G is graph $G^* = (F, E')$ where F is the set of faces of G, and E' is the set of pairs of faces that share an edge. We select one face F_0 as the *outer face.*

For a simple cycle $C = (v_1, \ldots, v_{k+1})$ we denote the set of faces that are surrounded by C as $Faces(C)$. More formally, let E'' be the set of pairs of faces that share an edge that is not on C then in (F, E'') has exactly two connected components. We denote as $Faces(C)$ the connected component of (F, E'') that does not contain the outer face F_0. A family of cycles $\mathcal{Z}$ is *laminar* iff for every $C, D \in \mathcal{Z}$ either $Faces(C) \subset Faces(D)$, or $Faces(D) \subset Faces(C)$, or $Faces(D) \cap Faces(C) = \varnothing$.

We use notation $\bullet$ to denote *contact* between two objects. More formally $u \bullet V$, if $u \subseteq V$. For example, we can have nodes and edges in contact with faces and cycles.

3.1.1 Uncrossable families of cycles and proper functions

Our algorithms apply to every family of cycles that satisfies the following:

Definition 3.1.1 (Uncrossing property [102]). *For any two cycles $C_1, C_2 \in \mathcal{C}$ such that there exists a path P_2 in C_2 which is edge-disjoint from C_1 and which intersects C_1 only at the endpoints of P_2, the following must hold. Let P_1 be a path in C_1 between the endpoints of P_2. Then either $P_1 \cup P_2 \in \mathcal{C}$ and $(C_1 \setminus P_1) \cup (C_2 \setminus P_2)$ contains a cycle in $\mathcal{C}$, or $(C_1 \setminus P_1) \cup P_2 \in \mathcal{C}$ and $(C_2 \setminus P_2) \cup P_1$ contains a cycle in $\mathcal{C}$.*

Many natural families of cycles satisfy the *uncrossing property.* Goemans and Williamson [102] showed this for FVS, D-FVS, BIP, and S-FVS. We show that a certain class of node-weighted connectivity problems in planar graphs can be expressed as problems of finding hitting sets for families of cycles satisfying the uncrossing property. To formalize this statement we introduce some definitions.

Definition 3.1.2 ((0, 1)-proper function). *A Boolean function $f : 2^V \to \{0,1\}$ is* proper *if $f(\emptyset) = 0$ and it satisfies the following two properties:*

1. *(Symmetry) $f(S) = f(V \setminus S)$.*
2. *(Disjointness) If $S_1 \cap S_2 = \emptyset$ and $f(S_1) = f(S_2) = 0$ then $f(S_1 \cup S_2) = 0$.*

These two properties imply the property known as *complementarity*: if $A \subseteq S$ and $f(S) = f(A) = 0$ then $f(S \setminus A) = 0$.

For a set $S \subseteq V$, let $\Gamma(S)$ be its boundary, i.e. the set of nodes not in S which have a neighbor in S, or formally $\Gamma(S) = \{v \in V | v \notin S, \exists u \in S : (u,v) \in E\}$. As observed by Demaine, Hajiaghayi and Klein [67], a wide class of node-weighted network design problems can be formulated as the following generic integer program, where $f : 2^V \to \{0,1\}$ is a proper function:

$$\text{Minimize: } \sum_{v \in V} w(v)x(v)$$

$$\text{Subject to: } \sum_{v \in \Gamma(S)} x(v) \geq f(S) \qquad \text{for all } S \subseteq V$$
$$x(v) \in \{0,1\} \qquad \text{for all } v \in V,$$

For example, for Node-Weighted Steiner Forest the corresponding $(0,1)$-proper function is defined as follows: $f(S) = 1$ iff there exists a pair of terminals (s_i, t_i), such that $|S \cap \{s_i, t_i\}| = 1$. The edge-weighted version of this program was introduced by Goemans and Williamson in [101]. Note that without loss of generality we can assume that the input graph is triangulated. Otherwise we add extra nodes of infinite cost inside each face and connect these new nodes by edges to all nodes on their faces without changing the cost of the optimum solution.

In Theorem 3.1.1 we will show that problems which can be expressed by a generic integer program above with some $(0,1)$-proper function f can also be expressed as problems of hitting uncrossable collections of cycles. We give some definitions and simplifying assumptions first.

Definition 3.1.3 (Active sets and boundaries). *For a proper function $f\colon 2^V \to \{0,1\}$ we say that sets S, such that $f(S) = 1$ are* active. *For an active set S we refer to its boundary $\Gamma(S)$ as* active boundary. *If an active boundary forms a simple cycle we call it* active simple boundary. *We denote the collection of all active simple boundaries as $\mathcal{C}_A^f$.*

Using this terminology, the generic integer program expresses the problem of finding a minimum weight hitting set for the collection of all active boundaries. Note that we can assume that all active singleton sets are included into the solution because each such set $\{s\}$ forms a boundary of its complement $V \setminus \{s\}$, which is active by symmetry, and thus $\{s\}$ has to be hit. We simplify the generic integer program using this observation so we can assume that all active boundaries don't contain active singleton sets. In particular, this implies by disjointntess of f that all active boundaries $\Gamma(S)$ are not active, namely $f(\Gamma(S)) = 0$.

We first show that finding a minimim weight hitting set for all active boundaries is equivalent to finding a minimum weight hitting set for $\mathcal{C}_A^f$ by showing that every active boundary contains an active simple boundary as a subset in Lemma A.2.1. Then we show that the family of active simple boundaries $\mathcal{C}_A^f$ satisfies the uncrossing property.

Theorem 3.1.1. *Let $G(V,E)$ be a triangulated planar graph. For every proper function $f\colon 2^V \to \{0,1\}$ the collection of active simple boundaries $\mathcal{C}_A^f$ forms an uncrossable family of cycles.*

Proof. Consider two active simple boundaries $\Gamma(S_1)$ and $\Gamma(S_2)$. If $\Gamma(S_2)$ crosses $\Gamma(S_1)$ then there exists a collection of edge-disjoint paths in $\Gamma(S_2)$ which we denote as P, such that each path $P_i \in P$ has only two nodes in common with $\Gamma(S_1)$. Each path $P_i \in P$ partitions $S_1 \setminus P_i$ into two parts which we denote as A_i^1 and A_i^2 respectively. Let's fix a path $P_i \in P$, such that at A_i^1 doesn't contain any other paths from P.

There are two cases: $A_i^1 \cap S_2 = \emptyset$ and $A_i^1 \subseteq S_2$. They are symmetric because if $A_i^1 \subseteq S_2$ we can replace the set S_2 by a set $S_2' = V \setminus S_2 \setminus \Gamma(S_2)$, ensuring that $A_i^1 \cap S_2 = \emptyset$. Note that the boundary doesn't change after such replacement, because $\Gamma(S_2) = \Gamma(S_2')$. By symmetry of f we have that $f(S_2) = f(V \setminus S_2) = 1$. Because $f(\Gamma(S_2)) = 0$ by disjointness we have $f(V \setminus S_2 \setminus \Gamma(S_2)) = f(S_2') = 1$, so S_2' is also an active set.

This is why it is sufficient to consider only the case when $A_i^1 \cap S_2 = \emptyset$. We will show the following auxiliary lemma:

Lemma 3.1.2. *Let $A_1, A, B \subseteq V$ be such that $A_1 \subseteq A$, $A_1 \cap B = \emptyset$ and $f(A) = f(B) = 1$. Then at least one of the following two statements holds:*

1. $f(A_1 \cup B) = f(A \setminus A_1) = 1$.
2. $f(A_1) = \max\left[f(B \setminus (A \setminus A_1)), f((A \setminus A_1) \setminus B)\right] = 1$.

The proof of the lemma follows from the properties of $(0,1)$-proper functions and is given in A.2.3

To show the uncrossing property for cycles $C_1 = \Gamma(S_1)$ and $C_2 = \Gamma(S_2)$ we select the paths in the definition of the uncrossing property as $P_1 = \Gamma(A_i^2) \setminus P_i$ and $P_2 = P_i$. Now we can apply Lemma 3.1.2 to sets A_i^1, S_1 and S_2, because $A_i^1 \subseteq S_1$, $A_i^1 \cap S_2 = \emptyset$ and $f(S_1) = f(S_2) = 1$. Thus, by Lemma 3.1.2 either $f(A_i^1 \cup S_2) = f(S_1 \setminus A_i^1) = 1$ or $f(A_i^1) = max(f(S_2 \setminus (S_1 \setminus A_i^1)), f((S_1 \setminus A_i^1) \setminus S_2)) = 1$. In the first case we have $f(A_i^2) = f(A_i^1 \cup S_2) = 1$ and thus both cycles $P_1 \cup P_2 = \Gamma(A_i^2)$ and $(C_1 \setminus P_1) \cup (C_2 \setminus P_2) = \Gamma(A_i^1 \cup S_2)$ are active simple boundaries. In the second case $f(A_1) = 1$ and thus the cycle $(C_1 \setminus P_1) \cup P_2 = \Gamma(A_1)$ is an active simple boundary. The cycle $(C_2 \setminus P_2) \cup P_1$ is not necessarily simple, but it forms a boundary of an active set $(S_2 \setminus (S_1 \setminus A_i^1)) \cup ((S_1 \setminus A_i^1) \setminus S_2)$. Thus, by Lemma A.2.1 it contains an active simple boundary, which is a cycle in $\mathcal{C}_A^f$.

□

3.2 Algorithm

3.2.1 Generic local-ratio algorithm

We will use a local-ratio analog of a generic primal-dual algorithm formulated by Goemans and Williamson [102] which we state as Algorithm 3.1

Algorithm 3.1 Generic local-ratio algorithm $(G(V,E), w, \mathcal{C})$.

```
1: $\bar{w} \leftarrow w$.
2: $S \leftarrow \{u \in V : \bar{w}(u) = 0\}$.
3: while $S$ is not a hitting set for $\mathcal{C}$ do
4:     $\mathcal{M} \leftarrow$ a collection of cycles returned by a violation oracle VIOLATION$(G, \mathcal{C}, S)$.
5:     $c_{\mathcal{M}}(u) \leftarrow |\{M \in \mathcal{M} : u \bullet M\}|$, for all $u \in V$.
6:     $\alpha \leftarrow \min_{u \in V \setminus S} \frac{\bar{w}(u)}{c_{\mathcal{M}}(u)}$.
7:     $\bar{w}(u) \leftarrow \bar{w}(u) - \alpha c_{\mathcal{M}}(u)$, for all $u \in V$.
8:     $S \leftarrow \{u \in V : \bar{w}(u) = 0\}$.
9: end while
10: return a minimal hitting set $H \subset S$ of $\mathcal{C}$.
```

Note that we don't need to specify the collection of cycles $\mathcal{C}$ explicitly. instead the generic algorithm requires that we specify an oracle VIOLATION$(G, \mathcal{C}, S)$ used in Step 4.

Given a graph G, collection of cycles $\mathcal{C}$ and a solution S if there are some cycles in $\mathcal{C}$ which are not hit by S this oracle should return a non-empty collection of such cycles, otherwise it should return the empty set. Such an oracle also allows to perform Step 3 and Step 10 without explicitly specifying $\mathcal{C}$.

The performance guarantee of the generic algorithm depends on the oracle used as described below. If $z: Z \to \mathbb{R}$ we use $z(Z)$ to denote $\sum_{a \in Z} z(a)$.

Theorem 3.2.1 (Local-ratio analog of Theorem 3.1 in [102])**.** *If the set $\mathcal{M}$ returned by a violation oracle used in Step 4 of the generic local-ratio Algorithm 3.1 satisfies that for any minimal solution $\breve{H}$:*

$$c_{\mathcal{M}}(\breve{H}) \le \gamma |M|,$$

then Algorithm 3.1 returns a hitting set H of cost $w(H) \le \gamma w(H^*)$, where H^* is the optimum solution.

We give the proof of this theorem for completeness in A.2.2.

The simplest violation oracles return a single cycle. Bar-Yehuda, Geiger, Naor and Roth [22] show that for FVS this approach can give a 10-approximation for planar graphs and Goemans and Willamson [102] improve it to a 5-approximation. They also analyzed an oracle, which returns a collection of all faces in $\mathcal{C}$, which are not hit by the current solution, and showed such oracle gives a 3-approximation for all families of cycles satisfying uncrossing property. Thus, by Theorem 3.1.1 such oracle gives a 3-approximation for all problems that we consider. We now give more complicated examples of violation oracles which give better approximation factors.

3.2.2 Minimal pocket violation oracles

The following oracle, introduced by Goemans and Williamson [102], returns a collection of faces in $\mathcal{C}$ inside a minimal *pocket* not hit by the current solution H.

Definition 3.2.1. *A* pocket *for a planar graph $G(V, E)$ and a cycle collection $\mathcal{C}$ is a set $U \subseteq V$ such that:*

1. *The set U contains at most two nodes with neighbors outside U.*
2. *The induced subgraph $G[U]$ contains at least one cycle in $\mathcal{C}$.*

Algorithm 3.2 Minimal-Pocket-Violation $(G, \mathcal{C}, S)$.

```
1: $\mathcal{C}_0 \leftarrow \{c \in \mathcal{C} : c$ not hit by $S\}$
2: Construct a graph $G_S$ by removing from $G$:
    3: All edges which do not belong to any cycle in $\mathcal{C}_0$.
    4: All vertices which are not adjacent to any edges.
5: Let $U_0$ be a pocket for $G_S$ and $\mathcal{C}_0$ which doesn't contain any other pockets.
6: return A collection of all cycles in $\mathcal{C}_0$ which are faces of $G_S[U_0]$.
```

As in the generic algorithm, we will not specify $\mathcal{C}$ and $\mathcal{C}_0$ explicitly, but will rather use an oracle to check relevant properties with respect to them. We show analysis of the approxiamtion factor obtained with this oracle in Section 3.3.

We will obtain a better approximation ratio by analyzing the following oracle in Section A.3.

Definition 3.2.2. *A* triple pocket *for a planar graph $G(V, E)$ and a cycle collection $\mathcal{C}$ is a set $U \subseteq V$ such that:*

1. *The set U contains at most three nodes with neighbors outside U.*
2. *The induced subgraph $G_S[U]$ has at least three faces in $\mathcal{C}$.*

The violation oracle Minimal-3-Pocket-Violation finds a minimal U_0 that is either a pocket or a triple pocket, and otherwise works like Minimal-Pocket-Violation.

3.3 Proof of 18/7 approximation ratio with pocket oracle

According to Theorem 3.2.1, to show that Algorithm 1 has approximation factor 18/7 it suffices to prove the following:

Theorem 3.3.1. *In every iteration of the generic local-ratio algorithm (Algorithm 3.1) with oracle* Minimal-Pocket-Violation *for every minimal hitting set $\breve{H}$ of $\mathcal{C}$ we have $c_{\mathcal{M}}(\breve{H}) \le \gamma|\mathcal{M}|$ for $\gamma = 18/7$.*

The rest of this section is the proof of this theorem. The strategy of the proof is a variant of amortized analysis. We consider an arbitrary minimal hitting set $\breve{H}$ of $\mathcal{C}$-cycles in the residual graph G_S (as defined in Algorithm 3.2). For every cycle in $\mathcal{M}$ we get 1 unit of credit, and for a node $h \in \breve{H}$ we get $c_{\mathcal{M}}(h)/\gamma$ units of debit (*i.e.* negative), and we need to show that overall balance is non-negative. We start by decomposing the balance into smaller parts which are simpler to analyze than the balance of the entire pocket. The goal is to limit the impact of the nodes in $\breve{H}$ for which witness cycle A_h is not in $\mathcal{M}$. We decompose the pocket into parts that have at most two such nodes (Section 3.3.1). Further analysis refers to one such part.

We further simply the analysis of the balance by applying a pruning rule, each application of the pruning rule makes the instance smaller while the balance decreases. Thus it is enough to prove the claim when the pruning rule cannot be applied. In particular, this proves the claim if pruning produces an instance with no credits and debits. (Section 3.3.2).

Finally (Section 3.3.3) we define objects called envelopes and we assign all credits and debits to the envelopes. Then we show that each envelope has a non-negative balance. The nature of the pocket oracle eliminates conceivable envelopes with negative balance. In the next section we show that we eliminate more types of envelopes with the oracle Minimal-3-Pocket-Violation which gives an approximation factor 12/5.

Before we proceed we need several definitions.

Definition 3.3.1 (See also [102])**.**

(a) *Given a hitting set $\breve{H}$ for $\mathcal{C}$ we say that $A \in \mathcal{C}$ is a* witness cycle *for $h \in \breve{H}$ if $A \cap \breve{H} = \{h\}$.*
(b) *If $\breve{H}$ is a minimal hitting set, we can select $\mathcal{A}_{\breve{H}} = \{A_h : \ h \in \breve{H}\}$, a* family of witness cycles *for $\breve{H}$.*
(c) *Given a pocket $G_S[U_0]$ with $\mathcal{M}$ being set of faces of $G_S[U_0]$ that are in $\mathcal{C}$ we define* debit graph*, a bipartite graph $\mathcal{G} = (\mathcal{M} \cup \breve{H}, \mathcal{E})$ with edges $\mathcal{E} = \{(M, h) \in \mathcal{M} \times \breve{H} : \ M \bullet h\}$.*

(d) *For $\mathcal{N} \subset \mathcal{M}$ we define $\mathcal{E}_{\mathcal{N}} = \{(M,h) \in \mathcal{E} :\ M \in \mathcal{N}\}$ and $balance(\mathcal{N}) = |\mathcal{N}| - |\mathcal{E}_{\mathcal{N}}|/\gamma$.*

Goemans and Williamson showed the following:

Lemma 3.3.2 (Lemma 4.2 in [102])**.** *For every collection of cycles $\mathcal{C}$ and every minimal hitting set $\breve{H}$ there exists a laminar family of witness cycles $\mathcal{A}_{\breve{H}}$.*

Observe also that the planar embedding of G_S defines a planar embedding of $\mathcal{G}$. We are going to use the fact that $\mathcal{G}$ is planar. By the definition, $c_{\mathcal{M}}(H) = |\mathcal{E}|$, so to prove Lemma 3.3.1 it suffices to show that

$$balance(\mathcal{M}) \geq 0 \tag{3.1}$$

We prove inequality (3.1) using mathematical induction on $|\mathcal{M} \cup \mathcal{A}_{\breve{H}}|$.

3.3.1 Complex witness cycles and decomposition of the debit graph

In this subsection we will show a sufficient condition for inequality (3.1) and thus for our theorem.

Definition 3.3.2. *If $A_g \in \mathcal{A}_h min$ and $A_g \notin \mathcal{M}$ we say that $A_g \in \mathcal{A}_{\breve{H}}$ is a* complex witness cycle *and that g is an outer (hit) node.*

Complex witness cycle A_h makes the analysis more complicated because there exist debits associated with pairs (M, h) but there is no credit for A_h. We reduce the problem of proving the non-negative balance of the debit graph to the problem of proving sufficiently high balances in simpler parts of that graph, where a part may have at most two complex witness cycles. There are two types of complex witness cycles:

Definition 3.3.3. *Let $\mathcal{C}_g = Faces(A_g) \cap \mathcal{C}$ and $\mathcal{M}_g = \mathcal{C}_g \cap \mathcal{M}$.*

If all nodes of a complex witness cycle A_g are in the pocket U_0 we say that A_g is a hierarchical witness cycle*. Otherwise both contact nodes of U_0 belong to A_g and we say that A_g is a* crossing witness cycle.

First we discuss how to handle the hierarchical witness cycles. If there are such cycles, one of them, say A_g, has a minimal set $Faces(A_g)$, and for such a cycle A_g we will show that

$$balance(\mathcal{M}_g) \geq 1 - 1/\gamma \tag{3.2}$$

As a consequence of this inequality we can simplify the debit graph $\mathcal{G}$ by removing $\mathcal{M}_g$ and inserting in its place A_g. After that replacement $\mathcal{M}_g$ becomes $\{A_g\}$ and $\mathcal{E}_{\mathcal{M}_g}$ becomes $\{(A_g, g)\}$, so $balance(\mathcal{M}_g)$ changes to $1 - 1/\gamma$. The replacement reduces $|\mathcal{A}_{\breve{H}} \cup \mathcal{M}|$ while the inequality (3.2) assures that the $balance(\mathcal{E})$ does not increase. This allows to invoke the inductive hypothesis.

We can repeat the simplification as long as there exists a hierarchical witness cycle. Note that we in the set of pairs $\mathcal{E}_{\mathcal{M}_g}$ (of the form "cycle M, hit node h") only g is a hit node with a complex witness cycle.

Now it remains to consider the case when for every hit node g that occurs in pairs of $\mathcal{E}$, if its witness cycle A_g is complex then A_g is a crossing witness cycle. Each such A_g contains a path $P_g \subset U_0$ between the two contact nodes of U_0, these paths cannot cross each other, thus they split pocket U_0 into *subpockets*; a subpocket has the boundary contained in two paths that either are of the form P_g or form a part of the outer face of $G[U_0]$.

Let $\mathcal{X}$ be the set of faces of $\mathcal{M}$ that are contained in a subpocket. It is sufficient to show that if $\mathcal{X} \neq \varnothing$ then

$$balance(\mathcal{X}) \geq 1 - 2/\gamma. \tag{3.3}$$

We established that it suffices two consider two types of simplified debit graphs and to prove that they respectively satisfy inequalities (3.2) and 3.3. We can describe a more general type of a debit graph that includes both types as special cases. In our first type, A_g is a hierarchical witness cycle that does not contain other hierarchical witness cycles in its interior and we can define W, the set of nodes of G_S that are on A_g or its interior. In our second type, P_g and P_h are two paths connecting the contacts of U_0 and W is the set of nodes of G_S that are on these two paths or in the interior of the cycle formed by these two paths; W may contains outer nodes only on these paths. Let $\mathcal{M}_W$ be the set of faces of $G[W]$ that are in $\mathcal{M}$, the inequalities (3.2) and (3.3) have LHS equal to $balance(\mathcal{M}_W)$.

Moreover, RHS of (3.2) and (3.3) also can be written with the same expression: if a is the number of the outer nodes in W than in both cases RHS $= 1 - a/\gamma$. The next lemma summarizes these observations.

Lemma 3.3.3. *To prove (3.1) it suffices to prove the following. Consider $W \subset U_0$ such that $\breve{H} \cap W$ contains $a \leq 2$ outer nodes, all on the outer face of $G[W]$. Let $\mathcal{M}_W$ be the set of faces in $\mathcal{M}$ that have all nodes in W. Then*

$$balance(\mathcal{M}_W) \geq 1 - a/\gamma \tag{3.4}$$

In the remainder of this section we will refer to W and a as introduced in this lemma, and we will assume that we have at most a outer nodes, all on the outer face of W.

3.3.2 Pruning

The following operation simplifies $\mathcal{M}_W$ and thus facilitates the proof of inequality (3.4).
Pruning $\mathcal{M}$ cycles of small degree. If a cycle $M \in \mathcal{M}_W \setminus \mathcal{A}_{\breve{H}}$ participates in at most two edges in $\mathcal{E}$, we remove M from $\mathcal{M}$. If $A_h \in \mathcal{M}_W$ for some $h \in \breve{H}$ and h participates in at most two edges in $\mathcal{E}$, including (A_h, h), we remove h from $\breve{H}$ and A_h from $\mathcal{M}$.

Lemma 3.3.4. *Each step of pruning decreases $|\mathcal{M}_W|$ by exactly 1 and $balance(\mathcal{M}_W)$ by at least $1 - 2/\gamma$.*

If $\mathcal{M}_W \neq \varnothing$ and applications of pruning lead to $\mathcal{M}_W = \varnothing$ then before pruning inequality (3.4) was true.

Proof. The first claim follows from the fact that a step of pruning decreases $|\mathcal{M}_W|$ by 1 and $\mathcal{E}_{\mathcal{M}_W}$ by at most 2.

The second claim for $a = 2$ follows from the first claim because we apply at least one step of pruning.

The second claim for $a = 1$ follows from the fact that before the last step of pruning we have $|\mathcal{M}_g| = 1$, *i.e.* $\mathcal{M}_g = \{M\}$ for some face M. If M is not a witness cycle then all witness cycles except A_g were eliminated, hence $\mathcal{E}_{\mathcal{M}_W}$ consists of exactly one pair (M, g). If M is a witness cycle A_h, then $\mathcal{E}_{\mathcal{M}_W}$ consists of exactly one pair (A_h, h). In both cases $balance(\mathcal{M}_g) = 1 - 1/\gamma$. $\square$

3.3.3 Envelopes

It remains to prove inequality (3.4) when pruning cannot be applied and $\mathcal{E}_{\mathcal{M}_W}$ is non-empty. We decompose $balance(\mathcal{M}_W)$ into parts that can be analyzed using Euler formula.

We partition the set of all faces of $G[W]$ into three parts as follows: $\mathcal{A} = \mathcal{M}_W \cap \mathcal{A}_{\breve{H}}$, $\mathcal{B} = \mathcal{M}_W \setminus \mathcal{A}_{\breve{H}}$ and $\mathcal{Z}$ is the set of faces in W which are not in $\mathcal{M}$. Consider the dual graph $G^* = (\mathcal{A} \cup \mathcal{Z}, E^*)$; we have an edge between two vertices of G^* if the corresponding faces share an edge. For every connected component C_i of G^* let E_i be the set of its *boundary edges* which are edges that are adjacent to one face in $C_i \subset \mathcal{A} \cup \mathcal{Z}$ and one face in $\mathcal{B}$. Every E_i forms a cycle (not necessarily simple) and we call such cycles *envelopes*.

Definition 3.3.4 (Envelopes in W)**.** Envelopes *in W are cycles consisting of boundary edges of connected components in the dual of $\mathcal{A} \cup \mathcal{Z}$. We assign each $h \in \mathcal{A}_{\breve{H}_W}$ to an envelope which contains h; we select the one which is on the boundary of the component of $\mathcal{A} \cup \mathcal{Z}$ that contains A_h. If h is an outer node, select the envelope which is on the boundary of the component of $\mathcal{A} \cup \mathcal{Z}$ that contains the other face of $G[W]$. An* outer envelope *is an envelope to which we assigned an outer node.*

We will split $balance(\mathcal{M}_W)$ into balances of envelopes; an edge (M, h) is assigned to the envelope where we assigned h together with its witness cycle A_h; below we explain how to assign faces from $\mathcal{B}$ to envelopes. We will show that each envelope has a non-negative balance and at least one envelope has a sufficient positive balance.

Envelopes do not have to be simple cycles, but balance for non-simple envelopes is positive, we omit the details in the preliminary version, instead, we assume that each envelope is a simple cycle.

Envelopes: normal form. For an envelope $\mathbf{S}$ we define *principal neighbors*, $\mathcal{B}$-faces that have edges on $\mathbf{S}$. Because we do not have pockets inside W, $\mathbf{S}$ must have at least 3 principal neighbors, except for the outer envelope. One can see that a traversal of $\mathbf{S}$ is a face of $\mathcal{G}$ of the form

$$(h_0, B_0, h_1 \ldots, h_{a-1}, B_{a-1}, h_0).$$

Envelopes: normalizing neighbors. First, suppose that some $B \in \mathcal{B}$ contains both h_i and h_{i+1} but $B \neq B_i$. This case is excluded, because if there are some $\mathcal{M}$ faces between B and B_i we would have a pocket, and if there are not, B_i would have at most two neighbors: h_i and h_{i+1}, and we have pruned $\mathcal{M}$-cycles with at most two neighbors in $\mathcal{E}_{\mathcal{M}_W}$. (Observe that h_i, h_{i+1} do not have to belong to $\breve{H}$.)

Now we modify $\mathcal{E}_{\mathcal{M}_W}$ to $\mathcal{E}'$ as follows:

- we eliminate pairs of the form (A_h, h);
- for each envelope $\mathbf{S}$ we contract nodes of $\breve{H}$ that were assigned to $\mathbf{S}$ to a single node $\mathbf{S}$.

The resulting graph is bipartite, but it can be a multi-graph. Therefore we modify it further: if B_i is a principal neighbor of $\mathbf{S}$ and (B, h_i), $(B, h_{i+1}) \in \mathcal{E}_{\mathcal{M}_W}$, in $\mathcal{E}'$ these two edges produce a single double edge. The edge set $\mathcal{E}'$ may remain a multigraph, *e.g.*

if $B = B_i$ the same as outer neighbor B_j then we have two double edges from B to $\mathbf{S}$. However, the edges from B to $\mathbf{S}$ cannot be consecutive in the circular ordering of $\mathbf{S}$, hence this multigraph does not have faces with two edges only.

Note that non-outer hit nodes assigned to an envelope $\mathbf{S}$ are contact nodes of $\mathbf{S}$; otherwise such a node h would belong to exactly two edges of $\mathcal{E}$: (A_h, h) and (B, h) where B is the principal neighbor that contains $\mathbf{S}$; in that case h and A_h would be eliminated by pruning.

Let $n_{\mathbf{S}}$ be the number of contact nodes of $\mathbf{S}$, and let $n_{\mathbf{S}} - \ell_{\mathbf{S}}$ be the number of hit nodes assigned to $\mathbf{S}$. Also, let $d_{\mathbf{S}}$ be the number of edges of the modified graph that are incident to $\mathbf{S}$.

If $\mathbf{S}$ is not the outer envelope, $n_{\mathbf{S}} \geq 3$.

Balance of envelopes We define *balance*$_{\mathbf{S}}$ as the part of *balance*$(\mathcal{E}_{\mathcal{M}_W})$ that can be attributed to $\mathbf{S}$. We will consider $\mathcal{E}_{\mathcal{M}_W}$ edges that are incident to H-nodes of $\mathbf{S}$, and $\mathcal{A}$-cycles of these nodes, and a "share" of B-cycles that can be attributed to $\mathbf{S}$ using $d_{\mathbf{S}}$ and the Euler formula.

Suppose that the modified graph has m nodes, d edges and f faces. Note that $d = \sum d_{\mathbf{S}}$. Also, $m = b + s$ where b is the number of faces in $\mathcal{B}$ and s is the number of envelopes.

Because each face has at least 4 edges and each edge is in two faces, we have $f \leq d/2$, hence the Euler formula implies $m = d - f + 2 \geq d/2 + 2$.

Thus we can allocate $d_{\mathbf{S}}/2$ nodes to an envelope $\mathbf{S}$, because one of these nodes is $\mathbf{S}$, we allocate $d_{\mathbf{S}}/2 - 1$ B-nodes, while to the outer envelopes we can allocate extra two $\mathcal{B}$-nodes.

To simplify the estimate of the number of pairs in $\mathcal{E}$ that are incident to $\mathbf{S}$ we assume first that $\ell_{\mathbf{S}} = 0$. For an non-outer envelope $\mathbf{S}$ we allocate $n_{\mathbf{S}}$ $\mathcal{A}$-faces, and this gives this estimate:

$$\begin{aligned}
&& \mathit{balance}_{\mathbf{S}} = && \\
&& n_{\mathbf{S}} + d_{\mathbf{S}}/2 - 1 - (2n_{\mathbf{S}} + d_{\mathbf{S}})/\gamma = && \\
&& \frac{3}{2}n_{\mathbf{S}} - 1 + (d_{\mathbf{S}} - n_{\mathbf{S}})/2 - (3n_{\mathbf{S}} + (d_{\mathbf{S}} - n_{\mathbf{S}}))/\gamma \geq && \gamma > 2 \\
&& \frac{3}{2}n_{\mathbf{S}} - 1 - 3n_{\mathbf{S}}/\gamma \geq && n_{\mathbf{S}} \geq 3 \\
&& \frac{7}{2} - 9/\gamma = 0 && \gamma = \frac{18}{7}
\end{aligned}$$

For every contact node h that is not a hit node (counted by $\ell_{\mathbf{S}}$) we decrease the number of credits in *balance*$(\mathbf{S})$ by 1, as we do not have the credit of A_h, and the number of debits by $3/\gamma$ as we do not have (A_h, h), (B, h) and (B', h) where B, B' are the principal neighbors adjacent to h on $\mathbf{S}$. Thus *balance*$(\mathbf{S}) \geq \ell_{\mathbf{S}}(3/\gamma - 1)$.

To estimate the balance of outer envelopes, observe that each outer envelope is in a separate connected component of the modified graph: all outer nodes are on the outer face of $G[W]$ so they are in the same connected component of G^*, hence an outer envelope is a connected component of edges of the boundary of that component of faces. In turn, $\mathcal{B}$-nodes of the modified graph are separated between the interiors of the outer envelopes. As a result, we can add 2 to the lower bound of each outer envelope.

Therefore an outer envelope $\mathbf{S}$ with a' outer nodes has balance at least

$$\begin{aligned}
&\frac{3}{2}n_{\mathbf{S}} - 1 - 3n_{\mathbf{S}}/\gamma + 2 - a'(1-1/\gamma) = \\
&3n_{\mathbf{S}}\left(\frac{1}{2} - 1/\gamma\right) + 1 - a'(1-1/\gamma) \geq && n_{\mathbf{S}} \geq 2 \\
&\quad 4 - 6/\gamma - a'(1-1/\gamma) > (2-a')(1-1/\gamma) && \gamma \geq 2
\end{aligned}$$

For $a' < 2$ this gives the desired claim. For $a' = 2$ and $n_{\mathbf{S}}$ one can improve the estimate to $2(1-2/\gamma)$. Thus we have proven inequality (3.4) which suffices to prove Theorem 3.3.1, *i.e.* we showed that oracle Minimal-Pocket-Violation guarantees approximation factor 18/7. We generalize this proof to show that oracle Minimal-3-Pocket-Violation gives approximation factor 12/5 in Theorem A.3.1, which is deferred to A.3.

3.3.4 Tight examples

We show instances of graphs, on which the primal-dual algorithm with oracles Minimal-Pocket-Violation and Minimal-3-Pocket-Violation gives 18/7 and 12/5 approximations respectively.

Our examples are for the Subset Feedback Vertex Set problem. Recall that in this problem we need to hit all cycles which contain a specified set of "special" nodes. Our examples are graphs with no *pockets* (or triple pockets), in which every face belongs to the family of cycles that we need to hit – this is ensured by selection of "special" nodes, which are marked with a star. The weights of vertices are assigned as follows. Given a node u with degree $d(u)$, its weight is $w(u) = d(u)$ if u is a solid dot and $w(u) = d(u) + \epsilon$ otherwise (for some negligibly small value of ϵ).

First we show an example for the oracle Minimal-Pocket-Violation in Figure 3.1. Because there are no pockets, the first execution of the violation oracle returns the collection of all faces in the graph. Thus, in each building block of Picture 3.1 (which shows 5 such blocks from left to right), the primal-dual algorithm selects the black dots with total weight 18 while stars also form a valid solution with weight $7 + 3\epsilon$. Hence the ratio will be arbitrarily close to 18/7, if we repeat the building block many times.

Similar family of examples for the oracle Minimal-3-Pocket-Violation is shown in Figure 3.2. In these examples there are no pockets or triple pockets, so the oracle Minimal-3-Pocket-Violation returns the collection of all faces in the graph. As above, the primal-dual algroithm selects the black dots with total weight 12 within each block, while the cost of the solution given by the stars is $5+2\epsilon$, so we can make the ratio arbitarily close to 12/5.

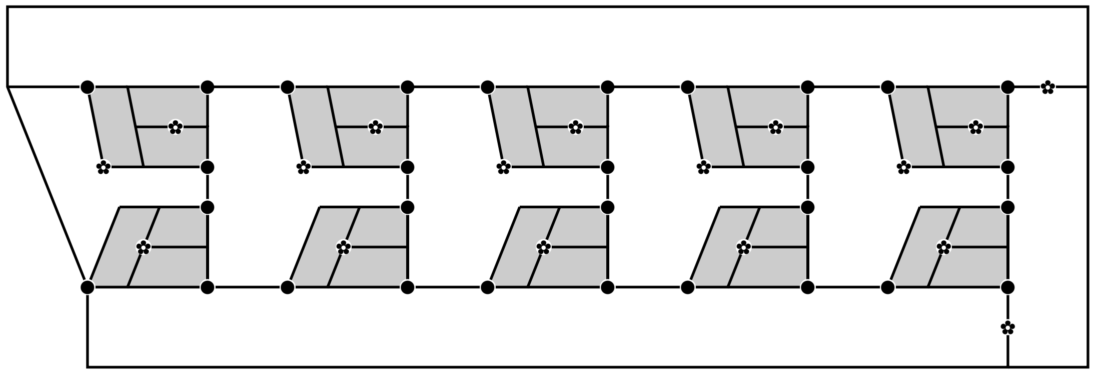


Figure 3.1: Family of instances of Subset Feedback Vertex Set with approximation factor 18/7 for the primal-dual algorithm with oracle Minimal-Pocket-Violation

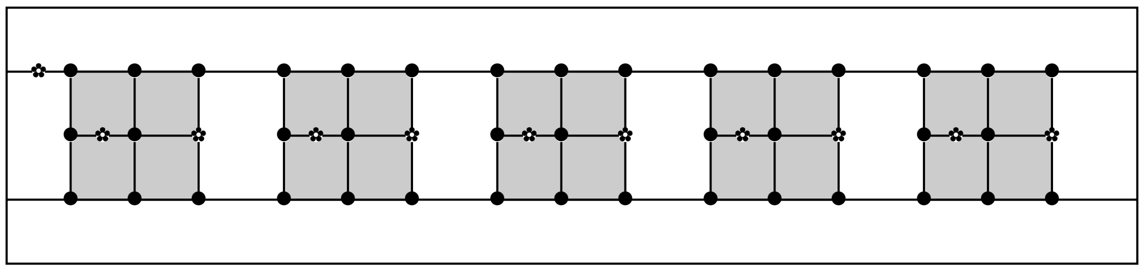


Figure 3.2: Family of instances of Subset Feedback Vertex Set with approximation factor 12/5 for primal-dual algorithm with oracle Minimal-3-Pocket-Violation

Part II

Concise Representations of Real Functions in Property Testing

Chapter 4

Concise Representations of Submodular Functions

4.1 Introduction

We investigate learning of submodular set functions, defined on the ground set $[n] = \{1, \ldots, n\}$. A set function $f : 2^{[n]} \to \mathbb{R}$ is *submodular* if one of the following equivalent definitions holds:

Definition 4.1.1. *1. For all $S, T \subseteq [n]$:*

$$f(S) + f(T) \geq f(S \cup T) + f(S \cap T).$$

2. For all $S \subset T \subseteq [n]$ and $i \in [n] \setminus T$:

$$f(S \cup \{i\}) - f(S) \geq f(T \cup \{i\}) - f(T).$$

3. For all $S \subseteq [n]$ and $i, j \in [n] \setminus S$:

$$f(S \cup \{i\}) + f(S \cup \{j\}) \geq f(S \cup \{i, j\}) + f(S).$$

Submodular set functions are important and widely studied, with applications in combinatorial optimization, economics, algorithmic game theory and many other disciplines. In many contexts, submodular functions are integral and nonnegative, and this is the setting we focus on. Examples of such functions include coverage functions[1], matroid rank functions, functions modeling valuations when the value of each set is expressed in dollars, cut functions of graphs[2], and cardinality-based set functions, i.e., functions of the form $f(S) = g(|S|)$, where g is concave.

[1]Given sets $A_1, \ldots, A_n$ in the universe U, a coverage function is $f(S) = |\cup_{j \in S} A_j|$.

[2]Given a graph G on the vertex set $[n]$, the cut function $f(S)$ of G is the number of edges of G crossing the cut $(S, [n]/S)$).

We study submodular functions $f : 2^{[n]} \to \{0, 1, \dots, k\}$, and give a learning algorithm for this class. To obtain our result, we use tools from several diverse areas, ranging from operations research to complexity theory.

Structural result. The first ingredient in the design of our algorithm is a structural result which shows that every submodular function in this class can be represented by a narrow pseudo-Boolean disjunctive normal form (DNF) formula, which naturally generalizes DNF for pseudo-Boolean functions. Pseudo-Boolean DNF formulas are well studied.

In the next definition and the rest of the paper, we use domains $2^{[n]}$ and $\{0, 1\}^n$ interchangeably. They are equivalent because there is a bijection between sets $S \subseteq [n]$ and strings $x_1 \dots x_n \in \{0, 1\}^n$, where the bit x_i is mapped to 1 if $i \in S$ and to 0 otherwise.

Definition 4.1.2 (Pseudo-Boolean DNF). *Let $x_1, \dots, x_n$ be variables taking values in $\{0, 1\}$. A* pseudo-boolean DNF of width k and size s *(also called a k-DNF of size s) is an expression of the form*

$$f(x_1, \dots, x_n) = \max_{t=1}^{s} \Big(a_t \Big(\bigwedge_{i \in A_t} x_i \Big) \Big(\bigwedge_{j \in B_t} \bar{x}_j \Big) \Big),$$

where a_t are constants, $A_t, B_t \subseteq [n]$ and $|A_t| + |B_t| \leq k$ for $t \in [s]$. A pseudo-boolean DNF is monotone *if it contains no negated variables, i.e., $B_t = \emptyset$ for all* terms *in the* max *expression. The class of all functions that have pseudo-Boolean k-DNF representations with constants $a_t \in \{0, \dots r\}$ is denoted DNF^{k,r}.*

It is not hard to see that every set function $f : 2^{[n]} \to \{0, \dots, k\}$ has a pseudo-Boolean DNF representation with constants $a_t \in \{0, \dots, k\}$, but in general there is no bound on the width of the formula.

Our structural result, stated next, shows that every submodular function $f : 2^{[n]} \to \{0, \dots, k\}$ can be represented by a pseudo-Boolean $2k$-DNF with constants $a_t \in \{0, \dots, k\}$. Our result is stronger for *monotone* functions, i.e., functions satisfying $f(S) \leq f(T)$ for all $S \subseteq T \subseteq [n]$. Examples of monotone submodular functions include coverage functions and matroid rank functions.

Theorem 4.1.1 (DNF representation of submodular functions). *Each submodular function $f : \{0, 1\}^n \to \{0, \dots, k\}$ can be represented by a pseudo-Boolean $2k$-DNF with constants $a_t \in \{0, \dots, k\}$ for all $t \in [s]$. Moreover, each term of the pseudo-Boolean DNF has at most k positive and at most k negated variables, i.e., $|A_t| \leq k$ and $|B_t| \leq k$. If f is monotone then its representation is a monotone pseudo-Boolean k-DNF.*

Note that the converse of Theorem 4.1.1 is false. E.g., consider the function $f(S)$ that maps S to 0 if $|S| \leq 1$ and to 1 otherwise. It can be represented by a 2-DNF as follows: $f(x_1 \dots x_n) = \max_{i,j \in [n]} x_i \wedge x_j$. However, it is not submodular, since version 3 of the definition above is falsified with $S = \emptyset, i = 1$ and $j = 2$.

Our proof of Theorem 4.1.1 builds on techniques developed by Gupta *et al.* [111] who show how to decompose a given submodular function into Lipschitz submodular functions. We first prove our structural result for monotone submodular functions. We use the decomposition from [111] to cover the domain of such a function by regions where the function is constant and then capture each region by a monotone term of width at most k. Then we decompose a general submodular function f into monotone regions, as in [111]. For each such region, we construct a monotone function which coincides with f on that region, does not exceed f everywhere else, and can be represented as a narrow pseudo-Boolean k-DNF by invoking our structural result for monotone submodular functions. This construction uses a monotone extension of submodular functions defined by Lovasz [146].

Learning. Our main result is a PAC-learning algorithm with membership queries under the uniform distribution for pseudo-Boolean k-DNF, which by Theorem 4.1.1 also applies to submodular functions $f : 2^{[n]} \to \{0, \dots, k\}$. We use a (standard) variant of the PAC-learning definition given by Valiant [193].

Definition 4.1.3 (PAC and agnostic learning under uniform distribution)**.** *Let U^n be the uniform distribution on $\{0,1\}^n$. A class of functions $\mathcal{C}$ is* PAC-learnable *under the uniform distribution if there exists a randomized algorithm $\mathcal{A}$, called a* PAC-learner*, which for every function $f \in \mathcal{C}$ and every $\epsilon, \delta > 0$, with probability at least $1-\delta$ over the randomness of $\mathcal{A}$, outputs a hypothesis h, such that*

$$\Pr_{x \sim U^n}[h(x) \neq f(x)] \leq \epsilon. \tag{4.1}$$

A learning algorithm $\mathcal{A}$ is proper *if it always outputs a hypothesis h from the class $\mathcal{C}$. A learning algorithm is* agnostic *if it works even if the input function f is arbitrary (not necessarily from $\mathcal{C}$), with ϵ replaced by $opt + \epsilon$ in (4.1), where opt is the smallest achievable error for a hypothesis in $\mathcal{C}$.*

Our algorithm accesses its input f via *membership queries*, i.e., by requesting $f(x)$ on some x in f's domain.

Theorem 4.1.2. *The class of pseudo-Boolean k-DNF formulas on n variables with constants in the range $\{0, \dots, r\}$ is PAC-learnable with membership queries under the uniform distribution with running time polynomial in n, $k^{O(k \log r/\epsilon)}$ and $\log(1/\delta)$, even in the agnostic setting.*

Our (non-agnostic) learning algorithm is a generalization of Mansour's PAC-learner for k-DNF [147]. It consists of running the algorithm of Kushilevitz and Mansour [142] for learning functions that can be approximated by functions with few non-zero Fourier coefficients, and thus has the same running time (and the same low-degree polynomial dependence on n). To be able to use this algorithm, we prove (in Theorem 4.4.1) that all functions in $\mathrm{DNF}^{k,r}$ have this property. The agnostic version of our algorithm follows from the Fourier concentration result in Theorem 4.4.1 and the work of Gopalan, Kalai and Klivans [108].

The key ingredient in the proof of Theorem 4.4.1 (on Fourier concentration) is a generalization of Håstad's switching lemma [115, 28] for standard DNF formulas to pseudo-Boolean DNF. Our generalization (formally stated in Lemma 4.3.1) asserts that a function $f \in \mathrm{DNF}^{k,r}$, restricted on large random subset of variables to random Boolean values, with large probability can be computed by a decision tree of small depth. (See Section 4.3 for definitions of random restrictions and decision trees.) Crucially, our bound on the probability that a random restriction of f has large decision-tree complexity is only a factor of r larger than the corresponding guarantee for the Boolean case.

Theorems 4.1.2 and 4.1.1 imply the following corollary.

Corollary 4.1.3. *The class of submodular functions $f\colon \{0,1\}^n \to \{0,\dots,k\}$ is PAC-learnable with membership queries under the uniform distribution in time polynomial in n, $k^{O(k \log k/\epsilon)}$ and $\log(1/\delta)$, even in the agnostic setting.*

Implications for testing submodularity. Our results give property testers for submodularity of functions $f : 2^{[n]} \to \{0,\dots,k\}$. A *property tester* [178, 103] is given oracle access to an object and a proximity parameter $\epsilon \in (0,1)$. If the object has the desired property, the tester *accepts* it with probability at least 2/3; if the object is ϵ-far from having the desired property then the tester *rejects* it with probability at least 2/3. Specifically, for properties of functions, ϵ-far means that a given function differs on at least an ϵ fraction of the domain points from any function with the property.

As we observe in Proposition A.4.1, a learner for a discrete class (e.g., the class of functions $f : 2^{[n]} \to \{0,\dots,k\}$) can be converted to a proper learner with the same query complexity (but huge overhead in running time). Thus, Corollary 4.1.3 implies a tester for submodularity of functions $f : 2^{[n]} \to \{0,\dots,k\}$ with query complexity polynomial in n and $k^{O(k \log k/\epsilon)}$, making progress on a question posed by Seshadhri [184].

4.1.1 Related work

Structural results for Boolean submodular functions. For the special case of Boolean functions, characterizations of submodular and monotone submodular functions in terms of simple DNF formulas are known. A Boolean function is monotone submodular if and only if it can be represented as a monotone 1-DNF (see, e.g., Appendix A in [21]). A Boolean function is submodular if and only if it can be represented as $\left(\bigvee_{i\in S} x_i\right) \bigwedge \left(\bigvee_{j\in T} \bar{x}_j\right)$ for $S, T \subseteq [n]$ [81].

Learning submodular functions. The problem of learning submodular functions has recently attracted significant interest. The focus on learning-style guarantees, which allow one to make arbitrary mistakes on some small portion of the domain, is justified by a negative results of Goemans *et al.* [100]. It demonstrates that every algorithm that makes a polynomial in n number of queries to a monotone submodular function (more specifically, even a matroid rank function) and tries to approximate it on all points in the domain, must make an $\Omega(\sqrt{n}/\log n)$ multiplicative error on some point.

Using results on concentration of Lipschitz submodular functions [41, 40, 194] and on noise-stability of submodular functions [56], significant progress on learning submodular functions was obtained by Balcan and Harvey [21], Gupta *et al.* [111] and Cheraghchi *et al.* [56]. These works obtain learners that *approximate* submodular functions, as opposed to learning them exactly, on an ϵ fraction of values in the domain. However, their learning algorithms generally work with weaker access models and with submodular functions over more general ranges.

Balcan and Harvey's algorithms learn a function within a given *multiplicative error* on all but ϵ fraction of the probability mass (according to a specified distribution on the domain). Their first algorithm learns *monotone* nonnegative submodular functions over $2^{[n]}$ within a *multiplicative factor of* $\sqrt{n}$ over *arbitrary* distributions using only *random examples* in polynomial time. For the special case of *product distributions* and *monotone* nonnegative submodular functions *with Lipschitz constant 1*, their second algorithm can learn *within a constant factor* in polynomial time.

Gupta *et al.* [111] design an algorithm that learns a submodular function with the range $[0, 1]$ within a given *additive error* α on all but ϵ fraction of the probability mass (according to a specified *product distribution* on the domain). Their algorithm requires membership queries, but works *even when these queries are answered with additive error* $\alpha/4$. It takes $n^{O(\log(1/\epsilon)/\alpha^2)}$ time.

Cheraghchi *et al.* [56] also work with *additive error*. Their learner is agnostic and only uses *statistical queries*. It produces a hypothesis which (with probability at least $1-\delta$) has the expected additive error $opt + \alpha$ with respect to a *product distribution*, where opt is the error of the best concept in the class. Their algorithm runs in time polynomial in $n^{O(1/\alpha)}$ and $\log(1/\delta)$.

Observe that the results in [111] and [56] directly imply an $n^{O(\log(1/\epsilon)k^2)}$ time algorithm for our setting, by rescaling our input function to be in $[0, 1]$ and setting the error $\alpha = 1/(2r)$. The techniques in [111] also imply $n^{O(k)}$ time complexity for non-agnostically learning submodular functions in this setting, for fixed ϵ and δ. To the best of our knowledge, this is the best dependence on n, one can obtain from previous work.

Chakrabarty and Huang [47] gave an exact learning algorithm for coverage functions, a subclass of monotone submodular functions. Their algorithm makes $O(n|U|)$ queries, where U is the size of the universe. (Coverage functions are defined as in Footnote 1 with additional nonnegative weight for each set, and $f(S)$ equal to the weight of $\cup_{j \in S} A_j$ instead of the cardinality.)

In a related line of work, focused on learning subadditive and fractionally subadditive functions with multiplicative approximation positive results are obtained by Balcan, Constantin, Iwata and Wang [20] and by Badanidiyuru, Dobzinski, Fu, Kleinberg, Nisan and Roughgarden [18].

Property testing submodular functions. The study of submodularity in the context of property testing was initiated by Parnas, Ron and Rubinfeld [162]. Seshadhri and Vondrak [185] gave the first sublinear (in the size of the domain) tester for submodularity of set functions. Their tester works for all ranges and has query and time complexity $(1/\epsilon)^{O(\sqrt{n}\log n)}$. They also showed a reduction from testing monotonicity to testing submodularity which, together with a lower bound for testing monotonicity given by Blais, Brody and Matulef [37], implies a lower bound of $\Omega(n)$ on the query complexity of testing submodularity for an arbitrary range and constant $\epsilon > 0$.

Given the large gap between known upper and lower bounds on the complexity of testing submodularity, Seshadhri [184] asked for testers for several important subclasses of submodular functions. The exact learner of Chakrabarty and Huang [47] for coverage functions, mentioned above, gives a property tester for this class with the same query complexity.

For the special case of Boolean functions, in the light of the structural results mentioned above, one can test if a function is monotone submodular with $O(1/\epsilon)$ queries by using the algorithm from [164] (Section 4.3) for testing monotone monomials.

4.2 Structural result

In this section, we prove Theorem 4.1.1 that shows that every submodular function over a bounded (nonnegative) integral range can be represented by a narrow pseudo-Boolean DNF. After introducing notation used in the rest of the section (in Definition 4.2.1), we prove the theorem for the special case when f is monotone submodular (restated in Lemma 4.2.1) and then present the proof for the general case. In the proof, we give a recursive algorithm for constructing pseudo-Boolean DNF representation which has the same structure of recursive calls as the decomposition algorithm of Gupta *et al.* [111] for the monotone case. Our contribution is in showing how to use these calls to get a monotone pseudo-Boolean k-DNF representation of the input function. For the non-monotone case the structure of recursive calls that we use is different from that of [111].

Lemma 4.2.1 (DNF representation of monotone submodular functions)**.** *Every monotone submodular function $f : \{0,1\}^n \to \{0,\dots,k\}$ can be represented by a pseudo-Boolean monotone k-DNF with constants $a_t \in \{0,\dots,k\}$ for all $t \in [s]$.*

Proof. Algorithm 4.1 below, with the initial call MONOTONE-DNF$(f,\emptyset)$, returns the collection $\mathcal{C}$ of terms in a pseudo-boolean DNF representation of f.

Algorithm 4.1 MONOTONE-DNF(f,S).

```
input  : Oracle access to $f : 2^{[n]} \to \{0,\dots,k\}$, argument $S \in 2^{[n]}$.
output: Collection $\mathcal{C}$ of monotone terms of width at most $k$.
1: $C \leftarrow (f(S) \cdot \bigwedge_{i \in S} x_i)$
2: for $j \in [n] \setminus S$ do
3:     if $f(S \cup \{j\}) > f(S)$ then
4:         $C \leftarrow C \cup$ MONOTONE-DNF$(f, S \cup \{j\})$.
5:     end if
6: end for
7: return $C$
```

First, note that the invariant $f(S) \geq |S|$ is maintained for every call MONOTONE-DNF(f,S). Since the maximum value of f is at most k, there are no calls with $|S| > k$. Thus, every term in the collection returned by MONOTONE-DNF$(f,\emptyset)$ has width at most k. By definition, all terms are monotone.

Next, we show that the resulting formula $\max_{C_i \in \mathcal{C}} C_i$ exactly represents f. For a clause $C_i \in \mathcal{C}$ and $S \in 2^{[n]}$ we use an indicator function $C_i(S) \to \{0,1\}$ such that $C_i(S) = 1$ iff C_i is satisfied by the assignment of values to its variables according to S. For all $Y \in 2^{[n]}$ we have $f(Y) \geq \max_{C_i \in \mathcal{C}} C_i(Y)$ by monotonicity of f. To see that for all Y we have $f(Y) \leq \max_{C_i \in \mathcal{C}} C_i(Y)$, fix an arbitrary Y and let $\mathcal{T} = \{Z \mid Z \subseteq Y, f(Z) = f(Y)\}$ and T be a set of the smallest size in $\mathcal{T}$. If there was a recursive call MONOTONE-DNF(f,T) then the term added by this recursive call would ensure the inequality. If $T = \emptyset$ then such a call was made. Otherwise, consider the set $\mathcal{U} = \{T \setminus \{j\} \mid j \in T\}$. By the choice of T, we have $f(Z) < f(T)$ for all $Z \in \mathcal{U}$. By submodularity of f (see Definition 4.1.1, part

2), this implies that the restriction of f on $T^{\downarrow}$ is a strictly increasing function. Thus, the recursive call MONOTONE-DNF(f, T) was made and the term added by it guarantees the inequality. $\square$

For a collection $\mathcal{S}$ of subsets of $[n]$, let $f_{\mathcal{S}}\colon \mathcal{S} \to \mathbb{R}$ denote the restriction of a function f to the union of sets in $\mathcal{S}$. We use notation $\mathbf{1}_{\mathcal{S}}\colon 2^{[n]} \to \{0,1\}$ for the indicator function defined by $\mathbf{1}_{\mathcal{S}}(Y) = 1$ iff $Y \in \bigcup\limits_{S \in \mathcal{S}} S$.

Definition 4.2.1 ($S^{\downarrow}$ and $S^{\uparrow}$)**.** *For a set $S \in 2^{[n]}$, we denote the collection of all subsets of S by $S^{\downarrow}$ and the collection of all supersets of S by $S^{\uparrow}$.*

Proof of Theorem 4.1.1. For a general submodular function, the formula can be constructed using Algorithm 4.2, with the initial call DNF$(f, [n])$. The algorithm description uses the function $f^{mon}_{S^{\downarrow}}$, defined next.

Definition 4.2.2 (Function $f^{mon}_{S^{\downarrow}}$)**.** *For a set $S \subseteq [n]$, define the function $f^{mon}_{S^{\downarrow}}\colon S^{\downarrow} \to \{0, \ldots, k\}$ as follows: $f^{mon}_{S^{\downarrow}}(Y) = \min_{Y \subseteq Z \subseteq S} f(Z)$.*

Note that if f is a submodular function then for every set $S \subseteq [n]$ the function $f_{S^{\downarrow}}$ is a submodular function.

Proposition 4.2.2 (Proposition 2.1 in [146])**.** *For every set $S \subseteq [n]$, if $f_{S^{\downarrow}}$ is a submodular function, then $f^{mon}_{S^{\downarrow}}$ is a monotone submodular function.*

Algorithm 4.2 DNF(f, S).

input : Oracle access to $f : 2^{[n]} \to \{0, \ldots, k\}$, argument $S \in 2^{[n]}$.
output: Collection $\mathcal{C}$ of terms, each containing at most k positive and at most k negated variables.

```
1: $C_{mon} \leftarrow$ MONOTONE-DNF$(f^{mon}_{S^{\downarrow}}, \emptyset)$;
2: $C \leftarrow \bigcup_{C_i \in C_{mon}} (C_i \cdot (\bigwedge_{i \in [n] \setminus S} \bar{x}_i))$;
3: for $j \in S$ do
       if $f(S \setminus \{j\}) > f(S)$ then
4:         $C \leftarrow C \cup$ DNF$(f, S \setminus \{j\})$.
5: return $C$ ;
```

Let $\mathcal{S}$ be the collection of sets $S \subseteq [n]$ for which a recursive call is made when DNF$(f, [n])$ is executed. For a set $S \in \mathcal{S}$, let $B(S) = \{j \mid f(S \setminus \{j\}) \leq f(S)\}$ be the set consisting of elements such that if we remove them from S, the value of the function does not increase. Let the *monotone region* of S be defined by $S^{\leq\downarrow} = \{Z \mid (S \setminus B(S)) \subseteq Z \subseteq S\} = S^{\downarrow} \cap (S \setminus B(S))^{\uparrow}$. By submodularity of f (Definition 4.1.1, part 2) the restriction $f_{S^{\leq\downarrow}}$ is a monotone nondecreasing function.

Proposition 4.2.3. *Fix $S \in \mathcal{S}$. Then $f(Y) \geq f^{mon}_{S^{\downarrow}}(Y)$ for all $Y \in S^{\downarrow}$. Moreover, $f(Y) = f^{mon}_{S^{\downarrow}}(Y)$ for all $Y \in S^{\leq\downarrow}$.*

Proof. By the definition of $f^{mon}_{S^\downarrow}$, we have $f^{mon}_{S^\downarrow}(Y) = \min_{Y \subseteq Z \subseteq S} f(Z) \le f(Y)$ for all $Y \in S^\downarrow$. Since the restriction $f_{S^{\le\downarrow}}$ is monotone nondecreasing, $f^{mon}_{S^\downarrow}(Y) = \min_{Y \subseteq Z \subseteq S} f(Z) = f(Y)$ for all $Y \in S^{\le\downarrow}$. $\square$

The following proposition is implicit in [111]. We give a proof for completeness.

Proposition 4.2.4. *For all functions $f : 2^{[n]} \to \{0, \dots, k\}$, the collection of all monotone regions of sets in $\mathcal{S}$ forms a cover of the domain, namely, $\cup_{S \in \mathcal{S}} S^{\le\downarrow} = 2^{[n]}$.*

Proof. The proof is by induction on the value $f([n])$ that the function f takes on the largest set in its domain. The base case of induction is $f([n]) = k$. In this case, $\mathcal{S}$ consists of a single set $S = [n]$, and the function f is monotone non-increasing on $S^\downarrow = S^{\le\downarrow}$. Now suppose that the statement holds for all f, such that $f([n]) \ge t$. If $f([n]) = t - 1$ then for every $Y \in 2^{[n]}$ there are two cases:

1. There is no Z of size $n - 1$ such that $f(Z) > f([n])$ and $Y \in Z^\downarrow$ then $Y \in [n]^{\le\downarrow}$ and thus Y is covered by the monotone region of $[n]$.

2. There exists a set Z of size $n - 1$ such that $f(Z) > f([n])$ and $Y \in Z^\downarrow$ then there exists a set $S \in \mathcal{S}$, such that $Y \in S^\downarrow$ by applying inductive hypothesis to $f_{Z^\downarrow}$, so Y is covered by $S^\downarrow$.

$\square$

Lemma 4.2.1 and Proposition 4.2.2 give that the collection of terms C_{mon}, constructed in Line 1 of Algorithm 4.2, corresponds to a monotone pseudo-Boolean k-DNF representation for $f^{mon}_{S^\downarrow}$. By the same argument as in the proof of Lemma 4.2.1, $|S| \ge n - k$ for all $S \in \mathcal{S}$, since the maximum of f is at most k. Therefore, Line 2 of Algorithm 4.2 adds at most $n - |S|$ negated variables to every term of C_{mon}, resulting in terms with at most k positive and at most k negated variables.

It remains to prove that the constructed formula represents f. For a set S, let C_S denote the collection of terms obtained on Line 2 of Algorithm 4.2. By construction, $C_S(Y) = f^{mon}_{S^\downarrow} \cdot \mathbf{1}_{S^\downarrow}(Y)$ for all $Y \in 2^{[n]}$. For every $Y \in 2^{[n]}$, the first part of Proposition 4.2.3 implies that $C_S(Y) = f^{mon}_{S^\downarrow}(Y) \cdot \mathbf{1}_{S^\downarrow}(Y) \le f(Y)$, yielding $\max_{S \in \mathcal{S}} C_S(Y) \le f(Y)$. On the other hand, by Proposition 4.2.4, for every $Y \in 2^{[n]}$ there exists a set $S \in \mathcal{S}$, such that $Y \in S^{\le\downarrow}$. For such S, the second part of Proposition 4.2.3 implies that $C_S(Y) = f^{mon}_{S^\downarrow}(Y) \cdot \mathbf{1}_{S^\downarrow}(Y) = f(Y)$. Therefore, f is equivalent to $\max_{S \in \mathcal{S}} C_S$. $\square$

4.3 Generalized switching lemma for pseudo-Boolean DNFs

The following definitions are stated for completeness and can be found in [158, 147].

Definition 4.3.1 (Decision tree)**.** *A decision tree T is a representation of a function $f \colon \{0,1\}^n \to \mathbb{R}$. It consists of a rooted binary tree in which the internal nodes are labeled by coordinates $i \in [n]$, the outgoing edges of each internal node are labeled 0 and 1, and the leaves are labeled by real numbers. We insist that no coordinate $i \in [n]$ appears more than once on any root-to-leaf path.*

Each input $x \in \{0,1\}^n$ corresponds to a computation path in the tree T from the root to a leaf. When the computation path reaches an internal node labeled by a coordinate $i \in [n]$, we say that T queries x_i. The computation path then follows the outgoing edge labeled by x_i. The output of T (and hence f) on input x is the label of the leaf reached by the computation path. We identify a tree with the function it computes.

The *depth* s of a decision tree T is the maximum length of any root-to-leaf path in T. For a function f, DT-depth(f) is the minimum depth of a decision tree computing f.

Definition 4.3.2 (Random restriction)**.** *A restriction ρ is a mapping of the input variables to $\{0,1,\star\}$. The function obtained from $f(x_1,\ldots,x_n)$ by applying a restriction ρ is denoted $f|_\rho$. The inputs of $f|_\rho$ are those x_i for which $\rho(x_i)=\star$ while all other variables are set according to ρ.*

A variable x_i is *live* with respect to a restriction ρ if $\rho(x_i)=\star$. The set of live variables with respect to ρ is denoted live(ρ). A random restriction ρ with parameter $p \in (0,1)$ is obtained by setting each x_i, independently, to $0,1$ or $\star$, so that $\Pr[\rho(x_i)=\star]=p$ and $\Pr[\rho(x_i)=1]=\Pr[\rho(x_i)=0]=(1-p)/2$.

We will prove the following generalization of the switching lemma [115, 28].

Lemma 4.3.1 (Switching lemma for pseudo-Boolean formulas)**.** *Let $f \in DNF^{k,r}$ and ρ be a random restriction with parameter p (i.e., $\Pr[\rho(x_i)=\star]=p$). Then*

$$\Pr[\textit{DT-depth}(f|_\rho) \geq s] < r \cdot (7pk)^s.$$

Proof. We use the exposition of Razborov's proof of the switching lemma for Boolean functions, described in [28], as the basis of our proof and highlight the modifications we made for non-Boolean functions.

Define $\mathcal{R}_n^\ell$ to be the set of all restrictions ρ on a domain of n variables that have exactly ℓ unset variables. Fix some function $f \in \mathrm{DNF}^{k,r}$, represented by a formula F, and assume that there is a total order on the terms of F as well as on the indices of the variables. A restriction ρ is applied to F in order, so that F_ρ is a pseudo-Boolean DNF formula whose terms consist of those terms in F that are not falsified by ρ, each shortened by removing any variables that are satisfied by ρ, and taken in the order of occurrences of the original terms on which they are based.

Definition 4.3.3 (Canonical labeled decision tree)**.** *The* canonical labeled decision tree *for F, denoted $T(F)$, is defined inductively as follows:*

1. *If F is a constant function then $T(F)$ consists of a single leaf node labeled by the appropriate constant.*

2. *If the first term C_1 of F is not empty then let F' be the remainder of F so that $F=\max(C_1,F')$. Let K be the set of variables appearing in C_1. The tree $T(F)$ starts with a complete binary tree for K, which queries the variables in K in the order of their indices. Each leaf v_σ in the tree is associated with a restriction σ which sets the variables of K according to the path from the root to v_σ. For each σ, replace the leaf node v_σ by the subtree $T(F_\sigma)$. For the unique assignment σ satisfying C_1,*

also label the corresponding node by L_σ equal to the maximum of the labels assigned to the predecessors of this node in the tree and the integer constant associated with the term C_1.

Note that the above definition is more general than a canonical decision tree for Boolean DNF formulas because it uses labels for some of the internal nodes in the tree to indicate that the paths from the parent node to these internal nodes restrict the value of the formula to be at least the value of the label. For the Boolean DNFs such labels are not needed and thus if we apply to them the definition above we get a standard definition of a canonical decision tree, as in [28]. Formally, for pseudo-Boolean DNF formulas the label L_σ of the internal node σ represents the fact that the value of the formula on the leaves in the subtree of σ is at least L_σ.

Using the terminology introduced above, we can state the switching lemma as follows.

Lemma 4.3.2. *Let F be a pseudo-Boolean formula, representing a function $f \in DNF^{k,r}$, $s \geq 0$, $p \leq 1/7$ and $\ell = pn$. Then*

$$\frac{|\{\rho \in \mathcal{R}_n^\ell \colon |T(F|_\rho)| \geq s\}|}{|\mathcal{R}_n^\ell|} < r(7pk)^s.$$

Proof. Let $stars(k, s)$ be the set of all sequences $\beta = (\beta_1, \ldots, \beta_t)$ such that for each $j \in [t]$, the coordinate $\beta_j \in \{\star, -\}^k \setminus \{-\}^k$ and such that the total number of $\star$'s in all the β_j is s.

Let $S \subseteq \mathcal{R}_n^\ell$ be the set of restrictions ρ such that $|T(F|_\rho)| \geq s$. We will define an injective mapping from S to the cartesian product $\mathcal{R}_n^{\ell-s} \times stars(k, s) \times [2^s] \times [r]$.

Let $F = \max_i C_i$. In the exposition below we use the same notation π to denote both a restriction and a path in the canonical labeled decision tree, which sets variables according to π. Suppose that $\rho \in S$ and π is the restriction associated with the lexicographically first path in $T(F|_\rho)$ of length at least s. Trim the last variables in π along the path π from the root so that $|\pi| = s$. Let c be the maximum label of the node on π (or zero, if none of the nodes on p_π are labeled). Partition the set of terms of F into two sets F' and F'', where F' contains all terms with constants $> c$ and F'' contains all terms with constants $\leq c$ (for Boolean formulas, $c = 0$ and $F = F'$). We will use the subformula F' and π to determine the image of ρ. The image of ρ is defined by following the path π in the canonical labeled decision tree for F_ρ and using the structure of the tree.

Let C_{ν_1} be the first term of F' that is not set to 0 by ρ. Since $|\pi| > 0$, such a term must exist and will not be an empty term (otherwise, the value of $F|_\rho$ is fixed to be $> c$). Let K be the set of variables in $C_{\nu_1}|\rho$ and let σ_1 be the unique restriction of the variables in K that satisfies $C_{\nu_1}|\rho$. Let π_1 be the part of π that sets the variables in K. We have two cases based on whether $\pi_1 = \pi$.

1. If $\pi_1 \neq \pi$ then by the construction of π, restriction π_1 sets all the variables in K. Note that the restriction $\rho\sigma_1$ satisfies the term C_{ν_1} but since $\pi_1 \neq \pi$ the restriction $\rho\pi_1$ does not satisfy term C_{ν_1}.

2. If $\pi_1 = \pi$ then it is possible that π does not set all of the variables in K. In this case we shorten σ_1 to the variables in K that appear in π_1.

Define $\beta_1 \in \{\star, -\}^k$ based on the fixed ordering of the variables in the term C_{ν_1} by letting the jth component of β_1 be $\star$ if and only if the jth variable in C_{ν_1} is set by σ_1. Since $C_{\nu_1}|_\rho$ is not the empty term, β_1 has at least one $\star$. From C_{ν_1} and β_1 we can reconstruct σ_1.

Now by the definition of $T(F|_\rho)$, the restriction $\pi \setminus \pi_1$ labels a path in the canonical labeled decision tree $T(F|_{\rho\pi_1})$. If $\pi_1 \neq \pi$, we repeat the argument above, replacing π and ρ with $\pi \setminus \pi_1$ and $\rho\pi_1$, respectively, and find a term C_{ν_2} which is the first term of F' not set to 0 by $\rho\pi_1$. Based on this, we generate π_2, σ_2 and β_2, as before. We repeat this process until the round t in which $\pi_1\pi_2 \dots \pi_t = \pi$.

Let $\sigma = \sigma_1\sigma_2 \dots \sigma_t$. We define $\xi \in \{0,1\}^s$ to be a vector that indicates for each variable set by π whether it is set to the same value as σ sets it. We define the image of ρ in the injective mapping as a quadruple, $\langle \rho\sigma_1 \dots \sigma_t, (\beta_1, \dots, \beta_t), \xi, c\rangle$. Because $\rho\sigma \in \mathcal{R}_n^{\ell-s}$ and $(\beta_1, \dots, \beta_t) \in stars(k,s)$ the mapping is as described above.

It remains to show that the defined mapping is indeed injective. We will show how to invert it by reconstructing ρ from its image. We use c to construct F' from F. The reconstruction procedure is iterative. In one stage of the reconstruction we recover $\pi_1 \dots \pi_{i_1}, \sigma_1 \dots \sigma_{i-1}$ and construct $\rho\pi_1 \dots \pi_{i-1}\sigma_i \dots \sigma_t$. Recall that for $i < t$ the restriction $\rho\pi_1 \dots \pi_{i-1}\sigma_i$ satisfies the term C_{ν_i}, but does not satisfy terms C_j for all $j < \nu_i$. This holds if we extend the restriction by appending $\sigma_{i+1} \dots \sigma_t$. Thus, we can recover ν_i as the index of the first term of F' that is not falsified by $\rho\pi_1 \dots \pi_{i-1}\sigma_i \dots \sigma_t$ and the consant corresponding to this term is at least c.

Now, based on C_{ν_1} and β_i, we can determine σ_i. Since we know $\sigma_1 \dots \sigma_i$, using the vector ξ we can determine π_i. We can now change $\rho\pi_1 \dots \pi_{i-1}\sigma_i \dots \sigma_t$ to $\rho\pi_1 \dots \pi_{i-1}\pi_i\sigma_{i+1} \dots \sigma_t$ using the knowledge of π_i and σ_i. Finally, given all the values of the π_i we reconstruct ρ by removing the variables from $\pi_1 \dots \pi_t$ from the restriction.

The computation in the following claim completes the proof of the switching lemma. □

□

Claim 4.3.3 ([28]). *For $p < 1/7$ and $p = \ell/n$, the following holds:*

$$\frac{|\mathcal{R}_n^{\ell-s}| \cdot |stars(k,s)| \cdot 2^s}{|\mathcal{R}_n^\ell|} < (7pk)^s.$$

Proof. We have $|\mathcal{R}_n^\ell| = \binom{n}{\ell}2^{n-\ell}$, so:

$$\frac{|\mathcal{R}_n^{\ell-s}|}{|\mathcal{R}_n^\ell|} \leq \frac{(2\ell)^s}{(n-\ell)^s}.$$

We use the following bound on $|stars(k,s)|$.

Proposition 4.3.4 (Lemma 2 in [28]). $|stars(k,s)| < (k/\ln 2)^s$.

Using Proposition 4.3.4 we get:

$$\frac{|S|}{|\mathcal{R}_n^\ell|} \leq \frac{|\mathcal{R}_n^{\ell-s}|}{|\mathcal{R}_n^\ell|} \cdot |stars(k,s)| \cdot 2^s$$

$$\leq \left(\frac{4\ell k}{(n-\ell)\ln 2} \right)^s$$
$$= \left(\frac{4pk}{(1-p)\ln 2} \right)^s .$$

For $p < 1/7$, the last expression is at most $(7pk)^s$, as claimed. $\square$

4.4 Learning pseudo-Boolean DNFs

In this section, we present our learning results for pseudo-Boolean k-DNF and prove Theorem 4.1.2.

Let R_r denote the set of multiples of $2/(r-1)$ in the interval $[-1,1]$, namely $R_r = \{-1, -1+2/(r-1), ..., 1-2/(r-1), 1\}$. First, we apply a transformation of the range by mapping $\{0,\ldots,r\}$ to R_r. Formally, in this section instead of functions $f\colon \{0,1\}^n \to \{0,\ldots,r\}$ we consider functions $f'\colon \{-1,1\}^d \to [-1,1]$, such that $f'(x'_1,\ldots,x'_n) = 2/(r-1)\cdot f(x_1,\ldots,x_n) - 1$, where $x'_i = 1-2x_i$. The mapping is one to one and thus a learning algorithm for the class of functions that can be represented by pseudo-Boolean DNF formulas of width k with constants in the range R_r implies Theorem 4.1.2. Thus, we will refer to this transformed class also as DNF^{k,r}.

For a set $S \subseteq [n]$, let χ_S be the standard Fourier basis vector and let $\hat{f}(S)$ denote the corresponding Fourier coefficient of a function f.

Definition 4.4.1. *A function g ϵ-approximates a function f if $\mathbb{E}[(f-g)^2] \leq \epsilon$. A function is M-sparse if it has at most M non-zero Fourier coefficients. The Fourier degree of a function, denoted deg(f), is the size of the largest set, such that $\hat{f}(S) \neq 0$.*

The following guarantee about approximation of functions in DNF^{k,r} by sparse functions is the key lemma in the proof of Theorem 4.1.2.

Theorem 4.4.1. *Every function $f \in$ DNF^{k,r} can be ϵ-approximated by an M-sparse function, where $M = k^{O(k\log(r/\epsilon))}$.*

Proof of Theorem 4.4.1. We generalize the proof by Mansour [147], which relies on multiple applications of the switching lemma. Our generalization of the switching lemma allows us to obtain the following parameters of the key statements in the proof, which bound the L_2-norm of the Fourier coefficients of large sets in Lemma 4.4.2 and the L_1-norm of the Fourier coefficients of small sets in Lemma 4.4.4.

Lemma 4.4.2. *For every function $f \in$ DNF^{k,r},*

$$\sum_{S\colon |S|>28k\log(2r/\epsilon)} \hat{f}^2(S) \leq \epsilon/2.$$

Proof. The proof relies on the following result.

Proposition 4.4.3 ([147, 158])**.** *Let $f\colon \{0,1\}^n \to \{-1,1\}$ and f_ρ be a random restriction with parameter p. Then for every $t \in [n]$,*

$$\sum_{|S|>t} \hat{f}^2(S) \leq \Pr_\rho[deg(f|_\rho) \geq tp/2].$$

Because $deg(f|_\rho) \leq \text{DT-depth}(f|_\rho)$ and thus $\Pr[deg(f|_\rho) \geq tp/2] \leq \Pr[\text{DT-depth}(f|_\rho) \geq tp/2]$. By using Lemma 4.3.1 and setting $p = 1/14k$ and $t = 28k\log(2r/\epsilon)$, we complete the proof. □

The main part of the proof of Theorem 4.4.1 is the following lemma, which bounds the L_1-norm of Fourier coefficients, corresponding to sets of bounded size.

Lemma 4.4.4. *For every function $f \in DNF^{k,r}$ and $\tau \in [n]$,*

$$\sum_{S\colon |S|\leq\tau} |\hat{f}(S)| \leq 4r(28k)^\tau = rk^{O(\tau)}.$$

Proof. Let $L_{1,t}(f) = \sum_{|S|=t} |\hat{f}(S)|$ and $L_1(f) = \sum_{t=0}^{n} L_{1,t}(f) = \sum_S |\hat{f}(S)|$.

We use the following bound on $L_1(f)$ for decision trees.

Proposition 4.4.5 ([142, 158])**.** *Consider a function $f\colon \{-1,1\}^n \to [-1,1]$, such that DT-depth$(f) \leq s$. Then $L_1(f) \leq 2^s$.*

We show the following generalization of Lemma 5.2 in [147] for DNF^{k,r}.

Proposition 4.4.6. *Let $f \in DNF^{k,r}$ and let ρ be a random restriction of f with parameter $p \leq 1/28k$. Then $\mathbb{E}_\rho\left[L_1(f|_\rho)\right] \leq 2r$.*

Proof. By the definition of expectation,

$$\begin{aligned}&\mathbb{E}_\rho[L_1(f)]\\ =&\sum_{s=0}^{n} \Pr[\text{DT-depth} f|_\rho = s]\\ &\cdot\mathbb{E}_\rho\left[L_1(f|_\rho) \mid \text{DT-depth}(f|_\rho) = s\right].\end{aligned}$$

By Proposition 4.4.5, for all ρ, such that DT-depth$(f|_\rho) = s$, it holds that $L_1(f) \leq 2^s$. By Lemma 4.3.1, $\Pr[\text{DT-depth}(f|_\rho) \geq s] \leq r(7pk)^s$. Therefore, $\mathbb{E}_\rho[L_1(f)] \leq \sum_{s=0}^{n} r(7pk)^s 2^s = r \cdot \sum_{s=0}^{n}(14pk)^s$. For $p \leq 1/28k$ the lemma follows. □

We use Lemma 5.3 from [147] to bound $L_{1,t}(f)$ by the value of $\mathbb{E}_\rho\left[L_{1,t}(f|_\rho)\right]$. Because in [147] the lemma is stated for Boolean functions, we give the proof for real-valued functions for completeness.

Proposition 4.4.7 ([147])**.** *For $f\colon \{0,1\}^n \to [-1,1]$ and a random restriction ρ with parameter p,*

$$L_{1,t}(f) \leq \left(\frac{1}{p}\right)^t \mathbb{E}_\rho\left[L_{1,t}(f)\right].$$

Proof. Consider a random variable $\mathcal{L}$ supported on $2^{[n]}$, such that for each x_i independently, $\Pr[x_i \in \mathcal{L}] = p$. The random variable $\mathcal{L}$ is the set of live variables in a random restriction with parameter p. We can rewrite $L_{1,t}$ as:

$$\begin{aligned} L_{1,t}(f) &= \sum_{|S|=t} |\hat{f}(S)| \\ &= \left(\frac{1}{p}\right)^t \mathbb{E}_{\mathcal{L}} \left[\sum_{S \subseteq \mathcal{L}, |S|=t} \left|\hat{f}(S)\right| \right]. \end{aligned} \tag{4.2}$$

For an arbitrary choice of $\mathcal{L}$ and a subset $S \subseteq \mathcal{L}$ we have:

$$\begin{aligned} |\hat{f}(S)| &= |\mathbb{E}_{x_1,\ldots,x_n} \left[f(x_1,\ldots,x_n)\chi_S(x_1,\ldots,x_n) \right]| \\ &\le \mathbb{E}_{x \notin \mathcal{L}} |\mathbb{E}_{x \in \mathcal{L}} \left[f(x_1,\ldots,x_n)\chi_S(x_1,\ldots,x_n) \right]| \\ &= \mathbb{E}_\rho \left[|\hat{f}|_\rho(S)| \mid live(\rho) = \mathcal{L} \right], \end{aligned}$$

where the last line follows from the observation that averaging over $x_i \notin \mathcal{L}$ is the same as taking the expectation of a random restriction whose set of live variables is restricted to be $\mathcal{L}$. Because the absolute value of every coefficient S is expected to increase, this implies that:

$$\begin{aligned} &\sum_{S \subseteq \mathcal{L}} \left|\hat{f}(S)\right| \\ \le &\mathbb{E}_\rho \left[\sum_{S \subseteq \mathcal{L}, |S|=t} |\hat{f}|_\rho(S)| \mid live(\rho) = \mathcal{L} \mid \right] \\ = &\mathbb{E}_\rho \left[L_{1,t}(f_\rho) | live(\rho) = \mathcal{L} \right]. \end{aligned}$$

Using this together with (4.2), we conclude that

$$\begin{aligned} &L_{1,t}(f) \\ &= \left(\frac{1}{p}\right)^t \mathbb{E}_{\mathcal{L}} \left[\sum_{S \subseteq \mathcal{L}, |S|=t} \left|\hat{f}(S)\right| \right] \\ &\le \left(\frac{1}{p}\right)^t \mathbb{E}_\rho \left[L_{1,t}(f|_\rho) \right]. \end{aligned}$$

□

Note that $\sum_{S\colon |S| \le \tau} |\hat{f}(S)| = \sum_{t=0}^{\tau} L_{1,t}(f)$. By setting $p = 1/28k$ and using Propositions 4.4.6 and 4.4.7, we get:

$$L_{1,t}(f) \le 2r(28k)^t.$$

Thus, $\sum_{S\colon |S| \le \tau} |\hat{f}(S)| \le 4r(28k)^\tau = rk^{O(\tau)}$, completing the proof of Lemma 4.4.4. □

Let $\tau = 28k\log(2r/\epsilon)$ and $L = \sum_{|S|\le\tau}|\hat{f}(S)|$. Let $G = \{S : |\hat{f}(S)| \ge \epsilon/2L \text{ and } |S| \le \tau\}$ and $g(x) = \sum_{S\in G}\hat{f}(S)\chi_S(x)$. We will show that g is M-sparse and that it ϵ-approximates f.

By an averaging argument, $|G| \le 2L^2/\epsilon$. Thus, function g is M-sparse, where $M \le 2L^2/\epsilon$. By Lemma 4.4.4, $L = rk^{O(\tau)} = k^{O(k\log(r/\epsilon))}$. Thus, $M = k^{O(k\log(r/\epsilon))}$, as claimed in the theorem statement.

By the definition of g and by Parseval's identity,

$$\begin{aligned} & \mathbb{E}[(f-g)^2] \\ = & \sum_{S\notin G}\hat{f}^2(S) \\ = & \sum_{S\colon |S|>\tau}\hat{f}^2(S) + \sum_{S\colon |S|\le\tau, |\hat{f}(S)|\le\epsilon/2L}\hat{f}^2(S). \end{aligned}$$

By Lemma 4.4.2, the first summation is at most $\epsilon/2$. For the second summation, we get:

$$\begin{aligned} & \sum_{S\colon |S|\le\tau, |\hat{f}(S)|\le\epsilon/2L}\hat{f}^2(S) \\ \le & \left(\max_{S\colon |\hat{f}(S)|\le\epsilon/2L}|\hat{f}(S)|\right)\left(\sum_{|S|\le\tau}|\hat{f}(S)|\right) \\ \le & \frac{\epsilon}{2L}\cdot L = \epsilon/2. \end{aligned}$$

This implies that $\mathbb{E}[(f-g)^2] \le \epsilon$ and thus g ϵ-approximates f. □

To get a learning algorithm and prove Theorem 4.1.2 we can use the sparse approximation guarantee of Theorem 4.4.1 together with Kushilevitz-Mansour learning algorithm (for PAC-learning) or the learning algorithm of Gopalan, Kalai and Klivans (for agnostic learning).

Proof of Theorem 4.1.2. We will use the learning algorithm of Kushilevitz and Mansour [107, 142], which gives the following guarantee:

Theorem 4.4.8 ([142]). *Let f be a function that can be ϵ-approximated by an M-sparse function. There exists a randomized algorithm, whose running time is polynomial in M, n, $1/\epsilon$ and $\log(1/\delta)$, that given oracle access to f and $\delta > 0$, with probability at least $1-\delta$ outputs a function h that $O(\epsilon)$-approximates f.*

Setting the approximation parameter in Theorem 4.4.8 to be $\epsilon' = \epsilon/Cr^2$ for large enough constant C and taking $M = k^{O(k\log(r/\epsilon'))}$ we get an algorithm which returns a function h that (ϵ/r^2)-approximates f. The running time of such algorithm is polynomial in n, $k^{O(k\log(r/\epsilon))}$ and $\log(1/\delta)$. By Proposition 4.4.9, if we round the values of h in every point to the nearest multiple of $2/(r-1)$, we will get a function h', such that $\Pr_{x\in U^n}[h'(x) \ne f(x)] \le \epsilon$, completing the proof.

Proposition 4.4.9. *Suppose a function $h : 2^{[n]} \to [-1,1]$ is an ϵ-approximation for $f\colon 2^{[n]} \to R_r$. Let g be the function defined by $g(x) = argmin_{y \in R_r} |h(x) - y|$, breaking ties arbitrarily. Then $\Pr_{x \in U^n}[g(x) \neq f(x)] \leq \epsilon \cdot (r-1)^2$.*

Proof of Proposition 4.4.9. Observe that $|f(x) - h(x)|^2 \geq 1/(r-1)^2$ whenever $f(x) \neq g(x)$. This implies

$$\begin{aligned} \Pr_{x \in U^n}[g(x) \neq f(x)] \leq \\ \Pr_{x \in U^n}[(r-1)^2 \cdot |f(x) - h(x)|^2 \geq 1] \leq \\ \mathbb{E}_{x \in U^n}[(r-1)^2 \cdot |f(x) - h(x)|^2] \leq \\ (r-1)^2 \cdot \mathbb{E}_{x \in U^n}[|f(x) - h(x)|^2] \leq \epsilon (r-1)^2. \end{aligned}$$

The last inequality follows from the definition of ϵ-approximation. □

Extension of our learning algorithm to the agnostic setting follows from the result of Gopalan, Kalai and Klivans.

Theorem 4.4.10 ([108]). *If every function f in a class C has an M-sparse ϵ-approximation, then there is an agnostic learning algorithm for C with running time poly$(n, M, 1/\epsilon)$.*

This completes the proof of Theorem 4.1.2. □

Chapter 5

Transitive-Closure Spanners and Testing Functions on Hypergrids

5.1 Introduction

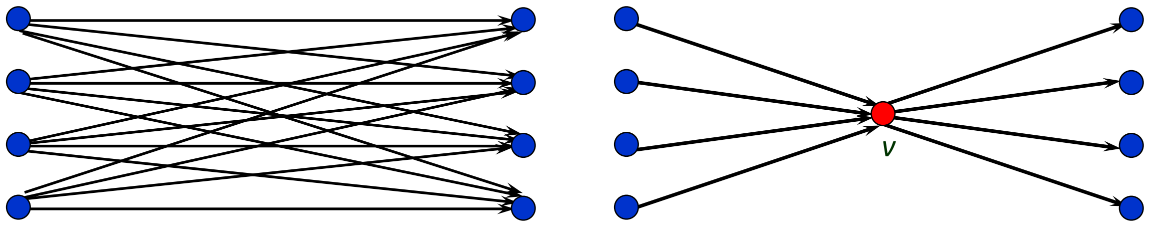
Graph spanners were introduced in the context of distributed computing by Awerbuch [16] and Peleg and Schäffer [165], and since then have found numerous applications. Our focus is on transitive-closure spanners, introduced explicitly in [36], but studied prior to that in many different contexts [51, 50, 198, 10, 53, 187, 39, 188, 189, 75, 117, 6].

Given a directed graph $G = (V, E)$ and an integer $k \geq 1$, a **k-transitive-closure spanner** (k-TC-spanner) of G is a directed graph $H = (V, E_H)$ such that: (1) E_H is a subset of the edges in the transitive closure of G; (2) for all vertices $u, v \in V$, if $d_G(u, v) < \infty$ then $d_H(u, v) \leq k$ and if $d_G(u, v) = \infty$ then $d_H(u, v) = \infty$, where $d_G(u, v)$ denotes the distance from u to v in G. That is, a k-TC-spanner is a graph with a small diameter that preserves the connectivity of the original graph. The edges of the transitive closure of G, added to G to obtain a TC-spanner, are called *shortcuts* and the parameter k is called the *stretch*.

TC-spanners have numerous applications, and there has been a lot of work on finding sparse TC-spanners for specific graph families. See [170] for a survey. In some applications of TC-spanners, in particular, to access control hierarchies [6, 66], the shortcuts can use *Steiner* vertices, that is, vertices not in the original graph G. The resulting spanner is called a *Steiner TC-spanner*.

Definition 5.1.1 (Steiner TC-spanner)**.** *Given a directed graph $G = (V, E)$ and an integer $k \geq 1$, a **Steiner k-transitive-closure spanner (Steiner k-TC-spanner)** of G is a directed graph $H = (V_H, E_H)$ such that: (1) $V \subseteq V_H$; (2) for all vertices $u, v \in V$, if $d_G(u, v) < \infty$ then $d_H(u, v) \leq k$ and if $d_G(u, v) = \infty$ then $d_H(u, v) = \infty$. Vertices in $V_H \backslash V$ are called* Steiner vertices.

For some graphs, Steiner TC-spanners can be significantly sparser than ordinary TC-spanners. For example, consider a complete bipartite graph $K_{\frac{n}{2},\frac{n}{2}}$ with $n/2$ vertices in each part and all edges directed from the first part to the second. Every ordinary 2-TC-spanner of this graph has $\Omega(n^2)$ edges. However, $K_{\frac{n}{2},\frac{n}{2}}$ has a Steiner 2-TC-spanner with n edges: it is enough to add one Steiner vertex v, edges to v from all nodes in the left part, and edges from v to all nodes in the right part. Thus, for $K_{\frac{n}{2},\frac{n}{2}}$ there is a factor of $\Theta(n)$ gap between the size of the sparsest Steiner 2-TC-spanner and the size of an ordinary 2-TC-spanner.



We focus on Steiner TC-spanners of directed *acyclic* graphs (DAGs) or, equivalently, partially ordered sets (posets). They represent the most interesting case in applications of TC-spanners. In addition, there is a reduction from constructing TC-spanners of graphs with cycles to constructing TC-spanners of DAGs, with a small loss in stretch ([170], Lemma 3.2), which also applies to Steiner TC-spanners.

The goal of this work is to understand the minimum number of edges needed to form a Steiner k-TC-spanner of a given graph G as a function of n, the number of nodes in G. More specifically, motivated by applications to access control hierarchies [6, 66] and property reconstruction [34, 128], described in Section 5.1.2, we study the relationship between the dimension of a poset and the size of its sparsest Steiner TC-spanner. The *dimension* of a poset G is the smallest d such that G can be embedded into a d-dimensional directed hypergrid via an order-preserving embedding. (See Definition 5.2.1). Atallah *et al.* [6], followed by De Santis *et al.* [66], use Steiner TC-spanners in key management schemes for access control hierarchies. They argue that many access control hierarchies are low-dimensional posets that come equipped with an embedding demonstrating low dimensionality. For this reason, we focus on the setting where the dimension d is small relative to the number of nodes n.

We also study the size of sparsest (Steiner) 2-TC-spanners of specific posets of dimension d, namely, d-dimensional directed hypergrids. Our lower bound on this quantity improves the lower bound of [34] and nearly matches their upper bound. It implies that our construction of Steiner 2-TC-spanners of d-dimensional posets is optimal up to a constant factor for any constant number of dimensions. It also has direct implications for property reconstruction. The focus on stretch $k = 2$ is motivated by this application.

Several classes of posets, in addition to low-dimensional hypergrids, are known to have small dimension. For example, if a *planar poset*, that is, a poset with a planar Hasse diagram, has *both* a minimum *and* a maximum element, it has dimension at most 2. A planar poset with *either* a minimum or a maximum element has dimension at most 3 [192]. (In general, however, a planar poset can have an arbitrary dimension [133]). One can also bound the dimension in terms of cardinality of the poset: every poset of cardinality $n \geq 4$ has dimension at most $n/2$ [118]. Also, if every element in an n-element poset has at most u points above it, then its dimension is at most $2(u + 1)\log n + 1$ [97]. Thus, posets with low dimension occur quite naturally in a variety of settings.

5.1.1 Results

5.1.1.1 Steiner 2-TC-spanners of Directed d-dimensional Grids.

The *directed hypergrid*, denoted $\mathcal{H}_{m,d}$, has vertex set[1] $[m]^d$ and edge set $\{(x, y) : \exists$ unique $i \in [d]$ such that $y_i - x_i = 1$ and if $j \neq i, y_j = x_j\}$. We observe (in Corollary 5.2.4) that for the grid $\mathcal{H}_{m,d}$, Steiner vertices do not help to create sparser k-TC-spanners. In [34], it was shown that for $m \geq 3$, sparsest (ordinary) 2-TC-spanners of $\mathcal{H}_{m,d}$ have size at most $m^d \log^d m$ and at least $\Omega\big(\frac{m^d \log^d m}{(2d \log\log m)^{d-1}}\big)$. They also give tight upper and lower bounds for the case of constant m and large d. Our first result is an improvement on the lower bound for the hypergrid for the case when m is significantly larger than d, i.e., the setting in the above applications.

Theorem 5.1.1. *Every (Steiner) 2-TC-spanner of $\mathcal{H}_{m,d}$ has $\Omega\Big(\dfrac{m^d(\ln m - 1)^d}{(4\pi)^d}\Big)$ edges.*

The proof of Theorem 5.1.1 constructs a dual solution to a linear programming relaxation of the 2-TC-spanner problem. We consider a linear program (LP) for the sparsest 2-TC-spanner of $\mathcal{H}_{m,d}$. Our program is a special case of a more general LP for the sparsest directed k-spanner of an arbitrary graph G, used in [36] to obtain an approximation algorithm for that problem. We show that for our special case the integrality gap of this LP is small and, in particular, does not depend on n.

Specifically, we find a solution to the dual LP by selecting initial values that have a combinatorial interpretation: they are expressed in terms of the *volume* of d-dimensional *boxes* contained in $\mathcal{H}_{m,d}$. For example, the dual variable corresponding to the constraint that enforces the existence of a length-2 path from u to v in the 2-TC-spanner is initially assigned a value inversely proportional to the number of nodes on the paths from u to v. The final sum of the constraints is bounded by an integral which, in turn, is bounded by an expression depending only on the dimension d.

We note that the best lower bound known previously [34] was proved by a long and sophisticated combinatorial argument that carefully balanced the number of edges that stay within different parts of the hypergrid and the number of edges that cross from one part to another. The recursion in the combinatorial argument is an inherent limitation of [34], resulting in suboptimal bounds even for constant d. In contrast, our linear programming argument can be thought of as assigning types to edges based on the volume of the boxes they define, and automatically balancing the number of edges of different types by selecting the correct coefficients for the constraints corresponding to those edges. It achieves an optimal bound for any constant number of dimensions.

Steiner TC-spanners of General d-dimensional Posets. We continue the study of the number of edges in a sparsest Steiner k-TC-spanner of a poset as a function of its dimension, following [6, 66]. We note that the only poset of dimension 1 is the directed line $\mathcal{H}_{n,1}$. TC-spanners of directed lines were discovered under many different guises. They were studied implicitly in [198, 51, 50, 10, 53, 75, 13] and explicitly in [39, 189]. Chandra, Fortune and Lipton [51, 50] implicitly showed that, for constant k, the size of

[1]For a positive integer m, we denote $\{1, \ldots, m\}$ by $[m]$.

Stretch k	Prior bounds on $S_k(G)$
$2d-1$	$O(n^2)$ [6]
$2d-2+t$ for $t \geq 2$	$O(n(\log^{d-1} n)\lambda_t(n))$ [6]
$2d+O(\log^* n)$	$O(n\log^{d-1} n)$ [6]
3	$O(n\log^{d-1} n \log\log n)$ for fixed d [66]

Stretch k	Our bounds on $S_k(G)$	
2	$O(n\log^d n)$	$\Omega\left(n\left(\frac{\log n}{cd}\right)^d\right)$ for a fixed $c > 0$
≥ 3		$\Omega(n\log^{\lceil (d-1)/k \rceil} n)$ for fixed d

Table 5.1: The size of the sparsest Steiner k-TC-spanner for d-dimensional posets on n vertices for $d \geq 2$

the sparsest k-TC-spanner of the directed line is $\Theta(n \cdot \lambda_k(n))$, where $\lambda_k(n)$ is the k^{th}-row inverse Ackermann function[2] .

Table 5.1 compares old and new results for $d \geq 2$. $S_k(G)$ denotes the number of edges in the sparsest Steiner k-TC-spanner of G. The upper bounds hold for all posets of dimension d. The lower bounds mean that there is an infinite family of d-dimensional posets for which all Steiner k-TC-spanners have the specified number of edges.

Atallah *et al.* constructed Steiner k-TC-spanners with k proportional to d. De Santis *et al.* improved their construction for constant d. They achieved $O(3^{d-t}nt\log^{d-1} n \log\log n)$ edges for odd stretch $k = 2t+1$, where $t \in [d]$. In particular, setting $t = 1$ gives $k = 3$ and $O(n\log^{d-1} n\log\log n)$ edges.

We present the first construction of Steiner 2-TC-spanners for d-dimensional posets. In our construction, the spanners have $O(n\log^d n)$ edges, and the length-2 paths can be found in $O(d)$ time. This result is stated in Theorem 5.2.2 (in Section 5.2). Our construction, like all previous constructions, takes as part of the input an explicit embedding of the poset into a d-dimensional grid. (Finding such an embedding is NP-hard [196]. Also, as mentioned previously, in the application to access control hierarchies, such an embedding is usually given.) The Steiner vertices used in our construction for d-dimensional posets are necessary to obtain sparse TC-spanners, as manifested by the example presented after the proof of Theorem 5.2.2.

Theorem 5.1.1 implies that there is an absolute constant $c > 0$ for which our upper bound for $k = 2$ is tight within an $O((cd)^d)$ factor, showing that no drastic improvement in the upper bound is possible. To obtain a bound in terms of the number n of vertices and dimension d, substitute n for m^d and $(\ln n)/d$ for $\ln m$ in the theorem statement. This works for all n larger than some constant to the power d and gives the following corollary.

Corollary 5.1.2. *There is an absolute constant $c > 0$ for which for all $d \geq 2$ and n larger than some constant to the power d, there exists a d-dimensional poset G on n vertices such that every Steiner 2-TC-spanner of G has $\Omega\big(n\big(\frac{\log n}{cd}\big)^d\big)$ edges.*

In addition, we prove a lower bound for all constant $k > 2$ and constant dimension d, which qualitatively matches known upper bounds. It shows that, in particular, ev-

[2]The *Ackermann function* [2] is defined by: $A(1,j) = 2^j$, $A(i+1,0) = A(i,1)$, $A(i+1,j+1) = A(i, 2^{2^{A(i+1,j)}})$. The inverse Ackermann function is $\alpha(n) = \min\{i : A(i,1) \geq n\}$ and the i^{th}-row inverse is $\lambda_i(n) = \min\{j : A(i,j) \geq n\}$. Specifically, $\lambda_2(n) = \Theta(\log n)$, $\lambda_3(n) = \Theta(\log\log n)$ and $\lambda_4(n) = \Theta(\log^* n)$.

ery Steiner 3-TC-spanner has size $\Omega(n \log n)$, and even with significantly larger constant stretch, every Steiner TC-spanner has size $n \log^{\Omega(d)} n$.

Theorem 5.1.3. *For all constant $d \geq 2$ and sufficiently large n, there exists a d-dimensional poset G on n vertices such that for all $k \geq 3$, every Steiner k-TC-spanner of G has $\Omega(n \log^{\lceil (d-1)/k \rceil} n)$ edges.*

This theorem (see Section 5.4) captures the dependence on d and greatly improves upon the previous $\Omega(n \log \log n)$ bound, which follows trivially from known lower bounds for 3-TC-spanners of a directed line.

The lower bound on the size of a Steiner k-TC-spanner for $k \geq 3$ is proved by the probabilistic method. We note that using the hypergrid as an example of a poset with large Steiner k-TC-spanners for $k > 2$ would yield a much weaker lower bound because $\mathcal{H}_{m,d}$ has a 3-TC-spanner of size $O((m \log \log m)^d)$ and, more generally, a k-TC-spanner of size $O((m \cdot \lambda_k(m))^d)$, where $\lambda_k(m)$ is the k^{th}-row inverse Ackermann function [34]. Instead, we construct an n-element poset embedded in $\mathcal{H}_{n,d}$ using the following randomized procedure: all poset elements differ on coordinates in dimension 1, and for each element, the remaining $d-1$ coordinates are chosen uniformly at random from $[n]$. We consider a set of partitions of the underlying hypergrid into d-dimensional boxes, and carefully count the expected number of edges in a Steiner k-TC-spanner that cross box boundaries for each partition. We show that each edge is counted only a small number of times, proving that the expected number of edges in a Steiner k-TC-spanner is large. We conclude that some poset attains the expected number of edges.

Organization. We explain applications of Steiner TC-spanners in Section 5.1.2. Section 5.2 gives basic definitions and observations. In particular, our construction of sparse Steiner 2-TC-spanners for d-dimensional posets (the proof of Theorem 5.2.2) is presented there. Our lower bounds constitute the main technical contribution of this paper. The lower bound for the hypergrid for $k = 2$ (Theorem 5.1.1) is proved in Section 5.3. The lower bound for $k > 2$ (Theorem 5.1.3) is presented in Section 5.4.

5.1.2 Applications

Numerous applications of TC-spanners are surveyed in [170]. We focus on two of them: property reconstruction, described in [34, 128], and key management for access control hierarchies, described in [6, 36, 66].

Property Reconstruction. Property-preserving data reconstruction was introduced by Ailon, Chazelle, Comandur and Liu [5]. In this model, a reconstruction algorithm, called a *filter*, sits between a *client* and a *dataset*. A dataset is viewed as a function $f : \mathcal{D} \to \mathcal{R}$. A client accesses the dataset using *queries* of the form $x \in \mathcal{D}$ to the filter. The filter *looks up* a small number of values in the dataset and outputs $g(x)$, where g must satisfy a pre-specified *structural* property (e.g., be monotone or have a low Lipschitz constant) and differ from f as little as possible. Extending this notion, Saks and Seshadhri [181] defined *local* reconstruction. A filter is *local* if the output function g does not depend on the order of the queries. Local filters can be used for distributed computations [181] and in private data analysis [128]. A filter is *nonadaptive* if its lookups do not depend on the answers to previous lookups.

Our results on TC-spanners are relevant to reconstruction of two properties of functions: monotonicity and having a low Lipschitz constant. Reconstruction of monotone functions was considered in [5, 34, 181]. A function $f : [m]^d \to \mathbb{R}$ is called *monotone* if $f(x) \leq f(y)$ for all $(x, y) \in E(\mathcal{H}_{m,d})$. Reconstruction of functions with low Lipschitz constant was studied in [128]. A function $f : [m]^d \to \mathbb{R}$ has Lipschitz constant c if $|f(x) - f(y)| \leq c \cdot |x - y|_1$. In [34], the authors proved that the existence of a local filter for monotonicity of functions with low lookup complexity implies the existence of a sparse 2-TC-spanner of $\mathcal{H}_{m,d}$. In [128], an analogous connection was drawn between local reconstruction of functions with low Lipschitz constant and 2-TC-spanners. Our improvement in the lower bound on the size of 2-TC-spanners of $\mathcal{H}_{m,d}$ directly translates into an improvement by the same factor in the lower bounds on lookup complexity of local nonadaptive filters for these two properties, showing they are optimal for any constant d.

Key Management for Access Control Hierarchies. Atallah *et al.* [6] used sparse Steiner TC-spanners to construct efficient key management schemes for access control hierarchies. An *access hierarchy* is a partially ordered set G of access classes. Each user is entitled to access a certain class and all classes reachable from the corresponding node in G.

One approach to enforcing the access hierarchy is to use a key management scheme of the following form [6, 66]. Each edge (i, j) has an associated public key $P(i, j)$, and each node i, an associated secret key k_i. Only users with the secret key for a node have the required permissions for the associated access class. The public and secret keys are designed so that there is an efficient algorithm A which takes k_i and $P(i, j)$ and generates k_j, but for each (i, j) in G, it is computationally hard to generate k_j without knowledge of k_i. Thus, a user can efficiently generate the required keys to access a descendant class, but not other classes. The number of runs of algorithm A needed to generate a secret key k_v from a secret key k_u is equal to $d_G(u, v)$. To speed this up, Atallah *et al.* suggest adding edges and nodes to G to increase connectivity. To preserve the access hierarchy represented by G, the new graph H must be a Steiner TC-spanner of G. The number of edges in H corresponds to the space complexity of the scheme, while the stretch k of the spanner corresponds to the time complexity.

We note that the time to find the path from u to v is also important in this application. In our upper bound, this time is $O(d)$, which for small d (e.g., constant) is likely to be much less than $2g(n)$, where $g(n)$ is the time to run algorithm A. This is because algorithm A involves the evaluation of a cryptographic hash function, which is expensive in practice and in theory.[3]

5.2 Definitions and Observations

For integers $j \geq i$, interval $[i, j]$ refers to the set $\{i, i + 1, \ldots, j\}$. Logarithms are always base 2, except for ln which is the natural logarithm.

Each DAG $G = (V, E)$ is equivalent to a poset with elements V and partial order $\preceq$, where $x \preceq y$ if y is reachable from x in G. Elements x and y are *comparable* if $x \preceq y$ or

[3]Any hash function which is secure against $\mathrm{poly}(n)$-time adversaries requires $g(n) \geq \mathrm{polylog}\, n$ evaluation time under existing number-theoretic assumptions.

$y \preceq x$, and *incomparable* otherwise. We write $x \prec y$ if $x \preceq y$ and $x \neq y$. The *hypergrid* $\mathcal{H}_{m,d}$ *with dimension* d *and side length* m was defined in the beginning of Section 5.1.1. Equivalently, it is the poset on elements $[m]^d$ with the *dominance order*, defined as follows: $x \preceq y$ for two elements $x, y \in [m]^d$ iff $x_i \leq y_i$ for all $i \in [d]$.

A mapping from a poset G to a poset G' is called an *embedding* if it respects the partial order, that is, all $x, y \in G$ are mapped to $x', y' \in G'$ such that $x \preceq_G y$ iff $x' \preceq_{G'} y'$.

Definition 5.2.1 (Poset dimension, [80])**.** *Let G be a poset with n elements. The* dimension *of G is the smallest integer d such that G can be embedded into the hypergrid $\mathcal{H}_{n,d}$.*

Dushnik and Miller [79] proved that for any $m > 1$, the hypergrid $\mathcal{H}_{m,d}$ has dimension exactly d.

Fact 5.2.1. *Each d-dimensional poset G with n elements can be embedded into a hypergrid $\mathcal{H}_{n,d}$, so that for all $i \in [d]$, the ith coordinates of images of all elements are distinct. Moreover, such an embedding can be obtained from an arbitrary embedding of G into $\mathcal{H}_{n,d}$ in time $O(dn \log n)$.*

Proof. Let G be a d-dimensional poset G with n elements. By Definition 5.2.1, it can be embedded into a hypergrid $\mathcal{H}_{m,d}$. For an element $x \in G$ let $(x_1, \ldots, x_d)$ be the d-dimensional vector of coordinates of x in the embedding above. If $m > n$, we can sort all these vectors in lexicographic order in time $O(dn \log n)$ since pairwise comparisons can be done in $O(d)$ time. This gives a linear extension of G, which we denote as L. Finally, for each dimension $i \in [d]$, in $O(n)$ time, we can go through the list of elements sorted by the ith coordinate, and reassign the ith coordinates, so that all of them are distinct, resolving ties according to the order of L.

It remains to show that this transformation, call it f, does not change the partial order of G. Consider a pair of elements $x \prec y$ in G. If $x_i < y_i$ for some $i \in [d]$ then $f(x)_i < f(y)_i$ because we did not change the order of distinct coordinates. If $x_i = y_i$ for some $i \in [d]$ then $f(x)_i < f(y)_i$ since x precedes y in L. Therefore, $f(x) \prec f(y)$. Finally, consider incomparable elements x and y of G. Since they are incomparable, $x_i < y_i$ while $x_j > y_j$ for some $i, j \in [d]$. Then $f(x)_i < f(y)_i$ while $f(x)_j > f(y)_j$, and consequently, $f(x)$ and $f(y)$ are incomparable, as required. □

Sparse Steiner 2-TC-spanners for d-dimensional Posets. We give a simple construction of sparse Steiner 2-TC-spanners for d-dimensional posets. For constant d, it matches the lower bound from Section 5.3 up to a constant factor. Note that the construction itself works for arbitrary, not necessarily constant, d.

Theorem 5.2.2. *Each d-dimensional poset G on n elements has a Steiner 2-TC-spanner H of size $O(n \log^d n)$. Given an embedding of G into the hypergrid $\mathcal{H}_{n,d}$, graph H can be constructed in time $O(dn \log^d n)$. Moreover, for all $x, y \in G$, where $x \prec y$, one can find a path in H from x to y of length at most 2 in time $O(d)$.*

Proof. Consider an n-element poset G embedded into the hypergrid $\mathcal{H}_{n,d}$. Transform it, so that for all $i \in [d]$, the ith coordinates of images of all elements are distinct. (See Fact 5.2.1.) In this proof, assume that the hypergrid coordinates start with 0, i.e., its vertex set is $[0, n-1]^d$. Let $\ell = \lceil \log n \rceil$ and $b(t)$ be the ℓ-bit binary representation of t, possibly with leading zeros. Let $p_i(t)$ denote the i-bit prefix of $b(t)$ followed by a single 1 and then $\ell - i - 1$ zeros. Let $lcp(t_1, t_2) = p_i(t_1)$, where i is the length of the longest common prefix of $b(t_1)$ and $b(t_2)$.

To construct a Steiner 2-TC-spanner (V_H, E_H) of G, we insert at most ℓ^d edges into E_H per each poset element. Consider a poset element with coordinates $x = (x_1, \dots, x_d)$ in the embedding. For each d-tuple $(i_1, \dots, i_d) \in [0, \ell - 1]^d$, let p be a hypergrid vertex whose coordinates have binary representations $(p_{i_1}(x_1), \dots, p_{i_d}(x_d))$. If $x \prec p$, we add an edge (x, p) to E_H; otherwise, if $p \prec x$ we add an edge (p, x) to E_H. Note that only edges between comparable points are added to E_H.

Observe that for $d > (2 \log n)/(\log \log n)$, the theorem is trivial since then $n \log^d n > n^3$, and the transitive-closure of G has $O(n^2)$ edges and can be computed in $O(n^3)$ time. For smaller d, $\lceil \log n \rceil^d = O(\log^d n)$ and, consequently, E_H contains $O(n \log^d n)$ edges and can be constructed in $O(dn \log^d n)$ time, as described, if bit operations on coordinates can be performed in $O(1)$ time.

For all pairs of poset elements $x = (x_1, \dots, x_d)$ and $y = (y_1, \dots, y_d)$, such that $x \prec y$, there is an intermediate point z with coordinates whose binary representations are $(lcp(x_1, y_1), \dots, lcp(x_d, y_d))$. By construction, both edges (x, z) and (z, y) are in E_H. Point z can be found in $O(d)$ time, since $lcp(x_i, y_i)$ can be computed in $O(1)$ time, assuming $O(1)$ time bit operations on coordinates. $\square$

Note that the Steiner vertices used in our construction for d-dimensional posets are necessary to obtain sparse TC-spanners. Recall our example of a bipartite graph $K_{\frac{n}{2},\frac{n}{2}}$ for which every 2-TC-spanner required $\Omega(n^2)$ edges. $K_{\frac{n}{2},\frac{n}{2}}$ is a poset of dimension 2, and thus, by the upper bound in Theorem 5.2.2, has a Steiner 2-TC-spanner of size $O(n \log^2 n)$. (As we mentioned before, for this graph there is an even better Steiner 2-TC-spanner with $O(n)$ edges.) To see that $K_{\frac{n}{2},\frac{n}{2}}$ is embeddable into a $[n] \times [n]$ grid, map each of the $n/2$ left vertices of $K_{\frac{n}{2},\frac{n}{2}}$ to a distinct grid vertex in the set of incomparable vertices $\{(i, n/2 + 1 - i) : i \in [n/2]\}$, and similarly map each right vertex to a distinct vertex in the set $\{(n + 1 - i, i + n/2) : i \in [n/2]\}$. It is easy to see that this is a proper embedding.

Equivalence of Steiner and non-Steiner TC-spanners for Hypergrids. Our lower bound on the size of 2-TC-spanners for d-dimensional posets of size n is obtained by proving a lower bound on the size of the Steiner 2-TC-spanner of $\mathcal{H}_{m,d}$ where $m = n^{1/d}$. The following lemma, used in Section 5.4, implies Corollary 5.2.4 that shows that sparsest Steiner and non-Steiner 2-TC-spanners of $\mathcal{H}_{m,d}$ have the same size.

Lemma 5.2.3. *Let G be a poset on elements $V \subseteq [m]^d$ with the dominance order and $H = (V_H, E_H)$ be a Steiner k-TC-spanner of G with minimal V_H. Then H can be embedded into $\mathcal{H}_{m,d}$.*

Proof. For each $s \in V_H - V$, we define $Prev(s) = \{x \in V : x \prec s\}$. If $Prev(s) = \varnothing$ then V_H is not minimal because H remains a Steiner k-TC-spanner of G when s is removed. We map each Steiner vertex s to $r(s)$, the replacement of s in $[m]^d$, whose ith coordinates for all $i \in [d]$ are $\max_{x \in Prev(s)} x_i$.

Consider an edge (x, y) in H. If $x, y \in V$ our embedding does not alter that edge. If $x \in V$, $y \in V_H - V$ then $x \in Prev(y)$ and $x \prec r(y)$ by the definition of r. If $x, y \in V_H - V$ then $Prev(x) \subseteq Prev(y)$ and the monotonicity of $\max(S)$ for sets implies $r(x) \preceq r(y)$.

Finally, if $x \in V_H - V$ and $y \in V$ then for each $z \in Prev(x)$ and each $i \in [d]$, we have $z_i \leq y_i$ because $z \prec x \prec y$, and this implies $r(x) \preceq y$. □

Corollary 5.2.4. *If $\mathcal{H}_{m,d}$ has a Steiner k-TC-spanner H, it also has a k-TC-spanner with the same number of nodes and at most the same number of edges.*

5.3 Lower Bound for 2-TC-spanners of the Hypergrid

In this section, we prove Theorem 5.1.1 that gives a nearly tight lower bound on the size of (Steiner) 2-TC-spanners of the hypergrids $\mathcal{H}_{m,d}$. By Corollary 5.2.4, we only have to consider non-Steiner TC-spanners.

Proof of Theorem 5.1.1. We start by introducing an LP relaxation for the sparsest 2-TC-spanner of an arbitrary graph. Our lower bound on the size of a 2-TC-spanner of $\mathcal{H}_{m,d}$ is obtained by finding a feasible solution to the dual program, which, by definition, gives a lower bound on the objective function of the primal.

An Integer Program for Sparsest 2-TC-spanner. For each graph $G = (V, E)$, we can find the size of a sparsest 2-TC-spanner by solving the following integer LP, a special case of an LP from [36] for directed k-spanners. For all vertices $u, v \in V$ satisfying $u \preceq v$, we introduce variables $x_{uv} \in \{0, 1\}$. For $u \neq v$, they correspond to potential edges in a 2-TC-spanner H of G. (We need variables x_{vv} for notational convenience in the last part of the proof.) For all vertices $u, v, w \in V$ satisfying $u \preceq w \preceq v$, we introduce auxiliary variables $x'_{uwv} \in \{0, 1\}$, corresponding to potential paths of length at most 2 in H. The integer LP is as follows:

$$\begin{array}{lll}
\text{minimize} & \displaystyle\sum_{u,v\colon u \preceq v} x_{uv} & \\
\text{subject to} & x_{uw} - x'_{uwv} \geq 0, x_{wv} - x'_{uwv} \geq 0 & \forall u, v, w\colon u \preceq w \preceq v; \\
& \displaystyle\sum_{w\colon u \preceq w \preceq v} x'_{uwv} \geq 1 & \forall u, v\colon u \preceq v.
\end{array}$$

Given a solution to the LP, we can construct a 2-TC-spanner $H = (V, E_H)$ of G of size not exceeding the value of the objective function by including (u, v) in E_H iff the corresponding variable $x_{uv} = 1$ and $u \neq v$. In the other direction, given a 2-TC-spanner $H = (V, E_H)$ of G, we can find a feasible solution of the LP with the value of the objective function not exceeding $|E_H| + |V|$. Let $E'_H = E_H \cup L$, where L is the set of loops (v, v) for all $v \in V$. Then we set $x_{uv} = 1$ iff $(u, v) \in E'_H$ and $x'_{uwv} = 1$ iff both $(u, w) \in E'_H$ and $(w, v) \in E'_H$. Therefore, the size of a sparsest 2-TC-spanner of G and the optimal value of the objective function of the LP differ by at most $|V|$. They are asymptotically equivalent because $|V| = O(|E_H|)$ for every weakly connected graph G.

A Fractional Relaxation of the Dual LP. Every feasible solution of the following fractional relaxation of the dual LP gives a lower bound on the optimal value of the objective function of the primal:

$$\text{maximize} \quad \sum_{u,v\colon u \preceq v} y_{uv}$$

$$\text{subject to} \quad \sum_{w\colon v\preceq w} y'_{uvw} + \sum_{w\colon w\preceq u} y''_{wuv} \le 1 \qquad \forall u,v\colon u\preceq v; \qquad (5.1)$$

$$y_{uv} - y'_{uwv} - y''_{uwv} \le 0 \qquad \forall u,v,w\colon u\preceq w\preceq v; \qquad (5.2)$$

$$y_{uv} \ge 0, y'_{uwv} \ge 0, y''_{uwv} \ge 0 \qquad \forall u,v,w\colon u\preceq w\preceq v.$$

Finding a Feasible Solution for the Dual. When the graph G is a hypergrid $\mathcal{H}_{n,d}$, we can find a feasible solution of the dual, which gives a lower bound on the objective function of the primal. To do that, we perform the following three steps. First, we choose initial values $\hat{y}_{uv}$ for the variables y_{uv} of the dual program and, in Lemma 5.3.1, give a lower bound on the resulting value of the objective function of the primal program. Second, we choose initial values $\hat{y}'_{uvw}$ and $\hat{y}''_{uvw}$ for variables y'_{uvw} and y''_{uvw} so that (5.2) holds. Finally, in Lemma 5.3.2, we give an upper bound on the left-hand side of (5.1) for all $u\preceq v$. Our bound is a constant larger than 1 and independent of n. We obtain a feasible solution to the dual by dividing the initial values of the variables (and, consequently, the value of the objective function) by this constant.

Step 1. For a vector $x=(x_1,\ldots,x_d)\in[0,m-1]^d$, let the *volume* $V(x)$ denote $\prod_{i\in[d]}(x_i+1)$. This corresponds to the number of hypergrid points inside a d-dimensional box with corners u and v, where $v-u=x$. We start building a solution to the dual by setting $\hat{y}_{uv}=\frac{1}{V(v-u)}$ for all $u\preceq v$. This gives the value of the objective function of the dual program, according to the following lemma.

Lemma 5.3.1. $\sum_{u,v\colon u\preceq v} \hat{y}_{uv} > m^d(\ln m-1)^d.$

Proof. Substituting $1/(V(v-u))$ for $\hat{y}_{uv}$, we get:

$$\begin{aligned}\sum_{u,v\colon u\preceq v}\hat{y}_{uv} &= \sum_{u,v\colon u\preceq v}\frac{1}{V(v-u)} = \sum_{l\in[m]^d}\prod_{i\in[d]}\frac{m-l_i+1}{l_i} = \left(\sum_{l\in[m]}\frac{m-l+1}{l}\right)^d \\ &> ((m+1)\ln(m+1)-m)^d > m^d(\ln m-1)^d.\end{aligned}$$

□

Step 2. The values of $\hat{y}'_{uvw}$ and $\hat{y}''_{uvw}$ are set as follows to satisfy (5.2) tightly (without any slack):

$$\hat{y}'_{uvw} = \hat{y}_{uw}\frac{V(v-u)}{V(v-u)+V(w-v)},\ \hat{y}''_{uvw} = \hat{y}_{uw}-\hat{y}'_{uvw} = \hat{y}_{uw}\frac{V(w-v)}{V(v-u)+V(w-v)}.$$

Step 3. The initial values $\hat{y}'_{uvw}$ and $\hat{y}''_{uvw}$ do not necessarily satisfy (5.1). The following lemma gives an upper bound on the left-hand side of all constraints in (5.1).

Lemma 5.3.2. *For all* $u\preceq v$, $\sum_{w\colon v\preceq w}\hat{y}'_{uvw} + \sum_{w\colon w\preceq u}\hat{y}''_{wuv} \le (4\pi)^d.$

Proof. Below we denote $v-u$ by $x^0=(x^0_1,\ldots,x^0_d)$, a d-dimensional vector of ones (1, ...,

1) by $\vec{1}$ and $\prod_{i\in[d]} dx_i$ by dx.

$$\begin{aligned}
\sum_{w\colon v\preceq w} \hat{y}'_{uvw} \quad + & \sum_{w\colon w\preceq u} \hat{y}''_{wuv} \\
= & \sum_{w\colon v\preceq w} \hat{y}_{uw}\frac{V(v-u)}{V(v-u)+V(w-v)} \\
+ & \sum_{w\colon w\preceq u} \hat{y}_{wv}\frac{V(v-u)}{V(u-w)+V(v-u)} \\
= & \sum_{w\colon v\preceq w} \frac{V(v-u)}{V(w-u)(V(v-u)+V(w-v))} \\
+ & \sum_{w\colon w\preceq u} \frac{V(v-u)}{V(v-w)(V(u-w)+V(v-u))} \\
< & \ 2\sum_{x\in[0,m]^d} \frac{V(x^0)}{V(x^0+x)(V(x^0)+V(x))} \\
\leq & \ 2^{2d+1}\sum_{x\in[1,m+1]^d} \frac{V(x^0)}{V(x^0+x)(V(x^0)+V(x))} \qquad (5.3) \\
< & \ 2^{2d+1}\int_{\mathbb{R}^d_+} \frac{V(x^0)dx}{V(x^0+x)(V(x^0)+V(x))} \qquad (5.4) \\
= & \ 2^{2d+1}\int_{\mathbb{R}^d_+} \frac{V^2(x^0)dt}{V(t)V(x_0)(V(x^0)+\prod\limits_i(t_i(x^0_i+1)+1))} \qquad (5.5) \\
< & \ 2^{2d+1}\int_{\mathbb{R}^d_+} \frac{V(x^0)dt}{V(t)(V(x^0)+\prod\limits_i t_i(x^0_i+1))} \\
= & \ 2^{2d+1}\int_{\mathbb{R}^d_+} \frac{dt}{V(t)(\vec{1}+V(t-1))}.
\end{aligned}$$

The first two equalities above are obtained by plugging in values of $\hat{y}'$ and $\hat{y}''$ from Steps 1 and 2 with appropriate indices. The first inequality is obtained by extending each sum to the whole subgrid. Here (5.3) holds because $\frac{1}{V(u)} \leq \frac{2^d}{V(u+1)}$ for all u, such that $u_i \geq 0$. In (5.4), the sum can be bounded from above by the integral because the summand is monotone in all variables. To get (5.5), we substitute x with t such that $x_i = t_i(x^0_i+1)$. Then $V(x_0+x) = V(t)V(x_0)$, and $dx = V(x_0)dt$. To obtain the last inequality, we substitute $V(x^0)$ for $\prod\limits_i(x^0_i+1)$.

Proposition 5.3.3. *Let $I_d = \int_{\mathbb{R}^d_+} \frac{dt}{V(t)(\vec{1}+V(t-1))}$. Then $I_d \leq \frac{\pi^d}{2}$ for all d.*

Proof. To bound the integral I_d, we first make a substitution $x_i = \frac{1-t_i}{1+t_i}$:

$$I_d = \int\limits_{[-1\ldots1]^d} \frac{dx}{\prod\limits_{1\le i\le d}(1+x_i) + \prod\limits_{1\le i\le d}(1-x_i)}.$$

Then we bound the denominator using the inequality $a+b \ge 2\sqrt{ab}$ and get

$$I_d \le \int\limits_{[-1\ldots1]^d} \frac{dx}{2\sqrt{\prod\limits_{1\le i\le d}(1+x_i) \times \prod\limits_{1\le i\le d}(1-x_i)}} = \frac{J^d}{2},$$

where J denotes the following integral:

$$J = \int\limits_{-1}^{1} \frac{dx}{\sqrt{1-x^2}} = \pi.$$

Therefore, $I_d \le \frac{\pi^d}{2}$, as claimed. □

Lemma 5.3.2 follows from Proposition 5.3.3. □

Finally, we obtain a feasible solution by dividing initial values $\hat{y}_{uv}$, $\hat{y}'_{uvw}$ and $\hat{y}''_{uvw}$ by the upper bound $(4\pi)^d$ from Lemma 5.3.2. Then Lemma 5.3.1 gives the desired bound on the value of the objective function:

$$\sum_{u,v:\, u\preceq v} \frac{\hat{y}_{uv}}{(4\pi)^d} > m^d \left(\frac{\ln m - 1}{4\pi}\right)^d.$$

This concludes the proof of Theorem 5.1.1. □

5.4 Our Lower Bound for k-TC-spanners of d-dimensional Posets for $k > 2$

In this section, we prove Theorem 5.1.3 that gives a lower bound on the size of Steiner k-TC-spanners of d-dimensional posets for $k > 2$ and $d \ge 2$.

Proof of Theorem 5.1.3. Unlike in the previous section, the poset which attains the lower bound is constructed probabilistically, not explicitly.

We consider n-element posets G embedded in the hypergrid $\mathcal{H}_{n,d}$, where the partial order is given by the dominance order $x \preceq y$ on $\mathcal{H}_{n,d}$. The elements of G are points $p_1, p_2, \ldots, p_n \in [n]^d$, where the first coordinate of each p_a is a. (By Fact 5.2.1, each d-dimensional poset with n elements can be embedded into $\mathcal{H}_{n,d}$, so that the first coordinates of all points are distinct.) Let $\mathcal{G}_d$ be a distribution on such posets G, where the last $d-1$ coordinates of each point p_a are chosen uniformly and independently from $[n]$.

Recall that $S_k(G)$ denotes the size of the sparsest Steiner k-TC-spanner of poset G. The following lemma gives a lower bound on the expected size of a Steiner k-TC-spanner of a poset drawn from $\mathcal{G}_d$.

Lemma 5.4.1. $\mathop{\mathbb{E}}_{G \leftarrow \mathcal{G}_d}[S_k(G)] = \Omega(n \log^{\lceil \frac{d-1}{k} \rceil} n)$ *for all* $k \geq 3$ *and constant* $d \geq 2$*.*

To simplify the presentation, we first prove the special case of Lemma 5.4.1 for 2-dimensional posets in Section 5.4.1. The general case is proved in Section 5.4.2. Since Lemma 5.4.1 implies the existence of a poset G, for which every Steiner k-TC-spanner has $\Omega(n \log^{\lceil (d-1)/k \rceil} n)$ edges, Theorem 5.1.3 follows. $\square$

5.4.1 The Case of $d = 2$

This section proves a special case of Lemma 5.4.1 for 2-dimensional posets, which illustrates many of the ideas used in the proof of the general lemma. In both proofs, we assume that $\ell = \log n$ is an integer.

Lemma 5.4.2 (Special case of Lemma 5.4.1)**.** $\mathop{\mathbb{E}}_{G \leftarrow \mathcal{G}_2}[S_k(G)] = \Omega(n \log n)$ *for all* $k \geq 3$ *and* $d = 2$*.*

Proof. To analyze the expected number of edges in a Steiner TC-spanner H of G, we consider ℓ partitions of $[n]^2$ into horizontal strips. We call strips *boxes* for compatibility with the case of general d.

Definition 5.4.1 (Box partition)**.** *For each* $i \in [\ell]$*, define sets of equal size that partition* $[n]$ *into* 2^i *intervals: the* j*th such set, for* $j \in [2^i]$*, is* $I^i_j = [(j-1)2^{\ell-i}+1, j2^{\ell-i}]$*. Given* $i \in [\ell]$*, and* $j \in [2^i]$*, the* box $\mathbb{B}(i,j)$ *is* $[n] \times I^i_j$ *and the* box partition $\mathbb{BP}(i)$ *is a partition of* $[n]^2$ *that contains boxes* $\mathbb{B}(i,j)$ *for all* $j \in [2^i]$*.*

For each odd j, we group boxes $\mathbb{B}(i,j)$ and $\mathbb{B}(i,j+1)$ into a *box-pair*. We call j the *index* of the box-pair and refer to $\mathbb{B}(i,j)$ and $\mathbb{B}(i,j+1)$ as the *bottom* and the *top* box in the box-pair. Recall that a poset G consists of elements $p_1, p_2, \ldots, p_n \in [n]^2$, where the first coordinate of each p_a is a. We analyze the expected number of edges in a Steiner TC-spanner H of G that cross from bottom to top boxes in all box-pairs. To do that, we identify pairs of poset elements (p_a, p_b), called *jumps*, that force such edges to appear. By Lemma 5.2.3, we can assume that all Steiner vertices of H are embedded into $\mathcal{H}_{n,2}$. Therefore, if p_a is in the bottom box and p_b is in the top box of the same box-pair then H must contain an edge from the bottom to the top box. To ensure that we count such an edge just once, we consider only p_a and p_b for which no other point p_c with $c \in (a,b)$ is contained in this box pair. Next we define *jumps* formally. This concept is also illustrated in Figure 5.1.

Definition 5.4.2 (Jumps)**.** *Given a poset* G*, embedded into* $\mathcal{H}_{n,2}$*, and an index* $i \in [\ell]$*, a* jump generated by the box partition $\mathbb{BP}(i)$ *is a pair* (p_a, p_b) *of elements of* G*, such that for some odd* $j \in [2^i]$*, the following holds:* $p_a \in \mathbb{B}(i,j)$*,* $p_b \in \mathbb{B}(i,j+1)$*, but* $p_c \notin \mathbb{B}(i,j) \cup \mathbb{B}(i,j+1)$ *for all* $c \in (a,b)$*. The set of jumps generated by all partitions* $\mathbb{BP}(i)$ *for* $i \in [\ell]$ *is denoted by* $\mathcal{J}$*.*

Next we establish that the number of jumps in a poset G is a lower bound on the number of edges in a Steiner TC-spanner of G (Proposition 5.4.3) and bound the expected number of jumps from below (Proposition 5.4.4).

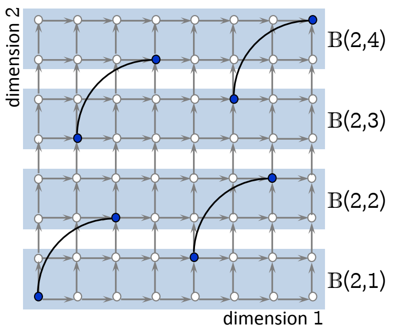


Figure 5.1: Box partition $\mathbb{BP}(2)$ and jumps it generates.

Proposition 5.4.3. *Let G be a poset, embedded into $\mathcal{H}_{n,2}$, and $H = (V_H, E_H)$ be a Steiner k-TC-spanner of G. Then $|E_H| \geq |\mathcal{J}|$.*

Proof. To prove the statement, we establish an injective mapping from $\mathcal{J}$ to E_H. By Lemma 5.2.3, we can assume that all Steiner vertices of H are embedded into $\mathcal{H}_{n,2}$. Given a jump (p_a, p_b), let j be the index of the box-pair containing p_a in the bottom box and p_b in the top box. We define $e(a,b) \in E_H$ by following an arbitrary path from p_a to p_b in H. This path is contained in the box-pair $\mathbb{B}(i,j) \cup \mathbb{B}(i,j+1)$. We define $e(a,b)$ as the edge on that path that starts in $\mathbb{B}(i,j)$ and ends in $\mathbb{B}(i,j+1)$.

It remains to show that the mapping $e(a,b)$ is injective. Consider an edge (u,v) of H with $u = (u_1, u_2)$ and $v = (v_1, v_2)$. Observe that there is a unique box-pair $\mathbb{B}(i,j) \cup \mathbb{B}(i,j+1)$ such that $v \in \mathbb{B}(i,j)$ and $u \in \mathbb{B}(i,j+1)$. (Indices i and j can be determined by finding the number of the form $j2^{\ell-i}$ in the interval $[u_2, v_2 - 1]$, such that $\ell - i$ is maximized.) At most one jump (a,b) satisfies $p_a \in \mathbb{B}(i,j)$, $p_b \in \mathbb{B}(i,j+1)$ and $a \leq u_1 \leq v_1 \leq b$, since the intervals $[a,b]$ are disjoint for all jumps in a box pair. □

Proposition 5.4.4. *When a poset G is drawn from the distribution $\mathcal{G}_2$, the expected size of $\mathcal{J}$ is at least $n(\ell-1)/4$.*

Proof. We first find the expected number of jumps generated by the partition $\mathbb{BP}(i)$ for a specific i. Let $\lambda_i(p_a)$ be the index j of the box-pair $\mathbb{B}(i,j) \cup \mathbb{B}(i,j+1)$ that contains p_a. This is well defined since box-pairs with respect to $\mathbb{BP}(i)$ partition $[n]^2$. Let $\rho_i(p_a)$ be 0 if p_a is in the bottom box of that box pair, and 1 otherwise. One can think of $\lambda_i(p_a)$ as the location of p_a, and of $\rho_i(p_a)$ as its relative position within a box-pair. Importantly, when G is drawn from $\mathcal{G}_2$, that is, the second coordinates of points p_a for all $a \in [n]$ are chosen uniformly and independently from $[n]$, then random variables $\rho_i(p_a)$ are independent and uniform over $\{0,1\}$ for all $a \in [n]$.

We group together points p_a that have equal values of $\lambda_i(p_a)$, and sort points within groups in increasing order of their first coordinate a. Since there are 2^{i-1} box-pairs, the number of groups is at most 2^{i-1}. Observe that random variables $\rho_i(p_a)$ within each group are uniform and independent because random variables $\lambda_i(p_a)$ and $\rho_i(p_a)$ are independent for all $a \in [n]$. Now, if we list $\rho_i(p_a)$ in the sorted order for all points in a particular group, we get a sequence of 0s and 1s. Two consecutive entries correspond to a jump iff they are 01. The last position in a group cannot correspond to the beginning of a jump. The number of positions that can correspond to the beginning of a jump in all groups is n minus the number of groups, which gives at least $n - 2^{i-1}$. For each such position, the probability that it starts a jump (i.e., the probability of 01) is 1/4. Thus, the expected number of jumps generated by the partition $\mathbb{BP}(i)$ is at least $(n - 2^{i-1})/4$.

Summing over all $i \in [\ell]$, we get the expected number of jumps in all partitions: $(n\ell - \sum_{i=1}^{\ell} 2^{i-1})/4 > n(\ell-1)/4 = \Omega(n \log n)$. □

Propositions 5.4.3 and 5.4.4 imply that, for a poset G drawn from $\mathcal{G}_2$, the expected number of edges in a Steiner TC-spanner of G is $\Omega(n \log n)$, concluding the proof of Lemma 5.4.2. □

5.4.2 The Case of Constant d

This section proves Lemma 5.4.1, a generalization of Lemma 5.4.2 and the main building block in the proof of Theorem 5.1.3.

Proof of Lemma 5.4.1. Generalizing the proof for $d = 2$ to arbitrary constant d, we consider ℓ^{d-1} partitions of $[n]^d$ into boxes, where $\ell = \log n$. As before, we assume ℓ is an integer. In this proof, let $\ell' = \lfloor \ell/(d-1) \rfloor$ and $d' = \lceil (d-1)/k \rceil$.

Definition 5.4.3 (Box partition). *Given vectors $\vec{\imath} = (i_1, \ldots, i_{d-1}) \in [\ell']^{d-1}$ and $\vec{\jmath} = (j_1, \ldots, j_{d-1}) \in [2^{i_1}] \times \cdots \times [2^{i_{d-1}}]$, the* box $\mathbb{B}(\vec{\imath}, \vec{\jmath})$ *is $[n] \times I_{j_1}^{i_1} \times \ldots \times I_{j_{d-1}}^{i_{d-1}}$, and the* box partition $\mathbb{BP}(\vec{\imath})$ *is a partition of $[n]^d$ that contains boxes $\mathbb{B}(\vec{\imath}, \vec{\jmath})$ for all eligible $\vec{\jmath}$.*

Next we generalize the definition of the set of jumps $\mathcal{J}$. We denote $(d-1)$-dimensional vectors $(0, \ldots, 0)$ and $(1, \ldots, 1)$ by $\vec{0}$ and $\vec{1}$, respectively. We say that a vector $\vec{\jmath}$ is *odd* if all of its coordinates are odd. Now we form box-pairs from boxes $\mathbb{B}(\vec{\imath}, \vec{\jmath})$ and $\mathbb{B}(\vec{\imath}, \vec{\jmath} + \vec{1})$ for odd vectors $\vec{\jmath}$. Analogously to the 2-dimensional case, we call $\mathbb{B}(\vec{\imath}, \vec{\jmath})$ the *bottom box* and $\mathbb{B}(\vec{\imath}, \vec{\jmath} + \vec{1})$ the *top box* in a box-pair.

Definition 5.4.4 (Jumps). *Given a poset G, embedded into $\mathcal{H}_{n,d}$, and an index vector $\vec{\imath} \in [\ell']^{d-1}$, a* jump generated by the box partition $\mathbb{BP}(\vec{\imath})$ *is a pair (p_a, p_b) of elements of G, such that for some odd vector $\vec{\jmath}$, the following holds: $p_a \in \mathbb{B}(\vec{\imath}, \vec{\jmath})$, $p_b \in \mathbb{B}(\vec{\imath}, \vec{\jmath} + \vec{1})$, but $p_c \notin \mathbb{B}(\vec{\imath}, \vec{\jmath}) \cup \mathbb{B}(\vec{\imath}, \vec{\jmath} + \vec{1})$ for all $c \in (a, b)$. The set of jumps generated by all partitions $\mathbb{BP}(\vec{\imath})$ for $\vec{\imath} \in [\ell']^{d-1}$ is denoted by $\mathcal{J}$.*

Next we generalize the definitions of location $\lambda_i(p_a)$ and relative position $\rho_i(p_a)$. Unlike in the 2-dimensional case, now some boxes (and, consequently, some points) do not belong to box-pairs. For each odd $\vec{\jmath}$, we group boxes $\mathbb{B}(\vec{\imath}, \vec{\jmath} + \vec{\alpha})$ for all $\vec{\alpha} \in \{0,1\}^{d-1}$ into a *megabox.* We call $\vec{\jmath}$ the *index vector* of the megabox, and refer to α as the *relative position* of a box in the megabox. Observe that megaboxes with respect to $\mathbb{BP}(\vec{\imath})$ partition $[n]^d$. Given $\vec{\imath}$, let $\lambda_{\vec{\imath}}(p_a)$ be the index vector $\vec{\jmath}$ of the megabox of p_a with respect to $\mathbb{BP}(\vec{\imath})$, and let $\rho_{\vec{\imath}}(p_a)$ be the relative position vector $\vec{\alpha}$ of the box of p_a in the megabox. In other words, to obtain $\lambda_{\vec{\imath}}(p_a)$, we take the index $\vec{\jmath}$ of the box $\mathbb{B}(\vec{\imath}, \vec{\jmath})$ containing p_a, and round its coordinates down to the nearest odd numbers. Then $\rho_{\vec{\imath}}(p_a) = \vec{\jmath} - \lambda_{\vec{\imath}}(p_a)$, where $\vec{\jmath}$ is the index of the box $\mathbb{B}(\vec{\imath}, \vec{\jmath})$ containing p_a.

Proposition 5.4.5. *Let G be a poset, embedded into $\mathcal{H}_{n,d}$, and $H = (V_H, E_H)$ be a Steiner k-TC-spanner of G. Then $|E_H| = \Omega(|\mathcal{J}|/\ell^{d-1-d'})$.*

Proof. To prove the statement, we establish a mapping from $\mathcal{J}$ to E_H that takes $O(\ell^{d-1-d'})$ jumps to one edge. By Lemma 5.2.3, we can assume that all Steiner vertices of H are embedded into $\mathcal{H}_{n,d}$.

First, we describe how to map a jump (p_a, p_b) to an edge $e(a, b) \in E_H$. Each such jump is generated by a box partition $\mathbb{BP}(\vec{\imath})$ for some $\vec{\imath}$. We follow an arbitrary path of length at most k in H from p_a to p_b, say, $(p_a = u_0, \ldots, u_k = p_b)$, and let $e(a, b)$ be an edge (u_{c-1}, u_c) with the maximum (over all $c \in [k]$) Hamming distance between $\rho_{\vec{\imath}}(u_{c-1})$ and $\rho_{\vec{\imath}}(u_c)$. Note that the maximum distance is at least d' because $\rho_{\vec{\imath}}(u_0) = \vec{0}$ and $\rho_{\vec{\imath}}(u_k) = \vec{1}$. That is, for $(u, v) = e(a, b)$, the difference $\rho_{\vec{\imath}}(v) - \rho_{\vec{\imath}}(u)$ is a vector in $\{0, 1\}^{d-1}$ with at least d' ones. In addition, the edge $e(a, b)$ belongs to the megabox of p_a and p_b.

Now we count the jumps (p_a, p_b) mapped to a specific edge (u, v). First, we find all such jumps generated by a single box partition $\mathbb{BP}(\vec{\imath})$. Observe that, for such a jump, p_a and p_b belong to the same megabox as u and v, i.e., $\lambda_{\vec{\imath}}(u)$. Moreover, interval $[a, b]$ contains $[u_1, v_1]$. Since intervals $[a, b]$ are disjoint for all jumps in a megabox, this uniquely determines $[a, b]$. Hence, there is at most one such jump.

It remains to count box partitions $\mathbb{BP}(\vec{\imath})$ which can generate a jump mapped to a specific edge (u, v). Recall that $\rho_{\vec{\imath}}(v) - \rho_{\vec{\imath}}(u)$ must be a vector in $\{0, 1\}^{d-1}$ with at least d' ones. There are less than 2^{d-1} such vectors. Consider one of these vectors, say, $\vec{\gamma}$. If for some $t \in [d - 1]$, $\gamma_t = 1$ then i_t is uniquely determined by the largest power of 2 that divides a number in $[u_t, v_t - 1]$. When $\gamma_t = 0$, there are at most ℓ' possible values of i_t because $\vec{\imath} \in [\ell']^{d-1}$. Since d is a constant, there are at most $2^{d-1}(\ell')^{d-1-d'} = O(\ell^{d-1-d'})$ possible vectors $\vec{\imath}$, such that $\mathbb{BP}(\vec{\imath})$ could have generated a jump (p_a, p_b).

Therefore, $O(\ell^{d-1-d'})$ jumps map to the same edge of E_H and, consequently, $|E_H| = \Omega(|\mathcal{J}|/\ell^{d-1-d'})$. □

Proposition 5.4.6. *When a poset G is drawn from the distribution $\mathcal{G}_d$, the expected size of $\mathcal{J}$ is $\Omega(\ell^{d-1}n)$.*

Proof. We first analyze the expected number of jumps generated by the partition $\mathbb{BP}(i)$ for a specific i. Under the distribution $\mathcal{G}_d$, the values $\rho_{\vec{\imath}}(p_a)$ are independent and uniform over $\{0, 1\}^{d-1}$ for all $a \in [n]$. Let P be the set of elements p_a in all the box-pairs, i.e., elements with $\rho_{\vec{\imath}}(p_a)$ equal to $\vec{0}$ and $\vec{1}$. The expected size of P is $n/2^{d-2}$. We group together elements p_a of P that have equal values of $\lambda_{\vec{\imath}}(p_a)$, and sort elements within groups in increasing order of their first coordinate a.

Observe that random variables $\rho_{\vec{\imath}}(p_a)$ within each group are uniform and independent because random variables $\lambda_{\vec{\imath}}(p_a)$ and $\rho_{\vec{\imath}}(p_a)$ are independent for all $a \in [n]$. Now, if we list $\rho_{\vec{\imath}}(p_a)$ in the sorted order for all elements in a particular group, we get a sequence of $\vec{0}$s and $\vec{1}$s. Two consecutive entries correspond to a jump iff they are $\vec{0}\,\vec{1}$. The last position in a group cannot correspond to the beginning of a jump. The expected number of positions that can correspond to the beginning of a jump in all groups is $n/2^{d-2}$ minus the expected number of groups. Let $m(\vec{\imath})$ denote the number of megaboxes with respect to a box partition $\mathbb{BP}(\vec{\imath})$. The number of groups is at most $m(\vec{\imath})$. On every position in the reordered sequence that is not the final position in its group, the expected number of jumps started is 1/4, so the expected number of jumps is at least $(n/2^{d-2} - m(\vec{\imath}))/4 = n/2^d - m(\vec{\imath})/4$.

The number of megaboxes in all box partitions is

$$\sum_{\vec{\imath} \in [\ell']^{d-1}} m(\vec{\imath}) = \sum_{\vec{\imath} \in [\ell']^{d-1}} \prod_{t=1}^{d-1} 2^{i_t - 1} = \left(\sum_{i_1=1}^{\ell'} 2^{i_1 - 1} \right)^{d-1} < 2^{\ell'(d-1)} \leq 2^{\ell} = n.$$

Therefore, the expected number of jumps generated by all box partitions is at least

$$(\ell')^{d-1}n/2^d - \frac{1}{4}\sum_{\vec{\imath}\in[\ell']^{d-1}} m(\vec{\imath}) \geq (\ell')^{d-1}n/2^d - n/4 = \Omega(\ell^{d-1}n).$$

The last equality holds because d is constant. □

By linearity of expectation, Propositions 5.4.5 and 5.4.6 imply that the expected number of edges in a Steiner TC-spanner H of G under the distribution $\mathcal{G}_d$ is

$$\Omega\big((\mathop{\mathbb{E}}_{G\leftarrow\mathcal{G}_d}|\mathcal{J}|)/\ell^{d-1-d'}\big) = \Omega(\ell^{d-1}n/\ell^{d-1-d'}) = \Omega(n\ell^{d'}) = \Omega(n\log^{\lceil(d-1)/k\rceil} n).$$

This concludes the proof of Lemma 5.4.1. □

Part III

Communication Complexity Methods in Summarization

Chapter 6

Lower bounds for Testing of Functions on Hypergrids

6.1 Introduction

We consider the problem of testing properties of functions over the hypergrid[1]: given oracle access to a function $f : [n]^d \to R$, for some (finite or infinite) set $R \subseteq \mathbb{R}$, and given a property $\mathcal{P}$ of functions mapping $[n]^d$ to R, what is the minimum number of queries to f that a randomized algorithm must make to distinguish with high probability between the case where f has the property $\mathcal{P}$ from the case where f is far[2] from having the same property? We focus on *nonadaptive* tests—algorithms that must fix all their queries before observing the value of the function on any of the queried inputs.

The problem of testing properties of functions has been studied extensively (see, for example, the surveys [176, 177] and the book [104]), but most of this research has been restricted to functions $f : [n] \to R$ on the line and to functions $f : \{0,1\}^d \to R$ on the hypercube. (These classes of functions correspond to the special cases of the hypergrid where $d = 1$ and $n = 2$, respectively.) The purpose of the current research is to generalize tools developed in these more specialized settings to improve our understanding of property testing of functions with general hypergrid domains. In particular, we show for the first time how the connection with communication complexity established in [37] can be applied to obtain lower bounds on functions with non-hypercube domains. We then use this method to obtain significantly stronger, and in many cases optimal, lower bounds on the number of queries required to nonadaptively test three of the most fundamental properties of functions on the hypergrid: monotonicity, convexity, and the Lipschitz property.

Monotonicity. The function $f : [n]^d \to R$ is *monotone* if for any two inputs $(x_1, \ldots, x_n), (y_1, \ldots, y_n) \in [n]^d$ that satisfy $x_1 \le y_1, \ldots, x_n \le y_n$, the function f satisfies $f(x_1, \ldots, x_n) \le f(y_1, \ldots, y_n)$.

[1]We use $[n]$ to denote the set $\{1, 2, \ldots, n\}$.

[2]See Section 6.2 for the formal definitions. For the purposes of this introduction, we say that f is far from having the property $\mathcal{P}$ if we need to modify the value of f on a constant fraction of its inputs to turn it into a function that does satisfy $\mathcal{P}$.

Functions on the hypergrid

	Our lower bounds	Previous lower bounds	Upper bounds
Monotonicity	$\Omega(d \log n)$	$\Omega(d)$ (adaptive, $n = 2$) [37]	$O(d \log n)$ [49]
Convexity	$\Omega(d \log n)$	—	—
Lipschitz	$\Omega(d \log n)$	$\Omega(d)$ (adaptive, $n = 2$) [127]	$O(d \log n)$ [49]

Functions on the line

	Our lower bounds	Previous lower bounds	Upper bounds
Monotonicity	$\Omega(\min\{\log r, \log n\})$	$\Omega(\min\{\log r, \log n\})$ (1.-s. err.) [87] $\Omega(\log n)$ (adaptive, $r \gg n$) [87, 94]	$O(\log n)$ [87]
Convexity	$\Omega(\log n)$ ($r = \Omega(n^2)$)	—	$O(\log n)$ [163]
Lipschitz	$\Omega(\min\{\log r, \log n\})$	$\Omega(\min\{\log r, \log n\})$ (1-s. err.) [127]	$O(\log n)$ [127]

Table 6.1: Query complexity bounds for testing properties of the function $f : [n]^d \to \mathbb{Z}$ (top) and of the function $f : [n] \to [r]$ (bottom). All the bounds are for nonadaptive tests with two-sided error unless marked otherwise.

Monotonicity testing is a classic problem in property testing that has been studied extensively for functions on the line [87, 94], on the hypercube [105, 75, 95, 37, 49, 48], on general partially ordered set domains [95], and on hypergrid domains as well: Dodis *et al.* [75] showed that we can test whether $f : [n]^d \to [r]$ is monotone with $O(d \log n \log r)$ queries. Ailon and Chazelle [4] gave an alternative algorithm with the incomparable query complexity $O(d2^d \log n)$. Very recently, Chakrabarty and Seshadhri [49] improved on both these results by showing that $O(d \log n)$ queries are sufficient for the task.

Prior to this work, however, the only known query complexity lower bounds for the problem of testing whether the function $f : [n]^d \to \mathbb{Z}$ is monotone were for two special cases: When $n = 2$, (i.e., for the hypercube), we know that $\Omega(d)$ queries are required to test monotonicity [37] and that this bound is optimal [48]. And when $d = 1$ and r is large enough, we know that $\Theta(\log n)$ queries are both necessary and sufficient for testing monotonicity [87, 94].

We give the first lower bound for testing monotonicity of functions on general hypergrid domains. Furthermore, the bound that we obtain is optimal for nonadaptive tests, since it matches the upper bound of Chakrabarty and Seshadhri [49].

Theorem 6.1.1. *Fix $\epsilon \in (0, \frac{1}{8}]$; $m, r \in \mathbb{N}$. Let $n = 2^m$. Any nonadaptive ϵ-test for monotonicity of functions $f : [n]^d \to [nd]$ must make $\Omega(d \log n)$ queries.*

The special case $d = 1$ of the theorem also gives the first nontrivial lower bound on the query complexity of two-sided error monotonicity tests for functions $f : [n] \to [r]$ on the line when r is subexponential in n.[3]

[3]Strictly speaking, our result gives the first lower bound in the two-sided error model for *any* finite r, but the Ramsey theory arguments of Fischer [94] can be extended to finite ranges when r is large enough.

Convexity. The function $f : [n]^d \to R$ is *convex* if for all $x, y \in [n]^d$ and all $\rho \in [0, 1]$ such that $\rho x + (1-\rho)y \in [n]^d$, the function f satisfies $f(\rho x + (1-\rho)y) \le \rho f(x) + (1-\rho)f(y)$.

Convexity testing is another classic problem in property testing. This problem was first studied by Parnas, Ron, and Rubinfeld [163], who showed that we can test if $f : [n] \to \mathbb{R}$ is convex with $O(\log n)$ queries. In the same paper, they proposed two open problems: to understand the testing of closely-related properties, and to examine the problem of testing convexity in the setting of functions $f : [n]^d \to \mathbb{R}$ on the hypergrid. While there has been much work on the first open problem—including results on testing submodularity [185, 171], convexity of images [169], and convexity of geometric sets in $\mathbb{R}^d$ [168]—our lower bound represents the first progress on the study of testing convexity on the hypergrid.

Theorem 6.1.2. *Fix $\epsilon \in (0, \frac{1}{8}]$; $m, r \in \mathbb{N}$. Let $n = 2^m$. Any nonadaptive ϵ-test for convexity of functions $f : [n]^d \to \mathbb{R}$ must make $\Omega(d \log n)$ queries.*

We also prove a matching lower bound for *separate convexity*, a closely-related (but weaker) property of functions on hypergrids (see Definition 6.4.5). Our results also hold for testing concavity.

The special case of our lower bound for $d = 1$ gives the first lower bound for testing convexity on the line.[4] This lower bound matches the query complexity of the nonadaptive test of Parnas, Ron, and Rubinfeld [163], showing that their algorithm and our lower bound are both optimal.

Lipschitz property.

The function $f : [n]^d \to R$ is *Lipschitz* if for any two inputs $(x_1, \ldots, x_n), (y_1, \ldots, y_n) \in [n]^d$, the function f satisfies $|f(x_1, \ldots, x_n) - f(y_1, \ldots, y_n)| \le \sum_{i=1}^n |x_i - y_i|$.

The problem of testing the Lipschitz property on functions with hypergrid domains has applications to data privacy and program checking [127, 74]. Notably, Dixit *et al.* [74] have used Lipschitz testers to construct privacy testers. Motivated by these applications, Jha and Raskhodnikova [127] initiated the study of testing whether functions are Lipschitz. They showed that testing if a function $f : \{0, 1\}^d \to [r]$ is Lipschitz can be done with $O(\min\{d^2, dr\})$ queries, that testing $f : [n] \to [r]$ for the same property can be done with $O(\log \min\{n, r\})$ queries, and that the latter bound is optimal for nonadaptive tests with one-sided error. Awasthi *et al.* [14] gave the first algorithms for testing the Lipschitz property for functions $f : [n]^d \to [r]$ on the hypergrid, showing that $O(\min\{d^{3/2} n \log n, dr \log r, dr \log n\})$ queries suffice for the task. Finally, Chakrabarty and Seshadhri [49] improved this bound for arbitrary ranges by showing that $O(d \log n)$ queries suffice for testing whether $f : [n]^d \to \mathbb{R}$ is Lipschitz.

We give the first lower bound for testing the Lipschitz property for functions with hypergrid domains.

Theorem 6.1.3. *Fix $\epsilon \in (0, \frac{1}{8}]$; $m, r \in \mathbb{N}$. Let $n = 2^m$. Any nonadaptive ϵ-test for the Lipschitz property of functions $f : [n]^d \to [R]$, where $R = \Omega(dn)$ must make $\Omega(d \log n)$ queries.*

[4]One might ask whether the lower bound for testing monotonicity on the line immediately implies a matching lower bound for testing convexity. It does not, and while it is certainly possible that a direct argument showing this implication exists, this argument would certainly not be trivial (c.f. [185]).

The lower bound in the theorem is optimal: it shows that no nonadaptive test can improve on the query complexity of Chakrabarty and Seshadhri's test in the hypergrid setting. The special case of the theorem when $d = 1$ is also optimal; showing that the algorithm of Jha and Raskhodnikova for the line is optimal, even if we allow two-sided error.

Our techniques.

We obtain our lower bounds by exploiting the connection between property testing and communication complexity discovered in [37]. This connection, which we describe in Section 6.2, gives a method for establishing reductions between property testing problems and (a special class of) communication problems. These reductions then let us build on the rich body of work in communication complexity to prove lower bounds in property testing. This approach has been particularly successful in establishing lower bounds for testing properties of functions over the hypercube [37, 45, 127], but until the present work it had yet to be applied to functions over other domains.

One important reason why the previous lower bounds were restricted to properties of functions over the hypercube is that a key ingredient in all these proofs is the use of parity functions—the set of functions $\chi_S : \{0,1\}^d \to \{0,1\}$, $S \subseteq [d]$, defined by $\chi_S(x) = \sum_{i \in S} x_i \pmod 2$. These functions form an orthonormal basis for the set of functions mapping $\{0,1\}^n$ to $\mathbb{R}$, a fact that is exploited in the reductions.

We prove our lower bounds for functions with hypergrid domains by replacing the use of parity functions with *Walsh functions*, a set of functions that forms an orthonormal basis of the functions mapping $[n]^d$ to $\mathbb{R}$. These functions offer different challenges in the construction of reductions and their analysis, but the resulting lower bounds are as clean and natural as the ones obtained for functions on the hypercube.

Remarks on adaptivity.

All the lower bounds introduced in this paper are for nonadaptive tests—that is, tests that must fix all their queries in advance, before observing the value of the function on any of the inputs that are queried. Interestingly, *all* the best known upper bounds on the query complexity of testing monotonicity, convexity, or the Lipschitz property (for functions over any domain) are achievable with nonadaptive tests and *nearly all* the lower bounds for testing these properties only apply to nonadaptive tests. In fact, apart from the result in [37], the only lower bound for monotonicity testing that applies to adaptive tests is obtained via an intricate argument of Fischer [94] that uses deep results in Ramsey theory to show that adaptivity cannot help in this setting.

In light of this general phenomenon, our results suggest two promising directions for future research: if one believes that the upper bounds can be improved in general, then our lower bounds imply that a completely new algorithmic approach which critically makes use of adaptivity will be required; conversely, if one believes that adaptivity does not help for these testing problems, then the proof of this statement will probably lead to a better understanding of the role of adaptivity in property testing and stronger connections between property testing and Ramsey theory or other areas of combinatorics.

Organization. The basic definitions and facts for property testing and communication complexity are introduced in Section 6.2. In Section 6.3, we prove our lower bounds for functions on the line; the more general lower bounds for functions with hypergrid domains are presented in Section 6.4.

6.2 Preliminaries

Property testing. The basic property testing definitions are as follows. For a more thorough introduction to the area, we recommend [176, 177].

Definition 6.2.1 (Relative distance to a property)**.** *Let $\mathcal{P}$ be a property (i.e., a set) of functions on a domain D, with range R and consider a function $f : D \to R$. The* relative distance *of f to the property is the minimum over all functions $g \in \mathcal{P}$ of the fraction of points in D on which f and g differ. We say f is ϵ-far from $\mathcal{P}$ if its relative distance from $\mathcal{P}$ is at least ϵ.*

Definition 6.2.2 (Property test [106, 178])**.** *Fix $\epsilon \in (0, 1)$. A* (two-sided *error,* adaptive) *ϵ-test for a property $\mathcal{P}$ is a randomized algorithm which, given oracle access to a function f, accepts with probability at least 2/3 if $f \in \mathcal{P}$, and rejects with probability at least 2/3 if f is ϵ-far from $\mathcal{P}$.*

A test has one-sided error *if it always accepts functions in $\mathcal{P}$. It is* nonadaptive *if the queries to f do not depend on the answers to the previous queries.*

Communication complexity.

In a (two-player) *communication game C*, Alice receives some input a, Bob receives some input b, and they must compute the value of some function $f_C(a, b)$ on their joint input. A *protocol* defines how Alice and Bob communicate. The maximum number of bits exchanged by Alice and Bob during the execution of a protocol over the possible inputs a and b is the *complexity* of the protocol. A randomized protocol is valid for f_C if for every input, the protocol computes f_C correctly with probability at least 2/3. The *communication complexity* of f_C is the minimum complexity of any protocol that is valid for f_C.

A number of different communication models have been extensively studied. We focus on the *one-way shared randomness* model. In this model, the only communication allowed is directed from Alice to Bob. Alice and Bob share access to a common source of randomness that can be used to determine the protocol. The communication complexity of f_C in the one-way shared randomness model is denoted $R^{A \to B}(f_C)$.

A fundamental function f_C studied in the one-way shared randomness model is Augmented Index. Alice's input to this function is a set $A \subseteq [t]$ while Bob's input is an index $i \in [t]$ and the set $B = A \cap [i-1]$. The output of Augmented Index is 1 if $i \in A$ and 0 otherwise. No randomized one-way communication protocol for this function does significantly better than the naïve protocol where Alice communicates her whole set to Bob.

Theorem 6.2.1 ([150])**.** *The one-way communication complexity of* Augmented Index *in the shared randomness model is $R^{A \to B}(\text{Augmented Index}) = \Theta(t)$.*

The connection between communication complexity and property testing is established via *combining operators*. An operator ψ that takes as input a and b, the inputs of Alice and Bob, respectively, and outputs a function $h(x) = \psi[a, b](x)$ is called a *one-bit one-way combining operator* if for all x in the domain of h, Bob can compute the value $h(x)$ with only one bit of communication from Alice. Next we summarize the conditions on the combining operator which are sufficient for a successful reduction from the AUGMENTED INDEX communication game to the problem of testing a given property.

Definition 6.2.3 (Reduction operator). *Let $t \in \mathbb{N}$ be a parameter. A combining operator $\psi[A, i, B]$ is called a* reduction operator *for (the* AUGMENTED INDEX *problem and) a property*[5] *$\mathcal{P}_t$ and a value $\epsilon_0 \in (0, 1)$ if it is a one-bit one-way combining operator and the function $h = \psi[A, i, B]$ satisfies the following two conditions for all valid inputs $A \subseteq [t]$, $i \in [t]$ and $B = A \cap [i - 1]$ to* AUGMENTED INDEX:

1. *If* AUGMENTED INDEX$(A, i, B) = 0$ *then $h \in \mathcal{P}_t$.*
2. *If* AUGMENTED INDEX$(A, i, B) = 1$ *then h is ϵ_0-far from $\mathcal{P}_t$.*

The following lemma is implicit in [37].

Lemma 6.2.2 (Reduction lemma). *If there exists a reduction operator for (the* AUGMENTED INDEX *problem and) a property $\mathcal{P}_t$ and a value $\epsilon_0 \in (0, 1)$ then for all $\epsilon \in (0, \epsilon_0]$, every nonadaptive ϵ-test for $\mathcal{P}_t$ requires $\Omega(t)$ queries.*

Proof. To prove the lemma, we reduce the AUGMENTED INDEX communication game to the problem of ϵ-testing property $\mathcal{P}_t$ and then apply Theorem 6.2.1.

Let $\psi[A, i, B]$ be a reduction operator. Consider a nonadaptive ϵ-test T for $\mathcal{P}_t$ that makes at most $q(t)$ queries for some $\epsilon \in (0, \epsilon_0]$. Then the following protocol for AUGMENTED INDEX uses $q(t)$ bits of communication from Alice to Bob. In this protocol, both players run the test T using shared randomness to find out the positions $x_1, \ldots, x_q$ queried by the test. Since T is nonadaptive, they can run it on any input of the right size. Then Alice sends to Bob $q \leq q(t)$ bits of information that Bob needs to compute $h(x_1), \ldots, h(x_q)$, where $h = \psi[A, i, B]$. One bit per query point is sufficient because ψ is a one-bit one-way combining operator. Bob answers the queries of T with $h(x_1), \ldots, h(x_q)$ and outputs 0 if T accepts and 1 otherwise. The correctness of the protocol for the cases when AUGMENTED INDEX(A, i, B) is 0 and 1, respectively, follows from the conditions 1 and 2 of Definition 6.2.3.

The reduction establishes that $R^{A \to B}(\text{AUGMENTED INDEX}) \leq q(t)$. Consequently, by Theorem 6.2.1, the query complexity of the test, $q(t)$, is $\Omega(t)$. □

6.3 Lower bounds on the line

We use two classes of functions on the domain $[2^m]$ (where[6] $m \in \mathbb{N}$) in the constructions that establish the lower bounds on the query complexity for testing properties of functions on the line: step functions and Walsh functions. Functions in both classes are constant on *blocks* of inputs in $[2^m]$, which we define next.

Definition 6.3.1 (Blocks). *Let $i \in \{0, \ldots, m\}$. For $k \in [2^{m-i}]$, the kth block of length 2^i is the set of integers $[2^i(k - 1) + 1, \ldots, 2^i k]$. We denote this block B^i_k.*

[5]In our reductions, the integer t is used to parameterize the size of the domain of functions under consideration. Specifically, for functions on the d-dimensional hypergrid domain $[n]^d$, we set $n = 2^{t/d}$.

[6]The parameter m is set to t in the application of the reduction lemma (Lemma 6.2.2) to testing functions on the line; in the case of d-dimensional hypergrids, it is set to t/d.

Observe that blocks of length 2^i partition $[2^m]$.

Definition 6.3.2 (Step functions). *For $i \in \{0, \dots, m\}$, the* step function *of block length 2^i is the function $s_i : [2^m] \to [2^{m-i}]$ defined by $s_i(x) = k$, such that $x \in B^i_k$. (Equivalently, $s_i(x) = \lfloor \frac{x-1}{2^i} \rfloor + 1$.)*

The step functions of block length 2^i are constant on each block B^i_k. Walsh functions indexed by i, which we define next, are equal to 1 on the first half of each block B^i_k and to -1 on the second half. In other words, whether they take the value 1 or -1 on input x is determined by the ith bit of the binary representation of $x-1$, denoted by $\mathsf{bit_i(x-1)}$, where the bits are numbered starting from the least significant.

Definition 6.3.3 (Walsh functions). *For $i \in [m]$, let $w_i : [2^m] \to \{-1, 1\}$ be the function defined by $w_i(x) = (-1)^{\mathsf{bit_i(x-1)}}$. For any $S \subseteq [m]$,* Walsh function *$w_S : [2^m] \to \{-1, 1\}$ corresponding to S is $w_S(x) = \prod_{i \in S} w_i(x)$. (If $S = \emptyset$ then $w_S(x) = 1$ for all x.)*

Also, we define $w_{m+1}(x) = 1$. (This is needed only in one of the proofs).

For two functions u, w we denote the the function $v(x) = u(x)w(x)$ by $u \times w$. We use $\| \cdot \|$ to denote the L_1-norm, that is, $\|u\| = \sum_x u(x)$, where the sum is over all x in the domain of u. With this notation, we can express the fundamental properties of Walsh functions that we will use in our proofs.

Proposition 6.3.1. *1. $\|w_S\| \geq 0$ for all sets $S \subseteq [m]$.*

2. For all sets $A, B \subseteq [m]$, Walsh function $w_{A \triangle B} : [2^m] \to \{-1, 1\}$ corresponding to the symmetric difference between A and B satisfies $w_{A \triangle B} = w_A \times w_B$.

6.3.1 Monotonicity

Theorem 6.3.2. *Fix $\epsilon \in (0, \frac{1}{4}]$; $m, r \in \mathbb{N}$. Let $n = 2^m$. Any nonadaptive ϵ-test for monotonicity of functions $f : [n] \to [r]$ requires $\Omega(\min(\log n, \log r))$ queries.*

Proof. To prove the lower bound of $\Omega(\log n)$ queries (for $n < r$), we apply the reduction lemma (Lemma 6.2.2) with the parameter t in the lemma set to m. To get the bound of $\Omega(\log r)$ queries (for $r \leq n$), we use the same proof with t set to $\lfloor \log_2(r-1) \rfloor$, except that the sets given to Alice and Bob reside in $\{m-t+1, \dots, m\}$ instead of $[m]$. Let ψ be the combining operator that receives Alice's set A, and Bob's index i and set B as input and returns the function $h : [2^m] \to \mathbb{Z}$ defined by

$$h(x) = 2s_i(x) + w_S(x),$$

where $S = A \triangle B = A \cap \{i, \dots, m\}$. The range of h is $[2 \cdot 2^{t-1} + 1] = [2^t + 1]$, i.e. it is equal to $[n+1]$ when $t = m$ and to $[r]$ when $t = \lfloor \log_2(r-1) \rfloor$.

By Proposition 6.3.1(2), $w_S = w_A \times w_B$. Bob knows B, so to determine $h(x)$ he only needs Alice to communicate a single bit—namely, the value $w_A(x)$. Thus, ψ is a one-bit one-way combining operator.

To prove that ψ is a reduction operator for monotonicity of functions on the line and $\epsilon_0 = 1/4$, it remains to show that it satisfies Items 1 and 2 of Definition 6.2.3. To demonstrate this, we prove the following lemma, which in addition to the required statements about h, also contains a statement about a related function h_- needed in Section 6.4.1.

Lemma 6.3.3. *Fix $i \in [m]$ and $S \subseteq \{i, \dots, m\}$. Consider the functions $h = 2s_i + w_S$ and $h_- = 2s_i - w_S$.*

1. *If $i \notin S$, then the functions h and h_- are monotone;*
2. *If $i \in S$, then the function h is $\frac{1}{4}$-far from monotone.*

Proof. Recall the definition of blocks (Definition 6.3.1). When $i \notin S$, i.e., $S \subseteq \{i + 1, \dots, m\}$, the functions s_i, w_S and $-w_S$ are constant on each block B^i_k (for $k \in [2^{m-i}]$). The value of of functions w_S and $-w_S$ can decrease (from 1 to -1) only between the blocks (i.e., if $w_S(x) > w_S(x+1)$ then $x \in B^i_k$ and $(x+1) \in B^i_{k+1}$ for some $k \in [2^{m-i} - 1]$). But the step function s_i increases by 1 on each subsequent block B^i_k. Thus, h and h_- are monotone (nondecreasing). This completes the proof of Item 1.

When $i \in S$, i.e., i is the smallest element in S, Walsh function w_S changes value in the middle of each block B^i_k. If this change is from 1 to -1, then w_S is 1/2-far from monotone on this block, and so is h because the step function s_i is constant on each B^i_k. Note that this change is from 1 to -1 for all blocks on which $w_{S\setminus\{i\}}$ evaluates to 1. By Proposition 6.3.1(1), $\|w_{S\setminus\{i\}}\| \geq 0$, so it happens for at least half of the blocks. Thus, h is $\frac{1}{4}$-far from monotone. □

By Lemma 6.3.3, the function h is monotone when $i \notin A$ and it is $\frac{1}{4}$-far from monotone when $i \in A$. That is, ψ is a reduction operator for monotonicity of functions of the form $f : [2^m] \to [t+1]$ and $\epsilon_0 = 1/4$. Then, by Lemma 6.2.2, any nonadaptive ϵ-test for this property, where $\epsilon \in (0, 1/4]$ requires $\Omega(t)$ queries. That is, when $r > n$, we get a bound of $\Omega(m) = \Omega(\log n)$, and when $r \leq n$, we get a bound of $\Omega(\log r)$. □

6.3.2 Convexity

Recall that the function $f : [n] \to \mathbb{R}$ is convex if for all $x, y \in [n]$ and all $\rho \in [0,1]$ such that $\rho x + (1-\rho)y$ is also an integer in $[n]$, the function f satisfies $f(\rho x + (1-\rho)y) \leq \rho f(x) + (1-\rho) f(y)$. Equivalently, we can define convexity in terms of the discrete derivative of functions on the line.

Definition 6.3.4 (Discrete derivative)**.** *The* discrete derivative *of the function $f : [n] \to \mathbb{R}$ is the function $f' : [n-1] \to \mathbb{R}$ defined by $f'(x) = f(x+1) - f(x)$.*

Definition 6.3.5 (Convexity, concavity)**.** *The function $f : [n] \to \mathbb{R}$ is* convex *(resp.,* concave*) if its derivative f' is a monotone nondecreasing (resp., nonincreasing) function.*

We present a lower bound for testing the convexity of functions on the line; the same bound also applies for testing concavity.

Theorem 6.3.4. *Fix $\epsilon \in (0, \frac{1}{8}]$ and $n = 2^m$ for some $m \geq 1$. Any nonadaptive ϵ-test for convexity of functions $f : [n] \to [r]$, where $r = \Omega(n^2)$, requires $\Omega(\log n)$ queries.*

Proof. We apply the reduction lemma (Lemma 6.2.2) with the parameter t in the lemma set to m. Our construction for the lower bound uses Walsh functions and *rising-step-size functions*, which are built from step functions (see Definition 6.3.2). The discrete derivatives of these new functions, which we call *double-step functions*, play a crucial role in our construction.

Definition 6.3.6 (Rising-step-size and double-step functions). *Fix $i \in [m]$. The* rising-step-size function $r_i : [n] \to [n^2]$ *is defined by $r_i(x) = s_i(x) + 2\sum_{y=1}^{x-1} s_i(y)$. Its discrete derivative, $r'_i(x) = s_i(x+1) + s_i(x)$, is called a* double-step function. *Equivalently (by Definitions 6.3.1 and 6.3.2), for all $k \in [2^{m-i}]$, function $r'_i(x)$ is equal to $2k$ on all but the last element x of the block B^i_k, and to $2k+1$ on the last element of B^i_k.*

Given Alice's set $A \subseteq [m]$ and Bob's index $i \in [m]$ and the prefix set $B = A \cap [i-1]$ the combining operator $\psi[A, i, B]$ returns the function

$$h(x) = r_i(x) + \frac{1}{2}(w_S(x) + 1),$$

where $S = A \triangle B = A \cap \{i, \dots, m\}$. Since $w_S = w_A \times w_B$, the operator ψ is a one-bit one-way combining operator. It remains to show that if $i \notin A$, then h is convex and that if $i \in A$, then h is 1/8-far from convex. We do so in the following lemma, which is also used in Section 6.4.3.

Lemma 6.3.5. *Fix $i \in [m]$ and $S \subseteq \{i, \dots, m\}$. The functions $h = r_i + \frac{1}{2}(w_S + 1)$ and $h_- = r_i - \frac{1}{2}(w_S + 1)$ satisfy the following properties.*

1. *If $i \notin S$, then h and h_- are convex;*
2. *If $i \in S$, then h is $\frac{1}{8}$-far from convex.*

Proof. First, consider the case where $i \notin S$. The discrete derivative of h is $h'(x) = r'_i(x) + \frac{1}{2}w'_S(x)$. It is sufficient to prove that h' is nondecreasing. Since $S \subseteq \{i+1, \dots, m\}$, the function w_S is constant on each block B^i_k (for $k \in [2^{m-i}]$). That is, for all but the last element x of a block B^i_k, the discrete derivative $w'(x) = 0$ and, consequently, $h'(x) = r'_i(x) = 2k$. Now consider $h'(x)$, where x is the last element of a block B^i_k. Recall that $r'_i(x) = 2k+1$. Since Walsh functions are ±1-valued, the value $\frac{1}{2}w'_S(x)$ is in $\{-1, 0, 1\}$. Thus, $h'(x) \in [2k, 2k+2]$, i.e., $h'(x-1) \le h'(x) \le h'(x+1)$. Therefore, h' is a nondecreasing function. The same argument shows that when $i \notin S$, the function h_- is also convex.

Now consider the case where $i \in S$. We start the analysis of this case by showing that for at least half of the blocks B^i_k, the derivative $w'_S(x) = -2$ on the 2^{i-1}th element of B^i_k (i.e., on the input $x = 2^i(k-1) + 2^{i-1}$.) Note that $w_S = w_i \times w_{S\setminus\{i\}}$. Proposition 6.3.1(1) gives that $\|w_{S\setminus\{i\}}\| \ge 0$, implying that $\Pr_x[w_{S\setminus\{i\}}(x) = 1] \ge 1/2$. Since $S \cap [i-1] = \emptyset$, the function $w_{S\setminus\{i\}}$ is constant within the blocks B^i_k. Thus, for at least half of these blocks it is a constant 1. For each block B^i_k, the function w_i is 1 on the first half of the block and -1 on the second half. Combining these observations, for half of the blocks B^i_k, the derivative of w_S on the middle point $x = 2^i(k-1) + 2^{i-1}$ of the block satisfies $w'_S(x) = w_S(x+1) - w_S(x) = w_{S\setminus\{i\}}(x+1) \cdot w_i(x+1) - w_{S\setminus\{i\}}(x) \cdot w_i(x) = -2$.

Let B^i_k be a block where $w'_S(x) = -2$ on the 2^{i-1}th element x of B^i_k. Note that $w'_S(x) = 0$ on all other inputs in the block apart from the last one because w_S is constant on all blocks B^{i-1}_j. Consider any three points $x, y, z \in B^i_k$ such that $x \le (k-1)2^i + 2^{i-1} < y < z$, namely, x is in the first half of the block B^i_k while y and z are in the second half. Then $h'(y) = h'(y+1) = \cdots = h'(z-1) = 2k$ so $(h(z) - h(y))/(z-y) = 2k$. However, $h'((k-1)2^i + 2^{i-1}) = 2k - 2$ so $(h(y) - h(x))/(y-x) < 2k$, which violates convexity. To fix convexity on all such triples, we must change the value of h on all the points $(k-1)2^i+1,\dots,(k-1)2^i+2^{i-1}$ in the first half of the block B^i_k, or on all but one point in the second half of B^i_k. Thus, we need to change at least 1/4 of the points in B^i_k. Since this is the case for at least half of all blocks, h is 1/8-far from convex. □

Lemma 6.3.5 completes the proof that ψ is a reduction operator for convexity and, thus, of the claimed lower bound for convexity. □

6.3.3 The Lipschitz property

Theorem 6.3.6. *Fix $\epsilon \in (0, \frac{1}{4}]$; $m, r \in \mathbb{N}$. Let $n = 2^m$. Any nonadaptive ϵ-test for the Lipschitz property of functions $f : [n] \to [r]$ requires $\Omega(\min(\log n, \log r))$ queries.*

Proof. To obtain the $\Omega(\log n)$ bound (for $r > n$), we apply the reduction lemma (Lemma 6.2.2) with the parameter t in the lemma set to m.

Definition 6.3.7 (Up-down staircase functions)**.** *For all $i \in \{0, 1, \dots, m\}$, let the* up-down staircase function *of block-length 2^i be the function $u_i : [2^m] \to [2^i]$, such that $u_i(1) = 1$ and the discrete derivative of u_i is*

$$u'_i(x) = \begin{cases} 0 & \textit{if } x \textit{ is divisible by } 2^i; \\ w_{i+1}(x) & \textit{otherwise.} \end{cases}$$

In other words (using Definition 6.3.1), function u_i takes values $1, \dots, 2^i$ on consecutive inputs from the block B^i_j if j is odd, and values $2^i, \dots, 1$ if j is even.

The combining operator ψ receives Alice's set A, and Bob's index i and set B as input and returns the function $h : [2^m] \to \mathbb{Z}$ defined by

$$h(x) = u_i(x) - \frac{1}{2}(w_S(x) + 1),$$

where $S = A \triangle B = A \cap \{i, \dots, m\}$. Since $w_S = w_A \times w_B$, the operator ψ is a one-bit one-way combining operator. It remains to show that if $i \notin A$, then $h(x)$ is Lipschitz and otherwise it is 1/4-far from Lipschitz. To demonstrate this, we prove a stronger lemma, which is also used in Section 6.4.3.

Lemma 6.3.7. *Fix $i \in [m]$ and $S \subseteq \{i, \dots, m\}$. Consider the functions $h(x) = u_i(x) - \frac{1}{2}(w_S(x) + 1)$ and $h_-(x) = u_i(x) - \frac{1}{2}(-w_S(x) + 1)$.*

1. *If $i \notin S$, then the functions h and h_- are Lipschitz;*

2. *If $i \in S$, then the function h is $\frac{1}{4}$-far from Lipschitz.*

Proof. If $i \notin S$, i.e., $S \subseteq \{i+1, \dots, m\}$, then the function w_S is constant on each block B^i_k (for $k \in [2^{m-i}]$). Let $w(x) = -\frac{1}{2}(w_S(x) + 1)$. Since Walsh functions are ± 1-valued, the discrete derivative $w'(x)$ is in $\{-1, 0, 1\}$ for all x, and $w'(x) = 0$ for all x *not* divisible by 2^i. By definition of the up-down staircase functions, $u'_i(x) \in \{-1, 0, 1\}$ for all x, and $u'_i(x) = 0$ for all x divisible by 2^i. Thus, $h' = u'_i + w'$ takes values only in $\{-1, 0, 1\}$, implying that h is Lipschitz. The proof that h_- is Lipschitz is analogous.

When $i \in S$, i.e., i is the smallest element in S, the rescaled Walsh function $w(x) = -\frac{1}{2}(w_S(x)+1)$ changes value in the middle of each block B_k^i. This change is either from -1 to 0 or vice versa. In the former case, the discrete derivative w' is 1 on the 2^{i-1}th element of the block, in the latter, it is -1. In both cases, it is 0 on all other elements of the block besides the last one. Next we show that if the former case occurs on a block with odd i (similarly, if the latter case occurs on a block with even i), then h is 1/2-far from Lipschitz on this block.

Consider the case when i is odd and w' is 1 on the 2^{i-1}th element of a block B_k^i. Since i is odd, u_i' takes value 1 on all but the last element of B_k^i. Then $h' = u_i' + w'$ is 2 on the 2^{i-1}th element of B_k^i, and 1 on all other elements of the block besides the last one. We pair up all elements of B_k^i as follows: each element x in the first half of the block is paired up with the element $x + 2^{i-1}$. The function h is not Lipschitz on each such pair: $h(x+2^{i-1}) - h(x) = \sum_{y=x}^{x+2^{i-1}-1} h'(y) = 2^{i-1}+1$. Thus, h is 1/2-far from Lipschitz on each such block. The other case (when i is even and w' is -1 on the 2^{i-1}th element of a block B_k^i) is analogous—the only difference is that h' takes negative values.

We can rephrase what we just proved as follows: the function h is 1/2-far from Lipschitz on all blocks B_k^i with $k \in [2^{m-i}]$, where $w_{S\setminus\{i\}}(x) = w_{i+1}(x)$ for all $x \in B_k^i$. Equivalently, $w_{S\setminus\{i\}}(x) \times w_{i+1}(x) = w_{(S\setminus\{i\})\triangle\{i+1\}}(x) = 1$ for all $x \in B_k^i$. By Proposition 6.3.1(1), $\|w_{(S\setminus\{i\})\triangle\{i+1\}}\| \geq 0$. Since $w_{(S\setminus\{i\})\triangle\{i+1\}}$ is constant on each block B_k^i, it is 1 on at least half of such blocks. Thus, h is 1/2-far from Lipschitz on at least half of the blocks B_k^i. That is, overall h is 1/4-far from Lipschitz. □

This completes the proof of the $\Omega(\log n)$ lower bound. To get the bound of $\Omega(\log \min\{n, r\})$, we use the same proof with t set to $\min\{m, \lfloor \log_2(r-1) \rfloor\}$. The range of h is $\{0, 1, \ldots, 2^t\}$, i.e., it has size $\min(n+1, r)$. □

6.4 Lower bounds on the hypergrid

In this section, we generalize the lower bounds for testing functions on the line to the hypergrid setting. To obtain our lower bounds for testing functions on the domain $[2^m]^d$, we give a reduction from the Augmented Index problem by applying the reduction lemma (Lemma 6.2.2) with the parameter t set to md. With this parameter setting, inputs to Augmented Index consist of subsets of $[md]$ and an index in $[md]$. We associate each such subset with a d-dimensional vector of subsets of $[m]$ and each such index with a d-dimensional vector of indices in $\{0, 1, \ldots, m\}$.

Definition 6.4.1 (Vector representation)**.** *Fix $m, d \in \mathbb{N}$. The d-dimensional vector corresponding to the set $S \in [md]$ is $\mathbf{S} = (\mathbf{S}_1, \ldots, \mathbf{S}_d)$, where $\mathbf{S}_j = \{\ell \in [m] : (j-1)m + \ell \in S\}$ for every $j \in [d]$. The d-dimensional vector corresponding to the index $i \in [md]$ is $\boldsymbol{i} = (\boldsymbol{i}_1, \ldots, \boldsymbol{i}_d)$, where $\boldsymbol{i}_j = \max\{0, \min\{m, i - (j-1)m\}\}$ for every $j \in [d]$.*

Equivalently, $\boldsymbol{i} = (m, \ldots, m, \boldsymbol{i}_{j^*}, 0, \ldots, 0)$, where $j^* = \lceil i/m \rceil$ and $\boldsymbol{i}_{j^*} = i - (j^*-1)m$. Observe that $i \in S$ iff $\boldsymbol{i}_{j^*} \in \mathbf{S}_{j^*}$. Recall that in the Augmented Index problem, Bob is given an element i, and he has to find out whether i is in Alice's set. Intuitively, in our reduction from Augmented Index to the problem of testing a property $\mathcal{P}$ (such as monotonicity) of d-dimensional functions, the function h returned by the combining operator will satisfy $\mathcal{P}$ on all axis-parallel lines in dimensions other than j^*. Most restrictions of h to lines in the special dimension j^* will behave as in the one-dimensional case: many of them will be far from $\mathcal{P}$ if i is in Alice's set; otherwise, all of them will be in $\mathcal{P}$. This suffices for the proofs for monotonicity, the Lipschitz property and for separate convexity (Definition 6.4.5), a property closely related (but not equivalent) to convexity. For convexity itself, it doesn't suffice to ensure that restrictions of the function on the axis-parallel lines are convex, so in this case if i is in Alice's set we construct the reduction in such a way that projections on all (not necessarily axis-parallel) lines are convex.

Next we extend the definitions of step functions and Walsh functions (namely, Definitions 6.3.2 and 6.3.3, respectively) to multiple dimensions.

Definition 6.4.2 (Componentwise sum)**.** *For a family of functions $f_i\colon [n] \to \mathbb{R}$, indexed by $i \in \{0, 1, \ldots, m\}$, and a vector $\boldsymbol{i} \in \{0, 1, \ldots, m\}^d$, the* componentwise sum $\hat{f}_{\boldsymbol{i}}\colon [n]^d \to \mathbb{R}$ *of f is defined by $\hat{f}_{\boldsymbol{i}}(x_1, \ldots, x_d) = \sum_{j=1}^d f_{\boldsymbol{i}_j}(x_j)$.*

Definition 6.4.3 (Step functions)**.** *The* step function *indexed by the d-dimensional vector $\boldsymbol{i} \in [m]^d$ is the componentwise sum $\hat{s}_{\boldsymbol{i}} : [2^m]^d \to [d2^m]$ defined by $\hat{s}_{\boldsymbol{i}}(x_1, \ldots, x_d) = \sum_{j=1}^d s_{\boldsymbol{i}_j}(x_j)$.*

Definition 6.4.4 (Walsh functions)**.** *The* Walsh function *indexed by the d-dimensional vector $\mathbf{S}$ of subsets of $[m]$ is the function $w_{\mathbf{S}} : [2^m]^d \to \{-1, 1\}$ defined by $w_{\mathbf{S}}(x_1, \ldots, x_d) = \prod_{j=1}^d w_{\mathbf{S}_j}(x_i)$.*

Next we extend Proposition 6.3.1 to the hypergrid setting.

Proposition 6.4.1. *1. $\|w_{\mathbf{S}}\| \geq 0$ for all d-dimensional vectors $\mathbf{S}$ of subsets of $[m]$.*

2. Fix $A, B \subset [md]$, and $S = A \triangle B$. Let $\mathbf{A}, \mathbf{B}, \mathbf{S}$ be the d-dimensional vector representations of sets A, B, S, respectively. Walsh function $w_{\mathbf{S}} : [2^m]^d \to \{-1, 1\}$ satisfies $w_{\mathbf{S}}(x) = w_{\mathbf{A}}(x) \cdot w_{\mathbf{B}}(x)$ for all $x \in [2^m]^d$.

Proof of Item 1. It is sufficient to prove that if the random variables $X_1, \ldots, X_d$ are i.i.d. and uniform over $[2^m]$ then $\Pr[w_{\mathbf{S}}(X_1, \ldots, X_d) = 1] \geq 1/2$. If $\mathbf{S}_j = \emptyset$ then $w_{\mathbf{S}_j}(X_j) = 1$. For all $j \in [d]$ such that $\mathbf{S}_j \neq \emptyset$, the random variables $w_{\mathbf{S}_j}(X_j) \in \{-1, 1\}$ are i.i.d. and uniformly distributed over $\{-1, 1\}$. Thus, $\Pr[w_{\mathbf{S}}(X_1, \ldots, X_d) = 1] = \Pr[\prod_{j \in [d]} w_{\mathbf{S}_j}(X_j) = 1] \geq 1/2$. □

Corollary 6.4.2. *Let $\mathbf{S}$ be the d-dimensional representation of $S \subseteq [md]$. The product $\prod_{k \in [d] \setminus \{j\}} w_{\mathbf{S}_k}(x_k)$, where $x_k \in [2^m]$ for all $k \in [d] \setminus \{j\}$, evaluates to 1 for at least half of the settings of variables x_k.*

Proof. Let $\mathbf{S}'$ be the $(d-1)$-dimensional vector $(\mathbf{S}_1, \ldots, \mathbf{S}_{j-1}, \mathbf{S}_{j+1}, \ldots, \mathbf{S}_d)$. Then:

$$\prod_{k \in [d] \setminus \{j\}} w_{\mathbf{S}_k}(x_k) = w_{\mathbf{S}'}(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_d).$$

By Proposition 6.4.1(1), this expression is 1 for at least half of the settings of x_k. □

6.4.1 Monotonicity

We extend our construction from Section 6.3.1 to prove Theorem 6.1.1.

Proof of Theorem 6.1.1. We use Lemma 6.2.2, giving a reduction with parameter $t = md$. Let $A \subseteq [md]$ be Alice's input and $i \in [md]$ and $B = A \cap [i-1]$ be Bob's input.

The combining operator ψ is defined as follows. It receives A, i, B as input. Then it computes $S = A \triangle B = A \cap \{i, \dots, md\}$ and the d-dimensional vectors $\boldsymbol{i}$ and $\mathbf{S}$ corresponding to i and S, respectively. (See Definition 6.4.1. Also recall definitions of the step functions $\hat{s}_{\boldsymbol{i}}$ and Walsh functions $w_{\mathbf{S}}$ on the hypergrid—specifically, Definitions 6.3.2, 6.4.2 and 6.4.4.) It returns the function $h : [n]^d \to \{d-1, \dots, dn+1\}$ defined by

$$h(x) = 2\hat{s}_{\boldsymbol{i}}(x) + w_{\mathbf{S}}(x).$$

By Proposition 6.4.1, $w_{\mathbf{S}} = w_{\mathbf{A}} \cdot w_{\mathbf{B}}$, where $\mathbf{A}$ and $\mathbf{B}$ are d-dimensional vector representations of A and B, respectively. Bob knows i and B and can compute their vector representations. To determine $h(x)$, he only needs Alice to communicate the bit $w_{\mathbf{A}}(x)$. Thus, ψ is a one-bit one-way combining operator.

Lemma 6.4.3 concludes the proof that ψ is a reduction operator for monotonicity and $\epsilon_0 = 1/8$, implying the theorem. □

Lemma 6.4.3. *Fix $i \in [md]$ and $S \subseteq \{i, \dots, dm\}$, and let $\boldsymbol{i}$ and $\mathbf{S}$, respectively, be their d-dimensional vector representations.*

1. *If $i \notin S$, then the function h is monotone;*

2. *If $i \in S$, then the function h is $\frac{1}{8}$-far from monotone.*

Proof. Let $j^* = \lceil i/m \rceil$. We will show that all line restrictions of h to dimensions other than j^* are monotone. If $i \notin S$, we will show that all line restrictions of h to dimension j^* are also monotone, so h itself is monotone. Conversely, if $i \in S$, we will show that at least half of the line restrictions of h to dimension j^* are 1/4-far from monotone, so h itself is 1/8-far from monotone.

Consider the restriction of $h = 2\hat{s}_{\boldsymbol{i}} + w_{\mathbf{S}}$ to a line in dimension $j \in [d]$, i.e., a function $\bar{h} : [2^m] \to \mathbb{N}$ defined by $\bar{h}(x_j) = h(\bar{x}_1, \dots, \bar{x}_{j-1}, x_j, \bar{x}_{j+1}, \dots, \bar{x}_d)$, where the values $\bar{x}_k \in [2^m]$ are fixed for all $k \in [d] \setminus \{j\}$. Then

$$\begin{aligned} \bar{h}(x_j) &= 2\sum_{k \neq j} s_{\boldsymbol{i}_k}(\bar{x}_k) + 2s_{\boldsymbol{i}_j}(x_j) + w_{\mathbf{S}_j}(x_j) \cdot \prod_{k \neq j} w_{\mathbf{S}_k}(\bar{x}_k) \\ &= 2s_{\boldsymbol{i}_j}(x_j) \pm w_{\mathbf{S}_j}(x_j) + c, \end{aligned} \tag{6.1}$$

where $\pm$ means "either + or -" and c is a constant independent of x_j.

If $j < j^*$ then $\mathbf{S}_j = \emptyset$, $\boldsymbol{i}_j = m$ and $\bar{h} = 2s_m \pm w_\emptyset + c = 2 \pm 1 + c$. And if $j > j^*$ then $\boldsymbol{i}_j = 0$, so $\bar{h}(x_j) = 2x_j \pm w_{\mathbf{S}_j}(x_j) + c$. In both cases, the function $\bar{h}$ is monotone.

Finally, if $j = j^*$ then $\boldsymbol{i}_j = i - (j-1)m$. In this case, $i \in S$ iff $\boldsymbol{i}_j \in \mathbf{S}_j$. If $\boldsymbol{i}_j \notin \mathbf{S}_j$ then, by (6.1) and Lemma 6.3.3, $\bar{h}(x_j)$ is monotone. Since all line restrictions of $h(x)$ are monotone, the overall function $h(x)$ is monotone. Now suppose $\boldsymbol{i}_j \in \mathbf{S}_j$. Consider the product $\prod_{k \neq j} w_{\mathbf{S}_k}(\bar{x}_k)$ that determines whether the expression $\pm$ in (6.1) is actually a plus or a minus. By Corollary 6.4.2, this product evaluates to 1 for at least half of the line restrictions $\bar{h}$ of h in dimension j. For those restrictions, $\bar{h}(x_j) = 2s_{\boldsymbol{i}_j}(x_j) + w_{\mathbf{S}_j}(x_j) + c$ and, since $\boldsymbol{i}_j \in \mathbf{S}_j$, Lemma 6.3.3 implies that $\bar{h}$ is $\frac{1}{4}$-far from monotone. Thus, at least half of the line restrictions of h in dimension j are 1/4-far from monotone. Since the domains of line restrictions of h in dimension j partition the domain of h, it implies that the overall function $h(x)$ is $\frac{1}{8}$-far from monotone. $\square$

6.4.2 Convexity

In this section, we give lower bounds for testing convexity and a related property called *separate convexity.*

Definition 6.4.5 (Separate convexity)**.** *The function $f\colon [n]^d \to \mathbb{R}$ is* separately convex *if for all $i \in [d]$ and all sets of values $(\bar{x}_1, \ldots, \bar{x}_{i-1}, \bar{x}_{i+1}, \ldots \bar{x}_d) \in [n]^{d-1}$, the one-dimensional function $\bar{f} : [n] \to \mathbb{R}$ defined by $\bar{f}(x_i) = f(\bar{x}_1, \ldots, \bar{x}_{i-1}, x_i, \bar{x}_{i+1}, \ldots, \bar{x}_d)$ is convex.*

Separate convexity is a weaker property than (standard) convexity: every convex function is also separately convex, but the converse is not true.

Theorem 6.4.4. *Fix $\epsilon \in (0, \frac{1}{8}]$; $m, r \in \mathbb{N}$. Let $n = 2^m$. Any nonadaptive ϵ-test for separate convexity of functions $f : [n]^d \to [r]$ where $r = \Omega(dn^2)$ must make $\Omega(d \log n)$ queries.*

The proofs of Theorem 6.4.4 and Theorem 6.1.2 have some common elements so we present them together.

Proof of Theorem 6.1.2 and Theorem 6.4.4. We apply Lemma 6.2.2 with parameter $t = md$. Let $A \subseteq [md]$ be the set received by Alice and let $i \in [md]$ and $B = A \cap [i-1]$ be Bob's input. Let $j^* = \lceil i/m \rceil$. Let $\mathbf{A}, \mathbf{B}$ and $\boldsymbol{i}$ be the d-dimensional vectors corresponding to A, B and i respectively. The combining operator ψ receives $\mathbf{A}$ and $\boldsymbol{i}$ as input and returns the function $h : [n]^d \to \mathbb{R}$ defined by

$$h(x_1, \ldots, x_d) = \alpha \left(\frac{1}{2} \left(w_{\mathbf{S}}(x) + 1 \right) + r_{\boldsymbol{i}_{j^*}}(x_{j^*}) \right) + \sum_{j=j^*+1}^{d} {x_j}^2$$

where $\mathbf{S}$ is a d-dimensional vector corresponding to $S = A \triangle B = A \cap \{i, \ldots, md\}$ and r is the family of rising-step-size function (Definition 6.3.6). The parameter $\alpha > 0$ will be selected later. For any $x \in [n]^d$, Bob only needs the single bit $w_{\mathbf{A}}(x)$ from Alice to compute $h(x)$, so ψ is a one-bit one-way combining operator.

To show that ψ is a reduction operator for convexity (resp., separate convexity) we need to show that if $i \notin S$ (or equivalently $\boldsymbol{i}_{j^*} \notin \mathbf{S}_{j^*}$) then h is convex (resp., separately convex) and otherwise h is $\frac{1}{8}$-far from convex (resp., separately convex).

We first show how to complete the proof using the following two lemmas and then present their proofs below.

Lemma 6.4.5. *If $\boldsymbol{i}_{j^*} \notin \mathbf{S}_{j^*}$ then :*

1. *For $\alpha = 1$ the function h is separately convex.*

2. There exists a value of $\alpha > 0$ such that the function h is convex.

Lemma 6.4.6. *If $\boldsymbol{i}_{j^*} \in \mathbf{S}_{j^*}$ then the function h is $\frac{1}{8}$-far from separately convex for all $\alpha > 0$.*

The proof of Theorem 6.4.4 for separate convexity is completed by setting $\alpha = 1$ and noting that the range of h is $[r]$ for $[r] = O(dn^2)$ because for every $k \in [m]$ the range of r_k is $O(n^2)$. In the proof of Theorem 6.1.2 for convexity we set α to the value from the second part of Lemma 6.4.5. Then Lemma 6.4.6 implies that h is $\frac{1}{8}$-far from convex because the distance to convexity is at least the distance to separate convexity. □

Proof of Lemma 6.4.5, Part 1. The proof follows by showing that every restriction of h to any dimension $j \in [d]$ is a convex function.

Every one-dimensional restriction $\bar{h}$ of h in dimension j^* can be expressed as $\bar{h}(x_{j^*}) = \alpha(r_{\boldsymbol{i}_{j^*}}(x_i) \pm \frac{1}{2} w_{\mathbf{S}_{j^*}}(x_{j^*})) + c$, where c is some constant independent of x_{j^*}. Because $\boldsymbol{i}_{j^*} \notin \mathbf{S}_{j^*}$ this function is convex by Lemma 6.3.5. For all $j < j^*$ every one-dimensional restriction $\bar{h}$ of h to dimension j is a constant function. For all $j > j^*$, the restrictions of h to dimension j can be expressed as $\bar{h}(x_j) = \pm\frac{1}{2}\alpha w_{\mathbf{S}_j}(x_j) + {x_j}^2 + c$. The derivative of the first term $w_{\mathbf{S}_j}$ satisfies that $|\frac{1}{2}\alpha w'_{\mathbf{S}_j}(x_j)| \le \alpha$ and the derivative of the second term is $2x_j$, so for $\alpha \le 1$ the derivative $\bar{h}'$ is a nondecreasing function and $\bar{h}$ is convex. Hence, the function h is separately convex for all $\alpha \le 1$. □

Proof of Lemma 6.4.5, Part 2. We show how to pick a parameter $0 < \alpha < 1$ such that the function h is convex. By definition, to prove convexity we need to show that for every pair of points $(x, y) \in [n]^d \times [n]^d$ and every $0 < \gamma < 1$ for which $z = \gamma x + (1-\gamma)y \in [n]^d$, we have that $h(z) \le \gamma h(x) + (1-\gamma)h(y)$.

The function h is independent of the first $j^* - 1$ coordinates, so that:

$$h(x) = h(y_1, \dots, y_{j^*-1}, x_{j^*}, \dots, x_d)$$

and

$$h(z) = h(y_1, \dots, y_{j^*-1}, z_{j^*}, \dots, z_d).$$

First, consider the case when for all $j > j^*$ it holds that $x_j = y_j$ so we have $x = (x_1, \dots, x_{j^*}, y_{j^*+1}, \dots, y_d)$. By Lemma 6.4.5 (Part 1), all the restrictions $\bar{h}$ of h to dimension j^* are convex, so in this case $h(z) \le \gamma h(x) + (1-\gamma)h(y)$.

Otherwise, fix an index $j > j^*$ such that $x_j \ne y_j$.

Proposition 6.4.7. *Define $\phi_{j^*}(x) = \sum_{t=j^*+1}^{d} {x_t}^2$. For all $n, d \ge 1$ there exists a value $\delta^*(n, d) > 0$ such that the inequality*

$$\phi_{j^*}(\gamma x + (1-\gamma)y) \le \gamma\phi_{j^*}(x) + (1-\gamma)\phi_{j^*}(y) - \delta^*(n, d)$$

holds for all pairs (x, y) such that $x_j \ne y_j$ for some $j > j^*$ and all $\gamma \in (0, 1)$ such that $\gamma x + (1-\gamma)y \in [n]^d$.

Proof. We have:

$$\phi_{j^*}(\gamma x + (1-\gamma)y) - \gamma\phi_{j^*}(x) - (1-\gamma)\phi_{j^*}(y)$$

$$\begin{aligned}
&= \sum_{t=j^*+1}^{d} (\gamma x_t + (1-\gamma) y_t)^2 - \gamma \sum_{t=j^*+1}^{d} {x_t}^2 - (1-\gamma) \sum_{t=j^*+1}^{d} {y_t}^2 \\
&= \sum_{t=j^*+1}^{d} \left((\gamma x_t + (1-\gamma) y_t)^2 - \gamma {x_t}^2 - (1-\gamma) {y_t}^2 \right) \\
&\leq \left((\gamma x_j + (1-\gamma) y_j)^2 - \gamma {x_j}^2 - (1-\gamma) {y_j}^2 \right) < 0.
\end{aligned}$$

The first inequality uses convexity of x^2. The second inequality uses its strict convexity and the fact that $x_j \neq y_j$.
Let $\delta(x, y, j, \gamma, n, d) = -\left((\gamma x_j + (1-\gamma) y_j)^2 - \gamma {x_j}^2 - (1-\gamma) {y_j}^2 \right) > 0$. Note that j and γ can take at most d and n^d different values respectively for any fixed pair (x, y). Thus there are at most dn^{3d} different valid tuples (x, y, j, γ). The claim follows by letting $\delta^*(n, d) = \min_{x,y,j,\gamma} \delta(x, y, j, \gamma, n, d)$. □

We set $\alpha = \frac{\delta^*(n,d)}{6(2n^2+1)}$. Using the notation introduced above,

$$h(x) = \alpha \left(\frac{1}{2} (w_{\mathbf{S}}(x) + 1) + r_{\boldsymbol{i}_{j^*}}(x_{j^*}) \right) + \sum_{j>j^*} {x_j}^2 = \alpha \left(\frac{1}{2} (w_{\mathbf{S}}(x) + 1) + r_{\boldsymbol{i}_{j^*}}(x_{j^*}) \right) + \phi_{j^*}(x).$$

Because the range of $c_{\boldsymbol{i}_{j^*}}$ is $[2n^2]$,

$$\begin{aligned}
h(z) - \gamma h(x) - (1-\gamma) h(y) &\leq \phi_{j^*}(z) - \gamma \phi_{j^*}(x) - (1-\gamma) \phi_{j^*}(y) + 3\alpha(2n^2+1) \\
&\leq -\delta^*(n, d) + 3\alpha(2n^2+1) = -\delta^*(n, d)/2 < 0,
\end{aligned}$$

where the inequalities follows from Proposition 6.4.7. This concludes the proof of the fact that h is convex. □

Proof of Lemma 6.4.6. If $\boldsymbol{i}_{j^*} \in \mathbf{S}_{j^*}$ then by Corollary 6.4.2 the product $\prod_{k \neq j^*} w_{\mathbf{S}_k}(x_k)$ evaluates to 1 for at least half of the line restrictions $\bar{h}$ of h to dimension j^*. For such restrictions, $\bar{h}(x_{j^*}) = \alpha(\frac{1}{2} w_{\mathbf{S}_{j^*}}(x_{j^*}) + r_{\boldsymbol{i}_{j^*}}(x_{j^*})) + c$, for some constant c. Lemma 6.3.5 implies that $\bar{h}$ is $\frac{1}{8}$-far from convex. The domains of the restrictions $\bar{h}$ of h in dimension j^* partition the domain of h, so we conclude that the function h is $\frac{1}{8}$-far from separately convex. □

6.4.3 The Lipschitz property

In this section, we extend our construction from Section 6.3.3 to prove Theorem 6.1.3.

Proof of Theorem 6.1.3. The starting point of the reduction is the same as in the proof of the lower bound for monotonicity in Section 6.4.1. We use the same notation for the parameters of the reduction from Augmented Index, Alice's and Bob's inputs, the set $S = A \triangle B = A \cap \{i, \ldots, md\}$ and the vector representation of these objects. Let $\hat{u}_{\boldsymbol{i}}$ be the componentwise sum (see Definition 6.4.2) of the up-down staircase functions u_i (see

Definition 6.3.7). The combining operator ψ returns the function

$$h(x) = \hat{u}_{\boldsymbol{i}}(x) - \frac{1}{2}(w_{\mathbf{S}}(x) + 1).$$

As in the proof of Theorem 6.1.1, ψ is a one-bit one-way combining operator. The next lemma completes the proof of the theorem. $\square$

Lemma 6.4.8. *Fix $i \in [md]$ and $S \subseteq \{i, \ldots, dm\}$, and let $\boldsymbol{i}$ and $\mathbf{S}$ be their respective d-dimensional vector representations.*

1. *If $i \notin S$, then the function h is Lipschitz;*
2. *If $i \in S$, then the function h is $\frac{1}{8}$-far from Lipschitz.*

Proof. Consider the restriction of h to a line in dimension $j \in [d]$, i.e., consider the single-variate function $\bar{h}(x_j) = h(\bar{x}_1, \ldots, \bar{x}_{j-1}, x_j, \bar{x}_{j+1}, \ldots, \bar{x}_d)$, where the values $\bar{x}_k \in [2^m]$ are fixed for all $k \in [d] \setminus \{j\}$. Then

$$\begin{aligned} \bar{h}(x_j) &= \sum_{k \neq j} u_{\boldsymbol{i}_k}(\bar{x}_k) + u_{\boldsymbol{i}_j}(x_j) - \frac{1}{2}\Big(w_{\mathbf{S}_j}(x_j) \cdot \prod_{k \neq j} w_{\mathbf{S}_k}(\bar{x}_k) + 1\Big) \\ &= u_{\boldsymbol{i}_j}(x_j) - \frac{1}{2}(\pm w_{\mathbf{S}_j}(x_j) + 1) + c, \end{aligned} \tag{6.2}$$

where $\pm$ means “either + or -” and c is a constant independent of x_j.

Let $j^* = \lceil i/m \rceil$. If $j < j^*$ then $\mathbf{S}_j = \emptyset$, $\boldsymbol{i}_j = m$ and $\bar{h} = u_{\boldsymbol{i}_j} - \frac{1}{2}(\pm 1 + 1) + c$. Since every up-down staircase function u_i is Lipschitz, and since a Lipschitz function plus a constant function is Lipschitz, the resulting function $\bar{h}$ is Lipschitz. If $j > j^*$ then $\boldsymbol{i}_j = 0$, so $\bar{h}(x_j) = 1 - \frac{1}{2}(\pm w_{\mathbf{S}_j}(x_j) + 1) + c$,, i.e., $\bar{h}$ is again a Lipschitz function because it is the sum of a Lipschitz function and a constant function.

Finally, if $j = j^*$ then $\boldsymbol{i}_j = i - (j-1)m$. In this case, $i \in S$ iff $\boldsymbol{i}_j \in \mathbf{S}_j$. If $\boldsymbol{i}_j \notin \mathbf{S}_j$ then, by (6.2) and Lemma 6.3.7, $\bar{h}$ is Lipschitz. Since all line restrictions of h are Lipschitz, the overall function h is Lipschitz. Now suppose $\boldsymbol{i}_j \in \mathbf{S}_j$. Consider the product $\prod_{k \neq j} w_{\mathbf{S}_k}(\bar{x}_k)$ that determines whether the expression $\pm$ in (6.1) is actually a plus or a minus. By Corollary 6.4.2, this product evaluates to 1 for at least half of the line restrictions $\bar{h}(x_j)$ of h in dimension j. For those restrictions, $\bar{h}(x_j) = u_{\boldsymbol{i}_j} + \frac{1}{2}(w_{\mathbf{S}_j} + 1)(x_j) + c$ and, since $\boldsymbol{i}_j \in \mathbf{S}_j$, Lemma 6.3.7 implies that $\bar{h}$ is $\frac{1}{4}$-far from Lipschitz. Thus, at least half of the line restrictions of h in dimension j are 1/4-far from Lipschitz. Since the domains of the line restrictions of h in dimension j partition the domain of h, the overall function h is $\frac{1}{8}$-far from Lipschitz. $\square$

Acknowledgment

The authors would like to thank Madhav Jha for his participation in the project. He declined to be a co-author, but we would like to acknowledge his contributions to the results presented here. We also thank Joshua Brody for insightful conversations about communication complexity.

Chapter 7

Beyond Direct-Sum with Application to Sketching

7.1 Introduction

We study the two-party communication complexity of solving multiple instances of a function $f(x, y)$. In this setting, Alice has $x_1, \ldots, x_k$, while Bob has $y_1, \ldots, y_k$, and they would like to communicate as few bits as possible in order to compute the list $(f(x_1, y_1), \ldots, f(x_k, y_k))$ with probability at least 2/3. We call this problem $f^k((x_1, \ldots, x_k), (y_1, \ldots, y_k))$. A natural protocol for f^k would be for Alice and Bob to run an independent protocol for each $i \in [k]$ to compute $f(x_i, y_i)$ with probability at least $1 - 1/(3k)$. Then, by a union bound, the entire list is computed correctly with probability at least 2/3. If we let $R_\delta(f)$ denote the minimal communication cost of a randomized protocol for computing f with probability at least $1 - \delta$, this gives us the upper bound $R_{1/3}(f^k) = O(kR_{1/(3k)}(f))$. A natural question is whether this is optimal.

A direct sum theorem in communication complexity states that solving k copies of f with probability at least 2/3 requires at least k times as much communication as solving a single copy with probability at least 2/3, that is, $R_{1/3}(f^k) = \Omega(kR_{1/3}(f))$. The direct sum problem is the focus of much work [46, 186, 123, 114, 25, 135, 43]. The direct sum theorem is known to hold for a number of specific functions, though it is not true for randomized private coin communication in general, as demonstrated by the Equality function. For this function, Alice and Bob have $x \in \{0, 1\}^k$ and $y \in \{0, 1\}^k$ respectively, and $f(x, y) = 1$ if $x = y$, otherwise $f(x, y) = 0$. In this case, $R_{1/3}(f^k) = \Theta(k)$ [90], yet $R_{1/3}(f) = \Theta(\log k)$ [143].

One of the most general known results about direct sums for communication is the following. Letting $D^{\mu}_{1/3}(f^k)$ denote the distributional complexity of f^k, that is, the minimal cost of a deterministic protocol for computing f^k which errs on at most a 1/3 fraction of inputs, weighted according to distribution μ, then $D^{\mu,r}_{1/3-\epsilon}(f^k) = \Omega(k(D^{\mu}_{1/3}(f) - r\log(1/\epsilon) - O(\sqrt{D^{\mu}_{1/3}(f)r})))$, where $D^{\mu,r}_{1/3-\epsilon}(f^k)$ denotes the minimal cost of a deterministic protocol for computing f^k with error probability at most $1/3 - \epsilon$ according to μ, and which uses at most r rounds of communication [43]. Moreover, this holds even if the protocol is only required to individually solve each of the n copies of f with probability at least $1/3 - \epsilon$, rather than to simultaneously solve all copies. Other related work includes direct product theorems, which state that any protocol which uses $o(kR_{1/3}(f))$ communication has success probability at most $\exp(-k)$ in solving f^k. Direct product theorems are known to hold for several functions, such as disjointness [135] and bounded round public-coin randomized communication [121, 122]. For more discussion on the latter two works, see below.

The starting point of our work is the following observation: even if a direct sum or direct product theorem were to hold for a function f, this is not enough to obtain optimal communication bounds, since one would only be able to conclude that:

$$\Omega(kR_{1/3}(f)) = R_{1/3}(f^k) = O(kR_{1/(3k)}(f)).$$

The ratio of the upper and lower bounds is $O(R_{1/(3k)}(f)/R_{1/3}(f))$, which can be as large as $\Theta(\log k)$. This $\Theta(\log k)$ factor is important in applications, which we describe below. Generic direct sum (or direct product) theorems are not suitable for addressing this gap, since such theorems do not take into account whether or not f becomes harder with smaller error probability, i.e., whether $R_\delta(f) \gg R_{\delta'}(f)$ for $\delta \ll \delta'$. For many functions, $R_{1/3}(f) = \Theta(R_0(f))$, e.g., for the disjointness problem, and there is nothing to prove in this case. Still for other functions, such as equality on n-bit strings, one can show that $R_\delta(f) = \Theta(\log 1/\delta + \log n)$, and so $R_\delta(f) \gg R_{\delta'}(f)$ for $\delta \ll \delta' \ll 1/n$.

Our Results: Our main theorem is a strengthening of the direct sum theorem for two-party randomized communication complexity to address this gap. We note that although our applications are for 1-way protocols, our main theorem holds for general 2-way communication. For that, we introduce the notion of deterministic protocols Π with the following "abortion" property.

Definition 7.1.1 (Protocols with abortion). *Consider a communication problem given by $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{Z}$ and a probability distribution μ over $\mathcal{X} \times \mathcal{Y}$. We say that a deterministic protocol Π_D (β, δ)-computes f with respect to μ if it satisfies the following (where $(X, Y) \sim \mu$):*

1. *(Abortion probability)* $\Pr[\Pi_D(X, Y) = \text{'abort'}] \leq \beta$
2. *(Success probability)* $\Pr[\Pi_D(X, Y) \neq f(X, Y) \mid \Pi_D(X, Y) \neq \text{'abort'}] \leq \delta$.

We can view randomized protocols as distributions over deterministic protocols (both for private-coin and public-coin protocols). We say that a randomized protocol Π (α, β, δ)-computes f with respect to μ if $\Pr_{\Pi_D \sim \Pi}[\Pi_D$ (β, δ)-computes $f] \geq 1 - \alpha$. The probability is taken over all randomness of the parties.

One should think of $\beta \gg \delta$. Notice that a protocol that (β, δ)-computes f is more powerful than a deterministic protocol which errs with probability at most $\approx \beta$ on the input distribution μ, since it "knows when it is wrong". On the other hand, it is less powerful than a deterministic protocol which errs with probability at most δ on distribution μ.

In our proofs we use the information complexity framework (see Section 7.2 for an introduction and basic definitions), developed and used for many important communication complexity problems [23, 25, 46, 114]. Our definitions are most closely related to those in [24] (see also [25]). Let λ be a distribution on $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ with marginals μ on $\mathcal{X} \times \mathcal{Y}$ and ν on $\mathcal{D}$. Let $(X, Y, D) \sim \lambda$ and suppose for any value of $d \in \mathcal{D}$ that X and Y are independent conditioned on $D = d$. The conditional information cost of Π under λ is defined as $\mathrm{I}(\Pi(X, Y); X, Y|D)$, where $(X, Y, D) \sim \lambda$. Let $\mathrm{IC}_{\mu,\alpha,\beta,\delta}(f|\nu)$ denote the minimum, over all protocols Π that (α, β, δ)-compute f, of $\mathrm{I}(X, Y; \Pi \mid D)$, where with some abuse of notation, Π is also used to denote the transcript of the protocol (that is, the set of messages exchanged together with the output). We also use the notation $\mathrm{IC}_{\mu,\delta}(f|\nu)$ to denote the minimum of $\mathrm{I}(X, Y; \Pi \mid D)$ over all randomized protocols Π, which $(0, 0, \delta)$-compute f (that is, which err with probability at most δ on every input, where the probability is over the random coins of Π). Notice that $\mathrm{I}(X, Y; \Pi \mid D) \leq \mathrm{H}(\Pi) \leq |\Pi|$ (where $|\Pi|$ is the maximum number of bits transmitted by Π over all inputs and choices of randomness), and so $\mathrm{IC}_{\mu,\delta}(f|\nu)$ is a lower bound on $R_\delta(f)$. As we will also be interested in 1-way protocols, we use $\mathrm{IC}^{\rightarrow}_{\mu,\alpha,\beta,\delta}(f|\nu)$ and $\mathrm{IC}^{\rightarrow}_{\mu,\delta}(f|\nu)$ to denote the above notions, where the minimum is taken over only 1-way protocols Π.

The following is our main theorem.

Theorem 7.1.1. *(Informal) For all $\delta \leq 1/3$,*

$$\mathrm{IC}_{\mu^k,\delta}(f^k|\nu^k) \geq \Omega(k)\, \mathrm{IC}_{\mu,\frac{1}{20},\frac{1}{10},\frac{\delta}{k}}(f|\nu),$$

and also $\mathrm{IC}^{\rightarrow}_{\mu^k,\delta}(f^k|\nu^k) \geq \Omega(k)\, \mathrm{IC}^{\rightarrow}_{\mu,\frac{1}{20},\frac{1}{10},\frac{\delta}{k}}(f|\nu)$.

As an example usage of our theorem, we can apply it to the Equality function f on k-bit strings. Namely, we are able to show for certain distributions μ and ν that $\mathrm{IC}^{\rightarrow}_{\mu,1/20,1/10,1/k}(f|\nu) = \Omega(\log k)$, matching the lower bound for protocols that are not allowed to abort. Our theorem therefore implies that $R^{\rightarrow}_{1/3}(f^k) = \Omega(k \log k)$, that is, the randomized 1-way complexity of solving k copies of Equality simultaneously is $\Omega(k \log k)$. This is matched by a trivial $O(k \log k)$ upper bound which solves Equality independently on each instance with probability $1 - O(1/k)$. To the best of our knowledge, no such result was known in the literature.

More importantly, we are able to apply our theorem to the augmented indexing problem on large domains with low error [126], denoted by $\mathrm{Ind}^a(k, N)$. In this problem, Alice has a list $x_1, x_2, \ldots, x_N$, each item belonging to the set $[k] = \{1, 2, \ldots, k\}$, while Bob has input $j \in [N], x_1, x_2, \ldots, x_{j-1}$, and $y \in [k]$. The function f evaluates to 1 if $x_j = y$, and otherwise it evaluates to 0. We consider 1-way protocols Π, where the message is sent from Alice to Bob. It is known that $R^{\rightarrow}_{1/k}(f) = \Theta(N \log k)$ [126]. We are able to show that for certain distributions μ and ν, we in fact have $\mathrm{IC}^{\rightarrow}_{\mu,1/20,1/10,1/k}(f|\nu) = \Omega(N \log k)$. Plugging this in to our main theorem, we obtain that $R^{\rightarrow}_{1/3}(f^k) = \Omega(kN \log k)$. Previously, it was only known that $R^{\rightarrow}_{1/3}(f^k) = \Omega(kN)$, which can be shown by using that $\mathrm{IC}^{\rightarrow}_{\mu,1/3}(f|\nu) = \Omega(N)$ [23], and applying a standard direct sum argument [24].

Our lower bound is optimal in light of a trivial upper bound in which Alice sends her entire input to Bob. The augmented indexing problem is known to have a long list of applications to data streams and sketching, some of the most recent applications appearing in [132, 129, 126], and so our lower bound on solving k copies of this problem applies to solving multiple copies of these problems, as described below.

Applications: [1] Our first application is to the sketching complexity [120, 149] of n-point Johnson-Lindenstrauss transforms. Here one wants to design a distribution over $k \times d$ matrices S so that given any n points $\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^n$ in $\mathbb{R}^d$, with probability at least $1 - \delta$, for all i and j, $\|S\mathbf{p}^i - S\mathbf{p}^j\|_2 = (1 \pm \epsilon)\|\mathbf{p}^i - \mathbf{p}^j\|_2$. See Definition 1 of [183], where this is referred to as $\mathrm{JLT}(\epsilon, \delta, n)$. Alon [6] has shown that this problem requires $k = \Omega(\frac{1}{\epsilon^2} \frac{1}{\log 1/\epsilon} \log \frac{n}{\delta})$ dimensions. Jayram and the second author show that for a constant number of points ($n = O(1)$), $k = \Omega(\epsilon^{-2} \log \frac{1}{\delta})$, which is also achievable by applying known Johnson-Lindenstrauss transforms [126]. We note that such work does not imply that for general n there is a lower bound of $k = \Omega(\epsilon^{-2} \log \frac{n}{\delta})$. Indeed, for all we knew, it could have been that $k = O(\frac{1}{\epsilon^2} \frac{1}{\log 1/\epsilon} \log \frac{n}{\delta})$, since there may be a better strategy than setting the failure probability to $O(\delta/n^2)$ and taking a union bound over the $\binom{n}{2}$ pairs of points. Our main theorem rules out this possibility, showing that $k = \Omega(\epsilon^{-2} \log \frac{n}{\delta})$ (Theorem 7.4.10). In fact, the main theorem shows that even if S is allowed to depend on the first $n/2$ points in an arbitrary way, the same lower bound still holds. In addition, we show that any encoding $\phi(\mathbf{p}^1), \ldots, \phi(\mathbf{p}^n)$ that allows pairwise ℓ_p-distance estimation for $p \in \{1, 2\}$ requires bit size $\Omega(n\epsilon^{-2} \log \frac{n}{\delta}(\log d + \log M))$, where M is the largest entry in absolute value of the vectors $\mathbf{p}^i$'s (Theorem 7.4.4); this is again optimal and achieved by known dimension reduction techniques [119].

A related problem is that of sketching matrix product, initiated in [183]. Here one wants to design a distribution over $n \times k$ matrices S, so that given $n \times n$ matrices A and B, one can "sketch" the matrices to obtain AS and $S^T B$ such that the matrix $C = ASS^T B$ approximates the product AB for some measure of error. Ideally, we would like k to be as small as possible, and obtain an entrywise error guarantee, namely, for all $i, j \in [n]$, we would like $|(AB)_{i,j} - C_{i,j}| \leq \epsilon \|A_i\|_2 \|B^j\|_2$, where A_i denotes the i-th row of A and B^j the j-th column of B. This notion of error has been used in several works, see the end of Section 1.1 of [160] for a discussion. In particular, Sárlos [183] achieves this error guarantee with $k = O(\epsilon^{-2} \log \frac{n}{\delta})$, for success probability $1 - \delta$. Later, Clarkson and the second author [58] were able to achieve $k = O(\epsilon^{-2} \log \frac{1}{\delta})$ with the weaker guarantee that $\|AB - C\|_F \leq \epsilon \|A\|_F \|B\|_F$. A natural question left open is whether $k = O(\epsilon^{-2} \log \frac{1}{\delta})$ is possible with the entrywise error guarantee. Using our main theorem, we show that this is not possible, namely that $k = \Omega(\epsilon^{-2} \log \frac{n}{\delta})$ is required in order to achieve the entrywise error guarantee (Theorem 7.4.12). We therefore separate the complexity of the two problems. Moreover, we show that sketches that satisfy the weaker guarantee that there is a procedure f outputting a matrix such that $|f(AS, B)_{i,j} - (AB)_{i,j}| \leq \epsilon \|A_i\| \|B^j\|$ for all matrices $A, B \in [\pm M]^{n \times n}$, then the bit size of AS is at least $\Omega(n \frac{1}{\epsilon^2} \log \frac{n}{\delta}(\log n + \log M))$, which is achieved in [183].

The final application we discuss is to multiple aggregation queries. While much of the data stream literature involves sequentially processing a large database to answer a single query, such as the number of distinct elements in a certain column or the join size of two tables, what one usually wants is to perform a sequence of such queries to different parts of the database. This issue was raised in [7], where the authors consider the setting of a

[1] All logarithms are base 2, unless otherwise specified. To simplify the notation, we assume throughout that quantities like $1/\epsilon^2$ and $1/\delta$ are always integral.

relation which holds multiple tables, each of which has multiple columns of attributes. The authors consider the problem of sketching each of the columns of attributes in each of the different tables, so that the storage size of the database can be reduced, yet at any later time, a user can ask for the join size along an attribute shared by two tables. They show that if the number of tables and attributes is $\mathrm{poly}(n)$, then each column can be compressed to $O(\epsilon^{-2}\log\frac{n}{\delta}\log M)$ bits, where M is an upper bound on the number of records in each table. It was left open whether or not this is optimal. Using our main theorem, we can show that $\Omega(\epsilon^{-2}\log\frac{n}{\delta}\log M)$ bits are in fact necessary (Theorem 7.4.13).

All of our results concerning linear sketches also hold for the *turnstile model* of data streaming [155, 9] and for more general data structures which, given their current state, and an update to the underlying vector, can produce a new state. Such data structures are sometimes referred to as *mergeable summaries* [3].

Our Techniques: Our starting point is the direct sum framework of [24]. There the authors show that for $(X, Y, D) \sim \lambda$ with $(X, Y) \sim \mu$ and $D \sim \nu$, if X and Y are independent conditioned on $D = d$ for any $d \in \mathcal{D}$, then $\mathrm{IC}_{\mu^k,\delta}(f^k|\nu^n) = \Omega(k\,\mathrm{IC}_{\mu,\delta}(f|\nu))$. To show this, they start with any randomized private coin protocol Π for f^k, with inputs $(X_1, Y_1), \dots, (X_k, Y_k)$ and values $D_1, \dots, D_k$, so that the X_i and Y_i are independent conditioned on D_i. They study the mutual information between the transcript and the inputs, conditioned on $D_1, \dots, D_k$, namely $I(\mathbf{X}, \mathbf{Y}; \Pi|\mathbf{D}) = H(\mathbf{X}, \mathbf{Y}|\mathbf{D}) - H(\mathbf{X}, \mathbf{Y}|\Pi, \mathbf{D})$, where $\mathbf{X} = (X_1, \dots, X_n)$, and $\mathbf{Y}$ and $\mathbf{D}$ are defined similarly. By the chain rule for mutual information,

$$\mathrm{I}(\mathbf{X}, \mathbf{Y}; \Pi \mid \mathbf{D}) = \sum_{i=1}^{k} \mathrm{I}(X_i, Y_i; \Pi \mid \mathbf{D}, \mathbf{X}_{<i}, \mathbf{Y}_{<i}),$$

where $\mathbf{X}_{<i}$ and $\mathbf{Y}_{<i}$ denote the first $i-1$ coordinates of $\mathbf{X}$ and $\mathbf{Y}$, respectively. For each summand, we further have

$$\begin{aligned}
&\mathrm{I}(X_i, Y_i; \Pi \mid \mathbf{D}, \mathbf{X}_{<i}, \mathbf{Y}_{<i}) = \\
&\sum_{\substack{\mathbf{x}_{<i}, \mathbf{y}_{<i}, \\ \mathbf{d}_{-i}}} \mathrm{I}(X_i, Y_i; \Pi \mid D_i, \mathbf{X}_{<i} = \mathbf{x}_{<i}, \mathbf{Y}_{<i} = \mathbf{y}_{<i}, \mathbf{D}_{-i} = \mathbf{d}_{-i}) \\
&\cdot \Pr[\mathbf{X}_{<i} = \mathbf{x}_{<i}, \mathbf{Y}_{<i} = \mathbf{y}_{<i}, \mathbf{D}_{-i} = \mathbf{d}_{-i}],
\end{aligned}$$

where $\mathbf{D}_{-i}$ denotes $\mathbf{D}$ with its i-th coordinate removed. The next step is the embedding step, which argues that for any choice of $\mathbf{x}_{<i}, \mathbf{y}_{<i}$, and $\mathbf{d}_{-i}$, $\mathrm{I}(X_i, Y_i; \Pi \mid D_i, \mathbf{X}_{<i} = \mathbf{x}_{<i}, \mathbf{Y}_{<i} = \mathbf{y}_{<i}, \mathbf{D}_{-i} = \mathbf{d}_{-i})$ is at least $\mathrm{IC}_{\mu,\delta}(f|\nu)$. This step works by building a protocol Π' for solving f by hardwiring the values $\mathbf{x}_{<i}, \mathbf{y}_{<i}$ and $\mathbf{d}_{-i}$ into Π'. Then given inputs (A, B) to Π' distributed according to μ, the parties set $X_i = A$, $Y_i = B$, and generate $\mathbf{X}_{>i}$ and $\mathbf{Y}_{>i}$ using private randomness without any communication. This is possible given the conditioning $\mathbf{D}_{-i} = \mathbf{d}_{-i}$. A randomized protocol Π for f^k, for every input, solves f in each coordinate simultaneously with probability at least $1-\delta$, and therefore Π' is a correct protocol for f with probability at least $1-\delta$. Moreover, this simulation guarantees that $\mathrm{I}(X_i, Y_i; \Pi \mid D_i, \mathbf{X}_{<i} = \mathbf{x}_{<i}, \mathbf{Y}_{<i} = \mathbf{y}_{<i}, \mathbf{D}_{-i} = \mathbf{d}_{-i}) = \mathrm{I}(A, B; \Pi' \mid D_i) \geq \mathrm{IC}_{\mu,\delta}(f|\nu)$.

Our main idea is to change the embedding step as follows. Observe that

$$\begin{aligned}1-\delta &\le \Pr(\Pi(\mathbf{X},\mathbf{Y}) = f^k(\mathbf{X},\mathbf{Y}))\\ &= \prod_{i=1}^{k} \Pr(\Pi_i(\mathbf{X},\mathbf{Y}) = f^k_i(\mathbf{X},\mathbf{Y}) \mid \Pi_{<i}(\mathbf{X},\mathbf{Y}) = f^k_{<i}(\mathbf{X},\mathbf{Y})),\end{aligned}$$

where $\Pi_i(\mathbf{X},\mathbf{Y})$ denotes the i-th coordinate of the output of Π, and $f^k_i(\mathbf{X},\mathbf{Y})$ the i-th coordinate of the output of f^k, and similarly define $\Pi_{<i}(\mathbf{X},\mathbf{Y})$ and $f^k_{<i}(\mathbf{X},\mathbf{Y})$. Hence, by averaging, most of the k terms in the product are at least $1-O\left(\frac{\delta}{k}\right)$. Qualitatively speaking, conditioned on Π succeeding on a typical prefix of the first $i-1$ coordinates, it is much more likely to succeed on the i-th coordinate than it would be without this conditioning.

This motivates the following change to the embedding step: since $\mathbf{x}_{<i}$ and $\mathbf{y}_{<i}$ are hardwired into Π, the parties know the value $f(x_j, y_j)$ for all $j < i$, and given the output of Π, can first verify whether or not $\Pi_{<i}(\mathbf{X},\mathbf{Y}) = (f(x_1,y_1),\ldots,f(x_{i-1},y_{i-1}))$. If this condition holds, then they can output $\Pi_i(\mathbf{X},\mathbf{Y})$ as usual. However, if this condition fails to hold, the parties output 'abort'. We prove that for a typical prefix $\mathbf{x}_{<i},\mathbf{y}_{<i}$, for most of the random seeds of the protocol and for most choices of random suffixes $\mathbf{X}_{>i}$ and $\mathbf{Y}_{>i}$, the following holds: the parties only abort with constant probability over the inputs $(A,B) \sim \mu$, and given that they do not abort, the output is correct with a very large $1-O(1/k)$ probability. Moreover, we still have that the information revealed by this protocol can be used to lower bound the term $\mathrm{I}(X_i, Y_i; \Pi \mid \mathbf{D}, \mathbf{X}_{<i}, \mathbf{Y}_{<i})$.

To complete the direct sum argument, we need a way of lower-bounding the information revealed by a protocol with this abortion property. For this, we directly bound the information revealed by designing an estimation procedure for predicting (X_i, Y_i) from the transcript of Π, and applying Fano's inequality. This part generalizes the approach used in [126], who show lower bounds for $(0,0,\delta)$-protocols. In our case we show that allowing constant probability of failure over the choice or randomness of the protocol and constant probability of abortion over the choice of the input doesn't change the asymptotic dependence of information cost as a function of the success probability over the choice of the input conditioned on non-abortion.

Other Related Work: In [121, 122], the authors show that for $O(1)$-round public-coin randomized communication complexity $R^{\mathrm{pub}}_{1-(1-\epsilon/2)^{\Omega(k\epsilon^2)}}(f^k) = \Omega\left(\epsilon k \left(R^{\mathrm{pub}}_{\epsilon}(f) - O\left(\frac{1}{\epsilon^2}\right)\right)\right)$, where $\epsilon > 0$ is arbitrary. One cannot apply this theorem to our problem, as one would need to set $\epsilon = 1/k$ to obtain our results, at which point the theorem gives a trivial bound. A similar problem occurs trying to apply the direct sum theorem of [124]. These are not drawbacks of these works, since their study is for a vastly different regime of parameters, namely, for constant ϵ, and for every relation f. We instead only consider functions f for which we can lower bound the conditional information cost of protocols with the abortion property. These are of particular importance for sketching and streaming applications and for these functions we obtain the first optimal bounds.

7.2 The Direct Sum Theorem

We give a summary of basic properties of the entropy of a discrete random variable X, denoted as $H(X)$, and the mutual information between two discrete random variables X and Y, denoted as $I(X;Y) = H(X) - H(X|Y)$, below (see Chapter 2 in [61] for the proofs):

Proposition 7.2.1. *1. Entropy span:* $0 \le H(X) \le \log|supp(X)|$.

2. $I(X;Y) \ge 0$ *because* $H(X|Y) \le H(X)$.

3. *Chain rule:* $I(X_1, X_2, \ldots, X_n; Y|Z) = \sum_{i=1}^{n} I(X_i; Y|X_1, \ldots, X_{i-1}, Z)$.

4. *Subadditivity:* $H(X,Y|Z) \le H(X|Z) + H(Y|Z)$, *where the equality holds if and only if* X *and* Y *are independent conditioned on* Z.

5. *Fano's inequality: Let* A *be a random variable, which can be used as "predictor" of* X, *namely there exists a function* g *such that* $\Pr[g(A) = X] \ge 1 - \delta$ *for some* $\delta < 1/2$. *If* $|supp(X)| \ge 2$ *then* $H(X|A) \le \delta \log(|supp(X)| - 1) + h_2(\delta)$, *where* $h_2(\delta) = \delta \log(1/\delta) + (1-\delta)\log\frac{1}{1-\delta}$ *is the binary entropy.*

We recall standard definitions from information complexity and introduce the information complexity for protocols with abortion, denoted as $\mathrm{IC}_{\mu,\alpha,\beta,\delta}(f|\nu)$, more formally. Given a communication problem $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{Z}$, consider the augmented space $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ for some $\mathcal{D}$. Let λ be a distribution over $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$, which induces marginals μ on $\mathcal{X} \times \mathcal{Y}$ and ν on $\mathcal{D}$. We say that ν *partitions* μ, if μ is a *mixture* of product distributions, namely for a random variable $(X,Y,D) \sim \lambda$, conditioning on any value of D makes the distribution of (X,Y) product.

To simplify the notation, a δ-*protocol* for f is one that for all inputs $(x,y) \in \mathcal{X} \times \mathcal{Y}$ computes $f(x,y)$ with probability at least $1-\delta$ (over the randomness of the protocol).

Definition 7.2.1. *Let* Π *be a protocol, which computes* f. *The* conditional information cost *of* Π *under* λ *is defined as* $\mathrm{I}(\Pi(X,Y); X, Y \mid D)$, *where* $(X,Y,D) \sim \lambda$. *The* conditional information complexity *of* f *with respect to* λ, *denoted by* $\mathrm{IC}_{\mu,\delta}(f|\nu)$, *is defined as the minimum conditional information cost of a* δ*-protocol for* f. *The* information complexity with aborts, *denoted by* $\mathrm{IC}_{\mu,\alpha,\beta,\delta}(f|\nu)$, *is the minimum conditional information cost of a protocol that* (α,β,δ)*-computes* f. *The analogous quantities* $\mathrm{IC}^{\rightarrow}_{\mu,\delta}(f|\nu)$ *and* $\mathrm{IC}^{\rightarrow}_{\mu,\alpha,\beta,\delta}(f|\nu)$ *are defined by taking the respective minimums over only one-way protocols. For bounded-round protocols with abortion the information costs* $\mathrm{IC}^{(r)}_{\mu,\delta}(f|\nu)$ *and* $\mathrm{IC}^{(r)}_{\mu,\alpha,\beta,\delta}(f|\nu)$ *are defined by taking the respective minimums over only* r*-round protocols.*

Our main theorem gives a lower bound the conditional information cost of a δ-protocol for k copies of a communication problem. More precisely, for a function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{Z}$ let $f^k : (\mathcal{X} \times \mathcal{Y})^k \to \mathbb{Z}^k$ denote its k-fold version $f^k(\mathbf{x},\mathbf{y}) = (f(x_1,y_1), f(x_2,y_2), \ldots, f(x_k,y_k))$.

Theorem 7.2.2. *Let* $\delta \le 1/3$. *Then for every function* $f\colon \mathcal{X} \times \mathcal{Y} \to \mathbb{Z}$ *and distribution* λ *on* $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ *with marginal* μ *on* $\mathcal{X} \times \mathcal{Y}$ *and marginal* ν *on* $\mathcal{D}$, *such that* μ *is partitioned by* ν, *it holds that* $\mathrm{IC}_{\mu^k,\delta}(f^k|\nu^k) \ge \Omega(k)\,\mathrm{IC}_{\mu,\frac{1}{20},\frac{1}{10},\frac{\delta}{k}}(f|\nu)$. *Moreover, this result also holds for 1-way protocols:* $\mathrm{IC}^{\rightarrow}_{\mu^k,\delta}(f^k|\nu^k) \ge \Omega(k)\,\mathrm{IC}^{\rightarrow}_{\mu,\frac{1}{20},\frac{1}{10},\frac{\delta}{k}}(f|\nu)$.

For the remaining part of the section we prove this theorem. Amplifying the success probability by repeating the protocol a constant number of times, it is easy to see that $\mathrm{IC}_{\mu^k,\delta}(f^k|\nu^k) = \Omega(1)\,\mathrm{IC}_{\mu^k,\delta/2000}(f^k|\nu^k)$, and similarly for one-way protocols (see Appendix A.5). Thus, without loss of generality we work with $(\delta/2000)$-protocols instead.

We focus on the first part of the theorem. For each $i \in [k]$, consider independent random variables $(X_i, Y_i, D_i) \sim \lambda$; to simplify the notation, we use W_i to denote the pair (X_i, Y_i). Let Π be a $(\delta/2000)$-protocol for f^k with private randomness R that achieves $\mathrm{I}(\Pi(\mathbf{W}, R); \mathbf{W}|\mathbf{D}) = \mathrm{IC}_{\mu^k,\frac{1}{2000}}(f^k|\nu^k)$. Our goal is to lower bound the mutual information $\mathrm{I}(\Pi(\mathbf{W}, R); \mathbf{W}|\mathbf{D})$ by $\frac{k}{8}\,\mathrm{IC}_{\mu,\frac{1}{20},\frac{\delta}{10},\frac{\delta}{k}}(f|\nu)$, by essentially showing that Π needs to compute most of the k instances with probability $1 - O(\delta/k)$.

To make this precise, the guarantee of the protocol gives that

$$\begin{aligned} 1 - \frac{\delta}{2000} &\le \Pr(\Pi(\mathbf{W}, R) = f^k(\mathbf{W})) \\ &= \prod_{i=1}^{k} \Pr(\Pi_i(\mathbf{W}, R) = f_i^k(\mathbf{W}) \mid \Pi_{<i}(\mathbf{W}, R) = f_{<i}^k(\mathbf{W})). \end{aligned}$$

Using the bound $p \le e^{-(1-p)}$ (valid for all $p \in [0, 1]$) to each term in the right-hand side, we can then use Markov's inequality to show that for at least half of the indices $i \in [k]$ we have the strong conditional guarantee

$$\begin{aligned} \Pr(\Pi_i(\mathbf{W}, R) = f_i^k(\mathbf{W}) &\mid \Pi_{<i}(\mathbf{W}, R) = f_{<i}^k(\mathbf{W})) \\ &\ge 1 - \frac{2\ln(1 - \frac{\delta}{2000})^{-1}}{k} \ge 1 - \frac{\delta}{200k}, \end{aligned} \tag{7.1}$$

where the last inequality uses the first-order approximation of ln at 1. We call these indices *good*. Moreover, using the chain rule, we can express the mutual information $\mathrm{I}(\Pi(\mathbf{W}, R); \mathbf{W} \mid \mathbf{D})$ in terms of the information revealed of each component of $\mathbf{W}$:

$$\mathrm{I}(\Pi(\mathbf{W}, R); \mathbf{W} \mid \mathbf{D}) = \sum_{i=1}^{k} \mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i}). \tag{7.2}$$

The idea is then, for each good index i, to obtain from Π a protocol that $(1/20, \delta/10, \delta/k)$-computes $f_i^k(\mathbf{W})$ and which reveals only $O(\mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i}))$ conditional information about W_i, effectively showing that $\mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i}) \ge \Omega(\mathrm{IC}_{\mu,\frac{1}{20},\frac{\delta}{10},\frac{\delta}{k}}(f|\nu))$. This is accomplished by simulating Π over some of its input. We show next that we can "hardwire" the first $i - 1$ inputs of Π while preserving the relevant properties of the protocol. Unfortunately hardwiring the last $k - i$ inputs of Π and its random seed (and thus leaving only input i free) might change the mutual information with W_i drastically; but we show that there is a large set of suffixes that still preserve most properties that we need. The existence of such suffixes is proved via the probabilistic method.

Lemma 7.2.3. *Consider a good index $i \in [k]$. Then there exists a prefix $\mathbf{w}_{<i} \in (\mathcal{X} \times \mathcal{Y})^{i-1}$ and a set G of fixings of the suffix $\mathbf{W}_{>i}$ and the random bits used by Π with the following properties:*

1. *(Low information cost)* $\mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i} = \mathbf{w}_{<i}) \leq 4\,\mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i})$.
2. *(Large set of fixings)* $\Pr((\mathbf{W}_{>i}, R) \in G) \geq 1 - \frac{1}{20}$.
3. *(Success probability) For every $(\mathbf{w}_{>i}, r)$ in G we have*
$$\Pr\left[\Pi_{<i}(\mathbf{w}_{<i} W_i \mathbf{w}_{>i}, r) \neq f^k_{<i}(\mathbf{w}_{<i} W_i \mathbf{w}_{>i})\right] \leq \frac{\delta}{10}.$$
4. *(Conditional success probability) For every $(\mathbf{w}_{>i}, r)$ in G we have*
$$\Pr[\Pi_i(\mathbf{w}_{<i} W_i \mathbf{w}_{>i}, r) \neq f^k_i(\mathbf{w}_{<i} W_i \mathbf{w}_{>i}) \mid \Pi_{<i}(\mathbf{w}_{<i} W_i \mathbf{w}_{>i}, r) = f^k_{<i}(\mathbf{w}_{<i} W_i \mathbf{w}_{>i})] \leq \frac{\delta}{k}.$$

Proof. We start by proving the following proposition.

Proposition 7.2.4. *Consider a good index $i \in [k]$. Then there exists $\mathbf{w}_{<i} \in (\mathcal{X} \times \mathcal{Y})^{i-1}$ such that the following hold:*

- $\mathrm{I}(\Pi(\mathbf{w}_{<i} \mathbf{W}_{\geq i}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i} = \mathbf{w}_{<i}) \leq 4\,\mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i})$
- $\Pr[\Pi_{<i}(\mathbf{w}_{<i} \mathbf{W}_{\geq i}, R) \neq f^k_{<i}(\mathbf{w}_{<i} \mathbf{W}_{\geq i})] \leq \frac{\delta}{500}$
- $\Pr[\Pi_i(\mathbf{w}_{<i} \mathbf{W}_{\geq i}, R) \neq f^k_i(\mathbf{w}_{<i} \mathbf{W}_{\geq i}) \mid \Pi_{<i}(\mathbf{w}_{<i} \mathbf{W}_{\geq i}, R) = f^k_{<i}(\mathbf{w}_{<i} \mathbf{W}_{\geq i})] \leq \frac{\delta}{50k}$.

Proof. We use the probabilistic method, so first we analyze the expected value of the quantities in the left-hand side of the above expression with respect to the random variable $\mathbf{W}_{<i}$.

For Item 1, it follows from the definition of conditional mutual information that

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathbf{W}_{<i}}[\mathrm{I}(\Pi(\mathbf{W}_{<i} \mathbf{W}_{\geq i}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i})] \\
&= \mathop{\mathbb{E}}_{\mathbf{W}_{<i}}\left[\mathop{\mathbb{E}}_{\mathbf{D}}[\mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i}) \mid \mathbf{W}_{<i}]\right] \\
&= \mathrm{I}(\Pi(\mathbf{W}, R); W_i | \mathbf{D}, \mathbf{W}_{<i}).
\end{aligned}$$

For Item 2, the product structure of μ^k and the guarantee of Π give

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathbf{W}_{<i}}\left[\Pr\left(\Pi_{<i}(\mathbf{W}_{<i} \mathbf{W}_{\geq i}, R) \neq f^k_{<i}(\mathbf{w}_{<i} \mathbf{W}_{\geq i})\right)\right] \\
&= \Pr(\Pi_{<i}(\mathbf{W}, R) \neq f^k_{<i}(\mathbf{W})) \\
&\leq \Pr(\Pi(\mathbf{W}, R) \neq f^k(\mathbf{W})) \leq \frac{\delta}{2000}.
\end{aligned}$$

For Item 3, we now use the fact that i is good to obtain

$$\sum_{\mathbf{w}_{<i}} \Pr\left[\Pi_i(\mathbf{w}_{<i}\mathbf{W}_{\geq i}, R) \neq f_i^k(\mathbf{w}_{<i}\mathbf{W}_{\geq i}) \mid \Pi_{<i}(\mathbf{w}_{<i}\mathbf{W}_{\geq i}, R) = f_{<i}^k(\mathbf{w}_{<i}\mathbf{W}_{\geq i})\right]$$
$$\cdot \Pr(\mathbf{W}_{<i} = \mathbf{w}_{<i} \mid \Pi_{<i}(\mathbf{W}, R) = f_{<i}^k(\mathbf{W}))$$
$$= \Pr[\Pi_i(\mathbf{W}, R) \neq f_i^k(\mathbf{W}) \mid \Pi_{<i}(\mathbf{W}, R) = f_{<i}^k(\mathbf{W})] \leq \frac{\delta}{200k}.$$

Although this last expectation is with respect to the distribution conditioned on $\Pi_{<i}(\mathbf{W}, R) = f_{<i}^k(\mathbf{W})$, because of the guarantee of Π, this conditioning does not change the distribution by much; more precisely, for every event E we have $\Pr(E) \leq \Pr(E \mid \Pi_{<i}(\mathbf{W}, R) = f_{<i}^k(\mathbf{W})) + \delta/2000 < \Pr(E \mid \Pi_{<i}(\mathbf{W}, R) = f_{<i}^k(\mathbf{W})) + 1/4$.

Using Markov's inequality to upper bound the probability of being 4 times larger than the expectation in each of the 3 items and taking a union bound, we obtain that the there is a $\mathbf{w}_{<i}$ satisfying the desired properties in the proposition. This concludes the proof. □ □

The proof of Lemma 7.2.3 then follows from Proposition 7.2.4 above and again from the application of Markov's inequality and the union bound. □ □

Now we use the protocol Π hardwiring $\mathbf{W}_{<i} = \mathbf{w}_{<i}$ (for a $\mathbf{w}_{<i}$ as above) and $\mathbf{D}_{-i} = \mathbf{d}_{-i}$ to obtain a protocol to $(1/20, \delta/10, \delta/k)$-compute f under the distribution μ. The idea is to simulate the inputs $\mathbf{W}_{>i}$ (conditioned on $\mathbf{D} = \mathbf{d}$) and run the protocol $\Pi(\mathbf{w}_{<i}W_i\mathbf{W}_{\geq i}, R)$, aborting whenever $\Pi_{<i}(\mathbf{w}_{<i}W_i\mathbf{W}_{\geq i}, R) \neq f_{<i}^k(\mathbf{w}_{<i}W_i\mathbf{W}_{\geq i})$.

Lemma 7.2.5. *Consider a good $i \in [k]$, let $\mathbf{w}_{<i}$ satisfy Lemma 7.2.3 and let $\mathbf{d}_{-i}$ be such that $\Pr(\mathbf{W}_{<i} = \mathbf{w}_{<i}, \mathbf{D}_{-i} = \mathbf{d}_{-i}) \neq 0$. Then there exists a protocol $\bar{\Pi}$ with input in $\mathcal{X} \times \mathcal{Y}$ and only private randomness $\bar{R}$ satisfying the following:*

- *$\bar{\Pi}$ $(\frac{1}{20}, \frac{\delta}{10}, \frac{\delta}{k})$-computes f with respect to the distribution μ*
- *For $(\bar{W}, \bar{D}) \sim \lambda$, $\mathrm{I}(\bar{\Pi}(\bar{W}, \bar{R}); \bar{W} \mid \bar{D}) = \mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid D_i, \mathbf{D}_{-i} = \mathbf{d}_{-i}, \mathbf{W}_{<i} = \mathbf{w}_{<i})$.*

Moreover, if Π is 1-way, then $\bar{\Pi}$ is also 1-way.

Proof. The protocol $\bar{\Pi}$ is constructed as follows. Suppose that Alice has input $x \in \mathcal{X}$ and Bob has input $y \in \mathcal{Y}$. Since ν partitions μ, Alice and Bob use their private randomness to sample respectively $\mathbf{X}'_{>i}$ and $\mathbf{Y}'_{>i}$ according to the distribution μ^{k-i} conditioned on $\mathbf{D}_{-i} = \mathbf{d}_{-i}$; more precisely, the random variable $(\mathbf{X}'_{>i}, \mathbf{Y}'_{>i})$ has the same distribution as $(\mathbf{X}_{>i}, \mathbf{Y}_{>i}) \mid (\mathbf{D}_{-i} = \mathbf{d}_{-i})$. They also use their private randomness to obtain a random variable R' with same distribution as the random coins used in Π.

Using these random variables, the players run the protocol $\Pi(\mathbf{w}_{<i}, (x, y), (\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R')$ to obtain estimates of the vector-valued function $f^k(\mathbf{w}_{<i}, (x, y), (\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}))$. Finally, since $\mathbf{w}_{<i}$ is known to Bob, he checks whether Π gave the correct values of the first $i - 1$ coordinates of $f^k(\mathbf{w}_{<i}, (x, y), (\mathbf{X}'_{>i}, \mathbf{Y}'_{>i})$, namely if

$$\Pi_j(\mathbf{w}_{<i}, (x, y), (\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R') = f_j^k(w_j)$$

for all $j < i$; if so, he outputs the estimate of $f(x,y) = f_i^k(\mathbf{w}_{<i}, (x,y), (\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}))$ given by $\Pi_i(\mathbf{w}_{<i}, (x,y), (\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R')$, and otherwise he aborts. Let

$$\bar{\Pi}(x, y, \bar{R}) = \Pi(\mathbf{w}_{<i}, (x,y), (\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R')$$

to denote the transcript exchanged with (and output of) this protocol.

We first analyze the information revealed by the protocol. Consider $(\bar{W}, \bar{D}) \sim \lambda$. Using the definition of our random variables and the product structure of λ^k, it follows by substitution of random variables that

$$\mathrm{I}(\bar{\Pi}(\bar{W}, \bar{R}); \bar{W} \mid \bar{D}) = \mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid D_i, \mathbf{D}_{-i} = \mathbf{d}_{-i}, \mathbf{W}_{<i} = \mathbf{w}_{<i}),$$

which gives the second part of the lemma.

For the correctness of the protocol, let the set G be defined as in Lemma 7.2.3. Take any $(\mathbf{w}_{>i}, r) \in G$; we claim that, conditioned on $((\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R') = (\mathbf{w}_{>i}, r)$, the protocol $\bar{\Pi}$ $(\delta/10, \delta/k)$-computes f (notice that conditioned on $((\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R') = (\mathbf{w}_{>i}, r)$ the protocol is indeed a deterministic one). Since the event $((\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R') \in G$ only depends on the randomness of the protocol, and since $\Pr(((\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R') \in G) \geq 1 - 1/20$, this implies that $\bar{\Pi}$ $(1/20, \delta/10, \delta/k)$-computes f.

To prove the claim, let E denote the event $((\mathbf{X}'_{>i}, \mathbf{Y}'_{>i}), R') = (\mathbf{w}_{>i}, r)$. It follows again from the definition of our random variables that the probability that $\bar{\Pi}(\bar{X}, \bar{Y}, \bar{R})$ aborts conditioned on E is equal to the probability that $\Pi_{<i}(\mathbf{w}_{<i}\mathbf{W}_{\geq i}, R) \neq f_{<i}^k(\mathbf{w}_{<i}\mathbf{W}_{\geq i})$ conditioned on $(\mathbf{D}_{-i}, \mathbf{W}_{>i}, R) = (\mathbf{d}_{-i}, \mathbf{w}_{>i}, r)$. Using the mutual independence between W_i, $\mathbf{W}_{>i}$ and R, this is the same as the probability that $\Pi_{<i}(\mathbf{w}_{<i}W_i\mathbf{w}_{>i}, r) \neq f_{<i}^k(\mathbf{w}_{<i}W_i\mathbf{w}_{\geq i})$; by definition of G (Item 3 of Lemma 7.2.3), this probability is at most $\delta/10$. Similarly, we obtain that

$$\begin{aligned}
&\Pr\left[\bar{\Pi}(\bar{W}, \bar{R}) \neq f(W') \mid \bar{\Pi} \text{ does not abort}, E\right] \\
&= \Pr\Big[\Pi_i(\mathbf{w}_{<i}\mathbf{W}_{\geq i}, R) \neq f_i^k(\mathbf{w}_{<i}\mathbf{W}_{\geq i}) \mid \Pi_{<i}(\mathbf{w}_{<i}\mathbf{W}_{\geq i}, R) \\
&= f_{<i}^k(\mathbf{w}_{<i}\mathbf{W}_{\geq i}), (\mathbf{D}_{-i}, \mathbf{W}_{>i}, R) = (\mathbf{d}_{-i}, \mathbf{w}_{>i}, r)\Big] \leq \frac{\delta}{k},
\end{aligned}$$

where the last inequality follows again from the product structure of λ^k, independence of R from the other random variables, and from the definition of G. This proves the claim and shows that $\bar{\Pi}$ $(1/20, \delta/10, \delta/k)$-computes f, giving the second item in the lemma.

Finally, notice that if Π is one-way then $\bar{\Pi}$ is also one-way. This concludes the proof of the lemma. □ □

The previous lemma (averaged out over all $\mathbf{d}_{-i}$), together with the first part of Lemma 7.2.3, gives that for every good index $i \in [k]$ we can lower bound $\mathrm{I}(\Pi(\mathbf{W}, R); W_i \mid \mathbf{D}, \mathbf{W}_{<i})$ by $\frac{1}{4}\mathrm{IC}_{\mu, \frac{1}{20}, \frac{\delta}{10}, \frac{\delta}{k}}(f|\nu)$. Since at least half of the i's in $[k]$ are good, plugging this bound on (7.2) gives that $\mathrm{IC}_{\mu^k, \frac{\delta}{2000}}(f^k|\nu^k) \geq \frac{k}{8}\mathrm{IC}_{\mu, \frac{1}{20}, \frac{\delta}{10}, \frac{\delta}{k}}(f|\nu)$, and similarly for the one-way information complexity. This concludes the proof of Theorem 7.2.2.

7.3 Lower Bounds for Protocols with Abortion

In this section we prove lower bounds on the information cost of one-way protocols with abortion. To illustrate the techniques, we first consider the equality problem. In order to make the argument more formal, we introduce the following formalization of one-way protocols. Alice has a (possibly random) function $M : \mathcal{X} \to \mathcal{M}$ and Bob has a (also possibly random) function $B : \mathcal{M} \times \mathcal{Y} \to \mathbb{Z}$ that depends on the received message and on its input, and $B(M(x), y)$ is the estimate for $f(x, y)$ output by Bob. Consider the augmented space $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ and let λ be a distribution on it that has marginal μ over $\mathcal{X} \times \mathcal{Y}$ and marginal ν over $\mathcal{D}$. Notice that, whenever ν partitions μ, the conditional information cost of the protocol (M, B) is given by $\mathrm{I}(M(X); X \mid D) = \mathrm{I}(M(X); X, Y \mid D)$, where $(X, Y, D) \sim \lambda$. For such distributions, $\mathrm{IC}^{\rightarrow}_{\mu,\delta}(f|\nu)$ is the minimum of $\mathrm{I}(M(X); X \mid D)$ over all one-way δ-protocols (M, B) for f.

7.3.1 Equality Problem

Let EQ^{ℓ} denote the equality problem: Alice and Bob have respectively the binary strings $\mathbf{x}$ and $\mathbf{y}$ of length ℓ and their goal is to check whether $\mathbf{x} = \mathbf{y}$ or not.

Lemma 7.3.1. *For $\ell = \log(1/20\delta)$, with $\delta \in (0, 1)$, there exists a distribution with marginals μ and ν, such that ν partitions μ and*

$$\mathrm{IC}^{\rightarrow}_{\mu,\frac{1}{20},\frac{1}{10},\delta}(\mathrm{EQ}^{\ell}|\nu) = \Omega\left(\log(1/\delta)\right).$$

Proof. To construct μ and ν, let D_0 be a random variable uniformly distributed on $\{0, 1\}$ and let $\mathbf{D}$ be a random variable uniformly distributed on $\{0, 1\}^{\ell}$. Let $(\mathbf{X}, \mathbf{Y})$ be a random variable supported on $\{0, 1\}^{\ell} \times \{0, 1\}^{\ell}$ such that, conditioned on $D_0 = 0$ we have $\mathbf{X}$ and $\mathbf{Y}$ distributed independently and uniformly on $\{0, 1\}^{\ell}$, and conditioned on $D_0 = 1$ we have $\mathbf{X} = \mathbf{Y} = \mathbf{D}$. Let μ be the distribution of $(\mathbf{X}, \mathbf{Y})$ and let ν be the distribution of $(D_0\mathbf{D})$. Note that ν partitions μ.

Consider a one-way protocol Π for EQ^{ℓ} and let M denote Alice's message function. Since $\mathbf{X}$ and $\mathbf{Y}$ are independent conditioned on $D_0\mathbf{D}$, we have

$$\mathrm{I}(M(\mathbf{X}); \mathbf{X}, \mathbf{Y} \mid D_0\mathbf{D}) = \mathrm{I}(M(\mathbf{X}); \mathbf{X} \mid D_0\mathbf{D}) = \mathrm{H}(\mathbf{X} \mid D_0\mathbf{D}) - \mathrm{H}(\mathbf{X} \mid M(\mathbf{X}), D_0\mathbf{D}).$$

Notice that $\mathrm{H}(\mathbf{X} \mid D_0\mathbf{D}) \geq \frac{1}{2}\,\mathrm{H}(\mathbf{X} \mid D_0 = 0, \mathbf{D}) = \frac{1}{2}\log(1/20\delta)$.

From Fano's inequality [61] we also have

$$\mathrm{H}(\mathbf{X} \mid M(\mathbf{X}), D_0\mathbf{D}) \leq \mathrm{H}(\mathbf{X} \mid M(\mathbf{X})) \leq 1 + p_e \log(|supp(\mathbf{X})|),$$

where $p_e = \min_g \Pr[g(M(\mathbf{X})) \neq \mathbf{X}]$ is the minimum error over all predictors g. Thus, to prove the lemma it suffices to show that if Π $(1/20, 1/10, \delta)$-computes EQ^{ℓ} then we can obtain a predictor with error at most $2/5$.

First assume Π is a deterministic protocol that $(1/10, \delta)$-computes EQ^{ℓ}. We say that an input $\mathbf{x}$ for Alice is *good* if $\Pi(\mathbf{x}, \mathbf{y}) = 1$ iff $\mathbf{x} = \mathbf{y}$; we claim that many inputs are good.

Note that the probability mass that our distribution assigns to every input $(\mathbf{x}, \mathbf{x})$ is

$$\begin{aligned} p_1 = &\Pr[D_0 = 0]\Pr[\mathbf{X} = \mathbf{Y} = \mathbf{x} \mid D_0 = 0] \\ &+ \Pr[D_0 = 1]\Pr[\mathbf{X} = \mathbf{Y} = \mathbf{x} \mid D_0 = 1] = 200\delta^2 + 10\delta. \end{aligned}$$

The probability assigned to every input $(\mathbf{x}, \mathbf{y})$ for $\mathbf{x} \neq \mathbf{y}$ is equal to $p_2 = 200\delta^2$. So the number of $\mathbf{x}$'s such that $\Pi(\mathbf{x}, \mathbf{x}) = \textit{abort}$ is at most $\Pr[\Pi = \textit{abort}]/p_1 = 1/(10p_1) \leq 1/(100\delta)$. Similarly, the number of $\mathbf{x}$'s such that there is at least one $\mathbf{y}$ where the protocol does not abort but makes a mistake is at most $\Pr[\Pi \neq \mathrm{EQ}^\ell, \Pi \neq \textit{abort}]/p_2 \leq \delta/(200\delta^2) = 1/(200\delta)$. Finally notice that if $\mathbf{x}$ does not satisfy either of these two conditions then $\mathbf{x}$ is good. This implies that there are at most $\frac{3}{200\delta}$ not good $\mathbf{x}$'s, and hence the probability that $\mathbf{X}$ is not good is at most $3/10$.

Now notice that if $\mathbf{x}$ is good then we can recover $\mathbf{x}$ itself from $M(\mathbf{x})$ using Π: simply find the unique $\mathbf{y}$ such that Bob outputs 1 upon receiving message $M(\mathbf{x})$. This then gives a predictor g with error probability $p_e \leq 3/10$ as desired.

For the case where Π only $(1/20, 1/10, \delta)$-computes EQ^ℓ, we can use the same argument as before and run Bob's part of the protocol over all $\mathbf{y}$ upon receiving message $M(\mathbf{x})$, but now we need Bob's private coins R^B to do it. This gives a predictor for $\mathbf{X}$ using $M(\mathbf{X})$ and R^B with error at most $3/10 + 1/20 \leq 2/5$, which shows that $\mathrm{H}(\mathbf{X} \mid M(\mathbf{X}), D_0\mathbf{D}) = \mathrm{H}(\mathbf{X} \mid M(\mathbf{X}), D_0\mathbf{D}, R^B) \leq 1 + \frac{2}{5}\log(1/20\delta)$. This concludes the proof. □ □

7.3.2 Augmented Indexing

In order to obtain the desired lower bound for our applications we need a generalization of EQ^ℓ, namely the augmented indexing problem on large domain with low error $\mathrm{Ind}^a(k, N)$, presented in the introduction.

Theorem 7.3.2. *Consider an integer k and a parameter δ such that k is at least a sufficiently large constant and $\delta \leq \frac{1}{20k}$. Then there is a distribution with marginals μ and ν such that ν partitions μ and $\mathrm{IC}^{\rightarrow}_{\mu, \frac{1}{20}, \frac{1}{10}, \delta}(\mathit{Ind}^a(k, N)|\nu) \geq \Omega(N \log k)$.*

In the remainder of this section we prove Theorem 7.3.2. To do so, we consider the following hard distribution for $\mathrm{Ind}^a(k, N)$. First we have the random variable $\mathbf{D}$ uniformly distributed in $[k]^N$ and a random variable D_0 taking value 0 or 1 with equal probability. The distribution of Alice's input is given by $\mathbf{X}$ and the distribution of Bob's input is given by $(I, \mathbf{Y}_{<I}, Y)$ as follows: when $D_0 = 1$, we set I uniformly at random from $[N]$, $\mathbf{Y}_{<I} = \mathbf{X}_{<I} = \mathbf{D}_{<I}$, $Y = X_I = D_I$ and $\mathbf{X}_{>I}$ uniformly in $[k]^{N-I}$; when $D_0 = 0$, we again set I uniformly at random from $[N]$, $\mathbf{Y}_{<I} = \mathbf{X}_{<I} = \mathbf{D}_{<I}$, $\mathbf{X}_{>I}$ uniformly in $[k]^{N-I}$, but now Y and X_I are picked independently and uniformly at random in $[k]$.

Let λ denote the joint distribution of $(\mathbf{X}, I, \mathbf{Y}_{<I}, Y, D_0, \mathbf{D}_{\leq I})$, with marginal μ over $(\mathbf{X}, I, \mathbf{Y}_{<I}, Y)$ and marginal ν over $(D_0, \mathbf{D}_{\leq I})$ (notice that the we use $\mathbf{D}_{\leq I}$ and not $\mathbf{D}$). We remark that μ is partitioned by ν.

Now we show that Theorem 7.3.2 holds with the distribution defined above. For that, consider a private-randomness one-way protocol given by Alice's message function M and Bob's output function B that $(1/20, 1/10, \delta)$-computes $\mathrm{Ind}^a(k, N)$ with respect to μ and has conditional information cost $\mathrm{I}(M(\mathbf{X});\mathbf{X} \mid D_0\mathbf{D}_{\leq I}) = \mathrm{IC}^{\rightarrow}_{\mu,\frac{1}{20},\frac{1}{10},\delta}(\mathrm{Ind}^a(k,N) \mid \nu)$. We show that the mutual information $\mathrm{I}(M(\mathbf{X});\mathbf{X} \mid D_0\mathbf{D}_{\leq I})$ is $\Omega(N \log k)$.

First, using the chain rule for mutual information, we express the above conditional information in terms of the conditional information of each X_i revealed by $M(\mathbf{X})$:

$$\mathrm{I}(M(\mathbf{X});\mathbf{X} \mid D_0\mathbf{D}_{\leq I}) = \sum_{i=1}^{N} \mathrm{I}(M(\mathbf{X});X_i \mid D_0\mathbf{D}_{\leq I},\mathbf{X}_{<i}) \tag{7.3}$$

$$= \sum_{i=1}^{N} \mathrm{H}(X_i \mid D_0\mathbf{D}_{\leq I},\mathbf{X}_{<i}) - \sum_{i=1}^{N} \mathrm{H}(X_i \mid M(\mathbf{X}),D_0\mathbf{D}_{\leq I},\mathbf{X}_{<i}).$$

We first claim that for each i, the term $\mathrm{H}(X_i \mid D_0\mathbf{D}_{\leq I},\mathbf{X}_{<i})$ is at least $(\frac{1}{2N} + \frac{i-1}{N})\log k$. To see this, notice that conditioned on $I = i$ and $D_0 = 0$, X_i is independent of $\mathbf{D}_{\leq I}$, and $\mathrm{H}(X_i \mid D_0 = 0,\mathbf{D}_{\leq I},\mathbf{X}_{<i}, I = i) = \log k$. Similarly, conditioned on $I < i$, X_i is independent of $\mathbf{D}_{\leq I}$ and hence $\mathrm{H}(X_i \mid D_0\mathbf{D}_{\leq I},\mathbf{X}_{<i}, I < i) = \log k$. Since the first event holds with probability $1/2N$ and the second holds with probability $(i-1)/N$, it follows that $\mathrm{H}(X_i \mid D_0\mathbf{D}_{\leq I},\mathbf{X}_{<i}) \geq (\frac{1}{2N} + \frac{i-1}{N})\log k$. Adding over all i's then gives that

$$\sum_{i=1}^{N} \mathrm{H}(X_i \mid D_0\mathbf{D}_{\leq I},\mathbf{X}_{<i}) \geq \frac{N}{2}\log k.$$

Now we need to upper bound the second summation in (7.3). For that, we will show that the guarantee of the protocol implies that $M(\mathbf{X})$ together with the prefix $\mathbf{X}_{<i}$ leads to a good predictor of X_i (for most i's); an application of Fano's inequality will then give the desired upper bound.

To make things more explicit, let R^A and R^B denote respectively Alice's and Bob's private randomness, and define $R = (R^A, R^B)$. To simplify the notation we use $\Pi(\mathbf{x}, j, y, r)$ to denote the transcript (and, as usual, also the output) of the protocol when Alice get $\mathbf{x}$, Bob gets $(j, \mathbf{x}_{<j}, y)$ and the random seed is $r = (r^A, r^B)$, namely $\Pi(\mathbf{x}, j, y, r) = B(M(\mathbf{X}, r^A), j, \mathbf{x}_{\leq j}, y, r^B)$. We also use $f(\mathbf{x}, j, y)$ to denote the function of the associated communication game, namely $f(\mathbf{x}, j, y)$ equals 0 if $x_j \neq y$ and 1 if $x_j = y$.

We first focus on tuples $(i, \mathbf{x}, r)$ that allows for a good predictor of x_i. To capture the bad tuples, let U_1 be the set of tuples $(i, \mathbf{x}, r)$ such that the protocol with random seed r aborts on the instances where Alice has input $\mathbf{x}$ and Bob has input $(i, \mathbf{x}_{<i}, \mathbf{x}_i)$ (so it is an 'equal' input), namely $U_1 = \{(i, \mathbf{x}, r) : \Pi(\mathbf{x}, i, x_i, r) = \text{'abort'}\}$. Also define U_2 as the tuples $(i, \mathbf{x}, r)$ where the protocol with random seed r makes a mistake (but does not abort) when Alice gets input $\mathbf{x}$ and Bob gets input $(i, \mathbf{x}_{<i}, y)$ for *some* y, namely $U_2 = \{(i, \mathbf{x}, r) : \exists y \text{ st } \Pi(\mathbf{x}, i, y, r) \neq f(\mathbf{x}, i, y) \text{ and } \Pi(\mathbf{x}, i, y, r) \neq \textit{abort}\}$. We say that a tuple $(i, \mathbf{x}, r)$ is *good* if it does not belong to either U_1 or U_2.

Notice that if $(i, \mathbf{x}, r)$ is good, then: (i) $\Pi(\mathbf{x}, i, x_i, r) = 1$; (ii) for every $y \neq x_i$, $\Pi(\mathbf{x}, i, y, r) \neq 1$. Good tuples render a good predictor for X_i.

Lemma 7.3.3. *For every index $i \in [N]$, there is a predictor g_i such that*

$$\Pr\left[g_i(M(\mathbf{X}, R^A), \mathbf{X}_{<i}) = X_i\right] \geq \Pr((i, \mathbf{X}, R) \textit{ is good}).$$

Proof. We are first going to use the protocol and the information $M(\mathbf{x}, r), \mathbf{x}_{<i}, r^B$ to estimate x_i as follows: let $\tilde{g}_i(M(\mathbf{x}, r^A), \mathbf{x}_{<i}, r^B)$ be any value y such that $\Pi(\mathbf{x}, i, y, r) = B(M(\mathbf{x}, r^A), i, \mathbf{x}_{<i}, y, r^B) = 1$. (If no such y exists, set the function value to any arbitrary value). It follows directly from the paragraph before the statement of the lemma that $\tilde{g}_i(M(\mathbf{x}, r), \mathbf{x}_{<i}, r^B) = x_i$ for all good $(i, \mathbf{x}, r)$, and hence

$$\begin{aligned}&\mathop{\mathbb{E}}_{R^B}\left[\Pr\left[\tilde{g}_i(M(\mathbf{X}, R^A), \mathbf{X}_{<i}, R^B) = X_i\right]\right]\\&= \Pr\left[\tilde{g}_i(M(\mathbf{X}, R^A), \mathbf{X}_{<i}, R^B) = X_i\right] \geq \Pr((i, \mathbf{X}, R) \text{ is good}).\end{aligned}$$

To remove the dependence on R^B, simply choose an outcome r^B such that

$$\Pr\left[\tilde{g}_i(M(\mathbf{X}, R^A), \mathbf{X}_{<i}, r^B) = X_i\right] \geq \Pr((i, \mathbf{X}, R) \text{ is good}),$$

and set $g_i(m, \mathbf{x}_{<i}) = \tilde{g}_i(m, \mathbf{x}_{<i}, r^B)$. □ □

Using this lemma and Fano's inequality [61], we obtain that

$$\sum_{i=1}^{N} \mathrm{H}(X_i \mid M(\mathbf{X}, R^A), D_0\mathbf{D}_{\leq I}, \mathbf{X}_{<i}) \leq N + \log k \sum_{i=1}^{N} \Pr((i, \mathbf{X}, R) \text{ is not good}).$$

Since we have assumed that k is at least a sufficiently large constant, it suffices to show that $\sum_{i=1}^{N} \Pr((i, \mathbf{X}, R) \text{ is not good}) \leq 9N/20$. The following lemma then concludes the proof.

Lemma 7.3.4. $\Pr((I, \mathbf{X}, R)$ *is not good*$) \leq 9/20$.

Proof. Using the union bound, we get that the probability that $(I, \mathbf{X}, R)$ is not good is at most the probability that it belongs to U_1 plus the probability that it belongs to U_2. We claim that $\Pr((I, \mathbf{X}, R) \in U_1) \leq 3/10$. Using the definition of U_1 and the fact that the random variable $(\mathbf{X}, I, X_I, R)$ has the same distribution as $(\mathbf{X}, I, Y, R)|(D_0 = 1)$, we get that

$$\begin{aligned}\Pr((I, \mathbf{X}, R) \in U_1) &= \Pr(\Pi(\mathbf{X}, I, X_I, R) = \textit{abort})\\&= \Pr(\Pi(\mathbf{X}, I, Y, R) = \textit{abort} \mid D_0 = 1)\\&= \Pr(\text{protocol aborts} \mid D_0 = 1).\end{aligned}$$

Furthermore, since the protocol $(1/20, 1/10, \delta)$-computes f, by union bound we see that the probability that it aborts is at most $3/20$. Therefore, using the fact that $\Pr(D_0 = 1) = 1/2$, we directly get that $\Pr((I, \mathbf{X}, R) \in U_1) \leq 3/10$.

Now we claim that the second term $\Pr((I, \mathbf{X}, R) \in U_2)$ is at most $3/20$. To do so, let C denote the event (which is solely determined by the random seed R) that the protocol

$(1/10, \delta)$-computes f. Given that C happens with probability at least $1/20$, to prove the claim it suffices to show $\Pr((I, \mathbf{X}, R) \in U_2 \mid C) \leq 1/10$. For that, let *Err* denote the event that $\Pi(\mathbf{X}, I, Y, R) \neq f(\mathbf{X}, I, Y)$ and $\Pi(\mathbf{X}, I, Y, R) \neq \textit{abort}$. Similar to the previous case, using the definition of U_2 and the fact that the random variable $(\mathbf{X}, I, y, R)|C$ has the same distribution as $(\mathbf{X}, I, Y, R)|(D_0 = 0, Y = y, C)$, we get

$$\begin{aligned}
&\Pr((I, \mathbf{X}, R) \in U_2 \mid C) = \\
&\Pr\left[\bigvee_{y\in[k]} \left(\Pi(\mathbf{X}, I, y, R) \neq f(\mathbf{X}, I, y) \text{ and } \Pi(\mathbf{X}, I, y, R) \neq \textit{abort}\right) | C\right] \\
&\leq \sum_{y\in[k]} \Pr\left(Err \mid D_0 = 0, Y = y, C\right) \\
&= k \cdot \mathop{\mathbb{E}}_{Y}\left[\Pr\left(Err \mid D_0 = 0, Y, C\right) \mid D_0 = 0, C\right] \\
&= k \cdot \Pr\left(Err \mid D_0 = 0, C\right),
\end{aligned}$$

where the second equality follows from the fact that $\Pr(Y = y \mid D_0 = 0, C) = \Pr(Y = y \mid D_0 = 0) = 1/k$ for all y.

By definition of C, we have that $\Pr(Err \mid C) \leq \delta$, so using the fact that $\Pr(D_0 = 0 \mid C) = \Pr(D_0 = 0) = 1/2$ we obtain that $\Pr(Err \mid D_0 = 0, C) \leq 2\delta$. Plugging this bound in the last displayed equation and using the fact that $\delta \leq 1/20k$, we get that $\Pr((I, \mathbf{X}, R) \in U_2 \mid C) \leq 1/10$ as desired. This concludes the proof of the lemma. □ □

7.4 Applications

For our application we will often assume that the dimension d of the vectors that we consider satisfies $d^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$ for some constant $\gamma > 0$ (where n is the number of copies of the communication problem), otherwise Alive can simply send her whole input to Bob.

All of our lower bounds come from a reduction to the same hard problem, which is an n-fold version of the augmented indexing problem with a further indexing on top of it.

7.4.1 Hard Problem

During our reductions it will be more convenient to work with a different reformulation of the augmented indexing problem $\mathrm{Ind}^a(u, N)$. In this new problem, Alice has a set $S \subseteq [1/(\epsilon^2\delta)]$ of size exactly $1/\epsilon^2$, where the i-th element is required to belong to the range $[\frac{(i-1)}{\delta} + 1, \frac{i}{\delta}]$ (so S selects an integral element from each interval $[\frac{(i-1)}{\delta} + 1, \frac{i}{\delta}]$ with $i \in [1/\epsilon^2]$). Bob has an element $k \in [1/(\epsilon^2\delta)]$ and also the set $S' \subseteq S$ consisting of the elements in S which are strictly smaller than k. Their goal is to decide whether k belongs to S or not. Denote this problem by $\mathrm{SetInd}(\epsilon, \delta)$.

We claim that the problem $\mathrm{SetInd}(\epsilon, \delta)$ is equivalent to the problem $\mathrm{Ind}^a(u, N)$ with $N = 1/\epsilon^2$ and universe size $u = 1/\delta$. To see this, given elements $x_1, x_2, \ldots, x_N$ in $[u]$, we can "concatenate" them to form the set $\{x_1, u + x_2, \ldots, (N-1)u + x^N\} \subseteq [1/(\epsilon^2\delta)]$.

Therefore, given an instance of $\text{Ind}^a(u, N)$ it is easy to construct an instance of $\text{SetInd}(\epsilon, \delta)$ (with the same yes/no answer) using this concatenation. Moreover, we can reverse this operation and use it to obtain the reverse mapping from an instance of $\text{SetInd}(\epsilon, \delta)$ to an instance of $\text{Ind}^a(u, N)$.

Using this correspondence, Theorem 7.3.2 directly gives the following.

Corollary 7.4.1. *Assume that δ is at most a sufficiently small constant. Then there is a distribution with marginals μ and ν such that ν partitions μ and $\text{IC}^{\rightarrow}_{\mu,\frac{1}{20},\frac{1}{10},\delta}(\text{SetInd}(\epsilon, \delta) \mid \nu) \geq \Omega(\frac{1}{\epsilon^2} \log \frac{1}{\delta})$.*

Now we consider the n-fold version of this problem: Alice and Bob receive n instances of $\text{SetInd}(\epsilon, \delta/n)$ and they want, with probability at least $1-\delta$, to solve all of them. Denote this problem by $n\text{SetInd}(\epsilon, \delta)$. Our direct sum theorem directly gives the following.

Corollary 7.4.2. *Assume that δ is at most a sufficiently small constant. Then there is a distribution with marginals μ and ν such that ν partitions μ and $\text{IC}^{\rightarrow}_{\mu^n,\delta}(nSetInd(\epsilon, \delta)|\nu^n) \geq \Omega(n\frac{1}{\epsilon^2} \log \frac{n}{\delta})$.*

Finally, we take an augmented indexing of r copies of this problem to obtain our hard problem $\text{Ind}(n\text{SetInd}(\epsilon, \delta), r)$. More precisely, an instance of $\text{Ind}(n\text{SetInd}(\epsilon, \delta), r)$ is obtained as follows: consider r instances $(\mathcal{S}_1^A, \mathcal{S}_1^B), \ldots, (\mathcal{S}_r^A, \mathcal{S}_r^B)$ of $n\text{SetInd}(\epsilon, \delta)$ (where $\mathcal{S}_i^A$ and $\mathcal{S}_i^B$ denote respectively Alice's and Bob's part of the input); then Alice receives $\mathcal{S}_1^A, \ldots, \mathcal{S}_r^A$ and Bob receives and index j and the collections $\mathcal{S}_1^A, \ldots, \mathcal{S}_{j-1}^A$ and $\mathcal{S}_j^B$; their goal is to solve the instance $(\mathcal{S}_j^A, \mathcal{S}_j^B)$.

The following lower bound follows from Corollary 7.4.2 and standard direct sum arguments; for completeness we present a proof in Section A.6.1 of the appendix.

Corollary 7.4.3. *Assume that δ is at most a sufficiently small constant. Then there is a distribution with marginals μ and ν such that ν partitions μ and $\text{IC}^{\rightarrow}_{\mu,\delta}(Ind(nSetInd(\epsilon, \delta), r)|\nu) \geq \Omega(r \cdot n\frac{1}{\epsilon^2} \log \frac{n}{\delta})$.*

7.4.2 Estimating Multiple ℓ_p Distances

Consider the following communication problem: Alice has n vectors $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^n \in [\pm M]^d$, Bob has n vectors $\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^n \in [\pm M]^d$, and their goal is to compute (with probability at least $1 - \delta$) approximations $(1 \pm \epsilon)\|\mathbf{u}^i - \mathbf{v}^i\|_p$ to the ℓ_p distances *for all* $i \in [n]$. Let $\ell_p(n, d, M, \epsilon)$ denote this problem.

Theorem 7.4.4. *Assume that n is at least a sufficiently large constant and that ϵ is at most a sufficiently small constant. Also assume that there is a constant $\gamma > 0$ such that $d^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$. Then $R^{\rightarrow}_{\delta}(\ell_p(n, d, M, \epsilon)) \geq \Omega\left(n\frac{1}{\epsilon^2} \log \frac{n}{\delta}(\log d + \log M)\right)$ for $p \in \{1, 2\}$.*

In the remaining part of this section we prove the above theorem. Since we can amplify the success probability of a protocol by repeating it and taking majority (see Section A.5), we will assume throughout that δ is at most a sufficiently small constant. We separately obtain the lower bound $\Omega(n\frac{1}{\epsilon^2} \log \frac{n}{\delta} \log d)$ when the alphabet M is small (Lemma 7.4.5) and the lower bound $\Omega(n\frac{1}{\epsilon^2} \log \frac{n}{\delta} \log M)$ when the alphabet is large (Lemma 7.4.7). It is easy to verify that together these lemmas imply Theorem 7.4.4.

Lower Bound for Small Alphabet Size. We consider the problem with $M = 1$ and prove the following.

Lemma 7.4.5. *Assume that n is at least a sufficiently large constant, δ is at most a sufficiently small constant and $\epsilon \leq 1/25$. Also assume that there is a constant $\gamma > 0$ such that $d^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$. Then $R^{\rightarrow}_{\delta}(\ell_p(n, d, 1, \epsilon)) \geq \Omega\left(n \frac{1}{\epsilon^2} \log \frac{n}{\delta} \log d\right)$ for $p \in \{1, 2\}$.*

To prove this lemma, we show how to use the n-fold ℓ_p approximation problem $\ell_p(n, d, 1, \epsilon/25)$ to solve the indexing problem $\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d)$, for some constant c. The main component of the reduction is the following lemma, which is a special case of Lemma 3.1 in [126]; although in [126] the authors present the lemma for instances of the problem $\text{Ind}^a(k, N)$, the equivalence between this problem and $\text{SetInd}(\epsilon, \delta)$ directly gives the following.

Lemma 7.4.6 ([126]). *Given $\epsilon, \eta \in (0, 1]$, consider subsets $S^1, S^2, \ldots, S^r$ of $[1/(4\epsilon^2\eta)]$, each of size $1/4\epsilon^2$ (assumed to be odd). Also consider an index $j \in [r]$ and an element $k \in [1/(4\epsilon^2\eta)]$ and let S' be the set consisting of all the elements of S^j that are smaller than k. Then there is an encoding of these objects, based on a random variable R, into vectors $\mathbf{u} = \mathbf{u}(S^1, S^2, \ldots, S^r, R)$ and $\mathbf{v} = \mathbf{v}(S^1, S^2, \ldots, S^{j-1}, S', j, k, R)$ with the following properties:*

1. *The vectors $\mathbf{u}$ and $\mathbf{v}$ belong to $\{0, 1\}^{d'}$, where $d' = O(10^r \frac{1}{\epsilon^2} \log \frac{1}{\eta})$.*
2. *If k does not belong to the set S^j, then with probability at least $1 - \eta$ we have $\|\mathbf{u} - \mathbf{v}\|_p^p \geq d' 10^{-j+1}(\frac{1}{2} - \frac{3\epsilon}{10})$ for all $p > 0$.*
3. *If k belongs to the set S^j, then with probability at least $1 - \eta$ we have $\|\mathbf{u} - \mathbf{v}\|_p^p \leq d' 10^{-j+1}(\frac{1}{2} - \frac{6\epsilon}{10})$ for all $p > 0$*

Using this lemma, the reduction of $\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d)$ (for some constant c to be determined) to $\ell_p(n, d, 1, \epsilon/25)$ (where Alice and Bob have shared randomness) is straightforward. Let Alice's instance for $\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d)$ be given by the sets $\{S_i^\ell\}_{i \in [n], \ell \in [c \log d]}$, where for a fixed ℓ the sets $S_1^\ell, S_2^\ell, \ldots, S_n^\ell$ correspond to the ℓ'th copy of the n-fold problem in the indexing of $\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d)$; unraveling the definition of the problem, we get that each S_i^ℓ is a subset of $[\frac{n}{4\epsilon^2\delta}]$ of size $1/4\epsilon^2$. Similarly, let Bob's instance be given by the index $j \in [c \log d]$, the elements $k_1, k_2, \ldots, k_n$, the sets $\{S_i^\ell\}_{i \in [n], \ell < j}$ and the sets $S'_1, S'_2, \ldots, S'_n$; again unraveling the definitions we have that for all i the set S'_i consists of all the elements of S_i^j less than k_i. The players want to decide whether $k_i \in S_i^j$ holds or not for all i.

For that, they evoke Lemma 7.4.6 with $\eta = \delta/n$ and use their inputs and shared randomness to make Alice compute $\mathbf{u}_i = \mathbf{u}_i(S_i^1, S_i^2, \ldots, S_i^{c \log d}, R)$ for each i, and make Bob compute $\mathbf{v}_i = \mathbf{v}_i(S_i^1, S_i^2, \ldots, S_i^{j-1}, S'_i, j, k, R)$ for each i. Notice that these vectors have $O(d^c \frac{1}{\epsilon^2} \log \frac{n}{\delta})$ coordinates, so we can use the fact $d^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$ to set the constant c to be small enough (depending on γ) so that these vectors have at most d coordinates. Then Alice and Bob use a protocol for $\ell_p(n, d, 2, \epsilon/25)$ to obtain with probability $1 - \delta$ an approximation $val_i = (1 \pm \frac{\epsilon}{10})\|\mathbf{u}_i - \mathbf{v}_i\|_p^p$ for all i. Based on Items 2 and 3 of Lemma 7.4.6, Bob then outputs that k_i belongs to S_i^j iff $val_i \leq d 10^{-j+1}(\frac{1}{2} - \frac{5\epsilon}{10})$.

It is easy to see that whenever both the guarantees of Lemma 7.4.6 hold for all n pairs $\{(\mathbf{u}_i, \mathbf{v}_i)\}_{i=1}^n$ and the protocol for $\ell_p(n, d, 1, \epsilon/25)$ succeeds, then Bob outputs the correct answer. Since this happens with probability at least $1 - 2\delta$, we obtain the lower bound $R_\delta^{\rightarrow}(\ell_p(n, d, 1, \epsilon/25)) \geq R_{2\delta}^{\rightarrow,\text{pub}}(\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d)))$, where shared randomness is allowed.

A well-know result relates the randomized complexity of private-randomness and shared-randomness protocols (using the assumption that δ is sufficiently small) [141]:

$$R_{4\delta}^{\rightarrow}(f) \leq R_{2\delta}^{\rightarrow,\text{pub}}(f) + O(\log I + \log \tfrac{1}{\delta}), \tag{7.4}$$

where I denotes the bit size of the input. Using this bound and employing our lower bound on $R_{4\delta}^{\rightarrow}(\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d))$ given by Corollary 7.4.3, we obtain that

$$\begin{aligned}
&R_{2\delta}^{\rightarrow,\text{pub}}(\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d)) \\
&\geq R_{4\delta}^{\rightarrow}(\text{Ind}(n\text{SetInd}(2\epsilon, \delta), c \log d)) - O\left(\log\left(\frac{n}{\epsilon\delta} + \log d\right)\right) \\
&\geq \Omega(n\tfrac{1}{\epsilon^2} \log \tfrac{n}{\delta} \log d),
\end{aligned}$$

where the last inequality uses the fact that n is at least a sufficiently large constant. This concludes the proof of Lemma 7.4.5.

Lower Bound for Large Alphabet Size. In this part we prove the following.

Lemma 7.4.7. *Assume that n is at least a sufficiently large constant, δ is at most a sufficiently small constant and $\epsilon \leq 1/75$. Also assume that $d \geq \Omega(\frac{1}{\epsilon^2} \log \frac{n}{\delta})$ and that there is a constant $\gamma > 0$ such that $M^{1-\gamma} \geq \frac{d}{\epsilon^3} \log \frac{n}{\delta}$. Then $R_\delta^{\rightarrow}(\ell_p(n, d, M, \epsilon)) \geq \Omega\left(\frac{1}{\epsilon^2} n \log n \log M\right)$ for $p \in \{1, 2\}$.*

For that, we need two specific statements of JL-type transforms.

Theorem 7.4.8. *[1] Let V be an arbitrary set of n points in $\mathbb{R}^d$ and consider $k \geq C\frac{1}{\epsilon^2} \log \frac{n}{\delta}$ for some sufficiently large constant C. Let S be a $k \times d$ matrix with entries picked independently uniformly from $\{-1/\sqrt{k}, 1/\sqrt{k}\}$. Then with probability at least $1 - \delta$ we have $\|Su - Sv\|_2^2 = (1 \pm \epsilon)\|u - v\|_2^2$ for all $u, v \in V$.*

Lemma 7.4.9. *($\ell_2 \rightarrow \ell_1$ JL) Let V be an arbitrary set of n points in $\mathbb{R}^d$ and consider $k \geq C\frac{1}{\epsilon^2} \log \frac{n}{\delta}$ for some sufficiently large constant C. Let S be a $k \times d$ matrix with entries picked independently uniformly from the centered normal distribution with standard deviation $1/k$. Then with probability at least $1 - \delta$ we have $\|Su - Sv\|_1 = (1 \pm \epsilon)\|u - v\|_2$ for all $u, v \in V$.*

Proof sketch. This result is essentially proved in [148]. More precisely, consider a vector $x \in \mathbb{R}^d$ with $\|x\| = 1$ and define $Y_i = kS_i x$, where S_i is the i-th row of S. By 2-stability of the normal distribution, Y_i is also normal with variance 1. The proof then follows exactly as in the proof of Theorem 5.1 of [148]. □ □

Again the lower bound is proved using a reduction from the indexing problem $\text{Ind}(n\text{SetInd}(\epsilon, \delta), r)$, but now with r set to $c \log M$, for some constant c to be determined later. Indeed, we simply modify the reduction above as follows, starting with the ℓ_2 case.

Assume for now that the players can use shared randomness. As before, the players evoke Lemma 7.4.6 with $\eta = \delta/2n$ and make Alice compute $\mathbf{u}_i = \mathbf{u}_i(S_i^1, S_i^2, \ldots, S_i^{c\log M}, R)$ for each i, and make Bob compute $\mathbf{v}_i = \mathbf{v}_i(S_i^1, S_i^2, \ldots, S_i^{j-1}, S_i', j, k, R)$ for each i. These vectors have $O(M^c \frac{1}{\epsilon^2}\log n)$ coordinates, which is $O(M)$ for small enough c by our assumption that $M^{1-\gamma} \geq \frac{d}{\epsilon^3}\log\frac{n}{\delta}$. Now the players use their shared randomness to apply the JL transform from Theorem 7.4.8 and obtain the vectors $\{\mathbf{u}_i'\}_i$ and $\{\mathbf{v}_i'\}_i$ satisfying the following: (i) with probability at least $1-\delta/2$ we have $\|\mathbf{u}_i' - \mathbf{v}_i'\|_2^2 = (1 \pm \frac{\epsilon}{20})\|\mathbf{u}_i - \mathbf{v}_i\|_2^2$ for all $i \in [n]$; (ii) the dimension of each of these vectors is $O(\frac{1}{\epsilon^2}\log\frac{n}{\delta})$, which is $O(d)$ due to the assumption $d \geq \Omega(\frac{1}{\epsilon^2}\log\frac{n}{\delta})$; (iii) all entries of these vectors belong to the set $0, \pm 1/\sqrt{k}, \ldots, \pm O(M/\sqrt{k})$.

Then Alice and Bob can use a protocol for $\ell_2(n, O(d), O(M), \epsilon/50)$ that succeeds with probability $1-\delta$ to compute $(1 \pm \frac{\epsilon}{20})$ approximations to the distances $\|\mathbf{u}_i' - \mathbf{v}_i'\|_2^2$ for all i and decide whether k_i belongs to S_i^j or not for every i just as before. It is easy to see that Alice and Bob will report the right answer with probability at least $1-2\delta$, and hence $R_\delta^{\rightarrow}(\ell_2(n, O(d), O(M), \epsilon/50)) \geq R_{2\delta}^{\rightarrow,\mathrm{pub}}(\mathrm{Ind}(n\mathrm{SetInd}(2\epsilon,\delta), c\log M)))$. Again using (7.4) and Corollary 7.4.3 concludes the proof of Lemma 7.4.7 for the case ℓ_2.

For the case of ℓ_1 distance again the players evoke Lemma 7.4.6 with $\eta = \delta/2n$ and make Alice compute $\mathbf{u}_i = \mathbf{u}_i(S_i^1, S_i^2, \ldots, S_i^{c\log M}, R)$ for each i, and make Bob compute $\mathbf{v}_i = \mathbf{v}_i(S_i^1, S_i^2, \ldots, S_i^{j-1}, S_i', j, k, R)$ for each i. Again that these vectors have $O(M^c\frac{1}{\epsilon^2}\log n) = O(\epsilon M/d)$ coordinates for small enough c (due to our assumption on M). Now for each i they use their shared randomness to obtain a matrix S with $d' = O(\frac{1}{\epsilon^2}\log\frac{n}{\delta}) = O(d)$ columns satisfying the guarantees from Lemma 7.4.9 (with approximation factor $(1 \pm \frac{\epsilon}{75})$ and success probability $1-\delta$). Then for all i Alice computes the vector $\tilde{\mathbf{u}}_i$ by taking $S\mathbf{u}_i$ and rounding each entry to the closest additive multiple of $\epsilon/75d'$, and Bob can compute $\tilde{\mathbf{v}}_i$ similarly. One can then verify that with probability $1-\delta$ we have $\|\tilde{\mathbf{u}}_i - \tilde{\mathbf{v}}_i\|_1 = (1 \pm \frac{2\epsilon}{75})\|\mathbf{u}_i - \mathbf{v}_i\|_2$ (see for instance Section A.7.1). Then Alice checks if $\|\tilde{\mathbf{u}}_i\|_\infty \leq 2\|\mathbf{u}_i\|_2^2$ (which is $O(\epsilon M/d)$) for all i; if so, she and Bob use a protocol for $\ell_1(n, O(d), O(M), \epsilon/75)$ to compute $(1\pm\epsilon/75)\|\tilde{\mathbf{u}}_i - \tilde{\mathbf{v}}_i\|_1$ for all i with probability $1-\delta$. It is easy to see that with probability at least $1-2\delta$ Alice and Bob compute an approximation $(1 \pm \frac{\epsilon}{10})\|\mathbf{u}_i - \mathbf{v}_i\|_2^2$ for all i, which can be used as before to solve their instance of $\mathrm{Ind}(n\mathrm{SetInd}(2\epsilon,\delta), c\log M))$. The proof of the lemma then follows just as in the ℓ_2 case.

7.4.3 Other Applications

The proof of the lower bound for the remaining applications is similar in spirit to that of Theorem 7.4.4, and are presented in Section A.7 of the appendix.

JL Transforms. The main result of this section is an optimal lower bound on the dimension of a JL transform.

Definition 7.4.1. *A family $\mathcal{F}$ of $k \times d$ matrices together with a distribution μ on $\mathcal{F}$ forms a* Johnson-Lindenstrauss transform *with parameters ϵ, δ, n, d (or JLT(ϵ, δ, n, d) for short), if the following holds for $S \sim \mu$: for any set V of n vectors in $\mathbb{R}^d$, for all $\mathbf{u}, \mathbf{v} \in V$ we have $(1-\epsilon)\|\mathbf{u}-\mathbf{v}\|^2 \leq \|S\mathbf{u} - S\mathbf{v}\|^2 \leq (1+\epsilon)\|\mathbf{u}-\mathbf{v}\|^2$ with probability at least $1-\delta$. We say that k is the* dimension *of the transform.*

Theorem 7.4.10. *Assume that n is at least a sufficiently large constant and that ϵ is at most a sufficiently small constant. Also assume that there is a constant $\gamma > 0$ such that*

$d^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$. *Then any JLT*$(\epsilon, \delta, n, d)$ *has dimension at least* $\Omega(\frac{1}{\epsilon^2} \log \frac{n}{\delta})$. *Moreover, this holds even if the guarantees of the transform only need to hold for vectors in* $\{-1, 0, 1\}^d$.

Sketching Multiple Inner Products. Consider the following communication problem: Alice has n vectors $\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^n \in [\pm M]^d$ and Bob has n vectors $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^n \in [\pm M]^d$. Alice needs to send sketches $S\mathbf{u}^1, S\mathbf{u}^2, \ldots, S\mathbf{u}^n$ of her vectors to Bob, who then has to output (with probability at least $1-\delta$) approximations $\langle \mathbf{u}^i, \mathbf{v}^i \rangle \pm \epsilon \|\mathbf{u}^i\| \|\mathbf{v}^i\|$ for *all* $i \in [n]$. Let $\mathrm{Ip}(n, d, M, \epsilon)$ denote this problem.

Theorem 7.4.11. *Assume that* n *is at least a sufficiently large constant and that* ϵ *is at most a sufficiently small constant. Also assume that there is a constant* $\gamma > 0$ *such that* $(d \log M)^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$. *Then* $R_\delta^{sketch}(\mathrm{Ip}(n, d, M, \epsilon)) \geq \Omega\left(n \frac{1}{\epsilon^2} \log \frac{n}{\delta} (\log d + \log M)\right)$.

Notice that the above lower bound requires the protocol to be a sketching one: otherwise one can apply a JL transform to reduce the dimension and use ℓ_2 sampling to solve $\mathrm{Ip}(n, d, M, \epsilon)$ with communication $\tilde{O}(n \frac{1}{\epsilon^2} \log \frac{n}{\delta} \log \log_{1+\epsilon} M)$ [153, 57].

Matrix Sketching. Given a matrix A, we use A_i to denotes its i-th row and use A^j to denote its j-th column.

Theorem 7.4.12. *Assume that* n *is a sufficiently large constant and that* ϵ *is at most a sufficiently small constant. Also assume that there is a constant* $\gamma > 0$ *such that* $n^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$. *Let* S *be a random* $n \times k$ *matrix which has an estimation procedure* f *outputting a matrix satisfying the following: for every pair of matrices* $A, B \in [\pm M]^{n \times n}$, *with probability at least* $1-\delta$ *we have* $f(AS, B)_{i,j} = (AB)_{i,j} \pm \epsilon \|A_i\| \|B^j\|$ *for all* $i, j \in [n]$. *Then the bit size of* AS *is at least* $\Omega(n \frac{1}{\epsilon^2} \log \frac{n}{\delta} (\log n + \log M))$. *Moreover, if the estimation is given by* $f(AS, B)_{i,j} = (ASS^T B)_{i,j}$, *then the dimension* k *is at least* $\Omega(\frac{1}{\epsilon^2} \log \frac{n}{\delta})$.

Database Joins. We refer the reader to [7] for more details about this application. Consider a database consisting of n tables and multiple attributes, with value domain $\mathcal{D}$. Let M denote the maximum number of records over all these tables. Given attribute j in table i, we use $f_i^j(d)$ to denote the number of records in table i whose value for attribute j is equal to d. We see f_i^j as a vector in $\{0, 1, \ldots, M\}^{|\mathcal{D}|}$. Given attribute j in table i and attribute j' in table i', the *join size* of these attributes is gives by the inner product $\langle f_i^j, f_{i'}^{j'} \rangle$. For simplicity, we assume that there is only one attribute j_i in each table i that we are interested in estimating join sizes. We have the following bounds for estimating these join sizes.

Theorem 7.4.13. *Assume that* n *is at least a sufficiently large constant and that* ϵ *is at most a sufficiently small constant. Consider linear sketches of the* n *frequency vectors* $f_i^{j_i}$ *which allow the join size estimation* $\langle f_i^{j_i}, f_{i'}^{j_{i'}} \rangle \pm \epsilon \|f_i^{j_i}\| \|f_{i'}^{j_{i'}}\|$ *for all* $i, i' \in [n]$ *with probability at least* δ. *Then we have the following lower bounds for the total bit size required by these sketches:*

- *If there is a constant* $\gamma > 0$ *such that* $|\mathcal{D}|^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$, *then we have the lower bound* $\Omega(n \frac{1}{\epsilon^2} \log \frac{n}{\delta} \log |\mathcal{D}|)$.
- *If* $d \geq \frac{n}{\epsilon^2 \delta}$ *and* M *is at least a sufficiently large constant, then we have the lower bound* $\Omega(n \frac{1}{\epsilon^2} \log \frac{n}{\delta} \log M)$.

As mentioned earlier, the bounds above actually lower bound the total size of computing a mergeable summary for the n tables.

Chapter 8

Optimal Round-Complexity of the Set Intersection Problem

Communication complexity [197] quantifies the communication necessary for two or more players to compute a function, where each player holds only a portion of the function's input. This model is widely studied, with applications in circuit complexity [172], combinatorial auctions [157], compressed sensing [17], data streams [9], and many other areas. We refer the reader to the book by Kushilevitz and Nisan [143] for a thorough treatment of the subject, which we briefly survey in Section 8.1.

One of the most well-studied problems in communication complexity is the disjointness function $\mathsf{DISJ}_k^n(S,T)$. In this problem, Alice has an input set $S \subseteq [n]$ of size at most k, Bob has an input set $T \subseteq [n]$ of size at most k, and $\mathsf{DISJ}_k^n(S,T) = 1$ iff $|S \cap T| = 0$. Håstad and Wigderson [116] showed that the randomized communication complexity with constant error probability, denoted $R(\mathsf{DISJ}_k^n)$, of this problem is $\Theta(k)$. The lower bound of $\Omega(k)$ follows by taking known lower bounds for set disjointness without a cardinality restriction on S and T, due to Kalysundaram and Schnitger [131], simplified by Razborov [173] and Bar-Yossef et al. [24], and combining them with a padding argument. The upper bound of $O(k)$ is due to a protocol given by Håstad and Wigderson, which they also remark was known and used many years ago [161].

In this paper we are interested in a seemingly much harder problem than the disjointness function. Namely, we are interested in recovering the *entire set intersection* $S \cap T$, rather than only deciding if $|S \cap T| = 0$. We call this problem the INT^k problem. Computing the intersection or the size of the intersection of two sets is a fundamental problem in computer science. Since communicating $|S|$ and $|T|$ can be done in one-round with a negligible amount of communication, a protocol for intersection gives a protocol for computing the size $|S \cup T|$ of the union with our communication/round tradeoff. This in turn gives a protocol for computing the exact Jaccard similarity $\frac{|S \cap T|}{|S \cup T|}$, exact Hamming distance, exact number of distinct elements, and exact 1-rarity and 2-rarity [65].

By the lower bound for the disjointness function, we have that $R(\mathrm{INT}^k) = \Omega(k)$, which holds for any number of rounds. Also, Alice and Bob can deterministically exchange their inputs using only $O(k \log(n/k))$ bits of communication, so the deterministic 1-round communication complexity $D^{(1)}(\mathrm{INT}^k)$ is $O(k\log(n/k))$. They can also first hash the elements in their sets to $O(\log k)$-bit strings, and exchange the hashed values, from which they can decide which elements are in the intersection with probability $1 - 1/k^C$, for an arbitrarily large constant $C > 0$. This means that the randomized 1-round communication complexity $R^{(1)}(\mathrm{INT}^k)$ is $O(k\log k)$, which is optimal since $R^{(1)}(\mathsf{DISJ}_k^n) = \Omega(k\log k)$ [64, 42].

A somewhat related problem is that of computing k copies of the equality function EQ_k^n. In this problem, Alice has k strings $x^1, \ldots, x^k \in \{0,1\}^n$, Bob has k strings $y^1, \ldots, y^k \in \{0,1\}^n$, and $f(x^1, \ldots, x^k, y^1, \ldots, y^k)$, and $\mathsf{EQ}_k^n(x^1, \ldots, x^k, y^1, \ldots, y^k)$ is a length-k bit vector, where the i-th bit is 1 iff $x^i = y^i$. Feder, Kushilevitz, Naor, and Nisan [90] show that $R(\mathsf{EQ}_k^n) = \Theta(k)$. One unfortunate aspect of their protocol is that the number of rounds they achieve is $\Omega(\sqrt{k})$, as their protocol seems to be inherently sequential.

We observe in Appendix A.9 that by hashing into buckets, given a protocol for EQ_k^n, we can build a protocol for the INT^k problem. Plugging in the protocol of Feder et al., we obtain a randomized protocol for INT^k with the optimal $O(k)$ bits of communication in Theorem A.9.1. However, the round complexity is $O(\sqrt{k})$. Another way of obtaining the optimal $O(k)$ bits of communication is to use a technique of Braverman and Rao to compress a protocol to its so-called internal information cost [43]. For the INT^k problem, the internal information cost is $O(k)$, and so this results in a protocol with the optimal $O(k)$ bits of communication, with a much smaller $O(\log k)$ number of rounds. It may seem plausible that one can combine the hashing technique we use in Appendix A.9 together with $O(k)$ invocations of the recent round-optimal protocols for EQ^n [44], each with error probability $O(1/k)$. However, with such low error probability one invocation of the protocol of [44] requires $\Omega(\log k)$ communication for any number of rounds, even though the expected communication for the simpler task of verifying that two unequal inputs are indeed not equal with error probability $O(1/k)$, can be smaller.

Our Results: In this paper we give a new randomized protocol for INT^k which achieves the optimal $O(k)$ bits of communication, and simultaneously achieves $O(\log^* k)$ number of rounds, where $\log^* k$ is the iterated logarithm function, that is the number of times the logarithm function must be iteratively applied before the result is at most 1. Our number of rounds provides a significant improvement on the earlier $O(\log k)$ rounds needed to achieve the optimal $O(k)$ bits of communication given in previous work.

We also provide a more refined tradeoff, showing that with $O(r)$ rounds, one can achieve communication $O(k \ \mathrm{ilog}^r k)$, where $\mathrm{ilog}^r k$ is the function obtained by iteratively applying the logarithm function r times (e.g., $\mathrm{ilog}^1 k = \log k$, $\log^2 k = \log\log k$, etc.). Our protocols work in the common random string model, but can be turned into constructive protocols (i.e., without using Newman's theorem) in the private random string model, incurring an additive $O(\log\log n)$ bits of communication with no increase in the number of rounds.

Next, we establish the first lower bound for the INT^k problem with r rounds. Namely, for any $r \geq 1$, we show that the r-round randomized communication complexity $R^{(r)}(\mathrm{INT}^k)$ is $\Omega(k\log^r k)$, which shows that our $O(r)$-round protocol has the best possible communication (up to a constant factor) as any r-round protocol. We also note that there is a simple $\Omega(\log\log n)$ lower bound in the private random string model, for any number of rounds, which follows by a reduction of INT^1 with sets drawn from the universe of size n to the

$\mathsf{EQ}_1^{\log n}$ problem. Hence, our protocol in the private random string model is also optimal.

Since EQ_k^n is also a special case of INT^k (Fact 8.1.1), we also significantly improve the round complexity of Feder et al. [90]. We further show that our tradeoff for EQ_k^n is optimal in the sense that our communication upper bound for $O(r)$ rounds matches what is best possible for any protocol with r rounds. a

Our Techniques: *Upper Bounds:* Our upper bounds use hashing and verification. First consider the following toy protocol: there is a hash function $h : [n] \to [k/\log k]$ that the two players share. For each $i \in [k/\log k]$, the players run a set intersection protocol on $S_i = \{x \in S \mid h(x) = i\}$ and $T_i = \{y \in T \mid h(y) = i\}$. To do so, note that with high probability, simultaneously for all $i \in [k/\log k]$, $|S_i| = O(\log k)$ and $|T_i| = O(\log k)$. Alice and Bob now agree on a hash function $g_i : [n] \to [\log^3 k]$. If Alice sends $g_i(x)$ to Bob for each $x \in S_i$, then Bob can compute $g_i(y)$ for each $y \in T_i$ and check if $g_i(y)$ is in the list of hashed elements that Alice sent. Bob can similary send the $g_i(y)$ values to Alice. Both parties therefore obtain candidate sets I_A and I_B, respectively, for the intersection $S_i \cap T_i$. The communication for a given $i \in [k/\log k]$ is $O(\log k \log\log k)$ and the correctness probability is $1 - \frac{1}{\Omega(\log k)}$. An important observation now is that I_A and I_B contain $S_i \cap T_i$ with probability 1. Therefore, if $I_A = I_B$, then in fact $I_A = I_B = S_i \cap T_i$. By spending an additional $O(\log k)$ bits of communication, Alice and Bob can run an equality test on I_A and I_B, which one should think of as a "verification test", which succeeds with probability $1 - \frac{1}{k^C}$, for an arbitrarily large constant $C > 0$. Whenever the equality test succeeds, Alice and Bob can conclude $I_A = I_B = S_i \cap T_i$, since all such equality tests simultaneously succeed with very high probability. For the values of $i \in [k/\log k]$ for which the corresponding equality test detects that $I_A \neq I_B$, then the players re-run the set intersection protocol on S_i and T_i. The expected number of re-runs for each $i \in [k/\log k]$ is less than 1, and so the overall expected communication is at most $2k/\log k \cdot O(\log k \log\log k) = O(k \log\log k)$, which can be made worst-case by terminating the protocol if it consumes more than a constant factor times its expected communication cost.

To improve the communication futher, we instead hash into k buckets using a hash function h, and build a "verification tree" with these k buckets as the leaves. The tree has r levels, where r is the number of rounds we seek to achieve. For $2 \leq i \leq r$, the nodes at distance i from the leaves have $\operatorname{ilog}^{r-i} k / \operatorname{ilog}^{r-i+1} k$ children, while the nodes at distance 1 (the parents of the leaves) have $\operatorname{ilog}^{r-1} k$ children. For a given $i \in [k]$, define S_i and T_i as before. For each $i \in [k]$, we run a set intersection protocol on S_i and T_i, now with only constant expected communication. For a node at distance 1 from the leaves, we have a candidate set intersection for each of its $\operatorname{ilog}^{r-1} k$ children. We concatenate these $\operatorname{ilog}^{r-1} k$ candidate intersections as strings, and verify they are equal with a single equality test. If the equality test succeeds, then we proceed to the next level in the tree. At a node v in a given level of the tree, we perform a single equality test on all candidate intersections of leaves in the subtree $T(v)$ rooted at v. If the equality test fails at v, we re-run the set intersection protocol at all leave in $T(v)$. By carefully choosing the correctness probabilty of the equality tests run at different levels in the tree, we are able to inductively show the expected communication until the root succeeds is $O(k)$, and the number of rounds is $O(r)$. Detailed description of the protocol and analysis is given in Section 8.2.2.

Lower Bounds: We prove our lower bounds by combining a recent improvement to the direct sum theorem in information complexity [152] with recent information cost lower bounds for the EQ problem [44]. The first observation is that INT^k is at least as hard as $\mathsf{EQ}_k^{n/k}$. We use the information complexity paradigm, which was recently surveyed in a PODS tutorial [125]. Our usage of this paradigm is most closely related to that developed in [24]. Roughly, the idea is to consider the mutual information $I(\Pi; X, Y) = H(\Pi) - H(\Pi|X, Y)$, where X, Y are the inputs to the two players in a communication problem f drawn from a distribution μ, Π is the transcript of the randomized protocol used, and H is the entropy function. Since $I(\Pi; X, Y) \leq H(\Pi) \leq |\Pi|$, the mutual information lower bounds the communication of Π. Setting $IC_\mu^{(r)}(f) = \min_{\text{correct } \Pi} I(\Pi; X, Y)$, where correct Π means a randomized r-round protocol Π which errs in computing $f(X, Y)$ with probability at most $1/3$ on every input (X, Y), one has $IC_\mu^{(r)}(f) \leq R^{(r)}(f)$. Letting f^s be the function in which one player has inputs $X_1, \ldots, X_s$ and the other has inputs $Y_1, \ldots, Y_s$, and the output is the s-tuple $(f(X_1, Y_1), f(X_2, Y_2), \ldots, f(X_s, Y_S))$, a direct sum theorem [24] shows $IC_{\mu^s}^{(r)}(f^s) \geq s \cdot IC_\mu^{(r)}(f)$, where $(X_1, Y_1), \ldots, (X_s, Y_s) \sim \mu^s$. Hence,

$$R^{(r)}(\mathrm{INT}^k) \geq R^{(r)}(\mathsf{EQ}_k^{n/k}) \geq IC_{\mu^k}^{(r)}(\mathsf{EQ}_k^{n/k}) \geq k \cdot IC_\mu^{(r)}(\mathsf{EQ}^{n/k}).$$

We now would like to show that $IC_\mu^{(r)}(\mathsf{EQ}^{n/k}) = \Omega(\text{ ilog}^r k)$ for $X, Y \sim \mu$. Ideally, we could appeal to a recent bound of [44] which shows that for μ the product uniform distribution on X, Y, $IC_\mu^{(r)}(\mathsf{EQ}^{n/k}) = \Omega((1-\delta)^3 \text{ ilog}^r((1-\delta)/\epsilon))$, where ϵ is the probability that the protocol declares that $X = Y$ when in fact $X \neq Y$, and δ is the probability that the protocol declares that $X \neq Y$ when in fact $X = Y$. Here the probability is over X and Y drawn from μ, as well as the randomness of the protocol. If ϵ and δ are fixed constants bounded away from 0 and 1, then we obtain $IC_\mu^{(r)}(\mathsf{EQ}^{n/k}) = \Omega(1)$, which is too weak.

We instead use the following improvement to the direct sum theorem of Molinaro et al. [152]. We will also need to use a distribution μ' different than μ above, as well as the notion of conditional information complexity. Our distribution μ' is such that $\Pr_{\mu'}[X \neq Y] \geq 1/3$ and $\Pr_\mu[X = Y] \geq 1/3$. There is also an auxiliary random variable $W \sim \nu$, so that conditioned on $W = w$ for any value w, X and Y are indepenent. Molinaro et al. show that $IC_{(\mu')^k}^{(r)}(f^s) \geq s \cdot IC_{\mu',1/20,1/10,1/s}^{(r)}(f)$, where $IC_{\mu',1/20,1/10,1/s}^{(r)}(f)$ is the minimum value of $I(\Pi; X, Y|W)$ over r-round randomized protocols Π which, with probability at least 19/20 over their randomness, result in a deterministic protocol which is allowed to output "abort" with probability at most 1/10, and given that it does not abort, it is correct with probability at least $1 - 1/s$. Here, the latter two probabilities are over input pairs $(X, Y) \sim \mu'$. Our main idea is to change such a protocol for the $\mathsf{EQ}^{n/k}$ problem: whenever it would output "abort", have it instead declare that $X \neq Y$. Then, conditioned on the private randomness resulting in a deterministic protocol with the property as described, which we call event $\mathcal{E}$, this modification makes the resulting deterministic protocol have the property that if $X \neq Y$, then the protocol outputs $X = Y$ with probability at most $(1/s)/\Pr_{\mu'}[X \neq Y] \leq 3/s$. However, if $X = Y$, then the protocol may output $X \neq Y$ with probability $1/10 + (1/s)/\Pr_{\mu'}[X = Y] \leq 1/10 + 3/s \leq 1/5$, where the latter follows for $s \geq 30$. Call the new protocol Π'. Then, conditioned on $\mathcal{E}$, Π' rarely (probability $O(1/s)$) makes a mistake when $X \neq Y$, while with constant probability it makes a mistake when

$X = Y$. This is exactly the property we need to apply [44]. We are able to show that

$$\begin{aligned} I(\Pi; X, Y|W) &= \Omega(I(\Pi; X, Y|W, \mathcal{E})) - 1 \\ &= \Omega(I(\Pi'; X, Y|W, \mathcal{E})) - 2 = \Omega(IC_\mu^{(r)}(\mathsf{EQ}^{n/k})) - 2 = \Omega(\text{ ilog}^r k), \end{aligned}$$

where the non-trivial step is showing that for $X, Y \sim \mu'$ and $W \sim \nu$, that $I(\Pi'; X, Y|W, \mathcal{E}) = \Omega(IC_\mu^{(r)}(\mathsf{EQ}^{n/k}))$, which we prove directly using the definition of μ and ν' given in [152], and their relationship with μ.

Recent Related Work: In very recent and independent work, Saglam and Tardos [180] give a tight communication bound for the DISJ_k^n problem of $R^{(1)}\ (\mathsf{DISJ}_k^n) = \Theta(k \text{ ilog}^{(r)} k$. As with the upper bounds in [44] (which are variants of those in yet an earlier version of that by Saglam and Tardos [179]), the authors are providing a protocol for an easier problem than INT^k. However, their lower bound also implies our lower bound, though the techniques used are very different. One possible advantage of our lower bound is we establish an information cost lower bound, which is used in direct sum theorems for communication cost (see Section 8.1 for definitions).

Recently, Braverman et. al [42] also give lower bounds on the exact communication cost of DISJ_k^n, showing that it is $\frac{2}{\ln 2} k \pm o(k)$, though the result does not establish a round versus communication tradeoff.

8.1 Definitions and preliminaries

In the two-player communication model there is a function $f : \mathcal{D} \times \mathcal{D} \to \mathcal{R}$, where $\mathcal{D}$ stands for domain and $\mathcal{R}$ for range. Alice and Bob are given respective inputs $x, y \in \mathcal{D}$, and need to jointly compute $f(x, y)$. They take turns exchanging bits to each other according to a protocol, and then the players write the output of the protocol on their output tape. For deterministic protocols, this output must equal $f(x, y)$ for every input pair x, y. We let $D(f)$ be the minimum, over deterministic protocols that compute f, of the maximum number of bits exchanged by the protocol over all inputs.

For randomized protocols, there are two well-studied and closely-related models. In the common random string model the players share an infinite string of independent unbiased coin tosses, and the players are otherwise deterministic. The correctness requirement is that for every input pair x, y, the output of Alice and Bob is equal to $f(x, y)$ with probability at least $1 - \delta$, for some specified $\delta > 0$, where the probability is taken over the shared random string. We let $R_\delta(f)$ be the minimum, over protocols in the common random string model satisfying the correctness protocol for f, of the maximum number of bits exchanged by the protocol over all inputs and shared random strings. For brevity, we let $R(f) = R_{1/3}(f)$. We note that a $2/3$ success probability can be amplified to $1 - \epsilon$ for an arbitrarily small constant $\epsilon > 0$ by incurring a constant factor overhead in communication.

In the private random string model, the players do not share a random string, but rather are allowed to be randomized using private randomness. By a result of Newman [156], any problem that can be solved in the common random string model can be solved in the private random string model, adding only $O(\log \log T)$ to the communication complexity, where T is the number of different inputs to the players. One unfortunate aspect of this reduction is that it is non-constructive in the sense that for each input length n, the protocol used is either hardwired an advice string that depends on n, or the players must search for the advice string, which doesn't require communication, but can result in unnecessary computation. We give our upper bounds in the common random string model, but describe how to translate them into constructive protocols in the private random string model, preserving optimality.

Besides the total communication, another well-studied resource is the total number of messages exchanged between the two players, known as the round complexity. In certain applications a server may not always be online, resulting in a significant delay between messages. There may also be undesirable overhead in the transmission of each message. Thus, it is important to not only achieve optimal communication, but also an optimal number of rounds for a given amount of communication. We use $D^{(r)}(f)$ and $R^{(r)}(f)$ to denote the deterministic and randomized communication complexity (in the common random string model) for protocols restricted to using at most r rounds.

Let EQ^n denote the communication problem of solving Equality on binary strings of length n. Let EQ^n_k denote the communication problem, corresponding to k independent instances of EQ^n. Let INT^k denote the communication problem of computing the intersection of two sets $S, T \subseteq [n]$, such that $|S|, |T| \leq k$.

A simple reduction from EQ^n_k to INT^k can be given as follows. For an instance $(x^1, \dots, x^k, y^1, \dots, y^k)$ of EQ^n_k an instance of INT^k is constructed by creating two sets of pairs $(1, x^1), \dots (k, x^k)$ and $(1, y_1), \dots (k, y^k)$. The size of the intersection between these two sets is exactly equal to the number of equal (x^i, y^i) pairs. This fact for DISJ^n_k can be also found in [44] as Lemma 6.3.

Fact 8.1.1 ([44])**.** *If there exists a protocol Π for INT^k, where the sets are drawn from a universe of size $N \geq k^c$ for $c > 2$ then there exists a protocol Π' for EQ^n_k with the same communication complexity and success probability for $n = \lfloor \log(\frac{N}{k}) \rfloor$.*

We will use the following fact about collision probability of a randomly chosen hash function.

Fact 8.1.2. *Given a subset $S \subseteq [n]$ for size $|S| \geq 2$, $i \geq 0$ and $t = \Theta(|S|^{i+2})$, a random hash function $h \colon [n] \to [t]$ has no collisions with probability at least $1 - 1/|S|^i$, namely for all $x, y \in S$ such that $x \neq y$ it holds that $h(x) \neq h(y)$. Moreover, a random hash function satisfying such guarantee can be constructed using only $O(\log n)$ random bits.*

8.2 Upper bound

8.2.1 Auxiliary protocols

We first describe auxiliary protocols Basic-Set-Intersection (Lemma 8.2.1) and Equality (Fact 8.2.3) that we use as building blocks in our main algorithm in Section 8.2.2. For a two-party communication protocol Π we denote the output of the protocol for the first party as $\Pi_A(x, y)$ and for the second party as $\Pi_B(x, y)$.

Lemma 8.2.1 (Protocol Basic-Set-Intersection(S,T))**.** *There exists a randomized protocol* Π *(with shared randomness), such that for any* $S,T \subset [n]$ *and an integer* $i \geq 1$*, the sets* $S' = \Pi_A(S,T)$ *and* $T' = \Pi_B(S,T)$ *satisfy the following properties:*

1. $S' \subseteq S, T' \subseteq T$.
2. *If* $S \cap T = \emptyset$ *then* $S' \cap T' = \emptyset$ *with probability 1.*
3. *If* $S \cap T \neq \emptyset$ *then* $(S \cap T) \subseteq (S' \cap T')$*. Also, with probability* $1 - 1/m^i$ *it holds that* $S' = T' = (S \cap T)$.

The total communication in the protocol is $O\left(i \cdot (|S| + |T|) \log(|S| + |T|)\right)$ *and the protocol can be executed in 4 rounds.*

Proof. The parties first exchange the sizes of their sets $|S|$ and $|T|$ and determine $m = |S| + |T|$. Using shared randomness they pick a random hash function $h\colon [n] \to [t]$, where $t = \Theta(m^{i+2})$. They exchange sets $h(S)$ and $h(T)$ using total communication $O(i \cdot m \log m)$. The outcome of the protocol is $\Pi_A(S,T) = h^{-1}(h(T)) \cap S$ and $\Pi_B(S,T) = h^{-1}(h(S)) \cap T$. Since exchanging the sizes of the sets takes two rounds and another two rounds are required to exchange $h(S)$ and $h(T)$, the total number of rounds of communication is 4.

By construction we have $S' = h^{-1}(h(T)) \cap S \subseteq S$ and similarly $T' \subseteq T$ so the first property holds. If $S \cap T = \emptyset$ then $S' \cap T' = (h^{-1}(h(T)) \cap S) \cap (h^{-1}(h(S)) \cap T) \subseteq (S \cap T) = \emptyset$ and the second property holds. Because $S \subseteq h^{-1}(h(S))$ and $T \subseteq h^{-1}(h(T))$ we have $S \cap T \subseteq (h^{-1}(h(T)) \cap S) \cap (h^{-1}(h(S)) \cap T) = S' \cap T'$, the first part of the third property. Moreover, if the hash function h has no collisions among $S \cup T$ then $S' = h^{-1}(h(T)) \cap S = T \cap S$ and $T' = h^{-1}(h(S)) \cap T = S \cap T$. The proof is completed using the analysis of collision probability given by Fact 8.1.2. □

Corollary 8.2.2. *If for the protocol* Π *for* Basic-Set-Intersection *in Lemma 8.2.1 it holds that* $\Pi_A(S,T) = \Pi_B(S,T)$ *then* $\Pi_A(S,T) = \Pi_B(S,T) = S \cap T$.

In our main protocol in Section 8.2.2 we will use an EQ^n test with the following guarantees to verify correctness of the protocol Basic-Set-Intersection. The following guarantee is achieved by a protocol, which uses a random hash function h into k bits.

Fact 8.2.3. *There exists a randomized (with shared randomness) protocol* Π *for* EQ^n *with the following properties. If* $x = y$ *then* $\Pi_A(x,y) = \Pi_B(x,y) = 1$ *with probability 1. If* $x \neq y$ *then* $\Pi_A(x,y) = \Pi_B(x,y) = 0$ *with probability at least* $1 - 1/2^k$*. The total communication in the protocol is* $O(k)$ *and it can be executed in two rounds.*

8.2.2 Main protocol

We will use the following definition of the iterated logarithm functions $\operatorname{ilog}^i z$. Let $\operatorname{ilog}^0 z = z$ and for an integer $i \geq 1$ let $\operatorname{ilog}^i z = \log\left(\operatorname{ilog}^{i-1} z\right)$.

Theorem 8.2.4. *For every integer* $r > 0$ *there exists an* $O(r)$*-round constructive randomized communication protocol (with shared randomness) for* INT^k *with total expected communication* $O(k \operatorname{ilog}^r k)$ *and success probability* $1 - 1/poly(k)$.

Proof. For $r = 1$ the parties use shared randomness to pick a hash function $h\colon [n] \to [m]$ for $m = k^c$, where $c > 2$. Then each of the parties uses $ck \log k$ bits to exchange $h(S)$ and $h(T)$ respectively. By Fact 8.1.2 the probability that h has a collision on a set $S \cup T$ is at most $1 - 1/\Theta(k^{c-2})$.

For $r > 1$ consider a tree $\mathcal{T}$ of depth r with the set of nodes at the i-th level for $0 \le i \le r$ denoted as L_i (these are the nodes at distance i from the leaves). Let the degree at the i-th level for $2 \le i \le r$ be equal to $d_i = \operatorname{ilog}^{r-i} k / \operatorname{ilog}^{r-i+1} k$ and the degree at the first level is $d_1 = \operatorname{ilog}^{r-1} k$. Note that this guarantees that the total number of leaves in the tree is k. For a node $v \in \mathcal{T}$, let $c(v)$ denote the set of children of v. For a node $v \in \mathcal{T}$, let $\mathcal{C}(v)$ denote the set of all leaves in the subtree of v. Note that for a node $v \in L_i$ the number of such leaves is $|\mathcal{C}(v)| = \operatorname{ilog}^{r-i} k$.

Definition 8.2.1 (Set assignment)**.** *A set assignment $\mathcal{A}$ to the leaves of $\mathcal{T}$ is a vector $\mathcal{A} = (\mathcal{A}_1, \dots, \mathcal{A}_k)$, consisting of k sets. We say that the set $\mathcal{A}_\ell$ is assigned to a corresponding leaf ℓ in $\mathcal{T}$.*

Definition 8.2.2 (Induced assignment)**.** *Let $\mathcal{A} = (\mathcal{A}_1, \dots, \mathcal{A}_k)$ be a set assignment for the leaves of $\mathcal{T}$. For every internal node $v \in \mathcal{T}$ denote an assignment induced at this vertex by $\mathcal{A}$ as $\mathcal{A}_v = \cup_{i \in \mathcal{C}(v)} \mathcal{A}_i$.*

Now we describe the protocol used by the parties. First, Alice and Bob use shared randomness to pick a hash function $h\colon [n] \to [k]$. Using this hash function they define initial assignments of sets S^{-1} and T^{-1} respectively as follows. For a leaf $\ell \in [k]$ of $\mathcal{T}$, let $S^{-1}_\ell = h^{-1}(h(\ell)) \cap S$ and $T^{-1}_\ell = h^{-1}(h(\ell)) \cap T$.

Then the protocol proceeds in r stages. In stage i for $0 \le i < r$ the parties construct new assignments to the leaves of $\mathcal{T}$, which induce new assignments on the internal nodes. We will show that after r stages the parties obtain an assignment to the leaves, such that with high probability the set induced by this assignment in the root of $\mathcal{T}$ is exactly $S \cap T$. We use notation S^i and T^i respectively for the i-th assignment that the parties make to the leaves of the tree. The description of the i-th stage is given as Algorithm 8.1. This completes the description of the protocol.

Algorithm 8.1 Protocol for INT^k. Stage i.

```
Input: Inputs of the parties $S, T \in [k]^k$, assignments $S^{i-1}, T^{i-1}$ from the previous stage.

1: For every $v \in L_i$ run the protocol EQUALITY$(S^{i-1}_v, T^{i-1}_v)$ with success probability $1 - 1/(\operatorname{ilog}^{r-i-1} k)^4$.
2: Let $\mathcal{F}$ be the set of vertices for which the equality protocol above returns $S^{i-1}_v \neq T^{i-1}_v$.
   We call these vertices *failed*.
3: For every $v \in \mathcal{F}$ and every leaf $u \in \mathcal{C}(v)$ run BASIC-SET-INTERSECTION$(S^{i-1}_u, T^{i-1}_u)$
   with success probability $1 - 1/(\operatorname{ilog}^{r-i-1} k)^4$ and assign $S^i_u = \Pi_A(S^{i-1}_u, T^{i-1}_u)$ and
   $T^i_u = \Pi_B(S^{i-1}_u, T^{i-1}_u)$ respectively.
4: For every $v \notin \mathcal{F}$ and every leaf $u \in \mathcal{C}(v)$ assign $S^i_u = S^{i-1}_u$ and $T^i_u = T^{i-1}_u$.
```

In the rest of the proof we first analyze the correctness probability of the protocol above (the key lemma is Lemma 8.2.5) and then total communication (Lemma A.10.1).

The proof of Theorem 8.2.4 is completed by observing that the protocol can be executed in $O(r)$ rounds.

Lemma 8.2.5. *After stage i for every leaf $u \in \mathcal{T}$ it holds that $S_u^i = T_u^i$ with probability at least $1 - 1/(\text{ ilog}^{r-i-1} k)^4$, taken over all the randomness of the protocol.*

Proof. If u is in the subtree of a node v, which is not *failed* at level i then we know that $S_v = T_v$ and thus $S_u = T_u$ for each $u \in \mathcal{C}(v)$ with probability at least $1 - 1/(\text{ ilog}^{r-i-1} k)^4$ by the guarantee of the EQUALITY(S_v, T_v) test. Otherwise, u is in the subtree of a failed node v at level i. In this case the claim follows because we run BASIC-SET-INTERSECTION protocol for this leaf with success probability at least $1 - 1/(\text{ ilog}^{r-i-1} k)^4$. □

We call a node $v \in L_i$ *correct* if after stage i it holds that $S_v^i = T_v^i$.

Corollary 8.2.6. *Every node $v \in L_i$ is correct with probability at least $1 - 1/(\text{ ilog}^{r-i-1} k)^3$. In particular, the root of the tree is correct with probability at least $1 - 1/k^3$.*

Proof. From Lemma 8.2.5 applied to the level i it follows that after the execution of stage i for every leaf $u \in \mathcal{C}(v)$ it holds that $S_u^i = T_u^i$ with probability at least $1 - 1/(\text{ ilog}^{r-i-1} k)^4$. Hence, by a union bound over all $\text{ilog}^{r-i} k$ such leaves with probability at least $1 - \text{ ilog}^{r-i} k/(\text{ ilog}^{r-i-1} k)^4 \geq 1 - 1/(\text{ ilog}^{r-i-1} k)^3$ we have $S_v^i = T_v^i$. □

The correctness proof of the protocol now follows from Corollary 8.2.6 together with the following invariant applied to the root of the tree after round $r - 1$.

Proposition 8.2.7. *If for a node $v \in \mathcal{T}$ Alice and Bob assign S_v^i and T_v^i to it respectively then if $S_v^i = T_v^i$ then $S_v^i = T_v^i = S_v \cap T_v$.*

Proof. Note that this invariant is maintained by BASIC-SET-INTERSECTION (Corollary 8.2.2). During the execution of the protocol the sets S_v' and T_v' only change when we apply BASIC-SET-INTERSECTION to the leaves in $\mathcal{T}$. Clearly, if the invariant is maintained for all leaves then it is also maintained for all internal nodes as well. □

In Appendix A.10 we show that the total expected communication in the protocol is $O(k \text{ ilog}^r k)$. Finally, the bound on the number of rounds of communication follows from the fact the communication in each of the r stages for the EQUALITY tests can be done in parallel in two rounds (Fact 8.2.3). After in four more rounds we can perform all BASIC-SET-INTERSECTION protocols in parallel (Lemma 8.2.1). This gives $6r$ rounds of communication. □

Bibliography

[1] Dimitris Achlioptas. Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4):671 – 687, 2003.

[2] W. Ackermann. Zum Hilbertshen aufbau der reelen zahlen. *Math. Ann.*, 99:118–133, 1928.

[3] Pankaj K. Agarwal, Graham Cormode, Zengfeng Huang, Jeff M. Phillips, Zhewei Wei, and Ke Yi. Mergeable summaries. In *PODS*, pages 23–34, 2012.

[4] Nir Ailon and Bernard Chazelle. Information theory in property testing and monotonicity testing in higher dimension. *Inf. Comput.*, 204(11):1704–1717, 2006.

[5] Nir Ailon, Bernard Chazelle, Seshadhri Comandur, and Ding Liu. Property-preserving data reconstruction. *Algorithmica*, 51(2):160–182, 2008.

[6] Noga Alon. Perturbed identity matrices have high rank: Proof and applications. *Combinatorics, Probability & Computing*, 18(1-2):3–15, 2009.

[7] Noga Alon, Phillip B. Gibbons, Yossi Matias, and Mario Szegedy. Tracking join and self-join sizes in limited storage. *J. Comput. Syst. Sci.*, 64(3):719–747, 2002.

[8] Noga Alon, Shlomo Hoory, and Nathan Linial. The Moore Bound for Irregular Graphs. *Graphs and Combinatorics*, 18:53–57, 2002.

[9] Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

[10] Noga Alon and Baruch Schieber. Optimal preprocessing for answering on-line product queries. Technical Report 71/87, Tel-Aviv University, 1987.

[11] Ingo Althöfer, Gautam Das, David Dobkin, Deborah Joseph, and José Soares. On sparse spanners of weighted graphs. *Discrete & Computational Geometry*, 9(1):81–100, 1993.

[12] Mikhail J. Atallah, Marina Blanton, and Keith B. Frikken. Key management for non-tree access hierarchies. In *SACMAT*, pages 11–18, 2006.

[13] Mikhail J. Atallah, Keith B. Frikken, and Marina Blanton. Dynamic and efficient key management for access hierarchies. In *ACM Conference on Computer and Communications Security*, pages 190–202, 2005.

[14] Pranjal Awasthi, Madhav Jha, Marco Molinaro, and Sofya Raskhodnikova. Testing Lipschitz functions on hypergrid domains. In *APPROX-RANDOM*, pages 387–398, 2012.

[15] Baruch Awerbuch. Communication-time trade-offs in network synchronization. In *PODC*, pages 272–276, 1985.

[16] Baruch Awerbuch. Communication-time trade-offs in network synchronization. In Michael A. Malcolm and H. Raymond Strong, editors, *PODC*, pages 272–276. ACM, 1985.

[17] Khanh Do Ba, Piotr Indyk, Eric Price, and David P. Woodruff. Lower bounds for sparse recovery. In *SODA*, pages 1190–1197, 2010.

[18] Ashwinkumar Badanidiyuru, Shahar Dobzinski, Hu Fu, Robert Kleinberg, Noam Nisan, and Tim Roughgarden. Sketching valuation functions. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '12, pages 1025–1035. SIAM, 2012.

[19] Vineet Bafna, Piotr Berman, and Toshihiro Fujito. A 2-approximation algorithm for the undirected feedback vertex set problem. *SIAM J. Discrete Math.*, 12(3):289–297, 1999.

[20] Maria-Florina Balcan, Florin Constantin, Satoru Iwata, and Lei Wang. Learning valuation functions. In Shie Mannor, Nathan Srebro, and Robert C. Williamson, editors, *COLT*, volume 23 of *JMLR Proceedings*, pages 4.1–4.24. JMLR.org, 2012.

[21] Maria-Florina Balcan and Nicholas J. A. Harvey. Learning submodular functions. In *STOC*, pages 793–802, 2011.

[22] Reuven Bar-Yehuda, Dan Geiger, Joseph (Seffi) Naor, and Ron M. Roth. Approximation algorithms for the vertex feedback set problem with applications to constraint satisfaction and bayesian inference. In *SODA'94*, pages 344–354, Philadelphia, 1994. SIAM.

[23] Ziv Bar-Yossef, T. S. Jayram, Robert Krauthgamer, and Ravi Kumar. The sketching complexity of pattern matching. In *APPROX-RANDOM*, pages 261–272, 2004.

[24] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, and D. Sivakumar. An information statistics approach to data stream and communication complexity. *J. Comput. Syst. Sci.*, 68(4):702–732, 2004.

[25] Boaz Barak, Mark Braverman, Xi Chen, and Anup Rao. How to compress interactive communication. In *STOC*, pages 67–76, 2010.

[26] Surender Baswana and Sandeep Sen. Approximate distance oracles for unweighted graphs in expected $\tilde{O}(n^2)$ time. *ACM Transactions on Algorithms*, 2(4):557–577, 2006.

[27] Mohammadhossein Bateni, Mohammadtaghi Hajiaghayi, and Dániel Marx. Approximation schemes for Steiner forest on planar graphs and graphs of bounded treewidth. *J. ACM*, 58:21:1–21:37, October 2011.

[28] Paul Beame. A switching lemma primer. In *Unpublished notes: http://www.cs.washington.edu/ homes/beame/papers/primer.ps*, 1994.

[29] Ann Becker and Dan Geiger. Optimization of Pearl's Method of Conditioning and Greedy-Like Approximation Algorithms for the Vertex Feedback Set Problem. *Artif. Intell.*, 83(1):167–188, 1996.

[30] Piotr Berman, Arnab Bhattacharyya, Elena Grigorescu, Sofya Raskhodnikova, David P. Woodruff, and Grigory Yaroslavtsev. Steiner transitive-closure spanners of low-dimensional posets. In *ICALP (1)*, pages 760–772, 2011.

[31] Piotr Berman, Arnab Bhattacharyya, Konstantin Makarychev, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Approximation algorithms for spanner problems and directed Steiner forest. *Inf. Comput.*, 222:93–107, 2013.

[32] Piotr Berman, Sofya Raskhodnikova, and Ge Ruan. Finding sparser directed spanners. In *FSTTCS*, pages 424–435, 2010.

[33] Piotr Berman and Grigory Yaroslavtsev. Primal-dual approximation algorithms for node-weighted network design in planar graphs. In *APPROX-RANDOM*, pages 50–60, 2012.

[34] Arnab Bhattacharyya, Elena Grigorescu, Madhav Jha, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Lower bounds for local monotonicity reconstruction from transitive-closure spanners. *SIAM J. Discrete Math.*, 26(2):618–646, 2012.

[35] Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. *SIAM J. Comput.*, 41(6):1380–1425, 2012.

[36] Arnab Bhattacharyya, Elena Grigorescu, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Transitive-closure spanners. *SIAM J. Comput.*, 41(6):1380–1425, 2012.

[37] Eric Blais, Joshua Brody, and Kevin Matulef. Property testing lower bounds via communication complexity. *Computational Complexity*, 21(2):311–358, 2012.

[38] Eric Blais, Sofya Raskhodnikova, and Grigory Yaroslavtsev. Lower bounds for testing properties of functions on hypergrid domains. *Electronic Colloquium on Computational Complexity (ECCC)*, 20:36, 2013.

[39] Hanls L. Bodlaender, Gerard Tel, and Nicola Santoro. Tradeoffs in non-reversing diameter. *Nordic J. Comput.*, 1(1):111 – 134, 1994.

[40] Stéphane Boucheron, Gábor Lugosi, and Pacal Massart. On concentration of self-bounding functions. *Electron. J. Probab.*, 14:1884–1899, 2009.

[41] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. A sharp concentration inequality with application. *Random Struct. Algorithms*, 16:277–292, May 2000.

[42] Mark Braverman, Ankit Garg, Denis Pankratov, and Omri Weinstein. From information to exact communication. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:171, 2012.

[43] Mark Braverman and Anup Rao. Information equals amortized communication. In *FOCS*, pages 748–757, 2011.

[44] Joshua Brody, Amit Chakrabarti, and Ranganath Kondapally. Certifying equality with limited interaction. *Electronic Colloquium on Computational Complexity (ECCC)*, 19:153, 2012.

[45] Joshua Brody, Kevin Matulef, and Chenggang Wu. Lower bounds for testing computability by small width OBDDs. In Mitsunori Ogihara and Jun Tarui, editors, *Theory and Applications of Models of Computation*, volume 6648 of *LNCS*, pages 320–331. Springer, 2011.

[46] Amit Chakrabarti, Yaoyun Shi, Anthony Wirth, and Andrew Chi-Chih Yao. Informational complexity and the direct sum problem for simultaneous message complexity. In *FOCS*, pages 270–278, 2001.

[47] Deeparnab Chakrabarty and Zhiyi Huang. Testing coverage functions. In *ICALP (1)*, pages 170–181, 2012.

[48] Deeparnab Chakrabarty and C. Seshadhri. A o(n) monotonicity tester for boolean functions over the hypercube. In *STOC*, pages 411–418, 2013.

[49] Deeparnab Chakrabarty and C. Seshadhri. Optimal bounds for monotonicity and Lipschitz testing over hypercubes and hypergrids. In *STOC*, pages 419–428, 2013.

[50] Ashok K. Chandra, Steven Fortune, and Richard J. Lipton. Lower bounds for constant depth circuits for prefix problems. In Josep Díaz, editor, *ICALP*, volume 154 of *Lecture Notes in Computer Science*, pages 109–117. Springer, 1983.

[51] Ashok K. Chandra, Steven Fortune, and Richard J. Lipton. Unbounded fan-in circuits and associative functions. *J. Comput. Syst. Sci.*, 30(2):222–234, 1985.

[52] Moses Charikar, MohammadTaghi Hajiaghayi, and Howard J. Karloff. Improved approximation algorithms for label cover problems. *Algorithmica*, 61(1):190–206, 2011.

[53] Bernard Chazelle. Computing on a free tree via complexity-preserving mappings. *Algorithmica*, 2:337–361, 1987.

[54] Chandra Chekuri, Guy Even, Anupam Gupta, and Danny Segev. Set connectivity problems in undirected graphs and the directed Steiner network problem. *ACM Trans. Algorithms*, 7(2):18, 2011.

[55] Mahdi Cheraghchi, Adam Klivans, Pravesh Kothari, and Homin K. Lee. Submodular functions are noise stable. In *SODA*, pages 1586–1592, 2012.

[56] Mahdi Cheraghchi, Adam Klivans, Pravesh Kothari, and Homin K. Lee. Submodular functions are noise stable. In *SODA*, pages 1586–1592, 2012.

[57] Kenneth L. Clarkson, Elad Hazan, and David P. Woodruff. Sublinear optimization for machine learning. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 449–457, Washington, DC, USA, 2010. IEEE Computer Society.

[58] Kenneth L. Clarkson and David P. Woodruff. Numerical linear algebra in the streaming model. In *STOC*, pages 205–214, 2009.

[59] Edith Cohen. Fast algorithms for constructing t-spanners and paths with stretch t. *SIAM J. Comput.*, 28(1):210–236, 1998.

[60] Edith Cohen. Polylog-time and near-linear work approximation scheme for undirected shortest paths. *J. ACM*, 47(1):132–166, 2000.

[61] Thomas M. Cover and Joy A. Thomas. *Elements of information theory (2 ed.)*. Wiley, 2006.

[62] Lenore Cowen. Compact routing with minimum stretch. *J. Algorithms*, 38(1):170–183, 2001.

[63] Lenore Cowen and Christopher G. Wagner. Compact roundtrip routing in directed networks. *J. Algorithms*, 50(1):79–95, 2004.

[64] Anirban Dasgupta, Ravi Kumar, and D. Sivakumar. Sparse and lopsided set disjointness via information theory. In *APPROX-RANDOM*, pages 517–528, 2012.

[65] Mayur Datar and S. Muthukrishnan. Estimating rarity and similarity over data stream windows. In *ESA*, pages 323–334, 2002.

[66] Alfredo De Santis, Anna Lisa Ferrara, and Barbara Masucci. New constructions for provably-secure time-bound hierarchical key assignment schemes. *Theor. Comput. Sci.*, 407(1-3):213–230, 2008.

[67] Erik Demaine, MohammadTaghi Hajiaghayi, and Philip Klein. Node-weighted Steiner tree and group Steiner tree in planar graphs. In *ICALP'09*.

[68] Erik D. Demaine and MohammadTaghi Hajiaghayi. Bidimensionality: new connections between fpt algorithms and ptass. In *SODA'05*, pages 590–601, Philadelphia, 2005. SIAM.

[69] Bistra Dilkina and Carla Gomes. Solving connected subgraph problems in wildlife conservation. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, volume 6140 of *LNCS*, pages 102–116. Springer Berlin / Heidelberg, 2010.

[70] Michael Dinitz, Guy Kortsarz, and Ran Raz. Label cover instances with large girth and the hardness of approximating basic k-spanner. In *ICALP (1)*, pages 290–301, 2012.

[71] Michael Dinitz and Robert Krauthgamer. Directed spanners via flow-based linear programs. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, STOC '11, pages 323–332, New York, NY, USA, 2011. ACM.

[72] Michael Dinitz and Gordon T. Wilfong. iBGP and constrained connectivity. In *APPROX-RANDOM*, pages 122–133, 2012.

[73] Irit Dinur and Samuel Safra. On the hardness of approximating minimum vertex cover. *The Annals of Mathematics*, 162(1):pp. 439–485, 2005.

[74] Kashyap Dixit, Madhav Jha, Sofya Raskhodnikova, and Abhradeep Thakurta. Testing the Lipschitz property over product distributions with applications to data privacy. In *TCC*, pages 418–436, 2013.

[75] Yevgeniy Dodis, Oded Goldreich, Eric Lehman, Sofya Raskhodnikova, Dana Ron, and Alex Samorodnitsky. Improved testing algorithms for monotonicity. In *RANDOM*, pages 97–108, 1999.

[76] Yevgeniy Dodis and Sanjeev Khanna. Designing networks with bounded pairwise distance. In *STOC*, pages 750–759, 1999.

[77] Y.G. Dorfman and G.I. Orlova. Finding the maximal cut in a graph. *Engineering Cybernetics*, pages 502–506, 1972.

[78] Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms.* Cambridge University Press, 2009.

[79] B. Dushnik and E.W. Miller. Concerning similarity transformations of linearly ordered sets. *Bulletin Amer. Math. Soc.*, 46:322–326, 1940.

[80] Ben Dushnik and E. W. Miller. Partially ordered sets. *Amer. J. Math.*, 63:600–610, 1941.

[81] Oya Ekin, Peter L. Hammer, and Uri N. Peled. Horn functions and submodular boolean functions. *Theor. Comput. Sci.*, 175(2):257–270, 1997.

[82] M. Elkin. Computing almost shortest paths. In *PODC*, pages 53–62, 2001.

[83] Michael Elkin and David Peleg. Strong inapproximability of the basic k-spanner problem. In *ICALP*, pages 636–647, 2000.

[84] Michael Elkin and David Peleg. The client-server 2-spanner problem with applications to network design. In *SIROCCO*, pages 117–132, 2001.

[85] Michael Elkin and David Peleg. Approximating k-spanner problems for $k > 2$. *Theor. Comput. Sci.*, 337(1-3):249–277, 2005.

[86] Michael Elkin and David Peleg. The hardness of approximating spanner problems. *Theory Comput. Syst.*, 41(4):691–729, 2007.

[87] F. Ergun, S. Kannan, S. R. Kumar, R. Rubinfeld, and M. Viswanathan. Spot-checkers. *J. Comput. Syst. Sci.*, 60(3):717–751, 2000.

[88] G. Even, J. (Seffi) Naor, B. Schieber, and M. Sudan. Approximating minimum feedback sets and multicuts in directed graphs. *Algorithmica*, 20:151–174, 1998.

[89] Guy Even, Joseph Naor, and Leonid Zosin. An 8-approximation algorithm for the subset feedback vertex set problem. *SIAM J. Comput.*, 30(4):1231–1252, 2000.

[90] Tomás Feder, Eyal Kushilevitz, Moni Naor, and Noam Nisan. Amortized communication complexity. *SIAM J. Comput.*, 24(4):736–750, 1995.

[91] Uriel Feige. A threshold of ln n for approximating set cover. *J. ACM*, 45(4):634–652, July 1998.

[92] Joan Feigenbaum, Sampath Kannan, Andrew McGregor, Siddharth Suri, and Jian Zhang. Graph distances in the data-stream model. *SIAM J. Comput.*, 38(5):1709–1727, 2008.

[93] Moran Feldman, Guy Kortsarz, and Zeev Nutov. Improved approximating algorithms for Directed Steiner Forest. In *SODA*, pages 922–931, 2009.

[94] Eldar Fischer. On the strength of comparisons in property testing. *Information and Computation*, 189(1):107–116, 2004.

[95] Eldar Fischer, Eric Lehman, Ilan Newman, Sofya Raskhodnikova, Ronitt Rubinfeld, and Alex Samorodnitsky. Monotonicity testing over general poset domains. In *STOC*, pages 474–483, 2002.

[96] Michael L. Fredman, János Komlós, and Endre Szemerédi. Storing a sparse table with 0(1) worst case access time. *J. ACM*, 31(3):538–544, 1984.

[97] Z. Füredi and J. Kahn. On the dimension of ordered sets of bounded degree. *Order*, 3:15–20, 1986.

[98] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness.* W. H. Freeman & Co., New York, 1990.

[99] Naveen Garg, Vijay V. Vazirani, and Mihalis Yannakakis. Approximate max-flow min-(multi)cut theorems and their applications. *SIAM J. Comput.*, 25:235–251, February 1996.

[100] Michel X. Goemans, Nicholas J. A. Harvey, Satoru Iwata, and Vahab S. Mirrokni. Approximating submodular functions everywhere. In *SODA*, pages 535–544, 2009.

[101] Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM J. Comput.*, 24(2):296–317, 1995.

[102] Michel X. Goemans and David P. Williamson. Primal-dual approximation algorithms for feedback problems in planar graphs. *Combinatorica*, 18:37–59, 1998.

[103] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

[104] Oded Goldreich, editor. *Property Testing: Current Research and Surveys*, volume 6390 of *Lecture Notes in Computer Science*. Springer, 2010.

[105] Oded Goldreich, Shafi Goldwasser, Eric Lehman, Dana Ron, and Alex Samorodnitsky. Testing monotonicity. *Combinatorica*, 20(3):301–337, 2000.

[106] Oded Goldreich, Shafi Goldwasser, and Dana Ron. Property testing and its connection to learning and approximation. *J. ACM*, 45(4):653–750, 1998.

[107] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32, 1989.

[108] Parikshit Gopalan, Adam Tauman Kalai, and Adam R. Klivans. Agnostically learning decision trees. In Cynthia Dwork, editor, *STOC*, pages 527–536. ACM, 2008.

[109] Martin Grötschel, Lászlό Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization.* Springer : Berlin [u.a.], 1988.

[110] Sudipto Guha, Anna Moss, Joseph Naor, and Baruch Schieber. Efficient recovery from power outage (extended abstract). In *STOC'99*, pages 574–582, 1999.

[111] Anupam Gupta, Moritz Hardt, Aaron Roth, and Jonathan Ullman. Privately releasing conjunctions and the statistical query barrier. In Lance Fortnow and Salil P. Vadhan, editors, *STOC*, pages 803–812. ACM, 2011.

[112] M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, and B. Reed. *Probabilistic methods for algorithmic discrete mathematics*, volume 16. Springer Verlag, 1998.

[113] Frank Hadlock. Finding a maximum cut of a planar graph in polynomial time. *SIAM J. Comput.*, 4(3):221–225, 1975.

[114] Prahladh Harsha, Rahul Jain, David A. McAllester, and Jaikumar Radhakrishnan. The communication complexity of correlation. *IEEE Transactions on Information Theory*, 56(1):438–449, 2010.

[115] Johan Håstad. Almost optimal lower bounds for small depth circuits. In *STOC*, pages 6–20, 1986.

[116] Johan Håstad and Avi Wigderson. The randomized communication complexity of set disjointness. *Theory of Computing*, 3(1):211–219, 2007.

[117] William Hesse. Directed graphs requiring large numbers of shortcuts. In *SODA*, pages 665–669, 2003.

[118] T. Hiraguchi. On the dimension of partially ordered sets. *Science Reports Kanazawa University*, 1:77–94, 1951.

[119] Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, May 2006.

[120] Piotr Indyk. Sketching, streaming and sublinear-space algorithms. 2007. Graduate course notes, available at `http://stellar.mit.edu/S/course/6/fa07/6.895/`.

[121] Rahul Jain. New strong direct product results in communication complexity. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:24, 2011.

[122] Rahul Jain, Attila Pereszlényi, and Penghui Yao. A direct product theorem for the two-party bounded-round public-coin communication complexity. In *FOCS*, pages 167–176, 2012.

[123] Rahul Jain, Jaikumar Radhakrishnan, and Pranab Sen. A direct sum theorem in communication complexity via message compression. In *ICALP*, pages 300–315, 2003.

[124] Rahul Jain, Pranab Sen, and Jaikumar Radhakrishnan. Optimal direct sum and privacy trade-off results for quantum and classical communication complexity. *CoRR*, abs/0807.1267, 2008.

[125] T. S. Jayram. Information complexity: a tutorial. In *PODS*, pages 159–168, 2010.

[126] T. S. Jayram and David P. Woodruff. Optimal bounds for Johnson-Lindenstrauss transforms and streaming problems with sub-constant error. In Dana Randall, editor, *SODA*, pages 1–10. SIAM, 2011.

[127] Madhav Jha and Sofya Raskhodnikova. Testing and reconstruction of Lipschitz functions with applications to data privacy. In Ostrovsky [159], pages 433–442. Full version available at http://eccc.hpi-web.de/report/2011/057/.

[128] Madhav Jha and Sofya Raskhodnikova. Testing and reconstruction of lipschitz functions with applications to data privacy. *SIAM J. Comput.*, 42(2):700–731, 2013.

[129] Hossein Jowhari, Mert Saglam, and Gábor Tardos. Tight bounds for Lp samplers, finding duplicates in streams, and related problems. In *PODS*, pages 49–58, 2011.

[130] Andrew B. Kahng, Shailesh Vaya, and Alexander Zelikovsky. New graph bipartizations for double-exposure, bright field alternating phase-shift mask layout. In *ASP-DAC'01*, pages 133–138, New York, 2001. ACM.

[131] Bala Kalyanasundaram and Georg Schnitger. The probabilistic communication complexity of set intersection. *SIAM J. Discrete Math.*, 5(4):545–557, 1992.

[132] Daniel M. Kane, Jelani Nelson, and David P. Woodruff. On the exact space complexity of sketching and streaming small norms. In *SODA*, pages 1161–1178, 2010.

[133] David Kelly. On the dimension of partially ordered sets. *Discrete Mathematics*, 35(1-3):135–156, 1981.

[134] Subhash Khot and Oded Regev. Vertex cover might be hard to approximate to within 2-epsilon. *J. Comput. Syst. Sci.*, 74(3):335–349, 2008.

[135] Hartmut Klauck. A strong direct product theorem for Disjointness. In *STOC*, pages 77–86, 2010.

[136] Philip Klein. *Flatworlds: Optimization Algorithms for Planar Graphs.*

[137] Philip N. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted Steiner trees. *J. Algorithms*, 19(1):104–115, 1995.

[138] Guy Kortsarz. On the hardness of approximating spanners. *Algorithmica*, 30(3):432–450, 2001.

[139] Guy Kortsarz and David Peleg. Generating sparse 2-spanners. *J. Algorithms*, 17:222–236, 1994.

[140] Guy Kortsarz and David Peleg. Generating low-degree 2-spanners. *SIAM J. Comput.*, 27(5):1438–1456, 1998.

[141] Ilan Kremer, Noam Nisan, and Dana Ron. On randomized one-round communication complexity. *Computational Complexity*, pages 21–49, 1999.

[142] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the Fourier spectrum. *SIAM J. Comput.*, 22(6):1331–1348, 1993.

[143] Eyal Kushilevitz and Noam Nisan. *Communication complexity.* Cambridge University Press, 1997.

[144] John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219 – 230, 1980.

[145] Xianyue Li, Xiao-Hua Xu, Feng Zou, Hongwei Du, Pengjun Wan, Yuexuan Wang, and Weili Wu. A PTAS for node-weighted Steiner tree in unit disk graphs. In *Combinatorial Optimization and Applications*, volume 5573 of *LNCS*, pages 36–48. Springer Berlin / Heidelberg, 2009.

[146] Laszlo Lovasz. Submodular functions and convexity. In *Mathematical Programming and the State of the Art*, pages 233–257, 1982.

[147] Yishay Mansour. An $O(n^{\log\log n})$ learning algorithm for DNF under the uniform distribution. *J. Comput. Syst. Sci.*, 50(3):543–550, 1995.

[148] Jiri Matousek. On variants of the Johnson-Lindenstrauss lemma. *Random Structures and Algorithms*, 33(2):142–156, 2008.

[149] A. McGregor. Data streams and linear sketches. 2007. STOC Workshop presentation, available at `http://people.cs.umass.edu/~mcgregor/stocworkshop/mcgregor.pdf`.

[150] Peter Bro Miltersen, Noam Nisan, Shmuel Safra, and Avi Wigderson. On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998.

[151] Carsten Moldenhauer. Primal-dual approximation algorithms for node-weighted Steiner forest on planar graphs. In *ICALP'11*, pages 748–759, 2011.

[152] M. Molinaro, D.P. Woodruff, and G. Yaroslavtsev. Beating the direct sum theorem in communication complexity with implications for sketching. In *Proc. 24th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2013.

[153] Morteza Monemizadeh and David P. Woodruff. 1-pass relative-error l_p-sampling with applications. In *SODA*, pages 1143–1160, 2010.

[154] Anna Moss and Yuval Rabani. Approximation algorithms for constrained node weighted steiner tree problems. *SIAM J. Comput.*, 37(2):460–481, 2007.

[155] S. Muthukrishnan. Data streams: algorithms and applications. *Found. Trends Theor. Comput. Sci.*, 1(2):117–236, August 2005.

[156] Ilan Newman. Private vs. common random bits in communication complexity. *Inf. Process. Lett.*, 39(2):67–71, 1991.

[157] Noam Nisan and Ilya Segal. The communication requirements of efficient allocations and supporting prices. *J. Economic Theory*, 129(1):192–224, 2006.

[158] Ryan O'Donnell. *Analysis of Boolean Functions (http://analysisofbooleanfunctions.org)*. 2012.

[159] Rafail Ostrovsky, editor. *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*. IEEE, 2011.

[160] Rasmus Pagh. Compressed matrix multiplication. In *ITCS*, pages 442–451, 2012.

[161] Itzhak Parnafes, Ran Raz, and Avi Wigderson. Direct product results and the gcd problem, in old and new communication models. In *STOC*, pages 363–372, 1997.

[162] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. On testing convexity and submodularity. *SIAM J. Comput.*, 32(5):1158–1184, 2003.

[163] Michal Parnas, Dana Ron, and Ronitt Rubinfeld. On testing convexity and submodularity. *SIAM J. Comput.*, 32(5):1158–1184, 2003.

[164] Michal Parnas, Dana Ron, and Alex Samorodnitsky. Testing basic boolean formulae. *SIAM J. Discrete Math.*, 16(1):20–46, 2002.

[165] David Peleg and Alejandro A. Schäffer. Graph spanners. *Journal of Graph Theory*, 13(1):99–116, 1989.

[166] David Peleg and Jeffrey D. Ullman. An optimal synchronizer for the hypercube. *SIAM J. Comput.*, 18(4):740–747, 1989.

[167] David Peleg and Eli Upfal. A trade-off between space and efficiency for routing tables. *J. ACM*, 36(3):510–530, 1989.

[168] Luis Rademacher and Santosh Vempala. Testing geometric convexity. In *FSTTCS*, pages 469–480, 2004.

[169] Sofya Raskhodnikova. Approximate testing of visual properties. In *RANDOM-APPROX*, pages 370–381, 2003.

[170] Sofya Raskhodnikova. Transitive-closure spanners: A survey. In Goldreich [104], pages 167–196.

[171] Sofya Raskhodnikova and Grigory Yaroslavtsev. Learning pseudo-Boolean k-DNF and submodular functions. In *SODA*, pages 1356–1368, 2013.

[172] Ran Raz and Avi Wigderson. Monotone circuits for matching require linear depth. *J. ACM*, 39(3):736–744, 1992.

[173] Alexander A. Razborov. On the distributional complexity of disjointness. *Theor. Comput. Sci.*, 106(2):385–390, 1992.

[174] Jan Remy and Angelika Steger. Approximation schemes for node-weighted geometric Steiner tree problems. In *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques*, volume 3624 of *LNCS*, pages 642–642. Springer Berlin / Heidelberg, 2005.

[175] Liam Roditty, Mikkel Thorup, and Uri Zwick. Roundtrip spanners and roundtrip routing in directed graphs. In *SODA*, pages 844–851, 2002.

[176] Dana Ron. Algorithmic and analysis techniques in property testing. *Foundations and Trends in Theoretical Computer Science*, 5(2):73–205, 2009.

[177] Ronitt Rubinfeld and Asaf Shapira. Sublinear time algorithms. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:13, 2011.

[178] Ronitt Rubinfeld and Madhu Sudan. Robust characterization of polynomials with applications to program testing. *SIAM Journal on Computing*, 25(2):252–271, 1996.

[179] Mert Saglam. Personal communication, 2013.

[180] Mert Saglam and Gábor Tardos. On the communication complexity of sparse set disjointness and exists-equal problems. *CoRR*, abs/1304.1217, 2013.

[181] Michael E. Saks and C. Seshadhri. Local monotonicity reconstruction. *SIAM J. Comput.*, 39(7):2897–2926, 2010.

[182] Alfredo De Santis, Anna Lisa Ferrara, and Barbara Masucci. Efficient provably-secure hierarchical key assignment schemes. In *MFCS*, pages 371–382, 2007.

[183] Tamás Sarlós. Improved approximation algorithms for large matrices via random projections. In *FOCS*, pages 143–152, 2006.

[184] C. Seshadhri. Question 2: Testing submodularity. In P. Indyk, A. McGregor, I. Newman, and K. Onak, editors, *Open Problems in Data Streams, Property Testing, and related topics, Bertinoro Workshop on Sublinear Algorithms (May 2011) and IITK Workshop on Algorithms for Processing Massive Data Sets (December 2009)*, page 3, 2011. Downloaded July 2, 2012 from `http://people.cs.umass.edu/~mcgregor/papers/11-openproblems.pdf`.

[185] C. Seshadhri and Jan Vondrák. Is submodularity testable? In *ICS*, pages 195–210, 2011.

[186] Ronen Shaltiel. Towards proving strong direct product theorems. *Computational Complexity*, 12(1-2):1–22, 2003.

[187] Mikkel Thorup. On shortcutting digraphs. In *WG*, pages 205–211, 1992.

[188] Mikkel Thorup. Shortcutting planar digraphs. *Combinatorics, Probability & Computing*, 4:287–315, 1995.

[189] Mikkel Thorup. Parallel shortcutting of rooted trees. *J. Algorithms*, 23(1):139–159, 1997.

[190] Mikkel Thorup and Uri Zwick. Compact routing schemes. In *ACM Symposium on Parallel Algorithms and Architectures*, pages 1–10, 2001.

[191] Mikkel Thorup and Uri Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, 2005.

[192] William T. Trotter and J.I. Moore. The dimension of planar posets. *Journal of Combinatorial Theory, Series B*, 22:54–67, 1977.

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

[194] Jan Vondrák. A note on concentration of submodular functions. *CoRR*, abs/1005.2791, 2010.

[195] Mihalis Yannakakis. Node-and edge-deletion np-complete problems. In *STOC'78*, pages 253–264, New York, 1978. ACM.

[196] Mihalis Yannakakis. The complexity of the partial order dimension problem. 3(3):351–358, 1982.

[197] Andrew Chi-Chih Yao. Some complexity questions related to distributive computing (preliminary report). In *STOC*, pages 209–213, 1979.

[198] Andrew Chi-Chih Yao. Space-time tradeoff for answering range queries (extended abstract). In *ACM STOC*, pages 128–136, 1982.

Appendix A

Appendix

A.1 Concentration results

We use the following standard concentration result on Poisson random variables, giving the proof for completeness.

Lemma A.1.1. *Let P be a Poisson random variable with parameter $\lambda \geq 1$. Then*

$$\Pr(P < \lfloor \lambda/e \rfloor) \leq e^{-\lambda/4};$$

$$\Pr(P > e\lambda) \leq e^{-\lambda}.$$

Proof. Let $T = \lfloor \lambda/e \rfloor$. Then

$$0P < \lambda/e = \sum_{t=0}^{T-1} \frac{\lambda^t e^{-\lambda}}{t!}.$$

The terms in the sum increase exponentially: $\frac{\lambda^{t+1}}{(t+1)!} \Big/ \frac{\lambda^t}{(t)!} = \frac{\lambda}{t+1} \geq e$. Hence,

$$\begin{aligned} 0P < \lambda/e &\leq \frac{\lambda^T e^{-\lambda}}{T!} \sum_{t=0}^{\infty} e^{-t} \leq \frac{\lambda^T e^{-\lambda}}{\sqrt{2\pi}(T/e)^T} \times 2 \leq e^{-(\lambda-T)} \left(\frac{\lambda}{T}\right)^T \\ &\leq e^{-(\lambda-T)} \left(\frac{\lambda}{\lambda/e}\right)^{\lambda/e} \leq e^{-\lambda(1-2/e)} \leq e^{-\lambda/4}. \end{aligned}$$

To estimate $T!$ we use Stirling's approximation.

Similarly, let $T' = \lceil e\lambda \rceil$. Then

$$0P > e\lambda = \sum_{t=T'}^{\infty} \frac{\lambda^t e^{-\lambda}}{t!}.$$

The terms in the sum decrease exponentially: $\frac{\lambda^{t+1}}{(t+1)!} \Big/ \frac{\lambda^t}{(t)!} = \frac{\lambda}{t+1} \leq 1/e$. Hence,

$$0P > e\lambda \leq \frac{2\lambda^{T'}e^{-\lambda}}{T'!} \leq \frac{2\lambda^{T'}e^{-\lambda}}{\sqrt{2\pi}(T'/e)^{T'}} \leq e^{-\lambda}\left(\frac{e\lambda}{T'}\right)^{T'} \leq e^{-\lambda}.$$

□

A.2 Omitted proofs

A.2.1 Lemma A.2.1

Lemma A.2.1. *Assume that (V, E) is a connected triangulated planar graph, f is a proper function and $\Gamma \subset V$ has these properties: (a) $f(A) = 0$ for every $A \subseteq \Gamma$ and (b) $f(B) = 1$ for some B that is a connected component of $V \setminus \Gamma$. Then there exists $C \subseteq \Gamma$ with properies (a) and (b) such that C is a simple cycle.*

Proof. We will show the existence of C by taking any minimal subset of Γ, which has properties (a) and (b). The proof of the lemma follows from the Claim A.2.4, Claim A.2.5 and Claim A.2.6.

Claim A.2.2. *There exist two active connected components in $V \setminus C$.*

Proof. Let B_0 be a connected component of $V \setminus \Gamma$ such that $f(B_0) = 1$ which exists by property (b). By symmetry of f, we have $f(V \setminus B_0) = 1$, and by disjointness of f, we have $f(V \setminus B_0 \setminus \Gamma) = 1$. By applying disjointness again, there exists a connected component B_1 of $V \setminus B_0 \setminus \Gamma$ such that $f(B_1) = 1$. □

We will denote two active components, whose existence is guaranteed by Claim A.2.2 as B_0 and B_1.

Claim A.2.3. *Each $c \in C$ is adjacent to both B_0 and B_1.*

Proof. Otherwise, suppose that c is not adjacent to B_i and let $C' = C \setminus \{c\}$. Then B_i is a connected component of $V \setminus C'$, hence C' has property (b). Because C' is a subset of C it has property (a) and we get a contradiction with minimiality of C. □

Claim A.2.4. *Each $c \in C$ has at most 2 neighbors in C.*

Proof. Otherwise, let c_1, c_2, c_3 be three distinct nodes of C that are neighbors of c. Contract B_0 and B_1 to a single node, this allows to define a minor with nodes in two groups: c, B_0, B_1 in one group, c_1, c_2, c_3 in the second group, and this minor contains $K_{3,3}$. □

Claim A.2.5. *Each $c \in C$ has at least 2 neighbors in C, or C forms a cycle of length 2.*

Proof. In triangulated planar graphs neighbors of a node form a (not necessarily simple) cycle. Consider the cycle of neighbors of $c \in C$. On this cycle we can traverse a group of nodes from B_0, say $b_0^1, \ldots, b_0^k$. Because the cycle contains also nodes from other connected components of $V \setminus C$, this groups must be preceeded and followed by a node from C, say c_0 and c_1. If $c_0 \neq c_1$, Claim A.2.5 is true. Otherwise (c, c_0) and (c, c_1) are two separate edges from c to the same node, and the cycle $C' = (c, c_0 = c_1, c)$ has in its interior the group of nodes of B_0 that we have discussed and no other neighbors of c. Thus the nodes from B_1 that are neighbors of c must be in the exterior or C'; consequently there are no nodes of C in the interior of C', as they would violate Claim A.2.3. In the same way we can argue that the are no nodes of C in the exterior of C' and thus we conclude that $C = C'$. □

Claim A.2.6. *The subgraph C is connected.*

Proof. This is obvious when C is a 2-cycle, so it remains to consider the case when each node in C has exactly 2 neighbors in C. If C is not connected it forms a set of disjoint simple cycles and nodes of different cycles are not adjacent. Each such cycle is adjacent to exactly two connected components of $V \setminus C$ and we can consider a cycle C' with the minimal interior I. If $f(I) = 1$ then C' satisfies property (b), which is a contradiction with the minimality of C. If $f(I) = 0$ then nodes of C' don't satisfy Claim A.2.3, because they are adjacent to at most one active connected component. □

□

A.2.2 Analysis of the generic local-ratio algorithm

Here we give the proof of Theorem 3.2.1

Proof. When Algorithm 3.1 returns the hitting set H we have $\bar{w}(H) = 0$. Thus the cost of this hitting set $w(H)$ is the sum of decreases of $\bar{w}(H)$ that are caused by Step 7, and the same applies to an optimum solution H^*. To show approximation ratio γ it suffices to show that anytime we decrease $\bar{w}(H)$ by some γx we also decrease $\bar{w}(H^*)$ by at least x.

The decrease of $\bar{w}(H^*)$ is $\alpha c_{\mathcal{M}}(H^*)$. We can estimate $c_{\mathcal{M}}(H^*)$ it as follows: if $u \in H^*$ is responsible for hitting some m cycles in $\mathcal{M}$ then $c_{\mathcal{M}}(u) \geq m$, thus $c_{\mathcal{M}}(H^*) \geq |\mathcal{M}|$. Thus the decrease of $\bar{w}(H^*)$ is at least $\alpha|\mathcal{M}|$.

The decrease of $\bar{w}(H)$ is $\alpha c_{\mathcal{M}}(H)$. Thus to show the approximation ratio γ it suffices to show that for **every** minimal hitting set $\breve{H}$ we have $c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|$. □

A.2.3 Uncrossing proper sets (Lemma 3.1.2)

Here we give the proof of Lemma 3.1.2.

Proof. For a set $S \subseteq V$ we use notation $\bar{S}$ to denote $V \setminus S$, the complement of S in V. We will show that if the second statement is false then the first one is true. There are two cases.

Case 1. Suppose for the sake of contradiction that $f(A_1) = 0$. Then because $f(A) = 1$ we have $f(A \setminus A_1) = 1$ by disjointness.

By symmetry we have $f(B) = f(\bar{B}) = 1$. Because $A_1 \subseteq \bar{B}$ and $f(A_1) = 0$ by disjointness we have $f(\bar{B} \setminus A_1) = 1$. By symmetry this implies that $f(\overline{\bar{B} \setminus A_1}) = f(A_1 \cup B) = 1$, because $\overline{\bar{B} \setminus A_1} = B \cup A_1$.

Case 2. Suppose for the sake of contradiction that $f(B\setminus(A\setminus A_1)) = f((A\setminus A_1)\setminus B) = 0$.

Note that $A \setminus ((A \setminus A_1) \setminus B) = A_1 \cup (A \cap B)$. From $f((A \setminus A_1) \setminus B) = 0$ and $f(A) = 1$ we thus conclude by disjointness that $f(A \setminus ((A \setminus A_1) \setminus B)) = f(A_1 \cup (A \cap B)) = 1$. Because

$A_1 \cap B = \emptyset$ the sets $B \setminus (A \setminus A_1)$ and $A_1 \cup (A \cap B)$ are disjoint. Using this together with the assumption that $f(B \setminus (A \setminus A_1)) = 0$ by disjointess we have that $f(B \setminus (A \setminus A_1) \cup A_1 \cup (A \cap B)) = f(A_1 \cup B) = 1$.

From the assumption that $f(B \setminus (A \setminus A_1)) = 0$ and $f(B) = 1$ by disjointness we have $f(B \setminus (B \setminus (A \setminus A_1))) = f(A \cap B) = 1$. Together with $f((A \setminus A_1) \setminus B) = 0$ by complementarity we have $f(A \setminus A_1) = 1$. □

A.3 Proof of 12/5 approximation ratio with triple pocket oracle

Theorem A.3.1. *In every iteration of the generic local-ratio algorithm (Algorithm 3.1) with oracle* Minimal-3-Pocket-Violation *for every minimal hitting set $\breve{H}$ of $\mathcal{C}$ we have $c_{\mathcal{M}}(\breve{H}) \leq \gamma|\mathcal{M}|$ for $\gamma = 12/5$.*

Proof. We change the proof of Theorem 3.3.1 in only a few aspects.

We need to consider a triple pocket, with contact nodes c, d, e. The crossing witness cycles contain paths between contact nodes, now we have 3 pairs of contact nodes rather than one, hence 3 possible families of crossing paths. Thus we need to change the analysis of subpockets. A subpocket defined by two crossing paths from the same family has two contact nodes, as before. However, one subpocket can be defined by crossing paths from different families and this subpocket may have 3 contact nodes.

The subpocket with 3 contact nodes can have a negative balance if it contains less than two faces in $\mathcal{M}$. Thus it is important that if we have such subpocket we must also have nother subpockets that bring the total number of faces in $\mathcal{M}$ to at least 3, and those subpockets have positive balance.

If the subpocket with 3 contact nodes contains exactly 1 face in $\mathcal{M}$ its balance is at least $1 - 3/\gamma$, while the balance of the other pockets is at least $2(1 - 2/\gamma)$, for the total of $3 - 7/\gamma = 1/12$. If the subpocket with 3 contact nodes contains exactly 2 faces in $\mathcal{M}$, its balance is at least $2 - 5/\gamma$ while the other pockets have balance at least $1 - 2/\gamma$ and the estimate of the total is the same.

The effects of pruning are estimated exactly as before.

The balance of envelopes is estimated similarly, but we can make assumptions based on the fact that an envelope cannot be used to define a triple pocket. For a non-outer envelope the critical cases are when $n_{\mathbf{S}} \leq 4$. The case when $n_{\mathbf{S}} \leq 2$ is excluded because we would define a pocket. The case when $n_{\mathbf{S}} = 3$ and $\ell_{\mathbf{S}} = 0$ is excluded because we would define a triple pocket. If $n_{\mathbf{S}} = 3$ and $\ell_{\mathbf{S}} \geq 1$ the balance is at least $(7/2 - 9/\gamma) + 3/\gamma - 1 = 5/2 - 6/\gamma = 5/2 - 5/2 = 0$. If $n_{\mathbf{S}} \geq 4$ then the balance $\frac{3}{2}n_{\mathbf{S}} - 1 - 3n_{\mathbf{S}}/\gamma = 5 - 12/\gamma = 0$.

The estimate of the balance of an outer envelope is similar to an estimate of a non-outer envelope, so we just need to examine the differences to see that it cannot be lower. We can increase the credit given to an outer envelope by 2, because of Euler formula applied to the connected component of the debit graph that contains that envelope: one can see that each outer envelope is in a separate component.

The impact of an outer node on the balance can occur in two ways. We can replace the status of a hit node on the envelope from non-outer to outer. This decreases the credits by 1, and debits by $1/\gamma$, hence we subtract $7/12$ from the balance. We can also add an outer hit node that is not a contact node, this does not change credits but adds $1/\gamma$ to the debit, hence we subtract $5/12$ from the balance. Thus the worst case is that the balance increases by $2 - 3 \times 7/12 = 1/4$. □

A.4 Converting a learner into a proper learner

Let $\mathcal{C}$ be a class of discrete objects represented by functions over a domain of "size" n.

Proposition A.4.1. *If there exists a learning algorithm L for a class $\mathcal{C}$ with query complexity $q(n,\epsilon)$ and running time $t(n,\epsilon)$, then there exists a proper learning algorithm L' for $\mathcal{C}$ with query complexity $q(n,\epsilon/2)$ and running time $t(n,\epsilon/2)+|\mathcal{C}|$.*

Proof. Given parameters n,ϵ and oracle access to a function f, the algorithm L' first runs L with parameters $n,\epsilon/2$ to obtain a hypothesis g. Then it finds and outputs a function $h \in \mathcal{C}$, which is closest to g, namely $h = argmin_{h' \in \mathcal{C}} dist(g,h')$. By our assumption that L is a learning algorithm, $dist(f,g) \le \epsilon/2$. Since $f \in \mathcal{C}$, we have $dist(g,h) \le dist(g,f) \le \epsilon/2$. By the triangle inequality, $dist(f,h) \le dist(f,g) + dist(g,h) \le \epsilon$. □

A.5 Information Cost When Amplifying Success Probability

Consider a function $f : \mathcal{X}\times\mathcal{Y} \to \mathbb{Z}$ and let λ be a distribution over $\mathcal{X}\times\mathcal{Y}\times\mathcal{D}$, with marginals μ over $\mathcal{X}\times\mathcal{Y}$ and ν over $\mathcal{D}$. We show that $\mathrm{IC}_{\mu,\delta^{\Omega(r)}}(f|\nu) \le r\,\mathrm{IC}_{\mu,\delta}(f|\nu)$. For that, take a δ-protocol Π for f which achieves $\mathrm{I}(\Pi; X, Y \mid D) = \mathrm{IC}_{\mu,\delta}(f|\nu)$, where $(X,Y,D) \sim \lambda$. Then let $\bar\Pi$ be the protocol on input (x,y) that runs r copies $\Pi(x,y,R_1), \Pi(x,y,R_2), \ldots, \Pi(x,y,R_r)$ with independent coins $R_1, R_2, \ldots, R_r$ and outputs the value obtained by the majority of the runs.

It is easy to see that $\bar\Pi$ outputs the correct answer with probability at least $1-\delta^{\Omega(r)}$. Moreover, by the chain rule for mutual information, we have

$$\begin{aligned}\mathrm{IC}_{\mu,\delta^{\Omega(r)}}(f|\nu) \le \mathrm{I}(\bar\Pi; X, Y \mid D) = \\ \sum_{i=1}^{r} \mathrm{I}\left[\Pi(X,Y,R_i); X, Y \mid D, \Pi(X,Y,R_1), \ldots, \Pi(X,Y,R_{i-1})\right].\end{aligned} \tag{A.1}$$

But we can expand the i-th term as

$$\begin{aligned}
&\mathrm{H}\left[\Pi(X,Y,R_i) \mid D, \Pi(X,Y,R_1), \ldots, \Pi(X,Y,R_{i-1})\right] \\
&\quad - \mathrm{H}\left[\Pi(X,Y,R_i) \mid D, \Pi(X,Y,R_1), \ldots, \Pi(X,Y,R_{i-1}), X, Y\right] \\
&\le \mathrm{H}\left[\Pi(X,Y,R_i) \mid D\right] \\
&\quad - \mathrm{H}\left[\Pi(X,Y,R_i) \mid D, \Pi(X,Y,R_1), \ldots, \Pi(X,Y,R_{i-1}), X, Y\right] \\
&= \mathrm{H}[\Pi(X,Y,R_i) \mid D] - \mathrm{H}[\Pi(X,Y,R_i) \mid D, X, Y] \\
&= \mathrm{I}[\Pi(X,Y,R_i); X, Y \mid D] = \mathrm{IC}_{\mu,\delta}(f|\nu),
\end{aligned}$$

where the first equality follows from the fact that, since the R_j's are independent, then conditioned on (X, Y) we have $\Pi(X, Y, R_i)$ independent from $\Pi(X, Y, R_1) \dots, \Pi(X, Y, R_{i-1})$. Plugging this bound on equation (A.1) gives that $\mathrm{IC}_{\mu,\delta^{\Omega(r)}}(f|\nu) \le r\,\mathrm{IC}_{\mu,\delta}(f|\nu)$.

A.6 Auxiliary Results for Lower Bounding Applications

Before proving the lower bound for our applications, we need to spell out some (standard) tools. In the next two subsections, we introduce the hard communication problem from there the lower bounds will come from. This hard problem is essentially based on constructing the n-fold version of augmented indexing and then doing an extra indexing over it. In the following subsection, we present, for completeness, an encoding of augmented indexing into vectors whose inner product depends whether the instance is yes/no; this was already present in the proof of Lemma 3.1 of [126].

A.6.1 Generic Indexing problems

A generic indexing problem can be defined as follows. Consider a function $f : \mathcal{X} \times \mathcal{Y} \to \mathcal{Z}$ and the associated (one-way) communication problem where Alice and Bob get respectively an element of $\mathcal{X}$ and $\mathcal{Y}$ and want to compute the value of the f over this pair; we use f to also denote this problem. Let $\mathrm{Ind}(f, N)$ denote the communication problem where Alice has input $x_1, x_2, \dots x_N \in \mathcal{X}$ and Bob has input $j \in [N]$, $x_1, x_2, \dots, x_{j-1} \in X$ and $y \in \mathcal{Y}$, and they want to compute $f(x_j, y)$. To simplify the notation, let $\tilde{\mathcal{X}} = \mathcal{X}^N$ denote the space of Alice's input, let $\tilde{\mathcal{Y}} = \bigcup_{i=0}^{N-1} (\mathbb{N} \times \mathcal{X}^i \times \mathcal{Y})$ denote the space of Bob's input.

It is folklore that the information complexity of an indexing problem $\mathrm{Ind}(f, N)$ is typically $\Omega(N)$ times the complexity of the base problem of computing f.

Lemma A.6.1. *Let λ be a probability distribution over $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ with marginal μ on $\mathcal{X} \times \mathcal{Y}$ and marginal ν on $\mathcal{D}$, such that μ is partitioned by ν. Then there is a distribution $\tilde{\lambda}$ over $\tilde{\mathcal{X}} \times \tilde{\mathcal{Y}} \times \tilde{\mathcal{D}}$ (where $\tilde{\mathcal{D}} = \mathbb{N} \times \mathcal{D}^N$) with the following property. Let $\tilde{\mu}$ denote the marginal of $\tilde{\lambda}$ on $\tilde{\mathcal{X}} \times \tilde{\mathcal{Y}}$ and let $\tilde{\nu}$ denote the marginal of $\tilde{\lambda}$ on $\tilde{\mathcal{D}}$. Then for all $\delta \in [0, 1]$*

$$\mathrm{IC}^{\rightarrow}_{\tilde{\mu},\delta}(\mathit{Ind}(f, N)|\tilde{\nu}) \ge N \cdot \mathrm{IC}^{\rightarrow}_{\mu,\delta}(f|\nu).$$

Moreover, $\tilde{\nu}$ partitions $\tilde{\mu}$.

To make the presentation self-contained, in the remaining part of this section we prove the above lemma. For that, we start by constructing the distribution $\tilde{\lambda}$. First let J be the uniform random variable over $[N]$, and, to makes things formal, let $D_0 = J$. Now let the random variables $\mathbf{X} = (X_1, X_2, \dots, X_N)$, Y and $\mathbf{D} = (D_1, D_2, \dots, D_N)$ have the following distribution: conditioned on $J = j$, we have $(X_j, Y, D_j) \sim \lambda$ and (X_i, D_i) distributed according to the marginal of λ on $\mathcal{X} \times \mathcal{D}$ for all $i \neq j$ (so the conditioning on J only specifies which variables will be correlated with Y). The distribution $\tilde{\lambda}$ is then defined as the distribution of the random variable $(\mathbf{X}, (J, \mathbf{X}_{<J}, Y), (D_0, \mathbf{D}))$. It is easy to see that $\tilde{\nu}$ partitions $\tilde{\mu}$.

Recall the notation for one-way protocols used in Section 7.3. Consider a private-randomness one-way δ-protocol (M, B) for $\mathrm{Ind}(f, N)$ (with Alice's and Bob's private coins respectively denoted by R^A and R^B) and attains $\mathrm{IC}^{\to}_{\tilde{\mu},\delta}(\mathrm{Ind}(f,N)|\tilde{\nu})$; that is, for the random variables above, we have $\mathrm{I}(M(\mathbf{X},R^A);\mathbf{X} \mid D_0\mathbf{D}) = \mathrm{IC}^{\to}_{\tilde{\mu},\delta}(\mathrm{Ind}(f,N)|\tilde{\nu})$. We start lower bounding the left-hand side.

First notice that the random variable $(\mathbf{X}, R^A, \mathbf{D})$ is independent of D_0. Therefore, we have

$$\begin{aligned}
&\mathrm{I}(M(\mathbf{X},R^A);\mathbf{X} \mid D_0\mathbf{D}) = \mathrm{I}(M(\mathbf{X},R^A);\mathbf{X} \mid \mathbf{D})\\
&= \sum_{j=1}^{N} \mathrm{I}(M(\mathbf{X},R^A);X_j \mid \mathbf{D},\mathbf{X}_{<j})\\
&= \sum_{j=1}^{N} \mathrm{I}(M(\mathbf{X},R^A);X_j \mid \mathbf{D}_{\geq j},\mathbf{X}_{<j})\\
&= \sum_{j=1}^{N} \sum_{\mathbf{d}_{>j},\mathbf{x}_{<j}} \mathrm{I}(M(\mathbf{X},R^A);X_j \mid D_j,\mathbf{D}_{>j}=\mathbf{d}_{>j},\mathbf{X}_{<j}=\mathbf{x}_{<j})\cdot\\
&\qquad\qquad \Pr(\mathbf{D}_{>j}=\mathbf{d}_{>j},\mathbf{X}_{<j}=\mathbf{x}_{<j})\\
&= \sum_{j=1}^{N} \sum_{\mathbf{d}_{>j},\mathbf{x}_{<j}} \mathrm{I}(M(\mathbf{x}_{<j}\mathbf{X}_{\geq j},R^A);X_j \mid D_j,\mathbf{D}_{>j}=\mathbf{d}_{>j})\cdot\\
&\qquad\qquad \Pr(\mathbf{D}_{>j}=\mathbf{d}_{>j},\mathbf{X}_{<j}=\mathbf{x}_{<j}), \qquad\qquad (A.2)
\end{aligned}$$

where the second equality follows from the chain rule for conditional mutual information, and the others follows from the product structure of $\mathbf{D}$ and $\mathbf{X}$ and independence from R^A. Now we lower bound each term in this expression using a standard simulation argument.

Claim A.6.2. *For every index $j \in [N]$ and fixing $\mathbf{d}_{>j}$ and $\mathbf{x}_{<j}$, there is a private-randomness one-way protocol $(\bar{M},\bar{B})$ with domain $\mathcal{X}\times\mathcal{Y}$ satisfying the following (where $\bar{R}^A$ and $\bar{R}^B$ denote Alice's and Bob's private coins respectively):*

- *$(\bar{M},\bar{B})$ is a δ-protocol for f.*
- *For the random variable $(\bar{X}_j,\bar{Y},\bar{D}_j)\sim\lambda$, we have:*

$$\mathrm{I}(\bar{M}(\bar{X}_j,\bar{R}^A);\bar{X}_j \mid \bar{D}_j) = \mathrm{I}(M(\mathbf{x}_{<j}\mathbf{X}_{\geq j},R^A);X_j \mid D_j,\mathbf{D}_{>j}=\mathbf{d}_{>j}).$$

Proof. The desired protocol $(\bar{M},\bar{B})$ is the following. Alice uses her private randomness $\bar{R}^A$ to obtain the random variable $\tilde{R}^A$ with the same distribution as R^A, and also the random variable $\tilde{\mathbf{X}}_{>j}$ with the same distribution as the conditioned random variable $\mathbf{X}_{>j} \mid (\mathbf{D}_{>j}=\mathbf{d}_{>j})$; Bob uses his private randomness $\bar{R}^B$ to obtain the random variable $\tilde{R}^B$ with same distribution as R^B. Then for every input $(x,y)\in\mathcal{X}\times\mathcal{Y}$, we set Alice's message to be $\bar{M}(x,\bar{R}^A) = M(\mathbf{x}_{<j}x\tilde{\mathbf{X}}_{>j},\tilde{R}^A)$ and Bob's output upon receiving message m is $\bar{B}(m,y,\bar{R}^B) = B(m,j,\mathbf{x}_{<j},y,\tilde{R}^B)$.

For every input $(x,y)\in\mathcal{X}\times\mathcal{Y}$, we can use the fact (M,B) is a δ-protocol for $\mathrm{Ind}(f,N)$ to obtain that

$$\begin{aligned}
1-\delta &\leq \Pr\left[B(M(\mathbf{x}_{<j}x\tilde{\mathbf{X}}_{>j},R^A),j,\mathbf{x}_{<j},y,R^B) = f(x,y)\right]\\
&= \Pr((\bar{M},\bar{B}) \text{ outputs } f(x,y)),
\end{aligned}$$

where the equality follows from the definition of our random variables. This gives the first part of the claim.

For the second part, let $(\bar{X}_j, \bar{Y}_j, \bar{D}_j) \sim \lambda$. By the definition of our random variables, $(\bar{X}_j \tilde{\mathbf{X}}_{>j}, \bar{D}_j, \tilde{R}^A)$ has the same distribution as $(\mathbf{X}_{\geq j}, D_j, R^A) \mid (\mathbf{D}_{>j} = \mathbf{d}_{>j})$, so by substitution we have

$$\begin{aligned}
&\mathrm{I}(\bar{M}(\bar{X}_j, \bar{R}^A); \bar{X}_j \mid \bar{D}_j) = \mathrm{I}(M(\mathbf{x}_{<j} \bar{X}_j \tilde{\mathbf{X}}_{>j}, \tilde{R}^A); \bar{X}_j \mid \bar{D}_j) \\
&= \mathrm{I}(M(\mathbf{x}_{<j} \mathbf{X}_{\geq j}, R^A); X_j \mid D_j, \mathbf{D}_{>j} = \mathbf{d}_{>j}),
\end{aligned}$$

which concludes the proof of the claim. □ □

Lemma A.6.1 then follows directly from the above claim and equation (A.2).

A.6.2 Encoding of Indexing Over Augmented Set Indexing

In this section we present the following encoding of Ind(SetInd$(\epsilon, \eta), r)$, which was already present in the proof of Lemma 3.1 in [126]. Notice that we consider the problem Ind(SetInd$(\epsilon, \eta), r)$ and not the n-fold problem Ind(nSetInd$(\epsilon, \eta), r)$, but we can use this encoding for each of the n copies present in the latter.

Lemma A.6.3. *Given $\epsilon, \eta \in (0, 1]$, consider subsets $S^1, S^2, \ldots, S^r$ of $[1/(\epsilon^2 \eta)]$, each of size $1/\epsilon^2$ (assumed to be odd). Also consider an index $j \in [r]$ and an element $k \in [1/(\epsilon^2 \eta)]$. Then there is an encoding of these objects, based on a random variable R, into vectors $\mathbf{u} = \mathbf{u}(S^1, S^2, \ldots, S^r, R)$, $\underline{\mathbf{u}} = \underline{\mathbf{u}}(S^1, S^2, \ldots, S^{j-1}, R)$ and $\mathbf{v} = \mathbf{v}(j, k, R)$ with the following properties:*

1. *The vectors $\mathbf{u}, \underline{\mathbf{u}}$ and $\mathbf{v}$ belong to $\{0,1\}^{2t \cdot \frac{10^r - 1}{9}}$, where $t = \frac{72}{\epsilon^2} \log \frac{1}{\eta}$.*
2. *$\|\mathbf{u} - \underline{\mathbf{u}}\|^2 \leq 2 \cdot 10^{r-j} t$ and $\|\mathbf{v}\|^2 = 10^{r-j} t$.*
3. *If k does not belong to the set S^j, then with probability at least $1 - \eta$ we have $\langle \mathbf{u} - \underline{\mathbf{u}}, \mathbf{v} \rangle \leq 10^{r-j} t (\frac{1}{2} + \frac{\epsilon}{12})$.*
4. *If k belongs to the set S^j, then with probability at least $1 - \eta$ we have $\langle \mathbf{u} - \underline{\mathbf{u}}, \mathbf{v} \rangle \geq 10^{r-j} t (\frac{1}{2} + \frac{2\epsilon}{12})$.*

To prove this lemma, we first define and analyze an encoding scheme for the case where we only have one set, i.e., $r = 1$.

So consider a set $S \subseteq [1/(\epsilon^2 \eta)]$. Let $\mathbf{X}$ be a uniform random vector in $\{-1, +1\}^{1/(\epsilon^2 \eta)}$. We define $\mathrm{enc}_1(S, \mathbf{X})$ to be the majority of the set $\{X_i\}_{i \in S}$; this is well-defined since $1/\epsilon^2$ is odd. We contrast this with the encoding $\mathrm{enc}_2(k, \mathbf{X})$ which is just the k-th component of $\mathbf{X}$.

Notice that if $k \notin S$, then the encodings are independent and hence

$$\Pr[\mathrm{enc}_1(S, \mathbf{X}) = \mathrm{enc}_2(k, \mathbf{X})] = \tfrac{1}{2}.$$

On the other hand, suppose $k \in S$. Then, using the fact that $\mathrm{enc}_1(S, \mathbf{X})$ depends on only $1/\epsilon^2$ coordinates of $\mathbf{X}$ (since $|S| = 1/\epsilon^2$), standard arguments involving the binomial coefficients give that

$$\Pr[\mathrm{enc}_1(S,\mathbf{X}) = \mathrm{enc}_2(k,\mathbf{X})] \geq \tfrac{1}{2}(1+\tfrac{\epsilon}{2}).$$

We repeat the above scheme to amplify the gap between the two cases. Let $\mathbb{X} = (\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^t)$ be a collection of $t = \frac{72}{\epsilon^2}\log\frac{1}{\eta}$ uniform i.i.d. random variables in $\{-1,+1\}^{1/(\epsilon^2\eta)}$. Define

$$\mathrm{enc}_1(S,\mathbb{X}) = (\mathrm{enc}_1(S,\mathbf{X}^1), \mathrm{enc}_1(S,\mathbf{X}^2), \ldots, \mathrm{enc}_1(S,\mathbf{X}^t)),$$

and

$$\mathrm{enc}_2(k,\mathbb{X}) = (\mathrm{enc}_2(k,\mathbf{X}^1), \mathrm{enc}_2(k,\mathbf{X}^2), \ldots, \mathrm{enc}_2(k,\mathbf{X}^t)).$$

Fact A.6.4 (Chernoff bounds, [78]). *Let $Y_1, Y_2, \ldots, Y_t$ be a collection of i.i.d. 0-1 Bernoulli random variables with success probability p. Set $\bar{Y} = \sum_{i=1}^t Y_i/t$. Then,*

$$\Pr[\bar{X} < p - h] < \mathit{exp}(-2h^2t), \text{ and}$$
$$\Pr[\bar{X} > p + h] < \mathit{exp}(-2h^2t).$$

In the above fact with $t = \frac{72}{\epsilon^2}\log\frac{1}{\eta}$ and $h = \epsilon/12$, we obtain that the tail probabilities are at most η. In the case $k \notin S$ we use $p = \frac{1}{2}$ to get

$$\Pr[\#(\mathrm{enc}_1(S,\mathbb{X}), \mathrm{enc}_2(k,\mathbb{X})) > t(\tfrac{1}{2} + \tfrac{\epsilon}{12})] \leq \eta, \tag{A.3}$$

where # denotes the number of coordinates where the vectors agree. In the case $k \in S$ we use $p = \frac{1}{2}(1+\frac{\epsilon}{2})$ to get

$$\Pr[\#(\mathrm{enc}_1(S,\mathbb{X}), \mathrm{enc}_2(k,\mathbb{X})) < t(\tfrac{1}{2} + \tfrac{2\epsilon}{12})] \leq \eta. \tag{A.4}$$

Finally, we convert the ± 1 vector $\mathrm{enc}_1(S,\mathbb{X})$ (resp. $\mathrm{enc}_2(k,\mathbb{X})$) into the 0/1 vector $\mathrm{enc}_1'(S,\mathbb{X})$ (resp. $\mathrm{enc}_2'(k,\mathbb{X})$) by replacing the occurrence of each 1 by the pattern 01 and the occurrence of each -1 by 10; so the new vectors have exactly twice as many coordinates as the original ones. Moreover, $\langle \mathrm{enc}_1'(S,\mathbb{X}), \mathrm{enc}_2'(k,\mathbb{X})\rangle = \#(\mathrm{enc}_1(S,\mathbb{X}), \mathrm{enc}_2(k,\mathbb{X}))$ and $\|\mathrm{enc}_1'(S,\mathbb{X})\| = \|\mathrm{enc}_2'(k,\mathbb{X}))\| = \sqrt{t}$. Setting $\mathbf{u} = \mathrm{enc}_1'(S,\mathbb{X})$, $\underline{\mathbf{u}} = 0$ and $\mathbf{v} = \mathrm{enc}_2'(k,\mathbb{X}))$ gives the desired encoding for the case where we have only one set S.

Now we adapt this encoding to handle multiple sets. For each $i \in [r]$, define the vector $\mathbf{u}^i \in \{0,1\}^{10^{r-i}2t}$ by appending 10^{r-i} copies of $\mathrm{enc}_1'(S^i,\mathbb{X})$. Then define the vector $\mathbf{u}$ by appending the vectors $\mathbf{u}^i$ for $i = 1,2,\ldots,r$. Also define the vector $\underline{\mathbf{u}}$ by appending the vectors $\mathbf{u}^i$ for $i = 1,2,\ldots,j-1$ and the appending 0's to obtain a vector with the same number of coordinates as $\mathbf{u}$.

Now for each $i \in [r]$ define the vector $\mathbf{v}^i$ to be equal to $\mathbf{0} \in \{0,1\}^{10^{r-i}2t}$ if $i \neq j$, and to be equal to 10^{r-j} copies of $\mathrm{enc}_2'(k,\mathbb{X})$ otherwise. Then define $\mathbf{v}$ by appending the vectors $\mathbf{v}^i$ for $i = 1,2,\ldots,r$.

It is easy to see that $\mathbf{u}$, $\underline{\mathbf{u}}$ and $\mathbf{v}$ have the desired properties. First, notice that these vectors have exactly $2t\sum_{i=1}^{r} 10^{r-i} = 2t \cdot \frac{10^r-1}{9}$ coordinates. Also,

$$\|\mathbf{u}-\underline{\mathbf{u}}\|^2 = \sum_{i=j}^{r} \|\mathbf{u}^i\|^2 = \sum_{i=j}^{r} 10^{r-i}t \leq 2 \cdot 10^{r-j}t$$

and $\|\mathbf{v}\|^2 = 10^{r-j}t$. Moreover,

$$\langle \mathbf{u}-\underline{\mathbf{u}}, \mathbf{v}\rangle = \langle \mathbf{u}^j, \mathbf{v}^j\rangle = 10^{r-j}\langle \mathrm{enc}_1'(S^j, \mathbb{X}), \mathrm{enc}_2'(k, \mathbb{X})\rangle.$$

Equations (A.3) and (A.4) conclude the proof of Lemma A.6.3.

A.7 Proof for Other Applications

A.7.1 Proof of Theorem 7.4.10

The proof follows the same line as the proof of Theorem 4.3 in [126], where we use a JL transform to provide a solution for the $(n/2)$-fold ℓ_2 estimation.

Consider an instance of $\ell_2(n/2, d, 1, 4\epsilon)$ where Alice has vectors $\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^{n/2} \in \{-1,0,1\}^d$ and Bob has vectors $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^{n/2} \in \{-1,0,1\}^d$. We consider the following shared-randomness protocol for this problem. Let $(\mathcal{F}, \mu)$ be a $\mathrm{JLT}(\epsilon, \delta, n, d)$ transform with dimension k as small as possible. The players use their shared randomness to agree upon a matrix S sampled from $\mathcal{F}$ according to μ. Then Alice computes $S\mathbf{u}^1, S\mathbf{u}^2, \ldots, S\mathbf{u}^{n/2}$ and stops if $\|S\mathbf{u}^i\|^2 > (1+\epsilon)\|\mathbf{u}^i\|^2$ for some i; otherwise, she rounds each entry of these vectors to the nearest additive multiple of $\epsilon/\sqrt{k}$ and sends the rounded vectors $\{\tilde{\mathbf{u}}^i\}_i$ to Bob. Bob then computes $S\mathbf{v}^1, S\mathbf{v}^2, \ldots, S\mathbf{v}^{n/2}$ and rounds their entries just as Alice did to obtain the vectors $\{\tilde{\mathbf{v}}^i\}_i$. Finally Bob outputs $\|\tilde{\mathbf{u}}^i - \tilde{\mathbf{v}}^i\|$ for each $i \in [n/2]$.

By the triangle inequality,

$$\|\tilde{\mathbf{u}}^i - \tilde{\mathbf{v}}^i\| = \|S\mathbf{u}^i - S\mathbf{v}^i\| \pm \left(\|\tilde{\mathbf{u}}^i - S\mathbf{u}^i\| + \|\tilde{\mathbf{v}}^i - S\mathbf{v}^i\|\right),$$

or using the definition of $\tilde{\mathbf{u}}^i$ and $\tilde{\mathbf{v}}^i$, $\|\tilde{\mathbf{u}}^i - \tilde{\mathbf{v}}^i\| = \|S\mathbf{u}^i - S\mathbf{v}^i\| \pm \epsilon$. But notice that whenever $\mathbf{u}^i = \mathbf{v}^i$, we have exactly $\|\tilde{\mathbf{u}}^i - \tilde{\mathbf{v}}^i\| = \|S\mathbf{u}^i - S\mathbf{v}^i\| = 0$, and $\mathbf{u}^i \neq \mathbf{v}^i$ implies $\|\mathbf{u}^i - \mathbf{v}^i\| \geq 1$ (because of the discrete domain $\{-1,0,1\}^d$). This then gives that

$$\|\tilde{\mathbf{u}}^i - \tilde{\mathbf{v}}^i\| = (1 \pm \epsilon)\|S\mathbf{u}^i - S\mathbf{v}^i\|. \tag{A.5}$$

Now suppose $\|S(\mathbf{u}^i - \mathbf{v}^i)\| = (1 \pm 3\epsilon)\|\mathbf{u}^i - \mathbf{v}^i\|$ for all i and also $\|S\mathbf{u}^i\|^2 = (1\pm\epsilon)\|\mathbf{u}^i\|^2$ for all i, which happens with probability at least $1-2\delta$. In this case, it follows directly from equation (A.5) that Bob outputs the desired estimate $(1 \pm 4\epsilon)\|\mathbf{u}^i - \mathbf{v}^i\|$ for all i. Moreover, Alice does not send too many bits: because the input vectors have entries in $\{-1,0,1\}$, $\|S\mathbf{u}^i\|^2 = (1\pm\epsilon)\|\mathbf{u}^i\|^2 \leq 2d$ and so every entry of $S\mathbf{u}^i$ (in absolute value) is upper bounded by $2d$; it then takes Alice $O\left(nk \log\left(\frac{dk}{\epsilon}\right)\right)$ bits to send all $\tilde{\mathbf{u}}^i$'s.

Therefore, the above protocol solves $\ell_2(n/2, d, 1, 4\epsilon)$ with probability at least $1-2\delta$ and communication $O\left(nk \log\left(\frac{dk}{\epsilon}\right)\right)$, using shared randomness. But using the lower bound that follows from Theorem 7.4.4 and equation (7.4) (and the fact that n is sufficiently large), we get $R_{2\delta}^{\to,\mathrm{pub}}(\ell_2(n/2, d, 1, 4\epsilon)) \geq \Omega\left(n\frac{1}{\epsilon^2}\log\frac{n}{\delta}\log d\right)$. The fact that $k \leq d$ and the assumption that (in particular) $d \geq 1/\epsilon$ give that $k \geq \Omega(\frac{1}{\epsilon^2} \log \frac{n}{\delta})$ as desired.

A.7.2 Proof of Theorem 7.4.11

As in Section 7.4.2, we obtain the lower bound by analyzing the case of small and large alphabet sizes separately, and we also assume without loss of generality that δ is at most a sufficiently small constant.

Lower Bound For Small Alphabet Size. In this section we consider $M = 1$ and obtain the following.

Lemma A.7.1. *Assume that n is at least a sufficiently large constant and that δ and ϵ are at most a sufficiently small constant. Also assume that there is a constant $\gamma > 0$ such that $d^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$. Then $R_\delta^{sketch}(\mathrm{Ip}(n, d, 1, \epsilon)) \geq \Omega(n \frac{1}{\epsilon^2} \log \frac{n}{\delta} \log d)$.*

As in the problem of approximating the ℓ_2 norm, the lower bound is again based on the problem $\mathrm{Ind}(n\mathrm{SetInd}(\epsilon, \delta), \log d)$, but now the main element in the reduction is the encoding provided by Lemma A.6.3.

We show how to use the n-fold inner product problem $\mathrm{Ip}(n, d, 1, \epsilon/25)$ to solve $\mathrm{Ind}(n\mathrm{SetInd}(\epsilon, \delta), c \log d)$, for a constant c depending on γ. As in Section 7.4.2, let Alice's instance for $\mathrm{Ind}(n\mathrm{SetInd}(\epsilon, \delta), c \log d)$ be given by the sets $\{S_i^\ell\}_{i \in [n], \ell \in [c \log d]}$ and let Bob's instance be given by the index $j \in [c \log d]$, the elements $k_1, k_2, \ldots, k_n$, the sets $\{S_i^\ell\}_{i \in [n], \ell < j}$ and the sets $S_1', S_2', \ldots, S_n'$ (although we will ignore the latter sets). They want to decide whether $k_i \in S_i^j$ holds or not for each $i \in [n]$.

To do so, independently for each $i \in [n]$ the players use the reduction from Lemma A.6.3 with success probability $\eta = \delta/n$ and $r = c \log d$ to make Alice obtain the vector $\mathbf{u}_i = \mathbf{u}_i(S_i^1, S_i^2, \ldots, S_i^{c \log d}, R)$ and Bob obtain the vectors $\underline{\mathbf{u}}_i = \underline{\mathbf{u}}_i(S_i^1, S_i^2, \ldots, S_i^{j-1}, R)$ and $\mathbf{v}_i = \mathbf{v}_i(j, k_i, R)$, using their shared randomness to simulate R. Notice that these vectors have at most $O(d^c \frac{1}{\epsilon^2} \log \frac{n}{\delta})$ coordinates, which is at most $O(d)$ for a sufficiently small c depending only on γ. Then the players can use a sketching protocol that solves $\mathrm{Ip}(n, O(d), 1, \epsilon/25)$ with success probability $1 - \delta$ to compute, for all $i \in [n]$, $val_i = \langle \mathbf{u}_i - \underline{\mathbf{u}}_i, \mathbf{v}_i \rangle \pm \frac{\epsilon}{25} \|\mathbf{u}_i - \underline{\mathbf{u}}_i\| \|\mathbf{v}_i\| = \langle \mathbf{u}_i - \underline{\mathbf{u}}_i, \mathbf{v}_i \rangle \pm \frac{\epsilon}{25} 10^{r-j} t$, where $t = \frac{72}{\epsilon^2} \log \frac{n}{\delta}$; they do so by having Alice sending Bob the linear sketches of the vectors $\mathbf{u}_i$'s, then Bob updating these sketches to obtain sketches of the vectors $\{\mathbf{u}_i - \underline{\mathbf{u}}_i\}_i$, and finally executing Bob's part of the protocol to approximate the values $\langle \mathbf{u}_i - \underline{\mathbf{u}}_i, \mathbf{v}_i \rangle$. Having these approximations at hand, for each $i \in [n]$ Bob outputs that $k_i \in S_i^j$ iff $val_i \geq 10^{r-j} t (\frac{1}{2} + \frac{3\epsilon}{24})$.

Since the guarantees from Lemma A.6.3 hold for all triples $(\mathbf{u}_i \underline{\mathbf{u}}_i, \mathbf{v}_i)$ for $i \in [n]$ with probability at least $1 - \delta$, it is easy to see that Bob outputs the correct answer with probability at least $1 - 2\delta$. This implies that:

$$R_\delta^{\mathrm{sketch}}(\mathrm{Ip}(n, O(d), 1, \epsilon/25)) \geq R_{2\delta}^{\rightarrow, \mathrm{pub}}(\mathrm{Ind}(n\mathrm{SetInd}(\epsilon, \delta), c \log d).$$

Corollary 7.4.3, equation (7.4) and the assumption that n is sufficiently large conclude the proof of Lemma A.7.1.

Lower Bound for Small Dimension. In this section we obtain the following bound.

Lemma A.7.2. *Assume that n is at least a sufficiently large constant and that δ and ϵ are at most a sufficiently small constant. Also assume that $d \geq \Omega(\frac{1}{\epsilon^2} \log \frac{n}{\delta})$ and that* *there is a constant* $\gamma > 0$ *such that* $M^{1-\gamma} \geq \frac{1}{\epsilon^2} \log \frac{n}{\delta}$. *Then* $R^{sketch}_{\delta}(\mathrm{Ip}(n, d, M, \epsilon)) \geq \Omega\left(n\frac{1}{\epsilon^2} \log \frac{n}{\delta} \log M\right)$.

As observed, for instance, in [183], JL transforms also approximate inner products.

Proposition A.7.3. *Let* $(\mathcal{F}, \mu)$ *be a JLT*(ϵ, δ, n, d). *Consider* $S \sim \mu$. *Then for every collection of* n *vectors* $\mathbf{u}^1, \mathbf{u}^2, \ldots, \mathbf{u}^n$ *in* $\mathbb{R}^d$, *with probability at least* $1 - \delta$ *we have* $\langle S\mathbf{u}^i, S\mathbf{u}^j \rangle = \langle \mathbf{u}^i, \mathbf{u}^j \rangle \pm \epsilon \|\mathbf{u}^i\| \|\mathbf{u}^j\|$ *for all* $i, j \in [n]$.

Lemma A.7.2 is then obtained from the previous reduction by applying the JL transform of Theorem 7.4.8 to the vectors $\mathbf{u}_i$, $\underline{\mathbf{u}}_i$ and $\mathbf{v}_i$, just as done in the second part of Section 7.4.2.

A.7.3 Proof of Theorem 7.4.12

To prove the first part of the theorem, notice that $(AB)_{i,i} \pm \epsilon \|A_i\| \|B^i\| = \langle A_i, B^i \rangle \pm \epsilon \|A_i\| \|B^i\|$. Therefore, a sketch S that allows (with probability at least $1 - \delta$) the approximation $(AB)_{i,j} \pm \|A_i\| \|B^j\|$ for all $i, j \in [n]$ can be used to solve the inner product problem $\mathrm{Ip}(n, n, M, \epsilon)$ with probability $1 - \delta$ and communication equal to the bit size of AS; the desired lower bound then follows directly from Theorem 7.4.11.

For the second part of the theorem, we can set $B = A^T$ to obtain $(AA^T)_{i,i} \pm \epsilon \|A_i\|^2 = (1 \pm \epsilon) \|A_i\|^2$, and hence the sketch S is a JLT(ϵ, δ, n, n); the lower bound $k \geq \Omega(\frac{1}{\epsilon^2} \log \frac{n}{\delta})$ then follows from Theorem 7.4.10.

A.7.4 Proof of Theorem 7.4.13

The lower bound of $\Omega(n\frac{1}{\epsilon^2} \log \frac{n}{\delta} \log |\mathcal{D}|)$ follows directly from Lemma A.7.1 (notice that the hard instances in this lemma are provided by $\{0, 1\}$ vectors). However, the lower bound $\Omega(n\frac{1}{\epsilon^2} \log \frac{n}{\delta} \log M)$ does not follow directly from Lemma A.7.2, because there the hard instances are given by vectors which can have negative coordinates. The latter lower bound comes from the following modification of the hard instances for inner products.

Lemma A.7.4. *Assume that* n *and* M *are at least a sufficiently large constant and assume that* δ *and* ϵ *are at most a sufficiently small constant. Also assume that* $d \geq n/(\epsilon^2 \delta)$. *Then* $R^{sketch}_{\delta}(\mathrm{Ip}(n, d, M, \epsilon)) \geq \Omega\left(n\frac{1}{\epsilon^2} \log \frac{n}{\delta} \log M\right)$. *Moreover, this holds even if the protocol only offers guarantees for vectors in* $\{0, 1, \ldots, M\}^d$.

Proof. Consider the problem $\mathrm{Ind}(n\mathrm{SetInd}(\epsilon, \delta), \log M)$. Let Alice's instance for this problem be given by the sets $\{S_i^\ell\}_{i \in [n], \ell \in [\log M]}$, and let Bob's instance be given by the index $j \in [\log M]$, the elements $k_1, k_2, \ldots, k_n$, the sets $\{S_i^\ell\}_{i \in [n], \ell < j}$ and the sets $S'_1, S'_2, \ldots, S'_n$. They want to decide whether $k_i \in S_i^j$ holds or not for each i. A trivial but important observation is that $k_i \in S_i^j$ iff the inner product $\langle \chi_{S_i^j}, e_{k_i} \rangle$ equals 1, where $\chi_{S_i^j} \in \{0, 1\}^{n/(\epsilon^2 \delta)}$ is the incidence vector of S_i^j and e_{k_i} is the k_i'th canonical vector.

To solve this problem, for each $i \in [n]$ Alice makes the vector $\mathbf{u}^i \triangleq \sum_{\ell=1}^{\log M} 10^{\log M - \ell} \chi_{S_i^\ell}$, and for every $i \in [n]$ Bob makes the vectors $\underline{\mathbf{u}}^i \triangleq \sum_{\ell=1}^{j-1} 10^{\log M - \ell} \chi_{S_i^\ell}$ and $\mathbf{v}^i \triangleq 10^{\log M - j} e_{k_i}$. Notice that the constructed vectors lie in $\{0, 1, \ldots, M'\}^d$, where $M' = M^{10}$ and $d = n/(\epsilon^2 \delta)$.

Then using the shared randomness, Alice runs a sketching protocol for $\mathrm{Ip}(n, d, M', \epsilon/4)$ to send Bob sketches $\mathcal{S}\mathbf{u}^1, \mathcal{S}\mathbf{u}^2, \ldots, \mathbf{u}^n$ that allows computation of n-fold $(\epsilon/4)$-approximations for dot products with probability at least $1-\delta$. Then Bob updates the sketches to obtain $\mathcal{S}(\mathbf{u}^1 - \underline{\mathbf{u}}^1), \mathcal{S}(\mathbf{u}^2 - \underline{\mathbf{u}}^2), \ldots, \mathcal{S}(\mathbf{u}^n - \underline{\mathbf{u}}^n)$, and use them to compute the inner product approximations $val_i = \langle(\mathbf{u}^i - \underline{\mathbf{u}}^i), \mathbf{v}^i\rangle \pm \frac{\epsilon}{4}\|\mathbf{u}^i - \underline{\mathbf{u}}^i\|\|\mathbf{v}^i\|$ for all $i \in [n]$, with probability at least $1-\delta$. Finally, for each i Bob reports that $k_i \in S_i^j$ iff $val_i \geq 10^{2(\log M - j)}/2$.

We claim that: (i) $\langle(\mathbf{u}^i - \underline{\mathbf{u}}^i), \mathbf{v}^i\rangle = 10^{2(\log M - j)}\langle\chi_{S_i^j}, e_{k_i}\rangle \pm \frac{10^{2(\log M - j)}}{9}$ and (ii) $\|\mathbf{u}^i - \underline{\mathbf{u}}^i\|\|\mathbf{v}^i\| \leq \frac{10^{2(\log M - j)+1}}{9\epsilon}$; since $\langle\chi_{S_i^j}, e_{k_i}\rangle$ equals 1 if $k_i \in S_i^j$ and 0 otherwise, these bounds show that the above protocol solves the instance of $\mathrm{Ind}(n\mathrm{SetInd}(\epsilon), \log M)$ with probability at least $1-\delta$. To prove (i), by bilinearity of inner products we have

$$
\begin{aligned}
\langle(\mathbf{u}^i - \underline{\mathbf{u}}^i), \mathbf{v}^i\rangle &= \sum_{\ell=j}^{\log M} 10^{2\log M - \ell - j}\langle\chi_{S_i^\ell}, e_{k_i}\rangle \\
&= 10^{2(\log M - j)}\langle\chi_{S_i^j}, e_{k_i}\rangle \pm \frac{10^{2(\log M - j)}}{9},
\end{aligned}
$$

where the inequality follows from the fact that $\langle\chi_{S_i^\ell}, e_{k_i}\rangle \leq 1$ for all i, ℓ. To prove (ii), notice that $|S_i^\ell| = 1/\epsilon^2$ and hence $\|\chi_{S_i^\ell}\| = 1/\epsilon$ for all i, ℓ. Using triangle inequality, we obtain that $\|\mathbf{u}^i - \underline{\mathbf{u}}^i\| \leq (1/\epsilon)\sum_{\ell=j}^{\log M} 10^{\log M - \ell} \leq \frac{10^{\log M - j + 1}}{9\epsilon}$. Since $\|\mathbf{v}^i\| = 10^{\log M - j}$, part (ii) directly follows.

The above protocol shows that $R_\delta^{\to,\mathrm{pub}}(\mathrm{Ind}(n\mathrm{SetInd}(\epsilon), \log M)) \leq R_\delta^{\mathrm{sketch}}(\mathrm{Ip}(n, d, M', \epsilon/4))$. Then Corollary 7.4.3 together with equation (7.4) gives that $R_\delta^{\mathrm{sketch}}(\mathrm{Ip}(n, d, M, \epsilon)) \geq \Omega(n\frac{1}{\epsilon^2}\log\frac{n}{\delta}\log M)$ as desired. □ □

A.8 Lower bound

Theorem A.8.1. *There exists a distribution on $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ with marginals μ on $\mathcal{X} \times \mathcal{Y}$ and ν on $\mathcal{D}$, such that ν partitions μ and:*

$$IC^{(r)}_{\mu^k,\delta}(EQ_k^{n/k}|\nu^k) = \Omega(k\ ilog^r k)$$

In the proof we will use the following modification of the strong direct sum theorem of the Theorem 7.2.2. The simulation procedure used in the proof of this theorem in preserves the number of rounds in the protocol, which allows us to state this theorem for bounded round protocols:

Theorem A.8.2 (Strong Direct Sum)**.** *Let $\delta \leq 1/3$. Then for every function $f\colon \mathcal{X}\times\mathcal{Y} \to \mathbb{Z}$ and distribution λ on $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ with marginal μ on $\mathcal{X} \times \mathcal{Y}$ and marginal ν on $\mathcal{D}$, such that μ is partitioned by ν, it holds that $\mathrm{IC}^r_{\mu^k,\delta}(f^k|\nu^k) \geq \Omega(k)\,\mathrm{IC}^r_{\mu,\frac{1}{20},\frac{1}{10},\frac{\delta}{k}}(f|\nu)$.*

Using the direct sum above it remains to show the following:

Lemma A.8.3. *There exists a distribution on $\mathcal{X} \times \mathcal{Y} \times \mathcal{D}$ with marginals μ on $\mathcal{X} \times \mathcal{Y}$ and ν on $\mathcal{D}$, such that ν partitions μ and:*

$$IC^{(r)}_{\mu,1/20,1/10,\delta/k}(EQ^{n/k}|\nu) = \Omega(\ ilog^r k)$$

Proof. In the proof we can use the same hard distribution as that used in [152]. Let $\ell = n/k$. To construct μ and ν, let D_0 be a random variable uniformly distributed on $\{0,1\}$ and let $\mathbf{D}$ be a random variable uniformly distributed on $\{0,1\}^\ell$. Let $(\mathbf{X},\mathbf{Y})$ be a random variable supported on $\{0,1\}^\ell \times \{0,1\}^\ell$ such that, conditioned on $D_0 = 0$ we have $\mathbf{X}$ and $\mathbf{Y}$ distributed independently and uniformly on $\{0,1\}^\ell$, and conditioned on $D_0 = 1$ we have $\mathbf{X} = \mathbf{Y} = \mathbf{D}$. Let μ be the distribution of $(\mathbf{X},\mathbf{Y})$ and let ν be the distribution of $(D_0\mathbf{D})$. Note that ν partitions μ. Also, this distribution satisfies that $\Pr[\mathbf{X} = \mathbf{Y}] \geq 1/3$ and $\Pr[\mathbf{X} \neq \mathbf{Y}] \geq 1/3$.

Let W be a random variable distributed according to ν. Let E be an indicator variable over the private randomness of Π which is equal to 1 if and only if conditioned on this private randomness Π satisfies that it aborts with probability at most $1/10$ and succeeds with probability at least $1-\delta/k$ conditioned on non-abortion. Given such protocol with abortion Π we transform it into a protocol Π' which never aborts, has almost the same information complexity and gives correct output on non-equal instances with high probability, while being correct on equal instances with constant probability. This is done by constructing Π' so that whenever Π outputs "abort", the output of Π' is $X \neq Y$, otherwise $\Pi = \Pi'$. Under the distribution μ conditioned on the event $E = 1$ the protocol Π' has the property that if $X \neq Y$, then it outputs $X = Y$ with probability at most $(1/k)/\Pr_\mu[X \neq Y] \leq 3/k$. However, if $X = Y$, then the protocol may output $X \neq Y$ with probability $1/10 + (1/k)/\Pr_{\mu'}[X = Y] \leq 1/10 + 3/k \leq 1/5$, where the latter follows for $k \geq 30$. Thus, conditioned on $E = 1$, the protocol Π' has failure probability $\epsilon = 1/k$ on non-equal instances $X \neq Y$, and constant failure probability $\delta = 1/5$ on equal instances $X = Y$, as desired. In this regime we can use the following theorem from [44]:

Lemma A.8.4 (Information Complexity Lower Bound for Equality [44])**.** *Let δ be the error on equal instances and ϵ be the error on non-equal instances. Let $\tilde{n} = \min\{n + \log(1-\delta), \log((1-\delta)/\epsilon)\}$. For $\delta \leq 1 - 8(\ ilog^{r-2}\tilde{n})^{-1/8}$ and a uniform distributions over the inputs μ_u it holds that:*

$$\mathrm{IC}^r_{\mu_u,\epsilon,\delta}(\mathrm{EQ}^n) = \Omega((1-\delta)^3\ ilog^{r-1}\tilde{n})$$

We have:

$$\begin{aligned} IC^{(r)}_{\mu,1/20,1/10,\delta/k}(EQ^{n/k}|\nu) &\geq I(\Pi;X,Y|W) = \Omega(I(\Pi;X,Y|W,E=1)) - 1 \\ &= \Omega(I(\Pi';X,Y|W,E=1)) - 2. \end{aligned}$$

Here the inequality is by definition of information compelxity and the equalities follows from Proposition 7.2.1 together with the fact that $H(E) \leq 1$, $\Pr[E = 1] = 19/20$, and the fact that the transcripts of the protocols Π and Π' only differ in a single bit. The right-hand side can be bounded using the following proposition.

Proposition A.8.5.

$$I(\Pi'; X, Y|W, E = 1)) = \Omega(IC^{(r)}_{\mu_u,1/k,1/5}(\mathsf{EQ}^{n/k})).$$

Proof. This follows from the construction of the distributions μ and ν that we use. If $D_0 = 0$ then $\mathbf{X} = \mathbf{Y}$ and the information revealed by Π is equal to zero. Otherwise, if $D_0 = 1$ then the distribution of $(\mathbf{X}, \mathbf{Y})$ is uniform. Because the latter happens with probability 1/2 we have $I(\Pi'; X, Y|W, E = 1)) \geq 1/2 \cdot IC^{(r)}_{\mu_u,1/k,1/5}(\mathsf{EQ}^{n/k}))$ as desired. □

Thus, we have $IC^{(r)}_{\mu,1/20,1/10,\delta/k}(EQ^{n/k}|\nu) = \Omega(IC^{(r)}_{\mu_u,1/k,1/5}(\mathsf{EQ}^{n/k}))$. The proof is completed by noting that setting $\epsilon = 1/k$ and $\delta = 1/5$ in Lemma A.8.4 gives $IC^{(r)}_{\mu_u,1/k/1,5}(\mathsf{EQ}^{n/k}) = \Omega(\text{ ilog}^r k)$. □

A.9 $O(\sqrt{k})$-round protocol with linear communication

Theorem A.9.1. *There exists an $O(\sqrt{k})$-round constructive randomized protocol for* INT^k *with success probability $1 - 1/poly(k)$. In the model of shared randomness the total expected communication is $O(k)$ and in the model of private randomness it is $O(k + \log\log n)$*

Proof. Let $m = k^c$ for a constant $c > 2$. First, the parties pick a random hash function $H: [n] \to [m]$, which gives no collisions on the elements in $S \cup T$ with probability at least $1 - 1/\Omega(k^{c-2})$. Thus, for the rest of the analysis we can assume $S, T \subseteq [m]$.

The parties pick a random hash function $h: [m] \to [k]$. For a set $U \subseteq [m]$ we use notation $U_i = h^{-1}(i) \cap U$ for the preimage of i in U. Using preimages S_i and T_i the parties construct a collection of instances of Equality, which contains an instance of Equality(s, t) for every $(s, t) \in S_i \times T_i$ for every $i \in [k]$.

Formally, for two sets of instances of a communication problem C,

$$C_1 = C(x_1, y_1), \ldots, C(x_i, y_i)$$

$$C_2 = C(x'_1, y'_1), \ldots, C(x'_j, y'_j)$$

let's denote their concatenation, which corresponds to solving C_1 and C_2 simultaneously as $C_1 \sqcup C_2 = (x_1, y_1), \ldots, (x_i, y_i), (x'_1, y'_1), \ldots (x'_j, y'_j)$. Let's denote as $E_i = \bigsqcup_{(s,t)\in(S_i\times T_i)} \mathrm{EQ}(s, t)$ the collection of instances of equality corresponding to hash value i. The collection of all instances constructed by the parties is $E = \bigsqcup_{i=1}^k E_i$.

The expected number of instances $\mathbb{E}[|E|]$ is given as:

$$\begin{aligned}\mathbb{E}[|E|] = \mathbb{E}\left[\sum_{i=1}^k |S_i||T_i|\right] &= \sum_{i=1}^k \mathbb{E}[|S_i||T_i|] \\ \leq \sum_{i=1}^k \mathbb{E}[|(S \cup T)_i|^2] &= \sum_{i=1}^k Var[|(S \cup T)_i|] + \mathbb{E}[|(S \cup T)_i|]^2 \qquad \text{(A.6)}\end{aligned}$$

Given that for a set Z, the random variable $|Z_i|$ is distributed according to a binomial distribution $B(|Z|, 1/k)$, for each i we have $Var[|\left(S \cup T\right)_i|] \leq 2k \cdot (1/k)(1 - 1/k) \leq 2$ and

$\mathbb{E}[|(S \cup T)_i|] \le 2$ so $\mathbb{E}[|E|] \le 6k$.

We use the following result of [90]:

Theorem A.9.2 ([90])**.** *There exists an $O(\sqrt{k})$-round constructive randomized protocol for EQ_k^n with success probability $2^{-\Omega(\sqrt{k})}$. In the public randomness model the expected total communication is $O(k)$ and in the private randomness model it is $O(k + \log n)$.*

In the shared randomness model the result now follows immediately. In the private randomness model the parties need to construct two random hash functions H and h, using Fact 8.1.2 with only $O(\log n) + O(\log k) = O(\log n)$ random bits. These bits are exchanged through the channel in the first round of the protocol and are added to the total communication, bringing it down to $O(k+\log n)$. To further reduce the communication we can use the hashing scheme of Fredman, Komlos and Szemeredi [96] as the first step of the protocol. In [96] it is shown that mapping elements $[n]$ by taking a remainder modulo a random prime $q = \tilde{O}(k^2 \log n)$ gives no collisions on a subset of size $O(k)$ with probability $1 - 1/poly(k)$. Applying this result to $S \cup T$ we can reduce the length of strings in the instances of equality down to $O(\log k + \log\log n)$. Thus, we can now specify the pairwise independent hash function using only $O(\log k+\log\log n)$ random bits. See Appendix A.1.1 in [132] for a detailed discussion.

□

A.10 Communication in the Set Intersection protocol

We analyze the total communication in the protocol. For a leaf $u \in \mathcal{T}$ let n_u denote the expected number of times the Basic-Set-Intersection protocol was run on the sets assigned to u.

Lemma A.10.1. *For every leaf $u \in \mathcal{T}$ it holds that $\mathbb{E}[n_u] = O(1)$.*

Proof. For a leaf u let's denote it's unique predecessor in level i as $p_i(u)$. Formally, $p_i(u) = v$ if and only if $v \in L_i$ and u is in the subtree of v. We can express $\mathbb{E}[n_u]$ as:

$$\begin{aligned}
\mathbb{E}[n_u] &= \sum_{i=0}^{r-1} \Pr[p_i(u) \text{ is failed}] \cdot (4 \ \mathrm{ilog}^{r-i} k) \\
&\le 1 + \sum_{i=1}^{r-1} d_i \cdot \Pr\left[v \text{ is an incorrect child of } p_i(u)\right] (4 \ \mathrm{ilog}^{r-i} k), \\
&\le 1 + \sum_{i=1}^{r-1} \frac{\mathrm{ilog}^{r-i} k}{\mathrm{ilog}^{r-i+1} k} \cdot \frac{1}{(\ \mathrm{ilog}^{r-i} k)^3} \cdot (4 \ \mathrm{ilog}^{r-i} k) = O(1)
\end{aligned}$$

where the first inequality holds by a union bound and the second by Corollary 8.2.6. □

The total expected communication in the protocol can be expressed as the sum of the total communication for Equality and Basic-Set-Intersection. The total communication for EQUALITY is:

$$\sum_{i=0}^{r-1} |L_i|(4 \text{ ilog}^{r-i} k) = O(k \text{ ilog}^r k) + \sum_{i=1}^{r-1} (k/ \text{ ilog}^{r-i} k) \cdot (4 \text{ ilog}^{r-i} k)$$
$$= O(k \text{ ilog}^r k) + O(rk) = O(k \text{ ilog}^r k).$$

The expected total communication for BASIC-SET-INTERSECTION is by Lemma 8.2.1 equal to:

$$\mathbb{E}\left[\sum_{i=1}^{k} (|S_i| + |T_i|) \log(|S_i| + |T_i|) \cdot n_i\right] = \sum_{i=1}^{k} \mathbb{E}\left[(|S_i| + |T_i|) \log(|S_i| + |T_i|)\right] \mathbb{E}[n_i],$$

where the equality follows from the independence of the random variables. Because for every i we have $\mathbb{E}[n_i] = O(1)$ by Lemma A.10.1, to complete the proof it is sufficient to show that $\mathbb{E}[(|S_i| + |T_i|) \log(|S_i| + |T_i|)] = O(1)$ and thus the total communication for BASIC-SET-INTERSECTION is $O(k)$. We have $\mathbb{E}[(|S_i|+|T_i|) \log(|S_i|+|T_i|)] \leq \mathbb{E}[(|S_i|+|T_i|)^2]$, where the right-hand side is constant by the same argument as used to bound each term in (A.6).

Vita

Grigory Yaroslavtsev

Grigory Yaroslavtsev is a Ph.D. candidate at the Department fo Computer Science and Enginnering at the Pennsylvania State University at University Park. Grigory joined Penn State in 2010 after completing his M.S. degree in Applied Mathematics and Physics from Academic University of the Russian Academy of Sciences, where he was the first student in a pilot class in theoretical computer science. In 2008 he got his B.S. degree in Engineering Physics from St. Petersburg State Polytechnical University. Grigory's research interests cover a broad spectrum of topics in large data processing, including but no limited to: approximation, parallel and online algorithms, learning theory and property testing, communication and information complexity, private data release. In 2012 he got the "Best Graduate Research Assistant Award" at the department of Computer Science and Engineering. During his time in graduate school he was supported by the University Graduate Fellowship and the College of Engineering Scholarship.