The Dissertation Committee for Jeffrey Stephen Ford
certifies that this is the approved version of the following dissertation:

Lower Bound Methods for Multiparty Communication Complexity

Committee:

Anna Gál, Supervisor

Greg Plaxton

Vijaya Ramachandran

György Turán

David Zuckerman

Lower Bound Methods for Multiparty Communication Complexity

by

Jeffrey Stephen Ford, BS, MSCS

Dissertation

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Doctor of Philosophy

The University of Texas at Austin

May 2006

To Jonobie

Acknowledgments

I would like to thank my advisor, Anna Gal, for all of the time she has spent encouraging me, cajoling me, and helping me shape my ideas into presentable form. Much of the work in this dissertation is joint work with her.

My wife's parents, Johnnie and O Baker, having been through a PhD program at UT themselves know more than anyone about the inevitable good times and bad times researching and writing a dissertation brings, and they have always shown me kindness as if I were their own son. My parents, Donald and Frances Ford, instilled in me an appreciation for learning and a love of education that propelled me to graduate studies. I know that I have made you proud.

Without a doubt, I must thank my wife, Jonobie Ford for her support in all matters personal, emotional, financial, and editorial. I'm sure she wasn't expecting to be married to a student for quite so long, but she has continued to encourage me all the way. As she chases her dreams, my sincere hope is that I can come close to supporting her in kind.

Jeffrey Stephen Ford

The University of Texas at Austin

May 2006

Lower Bound Methods for Multiparty Communication Complexity

Publication No. _____

Jeffrey Stephen Ford, Ph.D.
The University of Texas at Austin, 2006

Supervisor: Anna Gál

Multiparty communication complexity is a measure of the amount of communication required to compute a function whose input is distributed among multiple parties. Proving lower bounds in the "Number on the Forehead" model of Chandra, Furst, and Lipton has been quite difficult, but results are of importance in many areas including time-space tradeoffs and lower bounds for VLSI and Boolean circuits, as well as constructions of universal traversal sequences and pseudorandom generators for space bounded computation. Currently, the largest known lower bounds for explicit functions are of the form $\Omega(n/2^k)$, where k is the number of players and n is the number of bits missed by each player. All of the previous lower bounds of this type utilize some version of the discrepancy method, thus they also apply to

the distributional and randomized multiparty communication complexity models.

We introduce three new measures, tensor weight, tensor norm, and tensor rank, each of which can be used to prove lower bounds on multiparty communication complexity. We prove that the discrepancy of a multidimensional tensor can be estimated using the weight of the tensor. This measure appears to be simpler to analyze and it gives close to optimal bounds on discrepancy. Our second measure, the norm, can be used to estimate the weight, and thus discrepancy. We show an interesting connection between tensor norm and Lagrange multipliers. The tensor rank method is not based on the discrepancy method. We also give a variant of tensor rank that provides lower bounds on probabilistic communication complexity.

We give an extension of the Hadamard property of matrices for tensors in multiple dimensions, and we present an $\Omega(n/2^k)$ lower bound on any k-party, n-bit communication problem represented by a Hadamard tensor. We also give explicit constructions of Hadamard tensors giving $\Omega(n/2^k)$ lower bounds for a new class of Boolean functions. We also examine all functions previously known to have $\Omega(n/c^k)$ lower bounds and show their place in a unifying framework of nearly Hadamard tensors.

Contents

Chapter 1

Introduction

Communication complexity, a measure of the amount of communication necessary to compute functions whose input is split among multiple parties, was introduced by Yao [43] in 1979. It has many applications in complexity theory, including time-space tradeoffs and lower bounds for VLSI and Boolean circuits, as well as constructions of universal traversal sequences and pseudorandom generators for space bounded computation. Kushilevitz and Nisan [25] have written an excellent survey of the area, and we use their terminology throughout the paper.

In this dissertation we propose several new methods for proving lower bounds on multiparty communication complexity. We also introduce an extension of Hadamard matrices into higher dimensions. Some of the work in this dissertation appears in [17] and [16]. Before describing the results in more detail, it is necessary to give some background information.

1.1 Communication Complexity

1.1.1 Two-Party Communication Complexity

Two-party communication complexity can be defined informally as a game between two players. For convenience we will adopt the usual names Alice and Bob for the two players. The players each receive part of the input to a function, and they must jointly compute the output of that function. They both have unlimited computational power, and they are allowed to send messages to each other. Their goal is to send as few messages as possible. In particular, Alice and Bob devise a message exchange protocol specific to the function being computed that uses the smallest amount of communication on the worst case input for that protocol.

More formally, consider a function $f : X \times Y \to Z$ with Alice receiving $x \in X$ and Bob receiving $y \in Y$. A *protocol* is a binary tree that specifies the behavior of the players and the output of the function. Each interior vertex v of the tree is labeled with a function, either $A_v : X \to \{0, 1\}$ or $B_v : Y \to \{0, 1\}$, and each leaf is labeled with an element of Z. The functions A_v correspond to those steps in the protocol where Alice sends a message bit, and the functions B_v correspond to those steps where Bob sends a message bit. The outputs of A_v and B_v are the bits sent. Given inputs $x \in X$ and $y \in Y$ the tree computes in the following natural way: Apply the function found at the root to x or y as appropriate and go to the left or right child according to the answer. Repeat this process until a leaf is reached. The answer given by the protocol on input (x, y) is the element of Z labeling that leaf. A tree that matches the output of f on all inputs *computes* f. The *complexity* of a protocol tree is the length of the longest path from the root to a leaf. The *communication complexity* of f (denoted $D(f)$) is the minimum complexity across all protocols that compute f.

Two variations on deterministic communication complexity are also widely studied, distributional complexity and randomized complexity. In the distributional

complexity model, a protocol is only required to be correct on some of the input. $D^{\mu}_{\epsilon}(f)$ denotes the complexity of the best protocol for f which is correct on a $1-\epsilon$ measure of the input using probability distribution μ. We omit the μ when using the uniform distribution. In the randomized complexity model, the protocol is allowed to use random bits, and for any input must be correct for most choices of those random bits. $R_{\epsilon}(f)$ denotes the complexity of the best randomized protocol which computes f with error at most ϵ (on the worst input).

1.1.2 Multiparty Communication Complexity

Communication complexity can be extended to allow for three or more players. With two players the input is partitioned into two parts. It is most natural to consider the two players to each have access to one of these parts, but we can alternately view each player as missing access to one part. The two common multiparty models use these two views. With k players the input is partitioned into k parts. In one model, usually called the "number in the hand" model, each player has access to one part of the input. In the other model introduced by Chandra, Furst, and Lipton [9] in 1983 and usually called the "number on the forehead" model, each player has access to all but one part of the input. (Imagine the players sitting around a table with information written on their foreheads. They can see all of the information except the parts on their own foreheads.) In both of these standard models, any message sent by one player is seen by all of the other players, as if messages were being written on a shared blackboard.

Although the number in the hand model may seem more natural, the number on the forehead model appears to be more important to complexity theory. It is also a much more difficult model in which to prove lower bounds. For the remainder of this paper, unless specified otherwise, when we refer to the multiparty communication complexity of a function, we assume the number on the forehead

model.

Similarly to the two-party case, a multiparty protocol can be defined formally as a binary tree with each of the interior vertices corresponding to the action of one player and each of the leaves corresponding to an output of the function. In particular, the player to speak at each point in the protocol is determined solely by the previous communication (and *not* by the input). The notions of protocols *computing* functions, *complexity* of protocols, and *communication complexity* of functions are all defined analogously to the two-party case. We use $D(f)$ to denote the multiparty communication complexity of function f. In general this depends on the specific partition of the input of f, but the partition we use will always be clear from context. Distributional complexity and randomized complexity are defined for multiparty communication complexity just as they are in the two-party case.

1.1.3 Upper and Lower Bounds

Consider any function $f : X \times Y \to Z$. The two-party communication complexity of f, $D(f)$ is trivially at most $\log |X| + \log |Z|$. The function can always be computed by Alice sending her entire input to Bob who sends back the answer. Many large classes of functions have been shown using various techniques to have two-party communication complexity lower bounds that match or are very close to this trivial upper bound. Generally these functions can be shown to have large multiparty communication complexity in the number in the hand model using the same techniques.

Similarly, in the number on the forehead model, any multiparty function can be computed by the trivial protocol of one player writing on the blackboard the entire piece of input from a second player's forehead followed by the second player, who can now see the entire input, writing the answer. The most common case we will consider is the computation of a Boolean function where each of k players is

missing n distinct bits from an input of nk bits. In this case the trivial protocol gives a multiparty communication complexity upper bound of $n + 1$ bits.

We show in Chapter 9 that most Boolean functions of this type have multiparty communication complexity at least n. Despite this existential result, the largest known lower bounds for specific functions being computed by a non-constant number of players k are all much smaller, of the form $\Omega(n/c^k)$ for various constants c. (See, for example, [4], [11], [12], [19], and [33].)

1.1.4 Circuit Lower Bounds

Multiparty communication complexity lower bounds are related to ACC circuit lower bounds. The complexity class ACC, introduced by Barrington [5], is composed of those languages decidable by unbounded fan-in, polylogarithmic depth, polynomial size circuit families with AND, OR, NOT, and MOD_m gates for an arbitrary fixed m. The complexity class ACC^0 adds the restriction that the circuits be of constant depth. Razborov [34] showed that ACC^0 restricted to MOD_2 gates does not contain MAJORITY. Smolensky [38] showed that for p and q powers of distinct primes, ACC^0 restricted to MOD_p gates does not contain MOD_q. It appears to be much harder to prove bounds for MOD gates with at least two distinct prime factors. For example, it remains an open problem to show an explicit function family that is not in ACC^0 restricted to MOD_6 gates.

Beigel and Tarui [6] improved a result of Yao [45] to show that any language in ACC^0 can be represented in a different form as a family of depth-2 circuits with output gates that compute symmetric functions and have quasi-polynomial fan-in from AND gates with polylogarithmic fan-in. Håstad and Goldmann [20] showed that a function represented by a depth-2 circuit of this form with fan-in $k - 1$ to the AND gates and fan-in N to the symmetric gate can be computed by k players with multiparty communication complexity $k \log N$. So any function in ACC^0 has

multiparty communication complexity at most polylogarithmic for a polylogarithmic number of players. One way to show that a function family is not in ACC^0 would be to prove a lower bound on its multiparty communication complexity of the form $\Omega(n/\mathrm{poly}(k))$, which gives greater than polylogarithmic communication complexity for a polylogarithmic number of players.

1.2 Previous Lower Bound Methods

Here we give an overview of the previously known lower bound techniques that form the background of our work. We include some of the standard theorems.

1.2.1 Two-Party Techniques

A correct protocol partitions the inputs according to its leaves. One way to bound communication complexity is to characterize these sets and then to show that any partition with these characteristics must be large. Obviously those inputs corresponding to a particular leaf must evaluate to the same output. We use the following formalization:

Definition 1.1 *Let $f : X \to Z$ be a function. For any $z \in Z$, all inputs $x \in X$ for which $f(x) = z$ are called* z-inputs. *A subset of the inputs, $X' \subseteq X$ is called* z-monochromatic *if all $x \in X'$ are z-inputs. A subset which is z-monochromatic for some z is also called* monochromatic.

The partition sets formed by the leaves exhibit a nice combinatorial property. At each step of the protocol either Alice or Bob communicates a message that depends only on the corresponding part of the input and the prior messages. This means that at each step the possible inputs that remain consistent with the prior messages are reduced in a way oblivious to Alice's input or oblivious to Bob's input. This allows us to use combinatorial rectangles to describe the behavior of protocols:

Definition 1.2 *A set S of ordered pairs is called a* rectangle *if there exist sets A and B such that S is the Cartesian product $A \times B$.*

Theorem 1.1 *[43] Let $f : X \times Y \to Z$ be a function. Let $\mathcal{P}$ be a protocol that computes f. Let v be a vertex of $\mathcal{P}$. The set of inputs of f which reach v is a rectangle.*

Proof: Since $\mathcal{P}$ computes f, each input of f follows a fixed path from the root of $\mathcal{P}$ to a leaf as $\mathcal{P}$ acts on that input. Consider the set of inputs which reach each vertex on the path from the root to v. All inputs begin at the root, so the set of inputs at the root is the rectangle $X \times Y$. Consider vertex $v' \neq v$ on the path from the root to v. Suppose the inputs that reach v' form the rectangle $X' \times Y'$. $\mathcal{P}$ specifies either a function $A_{v'} : X \to \{0,1\}$ or $B_{v'} : Y \to \{0,1\}$. Suppose the function is $A_{v'}$ taking input from X and independent of Y. There must be a subset $X'' \subseteq X$ such that the inputs which reach the two children of v' are $X'' \times Y'$ and $(X' \setminus X'') \times Y'$. These are both rectangles. Similarly, if the function is $B_{v'}$, then Y' is partitioned, but the inputs reaching both children are rectangles. Regardless of which function appears, the next vertex on the path is reached by a set of inputs which form a rectangle. By induction along the path the inputs reaching v form a rectangle. ∎

Now we can give a very general lower bound for two-party communication complexity:

Theorem 1.2 *[43] Let $f : X \times Y \to Z$ be a function. Let $P(f)$ be the smallest number of monochromatic rectangles necessary to partition $X \times Y$. Then*

$$D(f) \geq \log P(f).$$

Proof: Any correct protocol computing f partitions the inputs into monochromatic rectangles at its leaves. Thus the number of leaves is at least $P(f)$. And so the depth of the protocol tree is at least $\log P(f)$. ∎

Many different techniques have been used to give lower bounds on this partition number. The simplest is to give an upper bound on the size of the largest monochromatic rectangle. We will present three other techniques that have counterparts in the multiparty model. Fooling sets (introduced by [43] and [27]) and discrepancy (introduced by [10]) have previously been generalized. We generalize matrix rank (introduced by [29]) for the first time to the multiparty model.

Fooling Sets

The first bound on the partition number is given by constructing a set of inputs that must each be in a different part of the partition:

Definition 1.3 *Let $f : X \times Y \to Z$ be a function. A subset of the input $S \subset X \times Y$ is called a* fooling set *if it is monochromatic and has the following property: For any distinct $(x_1, y_1), (x_2, y_2) \in S$, $f(x_1, y_2) \neq f(x_1, y_1)$ or $f(x_2, y_1) \neq f(x_1, y_1)$.*

Theorem 1.3 *[27] Let $f : X \times Y \to Z$ be a function. If S is a fooling set of f then*

$$D(f) \geq \log |S|.$$

Proof: Consider any partition of the inputs into monochromatic rectangles. Let (x_1, y_1) and (x_2, y_2) be distinct elements of a fooling set S. By the definition of rectangle, if (x_1, y_1) and (x_2, y_2) are both in a rectangle then (x_1, y_2) and (x_2, y_1) are also in the rectangle. But by the definition of fooling set, $\{(x_1, y_1), (x_1, y_2), (x_2, y_1)\}$ is not monochromatic. So (x_1, y_1) and (x_2, y_2) must be in different rectangles in the partition. Thus the number of elements in a fooling set gives a lower bound on the number of monochromatic rectangles necessary to partition the inputs of a function. Applying Theorem 1.2 gives the bound on communication complexity. ∎

This theorem can be generalized by using a fooling set for each of the outputs of the function being computed since the inputs in each of these fooling sets must

be in different partition elements. In the common case of Boolean functions, this distinction is not very important, as the lower bound given by the largest fooling set is at most 1 less than the lower bound given by the largest fooling sets for both outputs. Fooling sets also give lower bounds on non-deterministic communication complexity.

Discrepancy

Another bound is given by measuring how large "almost monochromatic" rectangles can be. It is typically defined for Boolean functions.

Definition 1.4 *Let $f : X \times Y \to \{0,1\}$ be a function, and let μ be a probability distribution on $X \times Y$. The* discrepancy *of f with respect to μ (denoted $\mathrm{Disc}_\mu(f)$) is the maximum over all rectangles R of the following:*

$$\left| \Pr_\mu\big[\big((x,y) \in R\big) \wedge \big(f(x,y) = 1\big)\big] - \Pr_\mu\big[\big((x,y) \in R\big) \wedge \big(f(x,y) \neq 1\big)\big] \right|.$$

Using this definition, the measure with respect to μ of the largest monochromatic rectangle is a lower bound on the discrepancy, but frequently the probability difference is maximized by a non-monochromatic rectangle. The discrepancy of a function gives a bound on its communication complexity:

Theorem 1.4 *[44] Let $f : X \times Y \to \{0,1\}$ be a function, and let μ be any probability distribution on $X \times Y$.*

$$D(f) \geq \log(1/\mathrm{Disc}_\mu(f)).$$

Proof: Any protocol that computes f using at most c bits of communication has at most 2^c leaves. Each of these leaves corresponds to a subset of $X \times Y$ that is a monochromatic rectangle. Each of these rectangles has measure at most $\mathrm{Disc}_\mu(f)$. On any input the set of leaves is reached with probability 1, so the sum over all

leaves of the measure of the input that reaches each leaf is 1. Thus $2^c \text{Disc}_\mu(f) \geq 1$ and so $c \geq \log(1/\text{Disc}_\mu(f))$. ∎

The discrepancy method actually gives lower bounds for two other measures of communication complexity, distributional complexity and randomized complexity. We state these two-party results without proof. If a protocol for function f must be correct on an $((1/2)+\epsilon)$ measure of the input under probability distribution μ, then the function has distributional complexity at least $\log(2\epsilon/\text{Disc}_\mu(f))$. More formally, $D^\mu_{(1/2)-\epsilon}(f) \geq \log(2\epsilon/\text{Disc}_\mu(f))$.

The randomized complexity of f for a protocol required to be correct with probability at least $((1/2)+\epsilon)$ will be at least the maximum value of $\log(2\epsilon/\text{Disc}_\mu(f))$ over all distributions μ. More formally, $R_{(1/2)-\epsilon}(f) \geq \max_\mu \log(2\epsilon/\text{Disc}_\mu(f))$.

Matrix Rank

Any problem in two-party communication complexity can be represented as a matrix with the rows labeled by the possible inputs for Alice and the columns labeled by the possible inputs for Bob. An entry in the matrix is given by applying the function to the inputs labeling its row and column. The following is one of the standard equivalent definitions of matrix rank:

Definition 1.5 *A matrix has* rank 1 *if it can be formed as the outer product of two vectors. The* rank *of a matrix is the minimum number of rank 1 matrices whose sum is the matrix.*

The rank of the matrix representing a function can be used to bound the communication complexity of the function:

Theorem 1.5 *[29] Let $f : X \times Y \to \{0,1\}$ be a function. Let A_f be the matrix representing f.*

$$D(f) \geq \log(\text{rank } A_f).$$

Proof: A 1-monochromatic rectangle over the inputs of f can be represented by a rank 1 matrix with the rows and columns labeled (as A_f) by the inputs of f. Let $\mathcal{P}$ be the best protocol for f. $\mathcal{P}$ partitions all of the inputs into monochromatic rectangles at its leaves. Summing the rank 1 matrices corresponding to the 1-monochromatic rectangles induced by the leaves gives A_f. Thus the number of leaves is at least the rank of A_f. The communication complexity of f is at least the logarithm of the number of leaves. Combining these statements gives the theorem. ∎

1.2.2 Multiparty Techniques

In the two-party model, many different explicit functions have been shown to require essentially the same amount of communication as that given by the trivial upper bound. In particular there are a large number of Boolean functions on pairs of n-bit strings with $\Omega(n)$ communication complexity. In the multiparty model, although we know by counting that most functions require communication close to the trivial upper bound, no explicit functions have been shown to have lower bounds of this magnitude.

Few of the methods for proving lower bounds in the two-party model have been successfully extended to multiparty complexity. Chandra, Furst, and Lipton [9] proved the first lower bounds using a technique similar to an extended fooling set. However their technique gives weak lower bounds. The best lower bounds, all of the form $\Omega(n/c^k)$ (for k players each having all but n bits of an input of length nk) have been proved using an extension of the discrepancy method originally devised by Babai, Nisan, and Szegedy[4].

Ramsey Theoretic Method

We describe the method of Chandra, Furst, and Lipton [9].

Definition 1.6 *Let $S = X_1 \times \cdots \times X_k$. A subset of S is called a* star *if it can be*

represented as

$$\{(x'_1, x_2, x_3, \ldots, x_k), (x_1, x'_2, x_3, \ldots, x_k), \ldots, (x_1, x_2, \ldots, x'_k)\}$$

where for $i \in \{1, \ldots, k\}$, $x_i, x'_i \in X_i$ *and* $x_i \neq x'_i$. *The element* $(x_1, x_2, x_3, \ldots, x_k)$ *is called the* center *of the star, but is not part of the star.*

Consider a protocol for a function $f : X_1 \times X_2 \times \cdots \times X_k \to Z$. If the set of inputs associated with a leaf contain all of the elements of a star, then the center of the star must also be in the set. At a step of the protocol where player i speaks, the protocol must have the same behavior on $(x_1, \ldots, x_{i-1}, x'_i, x_i, \ldots, x_k)$ as on $(x_1, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_k)$. If a star is monochromatic, but the center of the star produces a different output, any valid protocol must separate the elements of the star among two or more of its leaves.

In particular, they considered the following function:

$$\text{EXACTLY–}N : \{1, 2, \ldots, N\}^k \to \{0, 1\}$$

where EXACTLY–$N(a_1, a_2, \ldots, a_k) = 1$ if and only if $a_1 + a_2 + \cdots + a_k = N$. Consider any 1-monochromatic star for this function. The center of the star will be a 0-input. A lower bound on the number of colors necessary to color the 1-inputs with no monochromatic stars provides a lower bound on the number of leaves of any protocol computing EXACTLY–N.

They relate this coloring to the following famous problem from Ramsey theory, using the answer to generate both a lower bound and a protocol giving a nearly matching upper bound:

> How many colors are necessary to color the set $\{1, \ldots, N\}^k$ such that for all vectors $(x_1, \ldots, x_k)$ and all integers $\lambda \neq 0$, if the set $\{(x_1, \ldots, x_k),$

$(x_1+\lambda, x_2, \ldots, x_k)$, $(x_1, x_2+\lambda, \ldots, x_k)$, $\ldots$, $(x_1, x_2, \ldots, x_k+\lambda)\}$ contains only elements from $\{1, \ldots, N\}^k$, it is not monochromatic?

This number is not known in the general case, but was bounded for $k = 3$, giving a three player protocol running in time $O(\sqrt{\log N})$ (but no matching lower bound). For $k \geq 3$ the method gives a lower bound of $\omega(1)$.

Discrepancy Method

Next we present the method of Babai, Nisan, and Szegedy[4]. To define discrepancy in the multiparty case we need to define a combinatorial construct serving the same purpose rectangles do for two-party communication complexity:

Definition 1.7 *Let $X_1 \times X_2 \times \cdots \times X_k$ be a set. A set $C_i \subseteq X_1 \times X_2 \times \cdots X_k$ is a* cylinder *with respect to player i if for all $x_{i_1}, x_{i_2} \in X_i$,*

$$(x_1, \ldots, x_{i-1}, x_{i_1}, x_{i+1}, \ldots, x_k) \in C_i \Leftrightarrow (x_1, \ldots, x_{i-1}, x_{i_2}, x_{i+1}, \ldots, x_k) \in C_i$$

In other words, membership in a cylinder with respect to player i does not depend on the ith coordinate.

Definition 1.8 *Let $X_1 \times X_2 \times \cdots \times X_k$ be a set. A set $C \subseteq X_1 \times X_2 \times \cdots \times X_k$ is a* cylinder intersection *if it can be expressed as $\bigcap_{i=1}^{k} C_i$ where each C_i is a cylinder with respect to player i.*

When a player writes on the blackboard in a multiparty protocol, the set of inputs consistent with the communication to date is the intersection of the previous set of consistent inputs and a cylinder with respect to the player writing. Thus any leaf of a multiparty protocol tree can be represented as a cylinder intersection. To define discrepancy we essentially use the two-party definition, replacing rectangles with cylinder intersections.

Definition 1.9 *Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{0,1\}$ be a function, and let μ be a probability distribution on $X_1 \times X_2 \times \cdots \times X_k$. We define the* bias *of a cylinder intersection $C \subseteq X_1 \times X_2 \times \cdots \times X_k$ on f with respect to μ as*

$$\left| \Pr_{\mu}\big[(\vec{x} \in C) \wedge (f(\vec{x}) = 1)\big] - \Pr_{\mu}\big[(\vec{x} \in C) \wedge (f(\vec{x}) \neq 1)\big] \right|.$$

The discrepancy *of f with respect to μ (denoted $\mathrm{Disc}_{\mu}(f)$) is the maximum over all cylinder intersections C of the bias of C on f with respect to μ.*

This is the theorem from Babai, Nisan, and Szegedy that establishes the discrepancy method:

Theorem 1.6 *[4] Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{0,1\}$ be a function. Let μ be a probability distribution on $X_1 \times X_2 \times \cdots \times X_k$.*

$$D(f) \geq \log(1/\mathrm{Disc}_{\mu}(f)).$$

The proof is analogous to the two-party result. As in the two-party case, similar bounds hold for distributional and randomized complexity, stated here without proof:

Theorem 1.7 *[4] Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{0,1\}$ be a function. Let μ be a probability distribution on $X_1 \times X_2 \times \cdots \times X_k$.*

$$D^{\mu}_{(1/2)-\epsilon}(f) \geq \log(2\epsilon/\mathrm{Disc}_{\mu}(f)).$$

Theorem 1.8 *[4] Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{0,1\}$ be a function. Let μ be a probability distribution on $X_1 \times X_2 \times \cdots \times X_k$.*

$$R_{(1/2)-\epsilon}(f) \geq \max_{\mu} \log(2\epsilon/\mathrm{Disc}_{\mu}(f)).$$

We will omit the μ from the discrepancy notation when working with the uniform distribution.

Discrepancy is extremely difficult to measure directly. Using discrepancy estimates, lower bounds have been found for the following functions:

Definition 1.10 *The k-wise generalized inner product (denoted GIP_n^k) is the function on k binary strings of length n that has the value 1 if the number of locations where all the strings are 1 is odd and has the value 0 otherwise.*

Definition 1.11 *The quadratic character of sum (denoted $\mathrm{QCS}_{p,k}$) is the function on k integers that has the value 1 if the sum of the integers is a quadratic residue mod p and has the value 0 otherwise.*

Babai, Nisan, and Szegedy [4] proved a lower bound of $\Omega(n/4^k)$ for GIP_n^k and $\Omega((\log p)/2^k)$ for $\mathrm{QCS}_{p,k}$. The GIP_n^k bound was improved by Chung and Tetali [12] to $\Omega(n/2^k)$, and Grolmusz [19] devised a protocol to compute GIP_n^k giving a nearly matching upper bound of $O(kn/2^k)$.

1.3 New Results

Chor and Goldreich [10] showed that any Boolean function defined by a Hadamard matrix has $\Omega(n)$ two-party communication complexity. Their proof uses a lemma by Lindsey (see [15] p. 88) that estimates the largest possible sum of entries in a sub-matrix of a Hadamard matrix. Lindsey's lemma implies upper bounds on the discrepancy of functions represented by Hadamard and "nearly" Hadamard matrices. The proof of Babai, Nisan, and Szegedy [4] generalized the proof of Lindsey to achieve multiparty bounds, but no one has previously generalized the Hadamard property to tensors. We give our attempt in this dissertation, and get the following main results:

- A new method for computing lower bounds of multiparty communication problems by estimating discrepancy with weight
- An extension of the Hadamard property of matrices into three or more dimensions
- A lower bound of $\Omega(n/2^k)$ on the multiparty communication complexity of functions represented by Hadamard tensors
- An explicit construction of a family of functions represented by tensors with the Hadamard property
- A unifying framework of nearly Hadamard tensors encompassing all functions previously known to have $\Omega(n/c^k)$ lower bounds

In addition we provide two new methods for computing lower bounds giving the following results:

- A new method for computing lower bounds of multiparty communication problems by estimating discrepancy with norm
- Insights into computing norm including an interesting relationship between the norm of a tensor and Lagrange multipliers
- A new method for computing lower bounds of multiparty communication problems by computing their "NOF-rank"
- An extension of "NOF-rank" giving a method to compute lower bounds for the probabilistic multiparty communication model

Finally we examine the discrepancy of some standard functions and the communication complexity of a random function giving these results:

- An exact computation of the discrepancy of PARITY under the uniform distribution

- A relationship between the discrepancy and monochromatic cylinder intersection size
- Comparisons between the number of functions and communication complexity protocols giving communication bounds for almost all functions

1.4 Organization and Chapter Dependencies

In Chapter 2 we give bounds on discrepancy for some simple functions. In Chapter 3 we present a new lower bound method based on tensor weight. In Chapter 4 we define an extension of Hadamard matrices into more than two dimensions and we give a communication complexity lower bound on functions represented by Hadamard tensors. In Chapter 5 we construct explicit functions with the Hadamard property. In Chapter 6 we give some extensions of the Hadamard property and show how to place some previous results into the Hadamard framework. In Chapter 7 we present a new lower bound method based on tensor norm. In Chapter 8 we generalize the rank lower bound method from two-party communication complexity to multiparty communication complexity. In Chapter 9 we use counting arguments to bound the two-party and multiparty communication complexity of arbitrary functions.

Chapter 4 depends on Chapter 3. Chapters 5 and Chapter 6 depend on chapter 4. The remaining chapters can be read independently and depend only on material in Chapter 1.

Chapter 2

Discrepancy and Monochromatic Cylinder Intersections

In this chapter we give some insights into computing discrepancy for some well-known functions. In general it is difficult to estimate discrepancy directly, hence later chapters will focus on less direct means of estimation. Throughout this chapter we assume that functions are from binary strings to Boolean values.

2.1 Isolated Inputs

We begin with the simplest case for multiparty discrepancy: The number of players is the same as the number of input bits, each player missing only one bit, and the computation is done with respect to the uniform distribution. Since each player is missing one bit of input, each player behaves the same at any stage in a protocol for two different input strings, the actual string being processed and the string with the opposite bit in the position that player cannot see. In general, regardless of how

much of the input each player is missing it is possible to partition the input strings relative to a player by grouping those that the player cannot distinguish. These groupings are important since in choosing a cylinder for the discrepancy definition, the cylinder for a particular player must either keep or reject all of the strings in any given set of the partition. Since the cylinder intersection in the discrepancy definition is mostly monochromatic, it is helpful to determine which inputs can be easily excluded from it without reducing the number of "good" inputs it contains. For terminology purposes, it is easier to define the opposite idea:

Definition 2.1 *An input x to a multiparty communication problem is* isolated *if for all players P_i there is an input x_i that from the point of view of P_i cannot be distinguished from x, but which produces a different output value.*

Here we do not require each player to miss only one bit, but in that special case, an input is isolated if changing any of its bits changes the value of the function computed on that input.

For an illustration of isolated inputs, we consider the THRESHOLD function, a generalized form of the MAJORITY function:

Definition 2.2 $\text{THRESHOLD}_t^n : \{0,1\}^n \to \{0,1\}$ *has output* 1 *when the input contains at least t ones and has output* 0 *otherwise.*

Lemma 2.1 *For $1 < t < n$, there are no isolated inputs in the multiparty communication problem corresponding to* THRESHOLD_t^n *where n players each miss one bit.*

Proof: Consider any 0-input of THRESHOLD_t^n. It contains at most $(t-1)$ ones. Assume the input is not the all-zero string. Consider any index which contains a one. The player who cannot see the bit at that index can see at most $(t-2)$ ones among the other bits. Thus regardless of the bit at that index, the player knows

that the input is a 0-input. Thus the input is not isolated. If the input is the all-zero string, then since we have restricted t to be at least 2, any player can see no ones, and so knows that the input is a 0-input regardless of the value of the unseen bit.

The proof is almost identical for 1-inputs, except that we consider the input from the point of view of a player whose unseen bit is a zero. Since that player sees at least t ones, the output is known to be 1 regardless of the unseen bit. The restriction that t be strictly smaller than n ensures that the all-one string is not isolated in the same way the restriction t be bigger than 1 was necessary for the all-zero string. ∎

This result means that there is a 0-monochromatic cylinder intersection consisting of all of the 0-inputs of THRESHOLD, and that all of the 1-inputs form a 1-monochromatic cylinder intersection. To form these, each player's cylinder excludes any string known to give the wrong output. Since none of the inputs are isolated, each of the "bad" inputs will be discarded by at least one player, and none of the "good" inputs will be discarded. The following theorem shows that monochromatic cylinder intersections suffice for discrepancy computations under certain conditions:

Theorem 2.2 *Let $f : \{0,1\}^k \to \{0,1\}$ be a function representing a k-player multiparty communication complexity problem with each player missing one bit. Let $C \subseteq \{0,1\}^k$ be a cylinder intersection with bias d (under the uniform distribution) on f. Then f has a monochromatic cylinder intersection with measure d (under the uniform distribution) over $\{0,1\}^k$.*

Proof: Since C is a cylinder intersection, we represent it as $\cap_{i=1}^{k} C_i$ where each C_i is a cylinder with respect to player i. Without loss of generality, assume that C has more 1-inputs than 0-inputs. Consider the set C_1' formed from C_1 by removing for each 0-input $b_1 b_2 \ldots\ b_k \in C$ both $b_1 b_2 \ldots\ b_n$ and $\bar{b_1} b_2 \ldots\ b_k$. C_1' remains a cylinder with respect to player 1. Let $C' = C_1' \cap C_2 \cap \cdots \cap C_k$. Since C' has no 0-inputs it is a 1-monochromatic cylinder intersection.

The bias of C on f is the difference between the number of 0-inputs in C and the number of 1-inputs in C divided by the total number of possible inputs to f. There are at least as many 1-inputs in C' as the difference between the number of 1-inputs and 0-inputs in C since the process of creating C' removed at most one 1-input for each 0-input removed. C' has no 0-inputs, so C' has a bias on f of at least d. That is, C' is a monochromatic cylinder intersection with measure d. ∎

2.2 Discrepancy of PARITY

We expand on the idea of isolated inputs to find an exact discrepancy for the PARITY function under the uniform distribution. Since reversing one bit of a string changes its parity, the PARITY function has only isolated inputs. We define PARITY explicitly as follows:

Definition 2.3 $\text{PARITY}^n : \{0,1\}^n \to \{0,1\}$ *outputs* 1 *when the number of ones in the input is odd and* 0 *otherwise.*

We also need to show discrepancy cannot grow under function composition:

Definition 2.4 *Let* $f : Y \to Z$ *and* $g : X \to Y$ *be functions with the domain of* f *the same as the range of* g. *The* composition *of* f *and* g *(denoted* $f \circ g$*) is the function which is equal to* $f(g(x))$ *for all* $x \in X$.

Lemma 2.3 *Given any two functions* $f, g : \{0,1\}^k \to \{0,1\}$,

$$\text{Disc}(f \circ g) \leq \text{Disc}(f).$$

Proof: Let C be a cylinder intersection of $f \circ g$. For any choice of input to g, C induces a cylinder intersection on the inputs of f. The bias of this induced cylinder intersection is at most $\text{Disc}(f)$. Thus the difference between the number of 1-inputs and 0-inputs in this induced cylinder intersection is at most $2^k\text{Disc}(f)$.

Summing over all choices of input to g, the difference between the number of 1-inputs and 0-inputs in C is at most $2^k(2^k\text{Disc}(f))$. The discrepancy of $f \circ g$ is at most this difference divided by the number of inputs to $f \circ g$ which is 2^{2k}. Thus $\text{Disc}(f \circ g) \leq \text{Disc}(f)$. ∎

Theorem 2.4 *The discrepancy of* PARITY *with k players under the uniform distribution is $(k-1)/(2k)$ when k is a power of two, regardless of the number of bits each player misses.*

Proof: First we consider the case where each player misses only one bit. According to Theorem 2.2, it suffices to prove that the largest monochromatic cylinder intersection contains a $((k-1)/(2k))$-fraction of the inputs. First we show that a higher fraction cannot be achieved. Let $C = \cap_{i=1}^{k} C_i$ be a monochromatic cylinder intersection where each C_i is a cylinder with respect to player i. Without loss of generality, assume C contains only strings of even parity. Suppose $b_1 b_2 \ldots b_k$ is an input string of odd parity. Since it is not in C, there is at least one cylinder, say C_j, which does not contain it. Since C_j does not contain $b_1 \cdots b_j \cdots b_k$ and it is a cylinder, it also does not contain $b_1 \cdots \bar{b_j} \cdots b_k$, a string of even parity. In this manner we can associate with each string of odd parity a string of even parity that is not in C. Each of these strings of even parity can be associated with at most k strings of odd parity, one per player. There are 2^{k-1} strings of odd parity, so there are at least $2^{k-1}/k$ strings of even parity that are not contained by C. Thus the measure of C is at most $(2^{k-1} - (2^{k-1}/k))/2^k$ which is $(k-1)/(2k)$.

One way to achieve the bound is to provide a list of $2^{k-1}/k$ binary strings of length k of even parity, each of which could be associated with k unique strings of odd parity. Then the cylinders formed by having each player exclude these strings of even parity would form an intersection that keeps all of the remaining strings of even parity, but excludes all strings of odd parity. Two strings of even parity have no overlap in the strings of odd parity they exclude if they have Hamming

distance greater than two. Thus it suffices to find $2^{k-1}/k$ binary strings of length k of even parity, every pair of which has Hamming distance at least 3. It is enough to generate that many binary strings of length $k-1$ without regard to parity, as those strings could all be converted to length k strings of even parity by concatenating the appropriate bit. This conversion would not reduce their Hamming distances. Since k is a power of two, Hamming codes (see e.g., [41]) provide exactly those strings, and the bound is achieved.

Next we consider the case where each player is missing $d > 1$ bits. PARITY with each player missing d bits is equivalent to the composition of d copies of PARITY with each player missing one bit. Thus, according to Lemma 2.3, the discrepancy is bounded from above by $(k-1)/(2k)$. We show this bound is achievable by extending the set of codewords from the case where each player misses only one bit. Generate a set of codewords of length kd by listing all strings of length kd which form one of the original length k codewords after taking the parity of every d bits. Each original codeword will extend to $(2^{d-1})^k$ codewords of length kd. Again, the cylinders formed by having each player exclude all of the codewords form an intersection that keeps the remaining strings of even parity, but excludes all strings of odd parity. Thus there are $2^{dk-1} - (2^{(d-1)k}2^{k-1}/k)$ strings of even parity in the cylinder intersection. This gives the required $(k-1)/2k$ fraction. ∎

Chapter 3

Tensor Weight

In this chapter we give a new method for estimating the discrepancy of multiparty communication complexity problems. We introduce a new measure called "weight" and prove bounds on discrepancy via weight. The discrepancy bound gives a multiparty communication complexity lower bound.

3.1 Representing Multiparty Communication Problems

A multiparty communication complexity problem can be represented by a tensor, the multidimensional analogue of a matrix. Each dimension of the tensor is associated with one player, and the entries are indexed along that dimension by the possible values of the part of the input that player misses. We use the notation $A(x_1, \ldots, x_k)$ (frequently abbreviated to $A(\vec{x})$) to denote the entry of the k-dimensional tensor A that is associated with inputs $x_1, \ldots, x_k$. In order to explain our methods efficiently, we must begin with a series of definitions of tensor properties.

Most of the functions we will be considering have inputs of the same size missed by each player, and are thus represented by tensors with an equal number of inputs associated with each dimension. We formalize that property as follows:

Definition 3.1 *Let A be a tensor in k dimensions. A has* order N *if in each of the k dimensions there are N values used to index that dimension.*

We take special note of two kinds of sub-tensor, defined as follows:

Definition 3.2 *Given a k-dimensional tensor A, a* line *of A is any vector formed by fixing all but one coordinate of A.*

Definition 3.3 *Given a k-dimensional tensor A, a* face *of A is any tensor formed by fixing one coordinate of A. (A face is a tensor in one less dimension.)*

Finally, we provide a way to build tensors from smaller tensors:

Definition 3.4 *Let A be a k-dimensional tensor with order N_1. Let B be a k-dimensional tensor with order N_2. The* Kronecker product *of A and B (denoted $A \otimes B$) is the tensor with order $N_1 N_2$ formed by replacing each entry of B with a copy of A multiplied by that entry.*

When attempting to characterize a function f represented by a tensor A_f in k dimensions, we will sometimes make use of a set of k smaller tensors, each in $k-1$ dimensions. To make the notation more readable, we will index both A_f and these smaller tensors by vectors of length k, with the understanding that for the i-th tensor in the set of smaller tensors, the i-th coordinate of the vector is ignored for indexing.

3.2 Weight Definitions

It is more convenient to consider Boolean functions to have output from $\{1, -1\}$ instead of from $\{0, 1\}$. The discrepancy of a function with output from $\{1, -1\}$ is given by the same definition as for functions with output from $\{0, 1\}$. Next we give an alternate definition of discrepancy under the uniform distribution and show its equivalence to the standard definition:

Definition 3.5 *Let $f : X_1 \times \cdots \times X_k \to \{1, -1\}$ be a function. Let A_f be the tensor which represents f. For any set S of lines of A_f, let A_f^S be the tensor formed from A_f by replacing all of the entries in the lines of S with 0's. The* discrepancy *of f under the uniform distribution is the maximal value over all sets S of $|A_f^S|/(\text{size}(A_f))$ where $|A_f^S|$ is the absolute value of the sum of the entries of A_f^S and $\text{size}(A_f)$ is the number of entries of A_f.*

The original discrepancy definition computed a maximum over cylinder intersections. Choosing the cylinder along a particular dimension that is used in a cylinder intersection is equivalent to replacing by 0 all entries in a set of lines along that dimension, and keeping those inputs whose tensor value is not 0. Since we are considering ± 1-valued functions, summing entries in the chosen cylinder intersection and dividing by size is the same as taking the difference in probabilities between 1-inputs and -1-inputs.

We consider a similar process sometimes referred to as the "switching lights game" which was first suggested for matrices in [39] and for tensors in [12] to generate the *weight* of the tensor. Instead of replacing the entries of a set of lines with 0's, we multiply the entries by -1. More formally, we give the following definitions:

Definition 3.6 *Let A be a tensor with ± 1 entries. A line of A is* flipped *by multiplying all of its entries by -1.*

Definition 3.7 *Let A be a tensor with ± 1 entries. The* excess *of A is the absolute value of the sum of the entries of A.*

Definition 3.8 *Let A be a tensor with ± 1 entries. Let A' be the tensor with maximum excess achievable by flipping an arbitrary set of lines of A. The* weight *of A (denoted $W(A)$) is the excess of A'.*

Note that there is always a set of lines which leads to a positive sum of entries with maximum excess. Given a set of lines which leads to a negative sum,

flipping along one dimension the compliment along that dimension of the lines in the set instead of the original lines gives a tensor with a positive sum of the same magnitude.

Although it might seem as though a simple algorithm exists to compute weight, no efficient one is known. In particular it is not enough to use a greedy algorithm which only considers lines with at least as many negative entries as positive. Consider the following matrix:

$$\begin{bmatrix} 1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 \end{bmatrix}.$$

This matrix has more positive entries than negative entries in every row and column, but by flipping the first two rows and the first three columns the number of positive entries is increased:

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 \\ 1 & 1 & -1 & 1 & 1 & 1 \\ -1 & -1 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

3.3 Weight Bounds

The weight of a tensor can be used to bound the discrepancy of the function it represents. In the proof of the theorem we use the following sign function:

$$\text{sign}(r) = \begin{cases} -1 & \text{if } r \text{ is negative,} \\ 1 & \text{otherwise.} \end{cases}$$

Theorem 3.1 *Consider a function* $f : X_1 \times \cdots \times X_k \to \{1, -1\}$ *where for* $i \in \{1, \ldots, k\}$, $|X_i| = N$. *Let* A_f *be the tensor with order* N *representing* f. *Then*

$$\text{Disc}(f) \leq W(A_f)/N^k.$$

Proof: A cylinder with respect to player i can be represented by a tensor in $(k-1)$ dimensions with order N restricted to entries from $\{0, 1\}$. (The element $\vec{x} \in X_1 \times \cdots \times X_k$ is in cylinder C_i if $C_i(\vec{x}) = 1$.) Let $C_1, C_2, \ldots, C_k$ be the tensor representations of a set of cylinders whose intersection has maximum bias on f, where each C_i represents a cylinder in the i-th dimension.

We define a set of related tensors with ± 1 entries as follows:

$$\hat{C}_k(\vec{x}) = \text{sign}\big(\sum_{x_k} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{k-1}(\vec{x})\big).$$

For all $i < k$,

$$\hat{C}_i(\vec{x}) = \text{sign}\big(\sum_{x_i} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{i-1}(\vec{x}) \hat{C}_{i+1}(\vec{x}) \cdots \hat{C}_k(\vec{x})\big).$$

By definition, the discrepancy of f is

$$\left| \sum_{\vec{x}} A_f(\vec{x}) C_1(\vec{x}) \cdots C_k(\vec{x}) \right| / N^k.$$

Since C_k does not depend on the k-th dimension, this is the same as

$$\left| \sum_{x_1,\ldots,x_{k-1}} C_k(\vec{x}) \left[\sum_{x_k} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{k-1}(\vec{x}) \right] \right| / N^k.$$

Since C_k has only ± 1 entries, this is no more than

$$\left| \sum_{x_1,\ldots,x_{k-1}} \left| \sum_{x_k} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{k-1}(\vec{x}) \right| \right| / N^k.$$

By definition of $\hat{C}_k$, this is the same as

$$\left| \sum_{x_1,\ldots,x_{k-1}} \hat{C}_k(\vec{x}) \left[\sum_{x_k} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{k-1}(\vec{x}) \right] \right| / N^k.$$

Rearranging to bring the new tensor back inside the sum gives

$$\left| \sum_{x_1,\ldots,x_k} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{k-1}(\vec{x}) \hat{C}_k(\vec{x}) \right| / N^k.$$

We repeat this process k times, at each step using the transformation

$$\begin{aligned} &\left| \sum_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_k} C_i(\vec{x}) \left[\sum_{x_i} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{i-1}(\vec{x}) \hat{C}_{i+1}(\vec{x}) \cdots \hat{C}_k(\vec{x}) \right] \right| \\ \leq \; &\left| \sum_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_k} \hat{C}_i(\vec{x}) \left[\sum_{x_i} A_f(\vec{x}) C_1(\vec{x}) \cdots C_{i-1}(\vec{x}) \hat{C}_{i+1}(\vec{x}) \cdots \hat{C}_k(\vec{x}) \right] \right|. \end{aligned}$$

This gives

$$\left| \sum_{\vec{x}} A_f(\vec{x}) C_1(\vec{x}) \cdots C_k(\vec{x}) \right| / N^k \leq \left| \sum_{\vec{x}} A_f(\vec{x}) \hat{C}_1(\vec{x}) \cdots \hat{C}_k(\vec{x}) \right| / N^k.$$

Each $\hat{C}_i$ represents flipping lines along dimension i so

$$\left|\sum_{\vec{x}} A_f(\vec{x})\hat{C}_1(\vec{x})\cdots\hat{C}_k(\vec{x})\right|/N^k \leq W(A_f). \quad \blacksquare$$

Using Theorems 1.6, 1.7, and 1.8 we get corresponding multiparty communication complexity lower bounds as follows:

Theorem 3.2 *Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{1, -1\}$ be a function where for all i, $|X_i| = N$, and let A_f be the tensor with order N representing f. Then*

$$D(f) \geq \log(N^k/W(A_f)).$$

Theorem 3.3 *Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{1, -1\}$ be a function where for all i, $|X_i| = N$, and let A_f be the tensor with order N representing f. Then*

$$D_{\frac{1}{2}-\epsilon}(f) \geq \log(2\epsilon N^k/W(A_f)).$$

Theorem 3.4 *Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{1, -1\}$ be a function where for all i, $|X_i| = N$, and let A_f be the tensor with order N representing f. Then*

$$R_{\frac{1}{2}-\epsilon}(f) \geq \log(2\epsilon N^k/W(A_f)).$$

The following shows that the discrepancy can not be too much smaller than the bound given by the weight:

Theorem 3.5 *Let $f : X_1 \times \cdots \times X_k \to \{1, -1\}$ be a function where for all i, $|X_i| = N$. Let A_f be the tensor with order N representing f. Then*

$$\mathrm{Disc}(f) \geq W(A_f)/(2^k N^k).$$

Proof: Consider a set of lines used to generate $W(A_f)$, that is, lines which give rise to a tensor of maximum excess when they are flipped. Partition the entries of A_f into two groups according to whether they are flipped by those lines from the set which are along the first dimension. Further partition each of those groups according to whether they are flipped by those lines from the set which are along the second dimension. Continue this process for each of the k dimensions giving 2^k groups, some of which may be empty. Since at each step we partitioned the entries according to a cylinder, each of the 2^k groups are cylinder intersections. At least one of these cylinder intersections must have entry sum with absolute value at least $W(A_f)/2^k$. Thus this cylinder intersection has a bias of at least $W(A_f)/(2^k N^k)$ on f, and so the discrepancy of f is at least $W(A_f)/(2^k N^k)$. ∎

There is a trivial upper bound of N^k on the weight of a k-dimensional tensor with order N. We give a general lower bound on weight and show that it is reasonably tight. It is well known that for any $N \times N$ matrix A, $W(A) \geq N^{3/2}/\sqrt{2}$. (See Theorem 5.1 in [2]; see also [15], [7].)

Theorem 3.6 *Let A be a k-dimensional tensor with order N. Then*

$$W(A) \geq N^{k-(1/2)}/\sqrt{2}.$$

Proof: Consider the set of matrices formed by fixing all but the first two dimensions of A. Each of the matrices has weight at least $N^{3/2}/\sqrt{2}$. They do not intersect, so the lines of each matrix can be flipped independently of the other matrices. This gives a tensor weight of at least $N^{k-2}(N^{3/2}/\sqrt{2})$. ∎

Theorem 3.7 *For any N and k, a $(1 - 2^{-kN^{k-1}})$-fraction of the k-dimensional tensors with order N have weight $O(\sqrt{k}N^{k-(1/2)})$.*

Proof: Consider the effect of flipping a particular set of lines on a random tensor A. The entries in the flipped tensor A' are all independent random variables with

equal probability of being 1 or -1. Thus, using Chernoff's bound,

$$\Pr[\text{sum of entries of } A' \geq cN^{k-(1/2)}] \leq e^{-c^2 N^{k-1}/2}.$$

The k players each have N^{k-1} lines available to flip, so there are $2^{k(N^{k-1})}$ sets of lines. Summing this probability over all of these sets gives the probability that one of them generates a tensor with entry sum above $cN^{k-(1/2)}$ is no more than

$$2^{k(N^{k-1})} e^{-c^2 N^{k-1}/2} = 2^{N^{k-1}(k - c^2 \log e/2)}.$$

Consider $c = 2\sqrt{(k/\log e)}$. Then the probability is at most $2^{-kN^{k-1}}$. Thus at least a $(1 - 2^{-kN^{k-1}})$-fraction of the tensors have weight at most $cN^{k-(1/2)}$. ∎

Given that almost all tensors have weight $O(\sqrt{k}N^{k-(1/2)})$, we examine how this affects communication complexity:

Theorem 3.8 *For any n, $k < n$, and $c < n$ a k-dimensional tensor with order $N = 2^n$ that has weight at most $c\sqrt{k}N^{k-(1/2)}$ represents a function with multiparty communication complexity $\Omega(n)$.*

Proof: According to Theorem 3.2, the function represented has communication complexity at least

$$\begin{aligned}
&\log(N^k/(c\sqrt{k}N^{k-(1/2)})) \\
&= \log(N^{1/2}/(c\sqrt{k})) \\
&= (\log(N) - \log(c\sqrt{k}))/2 \\
&= n/2 - \log(c\sqrt{k}) \\
&= \Omega(n). \quad ∎
\end{aligned}$$

Thus almost all functions have weight low enough to give an asymptotically optimal multiparty communication complexity lower bound. Although no explicit

functions are known to have this low weight, the method appears capable of producing interesting lower bounds.

Chapter 4

Hadamard Tensors

Hadamard tensors are the first known extension of Hadamard matrices into three or more dimensions. Hadamard matrices are known to have minimal weight [7]. It is our hope that extending the property will lead to tensors with minimal weight, which would give good lower bounds on multiparty communication complexity.

In this chapter we will give the extension and show a lower bound on the multiparty communication complexity of functions represented by Hadamard tensors. In the next chapter we will give an explicit construction of a family of Hadamard tensors and examine some of their properties. Finally, in Chapter 6, we will examine how some previous lower bounds fit into a "nearly" Hadamard framework.

4.1 Extending the Hadamard Property

Hadamard matrices provide examples of functions which are difficult to compute in the two-party communication complexity model. In particular, Chor and Goldreich [10] showed that any function represented by a Hadamard matrix with order 2^n requires $\Omega(n)$ bits of communication to compute. Hadamard matrices also provide boundary examples for many other quantities in linear algebra. In particular they

have maximal rank and minimal norm. If this property can be extended to an object capable of representing multiparty communication problems, then these functions are likely to require high communication complexity. The Hadamard property has not previously been extended beyond two dimensions. In fact, Chung and Tetali [12] mention that it is not clear how to generalize Hadamard matrices to tensors. We present our attempt in this chapter.

The Hadamard property has a standard definition for matrices:

Definition 4.1 *A square matrix restricted to* ± 1 *entries is* Hadamard *if the inner product of any two distinct rows is 0.*

It is equivalent to state the condition for columns: The product of any two distinct rows is 0 if and only if the product of any two distinct columns is 0. To extend the property to more dimensions, we take an inner product of an increased number of lines according to the following definition:

Definition 4.2 *Let A be a k-dimensional tensor with order N restricted to* ± 1 *entries. Choose a dimension i. For every dimension $j \neq i$, choose two distinct indices y_j and z_j. Selecting for each $j \neq i$ either y_j or z_j gives a line of A along dimension i. Consider the set of 2^{k-1} lines that can be formed in this way. The absolute value of the sum of the entries of the vector formed by taking the entrywise product of these lines is called a* line product. *A is* Hadamard *if for any choices of i, y_j's, and z_j's the line product is 0.*

Just as the Hadamard condition on the rows of a matrix implies the condition on the columns, we show the following for Hadamard tensors:

Theorem 4.1 *Let A be a k-dimensional tensor with order N restricted to* ± 1 *entries. Suppose for a particular dimension i that all of A's line products are 0. Then A is Hadamard.*

Proof: Suppose all line products using dimension i_1 are 0. Choose a different dimension i_2. Choose for all other dimensions j different from i_1 and i_2 two distinct indices y_j and z_j. Consider the matrix formed with rows indexed by dimension i_1 and columns by dimension i_2 where each entry is the product of the 2^{k-2} entries of A corresponding to that coordinate and the choices of y_j's and z_j's. Since we have assumed all line products using i_1 are 0, the inner product of any two of the columns of this matrix is 0. Thus the inner product of any two rows of this matrix is also 0, and all line products using dimension i_2 are also 0. ∎

Here is a different way to view the same property:

Definition 4.3 *A tensor in three dimensions is* Hadamard *if the entrywise product of any two distinct faces along the same dimension is a Hadamard matrix. A tensor in more than three dimensions is* Hadamard *if the entrywise product of any two distinct faces along the same dimension is a Hadamard tensor (in one less dimension).*

Lemma 4.2 *The two definitions of the Hadamard property for tensors are equivalent.*

Proof: Let A be a k-dimensional tensor. According to the original Hadamard definition and Theorem 4.1 A is Hadamard if and only if for all choices of y_i's and z_i's

$$\sum_{x_k \in X_k} \prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} A(x_1, x_2, \ldots, x_k) \; = 0.$$

Without loss of generality we consider a face product along the $(k-1)$-th dimension. Let $A^{y_{k-1}}$ be the face of A with the $(k-1)$-th dimension fixed to be y_{k-1}. Let $A^{z_{k-1}}$ be the face of A with the $(k-1)$-th dimension fixed to be z_{k-1}. The new definition specifies that A is Hadamard if and only if for any choices of y_i's

and z_i's

$$\sum_{x_k \in X_k} \prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-2} \in \{y_{k-2}, z_{k-2}\}}} A^{y_{k-1}}(x_1, x_2, \ldots, x_{k-2}, x_k) A^{z_{k-1}}(x_1, x_2, \ldots, x_{k-2}, x_k) \ = 0.$$

These sums are the same. ∎

Lemma 4.3 *A tensor with the Hadamard property retains that property under the arbitrary flipping of its lines.*

Proof: This follows by induction from the characterization of Hadamard tensors given by Lemma 4.2. The result holds for matrices since after flipping a row or column any row inner product or column inner product that was 0 remains 0.

Suppose the result holds for tensors of dimension $k-1$. Let A be a k-dimensional tensor. Consider any face product $A^{y_i} \circ A^{z_i}$ (where A^{x_i} denotes the face formed by fixing the i-th coordinate to x_i). Flipping a line of A may miss A^{y_i} and A^{z_i} entirely, intersect both in one entry, or flip an entire line of A^{y_i} or A^{z_i}. In the first case the face product is unaffected. In the second case the face product is unchanged since the corresponding entry is negated twice. In the third case the face product has a line flipped. By the induction hypothesis this is still a Hadamard tensor. ∎

The Hadamard property is preserved for matrices under Kronecker product. Thus it is easy to construct large Hadamard matrices. For example, the inner product function (frequently called IP) on two strings of length n is -1 if the number of indices of the strings both containing ones is odd and 1 otherwise. The matrix representing this function is Hadamard and can be constructed by repeatedly taking Kronecker products with the matrix representing the AND function: $\left[\begin{smallmatrix} -1 & 1 \\ 1 & 1 \end{smallmatrix}\right]$.

Unfortunately our Hadamard property is not preserved by Kronecker product for higher-dimensional objects. To show this, we take three steps. First we show

a process by which tensors can be placed in canonical form by flipping lines. Next we show that any tensor achievable by flipping lines in two tensors and then taking their Kronecker product is also achievable by taking the product first, then flipping lines. Finally we exhibit a line product of the Kronecker product of two tensors in canonical form that is not 0. Thus we prove the more general theorem that no tensor which can be decomposed as the Kronecker product of two smaller tensors has the Hadamard property.

Definition 4.4 *Let A be a k-dimensional tensor restricted to ± 1 entries. We say that tensor A' is the* canonical form *of A if it can be derived from A by flipping lines of A, and it has a 1 entry at every position indexed by 0 in at least one dimension.*

Lemma 4.4 *Let A be a k-dimensional tensor restricted to ± 1 entries. Then A has a canonical form.*

Proof: We show that there is a series of flips achieving the canonical form. In order, from the first dimension to the last, for each -1 entry corresponding to index 0, flip the corresponding line.

After processing the first dimension, there will be no -1 entries corresponding to index 0 in the first dimension, as all of these entries will have been flipped. None of them will have been flipped more than once since all lines corresponding to a particular dimension contain distinct entries.

Assume that there are no -1 entries corresponding to index 0 in the first i dimensions. Consider the effect of flipping lines corresponding to -1 entries with index 0 in the $(i+1)$-th dimension. None of the entries will be indexed by 0 in the first i dimensions. Thus the corresponding lines do not contain any entries indexed by 0 in the first i dimensions. (Along a line associated with dimension $(i+1)$, only the $(i+1)$-th index changes.) Thus this step removes all -1 entries corresponding to index 0 in the $(i+1)$-th dimension without disturbing entries corresponding to index 0 in the first i dimensions. ∎

Lemma 4.5 *Let A and B be tensors. Let C be a tensor $A' \otimes B'$ where A' is a tensor formed by flipping lines of A and B' is formed by flipping lines of B. Then C can also be formed by flipping lines of $A \otimes B$.*

Proof: Each line of A corresponds to several lines in the Kronecker product, one for each entry of B. However any line in the Kronecker product corresponds to the same line of A repeated several times. Thus flipping all lines corresponding to a line of A in the Kronecker product is equivalent to flipping the line of A originally.

Each line of B corresponds to a sub-tensor in the Kronecker product containing several copies of A or its opposite, one for each entry in B. Flipping all entries in all of these copies is possible in the Kronecker product since they are made up of lines in the larger tensor. This is equivalent to flipping a line of the original B. ∎

Just to be clear, this lemma is one-directional. It is not possible in general to flip lines in the tensor product and have the same tensor achievable by flipping lines in the original. For an easy example of this, imagine flipping only one line in the tensor product. This does not correspond to any set of line flips in the original tensors.

Lemma 4.6 *Let A and B be tensors in canonical form. Then $A \otimes B$ is not a Hadamard tensor.*

Proof: We show this by giving a non-zero line product. To specify a line product, we must give for all but one dimension two inputs for that dimension. In the case of the Kronecker product of A and B, we specify the values for A and B in all but one dimension. Then a line takes on all combinations of values possible for both A and B in the unspecified dimension.

Consider the following line product of $A \otimes B$: We take the product along the first dimension. In the second dimension we choose the entry corresponding to index 0 in A and 0 in B and we choose an entry corresponding to index 0 in A and

an arbitrary non-zero index of B. We choose similarly for the third dimension. Our first entry is the one corresponding to index 0 in A and 0 in B, but our second entry corresponds to 0 in B and an arbitrary non-zero index of A. We make arbitrary choices if there are any other dimensions.

Notice that every entry of every line in the line product corresponds to a zero index of A (in the second dimension) and a zero index of B (in the third dimension). Since A and B are in canonical form, this means that these entries are all 1. Thus the line product is positive. ∎

It may seem that lack of closure under Kronecker product disqualifies our Hadamard definition. However it does not seem possible to preserve *all* of the properties of Hadamard matrices. The smallest interesting tensor is the 2x2x2 cube. Assuming the Hadamard property is preserved by line flips, then there are only two cases of the 2x2x2 cube to consider, an even number of -1's and an odd number of -1's. One of the widely studied multiparty problems, generalized inner product, is an extension of inner product which is represented by a Hadamard matrix, and it is the (possibly repeated) Kronecker product of the 2x2x2 cube with itself. GIP is known to have multiparty communication complexity far less than linear, and it is closed under Kronecker product. Thus it must be the case that the correct extension of Hadamard matrices either does not have a 2x2x2 version, or that it cannot keep both the line flipping property and the Kronecker product property.

In a similar way, it may not be possible to keep both the communication complexity properties of Hadamard matrices and their Kronecker product closure. Even though our extension, is not closed under Kronecker product, none of the functions represented by Hadamard tensors are known to have low multiparty communication complexity, leaving the possibility that it is the correct generalization with respect to communication complexity properties.

4.1.1 Constructivity and Largeness of Hadamard Tensors

As one of the goals for studying Hadamard tensors is to eventually get circuit lower bounds for ACC^0 it is natural to ask whether the property is "natural" in the sense of Razborov and Rudich [35]. According to their definitions, a property is "constructive" if it can be decided in time polynomial in the truth table size of the function. A property has "largeness" if the probability that a random function has the property is more than inverse polynomial in the truth table size.

We can show that the Hadamard property is constructive since in the time bound required we can compute all of the necessary line products:

Lemma 4.7 *Let $f : (\{0,1\}^n)^k \to \{1,-1\}$ be a function, and let A_f be the k-dimensional tensor with order 2^n representing f. Then for any choice of dimension, there are $(2^{n-1}(2^n-1))^{k-1}$ ways to choose entries of A_f to form a line product along that dimension.*

Proof: In each dimension there are 2^n entries, giving $\binom{2^n}{2}$ ways to choose two distinct entries along that dimension. $\binom{2^n}{2}$ simplifies to $2^{n-1}(2^n-1)$. Since the line product is taken along one dimension, we choose entries along $(k-1)$ dimensions, giving the desired result. ∎

Theorem 4.8 *The Hadamard property is constructive.*

Proof: To show a function is represented by a Hadamard tensor it suffices to check all line products along one dimension. The previous lemma shows that for $f : (\{0,1\}^n)^k \to \{1,-1\}$ the number of line products to compute is polynomial in 2^{nk}, the size of the truth table of f. Computing a line product requires multiplying 2^{k-1} entries together 2^n times, and then adding the 2^n results so each computation is polynomial in 2^{nk}. Thus all line products can be checked in the desired time. ∎

On the other hand, the Hadamard property is not large, as we show by proving a bound which is smaller than inverse polynomial in 2^{nk}:

Theorem 4.9 *Let A be a random k-dimensional tensor with order 2^n restricted to ± 1 entries.*

$$\Pr[A \text{ is Hadamard}] \leq 2^{-(2^n-n+1)(2^{(n-1)(k-1)})}.$$

Proof: Consider the lines along an arbitrary dimension. There are $2^{n(k-1)}$ different lines. A line product uses 2^{k-1} of the lines, so it is possible to form $2^{n(k-1)}/2^{k-1} = 2^{(n-1)(k-1)}$ line products without using any line more than once. The values of these line products in a random tensor are independent.

Now consider the value of a random line product. Taking the entrywise product of the chosen lines with random entries gives a line with random entries, so the probability that the line product is zero is the same as the probability that a fair coin flipped 2^n times has exactly 2^{n-1} heads. This probability is exactly $\binom{2^n}{2^{n-1}}/2^{2^n}$. This value is no more than $2^{n-1}/2^{2^n}$.

To be Hadamard all line products must be zero. The probability that $2^{(n-1)(k-1)}$ independent line products is zero is no more than $(2^{n-1}/2^{2^n})^{2^{(n-1)(k-1)}}$ which is $2^{-(2^n-n+1)(2^{(n-1)(k-1)})}$. ∎

Razborov and Rudich showed that combinatorial properties that are both constructive and large cannot be used to prove good lower bounds on some circuit classes. The fact that the Hadamard property is not large gives evidence that it may be useful for lower bound proofs.

4.2 Discrepancy of Hadamard Tensors

The weight of a tensor can be bounded using its face products. First we need the following definition:

Definition 4.5 *Let A be a k-dimensional tensor. Consider the set of tensors formed by taking the entrywise product of two distinct, arbitrary faces of A along the same dimension after flipping arbitrary lines of A. Denote by A^* the tensor in that set of*

maximal excess. (Note that A^* is a $(k-1)$-dimensional tensor.)

Lemma 4.10 *For any k-dimensional tensor A with order N,*

$$W(A)^2 \leq N^{k-1}(N^k + (N^2 - N)(W(A^*))).$$

Proof: Let A be a k-dimensional tensor with order N. Let A' be the tensor formed by flipping lines of A so that the excess of A' is the weight of A. By definition,

$$|W(A)|^2 = \Big| \sum_{\vec{x} \in \{1,\ldots,N\}^k} A'(\vec{x}) \Big|^2.$$

Using the Cauchy-Schwarz inequality, this is at most

$$N^{k-1} \sum_{x_1,\ldots,x_{k-1}} \left(\sum_{x_k} A'(\vec{x}) \right)^2.$$

Rearranging terms gives

$$N^{k-1} \sum_{x_1,\ldots,x_{k-1}} \left(\sum_{x_{k_1},x_{k_2}} A'(x_1,\ldots,x_{k-1},x_{k_1}) A'(x_1,\ldots,x_{k-1},x_{k_2}) \right).$$

When $x_{k_1} = x_{k_2}$, this gives a product of a face of A' with itself. This happens N times, each with entry sum N^{k-1}. So the expression can be reduced to

$$N^{k-1} \left(N^k + \sum_{x_{k_1} \neq x_{k_2}} \sum_{x_1,\ldots,x_{k-1}} A'(x_1,\ldots,x_{k-1},x_{k_1}) A'(x_1,\ldots,x_{k-1},x_{k_2}) \right).$$

Using $W(A^*)$ as a bound on the excess of the face products gives a reduction to

$$N^{k-1}\big(N^k + (N^2 - N)(W(A^*))\big). \quad \blacksquare$$

Since we can estimate the weight of a tensor from the weight of its face products, and we know that Hadamard tensors have Hadamard face products we get a bound on the weight of a Hadamard tensor:

Theorem 4.11 *Let A be a k-dimensional Hadamard tensor with order N.*

$$W(A) \leq \phi N^{k-(1/2^{k-1})}$$

where $\phi = (1+\sqrt{5})/2$.

Proof: Let A be a three dimensional Hadamard tensor with order N. Then A^* is a Hadamard matrix. The maximal excess of a Hadamard matrix is $N^{3/2}$ [7].

$$\begin{aligned} W(A) &\leq (N^2(N^3 + (N^2 - N)(N^{3/2})))^{1/2} \\ &\leq (N^2(2N^{7/2}))^{1/2} \\ &\leq \sqrt{2}N^{3-(1/4)} \\ &\leq \phi N^{3-(1/4)}. \end{aligned}$$

Suppose the theorem holds for all $(k-1)$-dimensional Hadamard tensors.

$$\begin{aligned} W(A) &\leq (N^{k-1}(N^k + (N^2 - N)(\phi N^{(k-1)-(1/2^{k-2})})))^{1/2} \\ &\leq (N^{k-1}((1+\phi)(N^{(k+1)-(1/2^{k-2})})))^{1/2} \\ &= ((1+\phi)N^{2k-(1/2^{k-2})})^{1/2} \\ &= \sqrt{1+\phi}N^{k-(1/2^{k-1})} \\ &= \phi N^{k-(1/2^{k-1})}. \quad \blacksquare \end{aligned}$$

4.2.1 Communication Complexity Lower Bounds

Using our new methods relating weight to discrepancy and multiparty communication complexity, these results follow:

Theorem 4.12 *Let* $f : (\{0,1\}^n)^k \to \{1,-1\}$ *be a function represented by a* k*-dimensional Hadamard tensor* A_f *with order* $N = 2^n$*. Then*

$$\mathrm{Disc}(f) \leq \phi N^{-1/2^{k-1}}.$$

Proof: This follows directly from Theorem 3.1 and Theorem 4.11. ∎

Theorem 4.13 *Let* $f : (\{0,1\}^n)^k \to \{1,-1\}$ *be a function represented by a* k*-dimensional Hadamard tensor* A_f *with order* $N = 2^n$*. Then*

$$D(f) = \Omega(n/2^k).$$

Proof: This follows directly from Theorem 4.11 and Theorem 3.2. ∎

Using Theorem 3.3 and Theorem 3.4 we also get the following lower bounds on distributional complexity and randomized complexity:

Theorem 4.14 *Let* $f : (\{0,1\}^n)^k \to \{1,-1\}$ *be a function represented by a* k*-dimensional Hadamard tensor* A_f *with order* $N = 2^n$*. Then*

$$D_{\frac{1}{2}-\epsilon}(f) = \Omega((n/2^k) + \log \epsilon).$$

Theorem 4.15 *Let* $f : (\{0,1\}^n)^k \to \{1,-1\}$ *be a function represented by a* k*-dimensional Hadamard tensor* A_f *with order* $N = 2^n$*. Then*

$$R_{\frac{1}{2}-\epsilon}(f) = \Omega((n/2^k) + \log \epsilon).$$

Chapter 5

Constructions of Hadamard Tensors

In this chapter we consider explicit constructions of Hadamard tensors.

5.1 Finite Field Multiplication Function

The first family of Hadamard tensors is formed from multiplying appropriate elements from a finite field:

Definition 5.1 *Let $f : \{0,1\}^n \to \{1,-1\}$ be a function. Let $x_1, \ldots, x_k$ be n-bit strings. Consider each of these strings as an element in the finite field $\mathrm{GF}(2^n)$ of polynomials $p(a)$ over $\mathrm{GF}(2)$, where the i-th bit of each input $(0 \leq i \leq n-1)$ indicates whether the element contains the term a^i. Define the the finite field multiplication function, $\mathrm{FFM}_n^k(f)(x_1, \ldots, x_k)$ to be $f(x_1 x_2 \cdots x_k)$, where $x_1 x_2 \cdots x_k$ indicates multiplication in the field.*

In particular, we will be interested in the case where f is a nontrivial additive character of the field. An additive character $\chi : \mathcal{F}_1 \to \mathcal{F}_2$ is a homomorphism such that for any $a, b \in \mathcal{F}_1$, $\chi(a)\chi(b) = \chi(a+b)$. These characters for $\mathrm{GF}(2^n)$ are the

functions that depend only on the parity of a nonempty subset of the input bits and that map the all-0 string to 1.

Although all finite fields of order 2^n are isomorphic, it is necessary to specify exactly which one is being used for the function to be well defined. We use the deterministic algorithm developed by Shoup [37] to construct the irreducible polynomial necessary to completely specify the multiplication operator. This choice is arbitrary; our results hold for any specification of the irreducible polynomials.

Theorem 5.1 *Let* $f : \{0,1\}^n \rightarrow \{1,-1\}$ *be a nontrivial additive character of* $\mathrm{GF}(2^n)$. $\mathrm{FFM}_n^k(f)$ *is Hadamard.*

Proof: First we show that the function gives a Hadamard matrix when $k = 2$. In the row corresponding to the 0 element of the field all entries are 1. Consider any other row of the matrix. It consists of the function f applied to the elements generated by multiplying a nonzero field element by every other field element. Thus it consists of applying f to all of the field elements. Since f is a nontrivial additive character, this gives half 1's and half -1's. Consider the inner product of two of the rows indexed by unique field elements i_1 and i_2. The element of the inner product corresponding to the column indexed by field element j is $f(i_1 j)f(i_2 j)$ which by the definition of f is the same as $f(i_1 j \oplus i_2 j)$ (viewing $i_1 j$ and $i_2 j$ as strings) which is the same as $f(i_1 j + i_2 j)$ using addition in the field. This is the same as $f((i_1 + i_2)j)$. Since $(i_1 + i_2)$ is an element of the field, the row formed by the entrywise product of rows i_1 and i_2 is the same as the row corresponding to $(i_1 + i_2)$. Elements in the field are their own additive inverses, so $(i_1 + i_2) \neq 0$, and this is not the all 1 row. Thus the entrywise product of rows i_1 and i_2 is a row with half 1's and half -1's, and the inner product of rows i_1 and i_2 is 0.

The general proof for $k \geq 2$ is similar. Consider this sum:

$$\sum_{x_k} \prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} f(x_1 x_2 \cdots x_k).$$

Multiplying entries in the tensor is equivalent to adding in the field, so this is the same as

$$\sum_{x_k} f(\sum_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} x_1 x_2 \cdots x_k)$$

which is the same as

$$\sum_{x_k} f((y_1 + z_1)(y_2 + z_2) \cdots (y_{k-1} + z_{k-1}) x_k).$$

Each of the terms $(y_i + z_i)$ is non-zero since elements are their own additive inverse, and for each i, $y_i \neq z_i$. The product of non-zero elements is non-zero. Thus for some fixed nonzero w, the sum is the same as $\sum_{x_k} f(x_k w)$. Multiplying a fixed nonzero element by all of the field elements generates all of the field elements, so this is the same as applying f to all of the field elements and taking the sum of its outputs, giving 0. ∎

5.1.1 FFM Considered as a Matrix

Since the FFM function is defined for as few as two players, it is interesting to consider the Hadamard matrices it produces. In fact, the function for two players is isomorphic to the inner product function:

Theorem 5.2 *Let* $f : \{0,1\}^n \to \{1,-1\}$ *be a nontrivial additive character of* $\mathrm{GF}(2^n)$. $\mathrm{FFM}_n^2(f)$ *is isomorphic to the inner product function* $\mathrm{IP} : \{0,1\}^n \times \{0,1\}^n \to \{1,-1\}$ *which is* 1 *when the number of indices where both inputs are*

1 *is odd and* -1 *otherwise.*

Proof: To begin, we prove the result for the function that is 1 only when the first bit of its input string is 1, call it f_0. In other words, $\mathrm{FFM}_n^2(f_0)$ is 1 when the polynomial representation of the product of the two field elements includes a constant term.

Any monomial a^m can be reduced using the irreducible polynomial to a representative polynomial of degree less than n. Consider the multiplication of a polynomial $p(a)$ by a fixed monomial a^i. The product contains a constant term if and only if an odd number of the monomials a^{i+j} contain a constant term in their reduced form where j ranges over the powers of a contained in $p(a)$. Consider the n-bit string S_{a^i} formed by the following rule: The jth bit of the string is 0 when a^{i+j} does not contain a constant term in its reduced form and 1 when it does. Then the constant term of the polynomial formed by multiplying a^i and $p(a)$ is the same as the inner product of $p(a)$ represented as a string and S_{a^i}.

For a fixed polynomial $q(a) = \sum_{i \in I} a^i$, form the string $S_{q(a)}$ by taking the bitwise parity of S_{a^i} for all $i \in I$. The constant term of the polynomial formed by multiplying $q(a)$ with arbitrary polynomial $p(a)$ is the same as the inner product of $p(a)$ represented as a string and $S_{q(a)}$.

Thus $\mathrm{FFM}_n^2(f_0)(x_1, x_2)$ is the same as the inner product of x_1 and S_{x_2}.

We must also show that S_{x_2} ranges over all n-bit strings. Suppose that for two polynomials $p(a)$ and $q(a)$, $S_{p(a)} = S_{q(a)}$. Then $S_{p(a)+q(a)}$ has all bits 0. So multiplying by $p(a) + q(a)$ never gives a constant term, only true if $p(a) + q(a) = 0$, and thus $p(a) = q(a)$. So every polynomial $p(a)$ generates a unique string, and since there are 2^n polynomials, all 2^n strings are generated.

To extend the proof to all f, define strings $S_{p(a),a^i}$ in the same manner as the $S_{p(a)}$ defined above, except that we base the string on the inclusion of the term a^i instead of the constant term. (The earlier strings are those of the form $S_{p(a),a^0}$.) The same proof with one of these strings shows the isomorphism for an f which

depends on a single monomial. Finally, define $S_{p(a),q(a)}$ as $\sum_{a^i \in q(a)} S_{p(a),a^i}$. The same proof with one of these strings shows the isomorphism for an f which gives the parity of the output bits corresponding to the terms of $q(a)$. ∎

5.1.2 Circuit Complexity of FFM

One motivation for finding lower bounds on multiparty communication complexity is to exhibit an explicit function family not contained in ACC^0. Our bound is not large enough to show FFM lies outside ACC^0, and the question is still open. We show its inclusion in a larger circuit class. It is easy to see that FFM belongs to uniform class P.

Theorem 5.3 *The family of functions given by* $\mathrm{FFM}_n^k(f)$ *as* f *ranges over a family of nontrivial additive characters is in* ACC *with depth* $\log k$.

To prove Theorem 5.3 we need a lemma about multiplication in a finite field:

Lemma 5.4 *Multiplication in a finite field is bilinear. That is, for every* n *there is a set of* n *binary matrices* $A^{(0)}, \ldots, A^{(n-1)}$ *with order* n *such that for any two field elements* x *and* y *viewed as binary vectors, the* t*-th bit of the product* xy *is given by* $xA^{(t)}y$. *(All operations are in* GF(2)*.)*

Proof: First consider a matrix A of polynomials in a whose entries are given by the simple formula $A_{i,j} = a^{(i+j)}$. For any two field elements x and y represented as binary vectors, their product represented as a polynomial is given by xAy. This product may not be in canonical form as it may contain powers of a above $(n-1)$. Consider the matrix A' which is the same as A except that the entries are placed in canonical form using the irreducible polynomial chosen for the representation of the field. Now $xA'y$ is the canonical representation in polynomial form of the product of x and y.

Create n binary matrices $A^{(0)}, \ldots, A^{(n-1)}$ where $A^{(t)}_{i,j}$ is 1 if $A'_{i,j}$ contains the term a^t and 0 otherwise. Then the product $xA^{(t)}y$ tells whether xy contains the term a^t, the t-th bit of the representation in vector form. ∎

Proof: (of Theorem 5.3) Multiplication of the finite field elements using these matrices can be done in constant depth, and k multiplications are sufficient, giving the $\log k$ depth. ∎

5.1.3 Other Hadamard Constructions

Next we consider a variation on the finite field multiplication function and show that it is also Hadamard:

Definition 5.2 *Let $x_1, \ldots, x_k$ be n-bit strings. Consider the first $(k-1)$ of these as an element in the finite field* GF(2^n) *of polynomials $p(a)$ over* GF(2). *Define the the finite field multiplication inner product function,* FFMIP$^k_n(x_1, \ldots, x_k)$ *to be $(x_1 x_2 \cdots x_{k-1}) \oplus x_k$ (where $x_1 x_2 \cdots x_k$ indicates multiplication in the field).*

Theorem 5.5 FFMIP *is Hadamard.*

Proof: Consider a tensor T representing FFMIP^k_n. According to Lemma 4.2, the tensor is Hadamard if the product of any two face products along a particular dimension is Hadamard. A face of T along the k-th dimension is just an instance of FFM since the k-th input selects which character function to apply to the product of the other inputs (except for the special case where the k-th input is all 0, representing the trivial character). The entrywise product of two distinct faces along this dimension is still a tensor representing FFM since the characters form a group, and thus the product of applying two characters to the polynomials is equivalent to applying a different character. In this composition it is no longer possible to get the trivial character since no two distinct characters compose to it. Thus the face product is Hadamard, and so T is Hadamard. ∎

It is conjectured that a Hadamard matrix exists for any order that is divisible by 4, and the smallest multiple of 4 for which the question is open is 668 [22]. Because Hadamard matrices are closed under Kronecker product a Hadamard matrix can be constructed for any order mn for which Hadamard matrices are known for orders m and n. It is open to show Hadamard tensors for any orders other than powers of two, or to construct Hadamard tensors based on some function other than finite field multiplication. One of the common constructions of Hadamard matrices is the Paley type Hadamard matrix based on quadratic characters of sums. As we see in the next chapter, the most natural extension of this matrix to higher dimensions is "almost" Hadamard.

Chapter 6

Relaxations of the Hadamard Property

We will now consider two relaxations of our Hadamard property, first tensors whose Hadamard line products are always small, and second tensors whose line products are small on average. This will allow us to examine some of the previous lower bounds in a Hadamard framework.

6.1 Small Line Products

Definition 6.1 *A matrix is* d-Hadamard *if the maximum inner product of two rows is* d*. A tensor is* d-Hadamard *if each of its line products is at most* d*.*

Theorem 6.1 *Let* A *be a* d*-Hadamard matrix with order* N*. Then*

$$W(A) \leq N^{3/2}(1+d)^{1/2}.$$

Proof: Let A' be a matrix formed by flipping lines of A. We begin by applying the Cauchy-Schwarz inequality to an expression for the excess of A'.

$$\begin{aligned}
\sum_{i=1}^{N}\sum_{j=1}^{N} A'(i,j) &\leq \left[(\sum_{i=1}^{N} 1)(\sum_{i=1}^{N}(\sum_{j=1}^{N} A'(i,j))^2)\right]^{1/2} && (6.1)\\
&= \left[N\sum_{i=1}^{N}(\sum_{j_1=1}^{N} A'(i,j_1))(\sum_{j_2=1}^{N} A'(i,j_2))\right]^{1/2} && (6.2)\\
&= \left[N\sum_{j_1=1}^{N}\sum_{j_2=1}^{N}\sum_{i=1}^{N} A'(i,j_1)A'(i,j_2)\right]^{1/2}. && (6.3)
\end{aligned}$$

For a fixed j_1 and j_2, the sum over i is the inner product of two rows of A'. Flipping rows or columns of A' does not affect any of these inner products except possibly in sign, so

$$\left[N\sum_{j_1=1}^{N}\sum_{j_2=1}^{N}\sum_{i=1}^{N} A'(i,j_1)A'(i,j_2)\right]^{1/2} \leq \left[N\sum_{j_1=1}^{N}\sum_{j_2=1}^{N}\left|\sum_{i=1}^{N} A(i,j_1)A(i,j_2)\right|\right]^{1/2}.$$

There are $(N^2 - N)$ ways to select different rows, each of which has inner product at most d. When $j_1 = j_2$ the inner product is N. Thus this sum is at most $(N^2 + (N^2 - N)d)$, and the excess of A' is at most

$$[N(N^2 + (N^2 - N)d)]^{1/2} \leq N^{3/2}(1 + d)^{1/2}. \quad \blacksquare$$

Lemma 6.2 *A k-dimensional tensor is d-Hadamard if and only if the entrywise product of any two distinct faces along the same dimension is a d-Hadamard tensor (in one less dimension).*

Proof: The proof mirrors that of Lemma 4.2. Let A be a k-dimensional tensor. According to the d-Hadamard definition A is d-Hadamard if and only if for all choices

of y_i's and z_i's

$$\sum_{x_k \in X_k} \prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} A(x_1, x_2, \ldots, x_k) \ < d.$$

Without loss of generality we consider a face product along the $(k-1)$-th dimension. Let $A^{y_{k-1}}$ be the face of A with the $(k-1)$-th dimension fixed to be y_{k-1}. Let $A^{z_{k-1}}$ be the face of A with the $(k-1)$-th dimension fixed to be z_{k-1}. The new definition specifies that A is d-Hadamard if and only if for any choices of y_i's and z_i's

$$\sum_{x_k \in X_k} \prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-2} \in \{y_{k-2}, z_{k-2}\}}} A^{y_{k-1}}(x_1, x_2, \ldots, x_{k-2}, x_k) A^{z_{k-1}}(x_1, x_2, \ldots, x_{k-2}, x_k) \ < d.$$

These sums are the same. ∎

Theorem 6.3 *Let A be a k-dimensional, d-Hadamard tensor with order N. Then*

$$W(A) \leq \phi N^{k-1/2^{k-1}} (1+d)^{1/2^{k-1}}.$$

Proof: The proof mirrors that of Theorem 4.11. According to Lemma 4.10, $W(A)^2 \leq N^{k-1}(N^k + (N^2 - N)(W(A^*)))$ where $A*$ is the tensor of maximal excess from the face products of A. Suppose $k = 3$. Then A^* is a d-Hadamard matrix. According to Theorem 6.1, the weight of A^* is at most $N^{3/2}(1+d)^{1/2}$. Thus,

$$\begin{aligned} W(A) &\leq (N^2(N^3 + (N^2 - N)(N^{3/2})(1+d)^{1/2}))^{1/2} \\ &\leq (N^2(2N^{7/2})(1+d)^{1/2})^{1/2} \\ &\leq \sqrt{2} N^{3-(1/4)} (1+d)^{1/4} \\ &\leq \phi N^{3-(1/4)} (1+d)^{1/4}. \end{aligned}$$

Suppose the theorem holds for all $(k-1)$-dimensional d-Hadamard tensors.

$$\begin{aligned} W(A) &\leq (N^{k-1}(N^k + (N^2 - N)(\phi N^{(k-1)-(1/2^{k-2})}))(1+d)^{1/2^{k-2}})^{1/2} \\ &\leq (N^{k-1}((1+\phi)(N^{(k+1)-(1/2^{k-2})}))(1+d)^{1/2^{k-2}})^{1/2} \\ &= ((1+\phi)N^{2k-(1/2^{k-2})}(1+d)^{1/2^{k-2}})^{1/2} \\ &= \sqrt{1+\phi}N^{k-(1/2^{k-1})}(1+d)^{1/2^{k-1}} \\ &= \phi N^{k-(1/2^{k-1})}(1+d)^{1/2^{k-1}}. \quad \blacksquare \end{aligned}$$

Theorem 6.4 *Let $f : (\{0,1\}^n)^k \rightarrow \{1,-1\}$ be a function represented by a k-dimensional, d-Hadamard tensor A_f with order n. Then*

$$D(f) \geq (n - \log(\phi(1+d)))/2^{k-1}.$$

Proof: This follows directly from Theorems 6.3 and 3.2. $\blacksquare$

Using Theorem 3.3 and Theorem 3.4 we also get the following lower bounds on distributional complexity and randomized complexity:

Theorem 6.5 *Let $f : (\{0,1\}^n)^k \rightarrow \{1,-1\}$ be a function represented by a k-dimensional d-Hadamard tensor A_f with order $N = 2^n$. Then*

$$D_{\frac{1}{2}-\epsilon}(f) = \Omega(((n - \log d)/2^k) + \log \epsilon).$$

Theorem 6.6 *Let $f : (\{0,1\}^n)^k \rightarrow \{1,-1\}$ be a function represented by a k-dimensional d-Hadamard tensor A_f with order $N = 2^n$. Then*

$$R_{\frac{1}{2}-\epsilon}(f) = \Omega(((n - \log d)/2^k) + \log \epsilon).$$

6.1.1 Lower Bound on QCS

Our definition of d-Hadamard tensors is motivated by the attempt to cast the QCS function of [4] in a Hadamard framework. (We consider the version with output from $\{1, -1\}$ instead of $\{0, 1\}$.) To show it is nearly Hadamard we begin with a technical lemma:

Lemma 6.7 *Let p be any odd prime. Let $y_1, z_1, \ldots, y_k, z_k$ be elements of Z_p where for each i, $y_i \neq z_i$. Let $S_1 = \{y_1, z_1\}$. For $i > 1$, let $SY_i = y_i + S_{i-1}$, $SZ_i = z_i + S_{i-1}$, and $S_i = (SY_i \cup SZ_i) \setminus (SY_i \cap SZ_i)$. In other words, S_i contains elements formed by adding either y_i or z_i to an element of S_{i-1} but not those formed both ways. Then S_k is nonempty.*

Proof: First we show that $|S_i|$ is even for all i. By definition, $|S_1|$ is 2. Suppose $|S_{i-1}|$ is even. Adding y_i to each element of S_{i-1} gives an even number of elements in SY_i, and adding z_i to each element of S_{i-1} gives an even number of elements in SZ_i. $|S_i| = |SY_i| + |SZ_i| - 2|SY_i \cap SZ_i|$. Since each of these terms is even, $|S_i|$ is even.

Suppose S_{i-1} is nonempty and that $SY_i = SZ_i$. Let $s \in SY_{i-1}$. We know $s + y_i \in SY_i$ and $s + y_i \in SZ_i$. This means $s + y_i - z_i \in S_{i-1}$. Thus $s + (y_i - z_i) + y_i \in SY_i$. So $s + (y_i - z_i) + y_i \in SZ_i$. So $s + 2(y_i - z_i) \in S_{i-1}$. And for any n, $s + n(y_i - z_i) \in S_{i-1}$. But since $y_i - z_i$ is not 0, it generates Z_p, and so all elements of Z_p are in S_{i-1}. This is a contradiction since there are an odd number of elements of Z_p, but our construction builds sets with an even number of elements in each S_i. ∎

Theorem 6.8 $\mathrm{QCS}_{p,k}$ *is $(2^k\sqrt{p})$-Hadamard.*

Proof: Choose $y_1, z_1, \ldots y_{k-1}, z_{k-1} \in \{0, 1, \ldots, p-1\}$ where for each i, $y_i \neq z_i$. Consider the following sum from the Hadamard definition:

$$\sum_{x_k} \prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} \mathrm{QCS}_{p,k}(x_1, x_2, \ldots, x_k).$$

Let $\chi : \{0, 1, \ldots, p-1\} \to \{1, 0, -1\}$ be the quadratic character function mapping 0 to 0, nonzero squares to 1, and everything else to -1. It is a multiplicative character on GF(p), meaning that $\chi(x)\chi(y) = \chi(xy)$ for any elements of GF(p). The QCS function is the same as χ for all inputs except those where $\sum_{i=1}^{k} x_i = 0$. However for any line, this occurs exactly once. So the sum above is no more than

$$2^{k-1} + \sum_{x_k} \prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} \chi(x_1 + \cdots + x_k)$$

where the first term accounts for the entries in the line that are 0 after switching from QCS to χ. Since χ is a multiplicative character, the summation is a summation over all values of x_k of χ applied to a polynomial in x_k of degree 2^{k-1}. According to Weil's character sum estimate [42], if we can show that this polynomial is not a square of some smaller polynomial, then the sum is bounded by $(2^{k-1} - 1)\sqrt{p}$. The polynomial we are considering is already factored into terms of the form $(x + a)$. Since we are ignoring the zeros, we can reduce this polynomial by removing any $(x + a)^2$ terms. Since factoring is unique, this means that the only way this is a square is if after this removal, no terms are left. Using the lemma above this cannot happen. So the original sum is bounded by $2^{k-1} + (2^{k-1} - 1)\sqrt{p}$ which is at most $2^k\sqrt{p}$. ∎

Theorem 6.9 *Let* $k \leq \log(p^{1/4} - p^{-1/2})$. *Then*

$$D(\mathrm{QCS}_{p,k}) = \Omega((\log p)/2^k).$$

Proof: $\mathrm{QCS}_{p,k}$ was shown to be $(2^k\sqrt{p})$-Hadamard in Theorem 6.8. We use Theorem 6.4 to get

$$D(\mathrm{QCS}_{p,k}) \geq (\log(p/(\phi(1+2^k\sqrt{p}))))/2^{k-1}.$$

The theorem follows by algebraic manipulation of the right hand side:

$$\begin{aligned}
&(\log(p/(\phi(1+2^k\sqrt{p}))))/2^{k-1} \\
&= (1/2^{k-1})[\log(p/(1+2^k\sqrt{p})) - \log\phi] \\
&\geq (1/2^{k-1})[\log(p^{1/4}) - \log\phi] \\
&= (\log p - 4\log\phi)/(2^{k+1}) \\
&= \Omega((\log p)/2^k). \quad \blacksquare
\end{aligned}$$

We also get lower bounds in the distributional and randomized models which follow directly from Theorems 6.8, 6.5, and 6.6:

Theorem 6.10

$$D_{(1/2)-\epsilon}(\mathrm{QCS}_{p,k}) = \Omega((\log p)/2^k + \log\epsilon).$$

Theorem 6.11

$$R_{(1/2)-\epsilon}(\mathrm{QCS}_{p,k}) = \Omega((\log p)/2^k + \log\epsilon).$$

6.2 Small Line Products on Average

Definition 6.2 *A matrix is* d-average Hadamard *if the expected value of the inner product of two rows chosen randomly under the uniform distribution is at most* d*. A tensor is* d-average Hadamard *if the expected value of a line product of the tensor chosen randomly under the uniform distribution is at most* d*.*

Both the Hadamard property and the d-Hadamard property are preserved by entrywise face products. This allowed us to get bounds on the weight of Hadamard and d-Hadamard tensors by showing matrix bounds and proceeding inductively using the Cauchy-Schwarz inequality. We will show a similar weight bound for d-average Hadamard tensors, but the property is no longer preserved by entrywise face products, so our argument will be slightly more complicated. We begin with some necessary lemmas:

Lemma 6.12 *Suppose* A *is a* d*-average Hadamard tensor. Let* A' *be any tensor formed by flipping lines of* A*. Then for any selection of coordinates, the line product of* A *using those coordinates is the same as the corresponding line product of* A'*.*

Proof: The proof is similar to that of Lemma 4.3. Consider any line product of A. Flipping an arbitrary line of A can affect this product in one of three ways. It may miss all of the lines in the product entirely in which case the product does not change. It may intersect two of the lines of the product in one point, causing no change in the product since the corresponding entry in the entrywise product of the lines is negated twice. Finally it may flip an entire line, causing the sum of entries in the entrywise product to change sign, but keep the same magnitude. In all of these cases the line product is unchanged. ∎

Lemma 6.13 *Suppose* A *is a* k*-dimensional tensor with order* N*. Let* c *be any*

constant, and let $1 \leq b \leq (k-1)$. *Then*

$$\sum_{\substack{x_1 \\ \cdots \\ x_{k-b}}} \sum_{\substack{y_{k-b+1} \\ \cdots \\ y_k}} \sum_{\substack{z_{k-b+1} \\ \cdots \\ z_k}} \Big(N^c \prod_{\substack{x_{k-b+1} \in \{y_{k-b+1}, z_{k-b+1}\} \\ \cdots \\ x_k \in \{y_k, z_k\}}} A(x_1, \ldots, x_k)\Big)$$

$$\leq \Big(\sum_{\substack{x_1 \\ \cdots \\ x_{k-b-1}}} \sum_{\substack{y_{k-b} \\ \cdots \\ y_k}} \sum_{\substack{z_{k-b} \\ \cdots \\ z_k}} \Big(N^{2c+b+k-1} \prod_{\substack{x_{k-b} \in \{y_{k-b}, z_{k-b}\} \\ \cdots \\ x_k \in \{y_k, z_k\}}} A(x_1, \ldots, x_k)\Big)\Big)^{1/2}.$$

Proof: We begin with

$$\sum_{\substack{x_1 \\ \cdots \\ x_{k-b}}} \sum_{\substack{y_{k-b+1} \\ \cdots \\ y_k}} \sum_{\substack{z_{k-b+1} \\ \cdots \\ z_k}} \Big(N^c \prod_{\substack{x_{k-b+1} \in \{y_{k-b+1}, z_{k-b+1}\} \\ \cdots \\ x_k \in \{y_k, z_k\}}} A(x_1, \ldots, x_k)\Big).$$

Using the Cauchy-Schwarz inequality, this is no more than

$$\Big(\Big(\sum_{\substack{x_1 \\ \cdots \\ x_{k-b-1}}} \sum_{\substack{y_{k-b+1} \\ \cdots \\ y_k}} \sum_{\substack{z_{k-b+1} \\ \cdots \\ z_k}} \Big(\sum_{x_{k-b}} N^c\Big)^2\Big)$$

$$\Big(\sum_{\substack{x_1 \\ \cdots \\ x_{k-b-1}}} \sum_{\substack{y_{k-b+1} \\ \cdots \\ y_k}} \sum_{\substack{z_{k-b+1} \\ \cdots \\ z_k}} \Big(\sum_{x_{k-b}} \prod_{\substack{x_{k-b+1} \in \{y_{k-b+1}, z_{k-b+1}\} \\ \cdots \\ x_k \in \{y_k, z_k\}}} A(x_1, \ldots, x_k)\Big)^2\Big)\Big)^{1/2}.$$

The first part of this is a repeated summation of N^{2c}, so this is equivalent to

$$\Big(N^{2c+b+k-1} \sum_{\substack{x_1 \\ \cdots \\ x_{k-b-1}}} \sum_{\substack{y_{k-b+1} \\ \cdots \\ y_k}} \sum_{\substack{z_{k-b+1} \\ \cdots \\ z_k}} \Big(\sum_{x_{k-b}} \prod_{\substack{x_{k-b+1} \in \{y_{k-b+1}, z_{k-b+1}\} \\ \cdots \\ x_k \in \{y_k, z_k\}}} A(x_1, \ldots, x_k)\Big)^2\Big)^{1/2}.$$

We can replace the squared summation with

$$\Big(\sum_{y_{k-b}} \prod_{\substack{x_{k-b+1}\in\{y_{k-b+1},z_{k-b+1}\}\\ \cdots \\ x_k\in\{y_k,z_k\}}} A(x_1,\ldots,x_{k-b-1},y_{k-b},x_{k-b+1},\ldots,x_k)\Big)$$
$$\Big(\sum_{z_{k-b}} \prod_{\substack{x_{k-b+1}\in\{y_{k-b+1},z_{k-b+1}\}\\ \cdots \\ x_k\in\{y_k,z_k\}}} A(x_1,\ldots,x_{k-b-1},z_{k-b},x_{k-b+1},\ldots,x_k)\Big).$$

Rearranging terms gives the desired result. ∎

Lemma 6.14 *Consider the sequence* $\{a_n\}$ *where* $a_0 = 0$ *and* $a_i = 2a_{i-1} + i + k - 2$ *for* $i > 0$. *Then for* $j \geq 0$, $a_j = (2^j - 1)k - j$.

Proof: The proof is by induction. The lemma holds for the first element in the sequence $a_0 = (2^0 - 1)k - 0 = 0$. Suppose $a_j = (2^j - 1)k - j$. Then

$$\begin{aligned} a_{j+1} &= 2a_j + (j+1) + k - 2 \\ &= 2((2^j - 1)k - j) + (j+1) + k - 2 \\ &= 2^{j+1}k - 2k - 2j + j + 1 + k - 2 \\ &= (2^{j+1} - 1)k - (j+1). \end{aligned}$$ ∎

Theorem 6.15 *Let* A *be a* k*-dimensional,* d*-average Hadamard tensor with order* N. *Then*

$$W(A) \leq N^{k-(1/2^{k-1})}(k+d)^{1/2^{k-1}}.$$

Proof: Let A' be the tensor with maximal excess formed by flipping lines of A. Repeated applications of Lemma 6.13, using Lemma 6.14 to compute the exponent

of N gives the following:

$$\begin{aligned} W(A') &= \sum_{x_1=1}^{N} \cdots \sum_{x_k=1}^{N} A'(x_1, \ldots, x_k) \\ &\leq \left(\sum_{\substack{y_2 \\ \vdots \\ y_k}} \sum_{\substack{z_2 \\ \vdots \\ z_k}} \left(N^{k2^{k-1}-2k+1} \sum_{x_1} \prod_{\substack{x_2 \in \{y_2, z_2\} \\ \vdots \\ x_k \in \{y_k, z_k\}}} A'(x_1, \ldots, x_k) \right) \right)^{(1/2^{k-1})}. \end{aligned}$$

This sum computes all of the line products of A', but also includes many cases where for at least one dimension i the same value is chosen for both coordinates y_i and z_i. Note that by Lemma 6.12 the line products of A' are the same as the line products of A. There are no more than kN^{2k-3} products where at least one dimension has matching coordinates. These all have sum N. The contribution of the line products is by assumption no more than $N^{2k-2}d$. Thus this sum is at most $N^{k-(1/2^{k-1})}(k+d)^{1/2^{k-1}}$. ∎

Theorem 6.16 *Let $f : (\{0,1\}^n)^k \to \{1,-1\}$ be a function represented by a k-dimensional d-average Hadamard tensor A_f with order 2^n. Then*

$$D(f) \geq (n - \log(k+d))/2^{k-1}.$$

Proof: This follows directly from Theorem 6.15 and Theorem 3.2. ∎

Using Theorem 3.3 and Theorem 3.4 we also get the following lower bounds on distributional complexity and randomized complexity:

Theorem 6.17 *Let $f : (\{0,1\}^n)^k \to \{1,-1\}$ be a function represented by a k-dimensional d-average Hadamard tensor A_f with order $N = 2^n$. Then*

$$D_{\frac{1}{2}-\epsilon}(f) = \Omega(((n - \log(k+d))/2^k) + \log \epsilon).$$

Theorem 6.18 *Let $f : (\{0,1\}^n)^k \to \{1,-1\}$ be a function represented by a k-*

dimensional d-average Hadamard tensor A_f with order $N = 2^n$. Then

$$R_{\frac{1}{2}-\epsilon}(f) = \Omega(((n - \log(k+d))/2^k) + \log \epsilon).$$

Chung [11] and Raz [33] state a sufficient condition for a function to have $\Omega(n/2^k)$ multiparty communication complexity (generalizing the method of [4]). We present this method and show the relation between their method and d-average Hadamard tensors. In particular, we can show that satisfying the condition of [11] and [33] is equivalent to requiring that the sum of the squares of the line products is small.

Their method relies on the following definitions:

Definition 6.3 *([33], [11]) Let $X_1, \ldots, X_k$ be sets. A* cube *with respect to these sets is a multi-set $\{a_1, b_1\} \times \{a_2, b_2\} \times \cdots \times \{a_k, b_k\}$ where for all i, $a_i, b_i \in X_i$. Note that a_i and b_i are not necessarily distinct.*

Definition 6.4 *([33], [11]) Given a function $f : X_1 \times X_2 \times \cdots \times X_k \to \{1, -1\}$, define the cube sign function, $S_f(D)$ to be*

$$\prod_{\substack{d_1 \in \{a_1, b_1\} \\ \cdots \\ d_k \in \{a_k, b_k\}}} f(d_1, d_2, \ldots, d_k).$$

Definition 6.5 *([33], [11]) Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{1, -1\}$ be a function. The cube expectation function, $\mathcal{E}(f)$, is defined to be $\mathbf{E}_D[S_f(D)]$ where D ranges over all cubes with respect to $X_1, \ldots, X_k$.*

Raz also gives a proof of the following theorem, originally proved by Chung, the reason for introducing the cube expectation function:

Theorem 6.19 *([33], [12]) Let $f : X_1 \times X_2 \times \cdots \times X_k \to \{1, -1\}$ be a function.*

$$C_{\frac{1}{2}-\epsilon}(f) = \Omega(2^{-k} \log(1/\mathcal{E}(f)) + \log \epsilon).$$

Notice that taking cube products in the cube sign function is very similar to taking line products in the Hadamard definition:

Lemma 6.20 *Let A be a k-dimensional tensor with order N restricted to ± 1 entries. For any line product of A, the square of the line product is the same as the sum of values of the corresponding cube products from the cube sign function.*

Proof: Consider the line formed by taking the entrywise product of the 2^{k-1} lines for the line product. Suppose that it has j 1's and $(N - j)$ -1's. Then the line product is $(2j - N)$. Consider taking all cube products using the same entries that were used for computing the line product and all pairs of entries along the dimension of the line product. (We consider here the choice of entries to be ordered and allow duplication.) There are N^2 cube products. Of these, $(j^2 + (N - j)^2)$ of them correspond to entries where the line has the same value and $2j(N - j)$ correspond to entries where the line has different values. Thus the sum of the cube products is $(j^2 + (N - j)^2 - 2j(N - j))$. This expands to $(4j^2 - 4jN + N^2)$ which is the square of $(2j - N)$. ∎

The cube products allow for repetition within a dimension. Although we do not allow repetition within line products, they are introduced in our proof relating line products to weight via the Cauchy-Schwarz inequality. For d-average Hadamard functions, essentially the same bounds on communication complexity are given by our methods as those given by computing the cube expectation function.

6.2.1 Lower Bound on GIP

By showing that the GIP function is average Hadamard, we get a bound on its communication complexity that matches that of [4]. To fit into the Hadamard framework, we consider the function with ± 1 output rather than $\{0,1\}$ output, specifically outputting 1 when the number of all 1 indices is even and -1 when the number of all 1 indices is odd. First we will show that the line formed by taking a line product of GIP (before summing the entries) is a line of GIP. Then we will show that a line generated by a random line product is distributed (almost) uniformly across the lines of GIP. Finally we will compute the expected value of the sum of entries in a line of GIP and use our theorems about average Hadamard tensors to get communication complexity lower bounds.

We will use the ± 1 version of the PARITY function:

Definition 6.6 *Let the* PARITY *function,* $\text{PARITY} : \{0,1\}^n \to \{1,-1\}$ *be the Boolean function which outputs* 1 *when the number of 1's in the input is even and* -1 *when the number of 1's is odd.*

When working with the input strings which have values from $\{0,1\}$, we use the $\oplus$ operator to denote bitwise XOR.

Using this definition, GIP_n^k maps input $(x_1, \ldots, x_k)$ to output $\text{PARITY}(x_1 \wedge \cdots \wedge x_k)$ where the $\wedge$ operator is performed bitwise.

Notice that for any two binary strings s and t of length n, the product of $\text{PARITY}(s)$ and $\text{PARITY}(t)$ is the same as $\text{PARITY}(s \oplus t)$. In other words, if both strings have an even parity or both have odd parity, then their bitwise XOR has even parity. If one string has even parity and one string has odd parity then their bitwise XOR has odd parity.

Lemma 6.21 *For any Boolean variables a, b, and c,*

$$(a \wedge c) \oplus (b \wedge c) = (a \oplus b) \wedge c.$$

Proof: This is easily seen by computing the truth tables. ∎

Lemma 6.22 *Consider the line formed in taking the line product of* GIP_n^k *using inputs* y_i *and* z_i *for each player* i *with* $1 \leq i \leq (k-1)$*. The entry corresponding to input* x_k *(for player* k*) is* $\mathrm{PARITY}\big((\bigwedge_{i=1}^{k-1}(y_i \oplus z_i)) \wedge x_k\big)$ *where all binary operations are done bitwise.*

Proof: By definition, the entry corresponding to x_k is

$$\prod_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} \mathrm{PARITY}\big(\bigwedge_{i=1}^{k} x_i\big).$$

Using the fact that multiplying outputs of PARITY is equivalent to taking bitwise XOR on the inputs, we can rewrite this as

$$\mathrm{PARITY}\Big(\bigoplus_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-1} \in \{y_{k-1}, z_{k-1}\}}} \bigwedge_{i=1}^{k} x_i\Big).$$

Expanding gives

$$\mathrm{PARITY}\Big(\bigoplus_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-2} \in \{y_{k-2}, z_{k-2}\}}} \big((y_{k-1} \wedge (\bigwedge_{i=1}^{k-2} x_i) \wedge x_k) \oplus (z_{k-1} \wedge (\bigwedge_{i=1}^{k-2} x_i) \wedge x_k)\big)\Big).$$

Using Lemma 6.21 to simplify gives

$$\mathrm{PARITY}\Big(\bigoplus_{\substack{x_1 \in \{y_1, z_1\} \\ \cdots \\ x_{k-2} \in \{y_{k-2}, z_{k-2}\}}} \big((y_{k-1} \oplus z_{k-1}) \wedge (\bigwedge_{i=1}^{k-2} x_i) \wedge x_k\big)\Big).$$

Repeating this process another $k-2$ times yields the desired result. ∎

Lemma 6.23 *The line formed by taking the line product of* GIP_n^k *using inputs* y_i

and z_i for each player i with $1 \leq i \leq (k-1)$ is the same as the line in the GIP *tensor corresponding to inputs $(y_i \oplus z_i)$ for each player i with $1 \leq i \leq (k-1)$.*

Proof: This follows directly from the previous lemma and the definitions. ∎

Lemma 6.24 *Consider the distribution of lines formed by taking line products of* GIP_n^k *using inputs y_i and z_i for each player i with $1 \leq i \leq (k-1)$ where the y_i's and z_i's are chosen uniformly at random except that $y_i \neq z_i$ for each i. This is the uniform distribution of lines of the* GIP *tensor along the k-th dimension such that none of the inputs $x_1, \ldots, x_{k-1}$, is all zero.*

Proof: For any fixed value for y_i, choosing a random $z_i \neq y_i$ distributes $y_i \oplus z_i$ uniformly across all input strings except the all-zero string. Doing this in each of the first $i-1$ dimensions distributes the lines uniformly across all the lines of GIP except those corresponding to all-zero strings in at least one dimension. ∎

Theorem 6.25 *The k-player, n-bit* GIP *function is $(2^n(1-(2^{n-1}/(2^n-1))^{k-1})^n)$-average Hadamard.*

Proof: We have established that computing the average line product is equivalent to computing the expected sum of entries in a line of the GIP tensor, excluding lines that correspond to the all-zero string in at least one dimension.

Consider the line of the tensor generated by using inputs $x_1, \ldots, x_{k-1}$ for the first $(k-1)$ players. If there is no index in the input strings $x_1, \ldots, x_{k-1}$ for which all of the strings have a 1, then the entire line is 1. Otherwise half of the line will be 1, and half will be -1 since the input strings for player k can be partitioned by their values on all but an index for which the first $(k-1)$ players' inputs are 1. Each pair of elements will have a 1 value and a -1 value since they agree equally on all but the special index.

Thus the expected value of a line is the probability of having an all-one line multiplied by 2^n, the number of 1's in the line. (The lines that are half 1 and half

-1 contribute nothing to the expected value.) A line is all-one if there is no index on which all of the first $(k-1)$ players' inputs have a 1. For a given index, the probability that all of the players' inputs are 1 is $(2^{n-1}/(2^n-1))^{k-1}$. (This is just over $(1/2)^{k-1}$ since we have excluded the all-zero string.) Thus the probability that on this index at least one player has a 0 is $1-(2^{n-1}/(2^n-1))^{k-1}$. And so the probability that no index is all-one is $(1-(2^{n-1}/(2^n-1))^{k-1})^n$. ∎

Theorem 6.26 $D(\mathrm{GIP}^{\mathrm{k}}_{\mathrm{n}}) = \Omega(n/4^k)$.

Proof: From Theorem 6.25 we know that GIP is $(2^n(1-(2^{n-1}/(2^n-1))^{k-1})^n)$-average Hadamard. By our definition, any d-average Hadamard function is also d'-average Hadamard where $d' > d$. Thus GIP is also $(2^n(1-(1/2)^{k-1})^n)$-average Hadamard.

Using Theorem 6.16, this gives that the communication complexity of GIP is at least

$$(1/2^{k-1})(n - \log(k + (2-(1/2^{k-2}))^n)).$$

This is at least

$$(1/2^{k-1})(n - \log k - \log((2-(1/2^{k-2}))^n)).$$

This is at least

$$(1/2^{k-1})(n - \log((2-(1/2^{k-2}))^n)) - 1.$$

This simplifies to

$$(1/2^{k-1})(n(1 - \log(2-(1/2^{k-2})))) - 1.$$

This further simplifies to

$$(1/2^{k-1})(n \log(2^{k-1}/(2^{k-1}-1))) - 1.$$

This further simplifies to

$$(1/2^{k-1})(n(k-1-\log(2^{k-1}-1))) - 1.$$

Since log is a concave function, it lies below any tangent line and we can use the approximation $\log(x+c) \leq \log(x) + c/x$. Thus the communication complexity is at least

$$(1/2^{k-1})(n(k-1-(k-1-1/2^{k-1}))) - 1.$$

This gives

$$(1/2^{k-1})(n/2^{k-1}) - 1.$$

Thus GIP $= \Omega(n/4^k)$. ∎

6.3 Embedded Tensors

The Hadamard bounds can be used for any function that contains an embedded Hadamard tensor. The multiplication of finite field elements is contained inside multiplication of Boolean matrices. To convert a matrix product to a Boolean output an operation must be performed. Previously Raz [33] considered the Boolean function defined by returning the upper left corner of the matrix and showed a bound of $\Omega(\sqrt{n}/2^k)$ lower bound on the k-party communication complexity for matrices of size $\sqrt{n} \times \sqrt{n}$. Babai, Hayes, and Kimmel [3] considered instead returning the trace (mod 2) of the product matrix. (The trace of a matrix is the sum of the elements along its diagonal.) They achieved a k-party lower bound of $\Omega(n/(k^2 2^k))$. We show an embedding that gives a proof of the same bound as Raz and a slightly weaker bound than that of Babai, Hayes, and Kimmel.

6.3.1 Lower Bound on Matrix Multiplication

Raz considered the function defined as follows: Each part of the input $x_i \in \{0,1\}^n$ is interpreted as a $\sqrt{n}$ by $\sqrt{n}$ matrix with $0,1$ entries. The function is defined by the bit in the upper left corner of the matrix obtained by taking the product (over GF(2)) of the k matrices.

We use the following standard representation of finite field elements [26]:

Lemma 6.27 *Let $p(a) = c_0 + c_1 a + \cdots + c_{n-1}a^{n-1} + a^n$ be an irreducible polynomial over* GF(2)*. Let A be the following matrix:*

$$\begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & c_0 \\ 1 & 0 & 0 & \cdots & 0 & c_1 \\ 0 & 1 & 0 & \cdots & 0 & c_2 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & c_{n-1} \end{bmatrix}.$$

The elements of GF(2^n) *can be represented as polynomials in A and I (the identity matrix) of degree at most $n-1$ where addition and multiplication in the field are computed using matrix addition and multiplication over* GF(2)*.*

So, for example, GF(2^3) consists of 0, I, A, $A+I$, A^2, A^2+I, A^2+A, and A^2+A+I. The output of the matrix multiplication function of Raz was the upper left corner of the product matrix. We show this corresponds to the parity function applied to the constant term of the polynomial.

Lemma 6.28 *Let A be a matrix of order n representing a finite field element as described in Lemma 6.27. Then for all $0 < k < n$, the first row of A^k begins with $n-k$ 0's.*

Proof: The first row of A does not contain a 1 except possibly in the last column. Given any j, the entry in the first row and j-th column of A^k for $k > 1$ is 1 only if

the entry in the first row and $(j+1)$-th column of A^{k-1} is 1. The statement follows by induction. ∎

Since the relevant powers of A contain a 0 in the upper left corner and I contains a 1 in the upper left corner, that bit indicates whether the matrix represents a polynomial with a constant term. Thus one of the FFM functions is embedded inside the Raz multiplication function.

Since the embedded sub-tensor is of order $2^{\sqrt{n}}$ and is Hadamard, we use our lower bound results to get an $\Omega(\sqrt{n}/2^k)$ lower bound on the k-party communication complexity of the function.

6.3.2 Lower Bound on TMP

Babai, Hayes, and Kimmel [3] considered the trace of the matrix product (represented by TMP). To show it contains FFM as a sub-tensor, we must correlate the trace of a matrix with a character f to get the correct version of FFM(f). We use the same representation of finite field elements as matrices. The key observation is that the trace of the sum of two field element matrices is the same as the sum of the traces of the two matrices. Remember that character functions are specified by giving a subset of the input bits on which parity is taken. We specify our character function as follows: Consider the matrices $\{I, A, A^2, \ldots, A^k\}$. Every matrix representing a field element can be represented as a sum of a subset of these matrices. Consider the subset of $\{I, A, A^2, \ldots, A^k\}$ containing the matrices that have trace 1. This always includes I, so the set is non-empty. Choose the character that corresponds to the parity of the intersection with this set. (Each of the field elements corresponds to one bit of the input string.) When the intersection is even, the element has trace 0 and when the intersection is odd the element has trace 1. Thus the FFM function with this character is embedded inside the TMP function. This gives the same lower bound as for Raz's multiplication function of $\Omega(\sqrt{n}/2^k)$.

Chapter 7

Tensor Norm

Although norm has been used to bound the discrepancy for two-party communication complexity problems ([24], [36]), it has not been previously extended to the multiparty model. Here we present a definition of norm that allows this extension.

7.1 Lower Bounds Using Norm

We begin with a standard norm definition for vectors, frequently called the ℓ_k-norm:

Definition 7.1 *For any vector $\vec{x}$, its* k-norm *(denoted $||\vec{x}||_k$) is the k-th root of the sum of the absolute values of the k-th powers of the entries of $\vec{x}$.*

We extend this definition to tensors:

Definition 7.2 *For any tensor A, its* k-norm *(denoted $||A||_k$) is the k-norm of the vector containing all of the entries of A.*

A k-dimensional tensor is *cylindrical* in the i-th dimension if its entries do not depend on the i-th index. (Tensors that represent cylinders in the multiparty communication game are cylindrical and are also restricted to 0/1 entries.)

Definition 7.3 *Let A be a k-dimensional tensor. The* NOF-norm *of A (denoted $||A||_{\text{NOF}}$) is*

$$\max_{T_1,\ldots,T_k:||T_i||_k=1} |A \circ T_1' \circ T_2' \circ \cdots \circ T_k'|.$$

Each T_i is a $(k-1)$-dimensional tensor. Each T_i' is the cylindrical extension of T_i into a k-dimensional tensor. The $\circ$ operator represents entrywise multiplication. $|T|$ is the absolute value of the sum of the entries of T.

We use the notation NOF-norm to emphasize that the norm relates to the number on the forehead model of computation. This definition is equivalent to one of the standard matrix norms called the spectral norm when $k = 2$. It is also equivalent to the following definition, an expression of a two step process:

Definition 7.4 *Let A be a k-dimensional tensor. The* NOF-norm *of A (denoted $||A||_{\text{NOF}}$) is*

$$\max_{T_1,\ldots,T_k:||T_i||_k=1} \max_{A'} |A' \circ T_1' \circ T_2' \circ \cdots \circ T_k'|.$$

A' ranges over all tensors that can be formed from A by flipping lines. Each T_i is a $(k-1)$-dimensional tensor restricted to non-negative entries. The rest of the notation is as in the previous definition except that absolute values are no longer necessary.

Note that this definition preserves the usual properties required for a function to be called a "norm". In particular, the NOF-norm of a tensor is always non-negative, and it is 0 if and only if the tensor has only zero entries, multiplying a tensor by a scalar multiplies its NOF-norm by the same scalar, and the triangle inequality is satisfied.

The smallest possible norm of an $N \times N$ matrix with ± 1 entries is $\sqrt{N}$, and the largest possible norm is N (see, e.g., [24]). We show the following bound on tensor NOF-norm:

Theorem 7.1 *Let A be a k-dimensional tensor with ± 1 entries and N entries per side. Then $N^{1-\frac{1}{k}} \leq ||A||_{\text{NOF}} \leq N$.*

Proof: First we show the upper bound. We use the following standard form of Hölder's inequality which holds for any sequences $S^{(1)}, \ldots, S^{(k)}$:

$$\big|\sum_i S_i^{(1)} \cdots S_i^{(k)}\big|^k \leq \big(\sum_i |S_i^{(1)}|^k\big) \cdots \big(\sum_i |S_i^{(k)}|^k\big)$$

$$\begin{aligned}
||A||_{\text{NOF}} &= \big|A \cdot X_1 \cdot X_2 \cdots X_k\big| \\
&= \big|\sum_{i_1}\sum_{i_2}\cdots\sum_{i_k} A_{i_1 i_2 \ldots i_k} X_{1 i_1 i_2 \ldots i_k} \cdots X_{k i_1 i_2 \ldots i_k}\big| \\
&\leq \sum_{i_1}\sum_{i_2}\cdots\sum_{i_k} \big|A_{i_1 i_2 \ldots i_k} X_{1 i_1 i_2 \ldots i_k} \cdots X_{k i_1 i_2 \ldots i_k}\big| \\
&= \sum_{i_1}\sum_{i_2}\cdots\sum_{i_k} |A_{i_1 i_2 \ldots i_k}|\, |X_{1 i_1 i_2 \ldots i_k}| \cdots |X_{k i_1 i_2 \ldots i_k}| \\
&= \sum_{i_1}\sum_{i_2}\cdots\sum_{i_k} |X_{1 i_1 i_2 \ldots i_k}|\, | \cdots |X_{k i_1 i_2 \ldots i_k}|.
\end{aligned}$$

Using Hölder's inequality, this is no more than

$$\big(\sum_{i_1,\ldots,i_k} |X_{1 i_1,\ldots,i_k}|^k\big) \cdots \big(\sum_{i_1,\ldots,i_k} |X_{k i_1,\ldots,i_k}|^k\big)\big)^{(1/k)}.$$

Each of these sums by definition has value N, and there are k of them giving a total value for the expression of N.

Given any A, we exhibit a set of cylindrical tensors that gives an NOF-norm of $N^{1-(1/k)}$. Choose a face of A. The appropriate player for this face chooses the cylindrical tensor with equal absolute values everywhere and with signs to make the product of this tensor and A be positive everywhere on this face. The other players use equal values in their cylindrical tensors for those entries which intersect the face and zero everywhere else. So after taking the product of A and all the tensors, it

will have positive entries on the face and zero entries everywhere else. The positive entries will be $N^{-(k-1)/k}(N^{-(k-2)/k})^{k-1}$ and there will be N^{k-1} of them. This gives a total of $N^{1-(1/k)}$. ∎

We prove the following theorem relating the NOF-norm to the weight:

Theorem 7.2 *Let A be a k-dimensional tensor with ± 1 entries and N entries per side. Then $||A||_{\text{NOF}} \geq W(A)/N^{k-1}$.*

Proof: Consider a set of tensors $X_1, \ldots, X_k$ which have the value $N^{-(k-1)/k}$ at every entry. These tensors have k-norm 1. Flip lines of A until its excess is $W(A)$. Now multiply by these tensors. This gives $||A||_{\text{NOF}} \geq W(A)/N^{k-1}$. ∎

Since tensor weight is a bound on discrepancy, we get the following corollary, proved for matrices by Shaltiel [36]:

Corollary 7.3 *Suppose $f : (\{0,1\}^n)^k \to \{1,-1\}$ is represented by a k-dimensional tensor A_f with $N = 2^n$ entries per side. Then* $\text{Disc}(f) \leq ||A_f||_{\text{NOF}}/N$.

Since discrepancy is a bound on communication complexity, we get the following corollary:

Corollary 7.4 *Suppose $f : (\{0,1\}^n)^k \to \{1,-1\}$ is represented by a k-dimensional tensor A_f with $N = 2^n$ entries per side. Then $D(f) \geq \log(N/||A||_{\text{NOF}})$.*

7.2 Norm and Kronecker Product

Some matrix measurements have the property that the measure applied to the Kronecker product of two matrices is the same as the product of the measure of the two original matrices. Thus lower bounds on two-party communication complexity can be derived on functions with large inputs using lower bounds on functions with small inputs. The following theorem is well known for matrices (see, for example, [36]):

Theorem 7.5 *Let A and B be matrices. Then*

$$||A \otimes B|| = ||A||\,||B||.$$

For tensors the situation is more difficult, but using a similar argument as for matrices, we show that the NOF-norm of the Kronecker product of two tensors cannot be smaller than the product of their NOF-norms. Later we give evidence that the NOF-norm of the Kronecker product may be larger than the product of their NOF-norms.

Theorem 7.6 *Let A and B be k-dimensional tensors.*

$$||A \otimes B||_{\text{NOF}} \geq ||A||_{\text{NOF}} ||B||_{\text{NOF}}.$$

Proof: Let $\{S_1, \ldots, S_k\}$ be a set of tensors where each tensor S_i is cylindrical in the i-th dimension, $||S_i||_k = 1$, and $||A||_{\text{NOF}} = |A \circ S_1 \circ \cdots \circ S_k|$. Let $\{T_1, \ldots, T_k\}$ be a similar set of tensors for B. Consider the set of tensors $\{S_1 \otimes T_1, \ldots, S_k \otimes T_k\}$. Since for every i both S_i and T_i are cylindrical in the i-th dimension, so is $S_i \otimes T_i$. In general, the k-norm is preserved by Kronecker product, as we can easily see by computing the k-norm of $(S_i \otimes T_i)$:

$$\begin{aligned}
&(||S_i \otimes T_i||_k)^k \\
&= \sum_{\vec{x}} |(S_i \otimes T_i)(\vec{x})|^k \\
&= \sum_{\vec{s},\vec{t}} |S_i(\vec{s})|^k |T_i(\vec{t})|^k \\
&= \sum_{\vec{s}} |S_i(\vec{s})|^k \left(\sum_{\vec{t}} |T_i(\vec{t})|^k \right) \\
&= \sum_{\vec{s}} |S_i(\vec{s})|^k \\
&= 1.
\end{aligned}$$

Consider the entrywise product $(A \otimes B) \circ (S_1 \otimes T_1) \circ \cdots \circ (S_k \otimes T_k)$. The tensor $(A \otimes B)$ is made up of many copies of A, each multiplied by an entry of B. In each of these copies each entry of A is multiplied by the corresponding entries of $S_1, \ldots, S_k$ and all of the entries is each copy are multiplied by one entry of $T_1, \ldots, T_k$ corresponding to the entry of B. Since each entry of A is multiplied by the corresponding entries of the tensors corresponding to the NOF-norm of A, each of these copies has entry sum $||A||_{\text{NOF}}$ multiplied by the corresponding entry of B and $T_1, \ldots, T_k$. Thus the total sum of entries is $||A||_{\text{NOF}}$ multiplied by the total of each entry of B multiplied by the corresponding entries of $T_1, \ldots, T_k$. But this is just $||B||_{\text{NOF}}$. Thus we have exhibited a set of cylindrical tensors with k-norm 1 that when multiplied by $A \otimes B$ entrywise gives entries which sum to $||A||_{\text{NOF}} ||B||_{\text{NOF}}$. ∎

7.3 Norm and Lagrange Multipliers

The method of Lagrange multipliers is commonly used to solve extremal problems. The method for finding maxima and minima of a function subject to constraints depends on the fact that at these points the gradient of the function must be proportional to the gradient of the constraints. Thus one way to solve these problems is by solving a series of equations formed by taking partial derivatives of the function and the constraints, introducing a variable, the Lagrange multiplier, to represent the ratio of the gradients. We begin by considering the NOF-norm of the simplest non-trivial tensor.

7.3.1 Norm of AND

The NOF-norm of the Kronecker product of two tensors cannot be less than the product of the two NOF-norms. In fact, it appears to be larger than the product. Consider the simplest non-trivial tensor. It is the 2x2x2 cube. Any 2x2x2 cube with an even number of 1's can be flipped to a cube with all 1's, and thus has maximal

NOF-norm. Any 2x2x2 cube with an odd number of 1's can be flipped to a cube with one -1 and seven 1's. We will estimate the NOF-norm of the cube with one -1. In general it is difficult to compute the exact norm of a matrix, and unsurprisingly it is even more difficult for tensors.

It is easiest to picture the cube by examining its matrix slices. In this view the cube is

$$\begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}.$$

The three tensors used in computing the NOF-norm (extended into three dimensions) are the following:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} a & b \\ c & d \end{bmatrix},$$

$$\begin{bmatrix} e & g \\ e & g \end{bmatrix} \begin{bmatrix} f & h \\ f & h \end{bmatrix},$$

$$\begin{bmatrix} i & i \\ j & j \end{bmatrix} \begin{bmatrix} k & k \\ l & l \end{bmatrix}.$$

By definition, the NOF-norm is the maximum value of

$$-aei + afk + bgi + bhk + cej + cfl + dgj + dhl$$

subject to the constraints

$$a^3 + b^3 + c^3 + d^3 = 1,$$
$$e^3 + f^3 + g^3 + h^3 = 1,$$
$$i^3 + j^3 + k^3 + l^3 = 1.$$

Using the method of Lagrange multipliers, any maximum must satisfy the constraints and these equations:

$$\begin{aligned}
-ei + fk &= \lambda_1(3a^2),\\
gi + hk &= \lambda_1(3b^2),\\
ej + fl &= \lambda_1(3c^2),\\
gj + hl &= \lambda_1(3d^2),\\
-ai + cj &= \lambda_2(3e^2),\\
ak + cl &= \lambda_2(3f^2),\\
bi + dj &= \lambda_2(3g^2),\\
bk + dl &= \lambda_2(3h^2),\\
-ae + bg &= \lambda_3(3i^2),\\
ce + dg &= \lambda_3(3j^2),\\
af + bh &= \lambda_3(3k^2),\\
cf + dh &= \lambda_3(3l^2).
\end{aligned}$$

If we multiply the first equation by a, the second equation by b, the third equation by c, and the fourth equation by d, and then sum the equations we get

$$-aei + afk + bgi + bhk + cej + cfl + dgj + dhl = 3\lambda_1(a^3 + b^3 + c^3 + d^3).$$

Since $a^3 + b^3 + c^3 + d^3 = 1$, $3\lambda_1$ is exactly the value we are trying to compute. When it is maximized, $3\lambda_1$ will be the NOF-norm. Using the same process on the middle four equations and on the last four equations, we get that $3\lambda_2$ and $3\lambda_3$ are also equal to the NOF-norm in the solution we want. Using λ to represent the NOF-norm, we

get the following updated equations:

$$\begin{aligned}
-ei + fk &= \lambda a^2, \\
gi + hk &= \lambda b^2, \\
ej + fl &= \lambda c^2, \\
gj + hl &= \lambda d^2, \\
-ai + cj &= \lambda e^2, \\
ak + cl &= \lambda f^2, \\
bi + dj &= \lambda g^2, \\
bk + dl &= \lambda h^2, \\
-ae + bg &= \lambda i^2, \\
ce + dg &= \lambda j^2, \\
af + bh &= \lambda k^2, \\
cf + dh &= \lambda l^2.
\end{aligned}$$

One solution to these equations is to set $a = e = i$, $b = c = f = g = j = k$, and $d = h = l$. As intuition to why this may be a correct solution, note that a, e, and i are part of the negative term in the norm equation. The values b, c, f, g, j, and k are part of terms including a, e, or i. The values d, h, and l are part of neither. So we would expect the first set to take on smaller value than the second set and the second set to take on smaller value than the third set. Further, it is common for extrema to occur when values known to be close are actually the same.

This gives three equations:

$$-a^2 + b^2 = \lambda a^2, \tag{7.1}$$

$$ab + bd + \lambda b^2, \tag{7.2}$$

$$d^2 + b^2 = \lambda d^2. \tag{7.3}$$

Solving the first equation for a gives $a = b/\sqrt{\lambda + 1}$. Solving the third equation for d gives $d = b/\sqrt{\lambda - 1}$. Substituting into the middle equation, and assuming b is not 0, this gives

$$\lambda = 1/\sqrt{\lambda + 1} + 1/\sqrt{\lambda - 1}.$$

Solving for λ numerically gives the approximate value $1.754 < \lambda < 1.755$ and corresponding values of $0.387 < a < 0.388$, $0.643 < b < 0.644$, and $0.741 < d < 0.742$.

This does not definitively answer the question about the NOF-norm of this tensor since although this is a local maximum there may be other solutions leading to a different global maximum. (The most obvious other solutions involve setting some of the variables to 0 and lead to local minima.) We have run simulations that suggest that this solution is, in fact, the global maximum.

It is known that the norm of symmetric matrices can be achieved using the same vector to multiply both the rows and the columns. If we assume that the tensors being used in the NOF-norm computation are symmetric, that is, $a = e = i$, $b = f = j$, $c = g = k$, and $d = h = l$, then similar calculations show that the solution above is the correct one. This gives further evidence that the estimate is correct.

Consider the Kronecker product of the tensor representing 2-bit, 3-player

AND with itself:

$$\begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & -1 & 1 & 1 \\ -1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}.$$

Using the same technique, one local maximum is achieved with value approximately 3.1811 using the cylindrical tensor for all three players which repeat the face

$$\begin{bmatrix} a & a & -b & b \\ -b & b & a & a \\ b & b & a & a \\ a & a & b & b \end{bmatrix}$$

where a is approximately 0.4386 and b is approximately 0.3437. Note that using this cylinder in all three directions causes each line to have either three or four positive entries, and that all lines with four positive entries are multiplied by the higher value, and all lines with three positive entries are multiplied by the lower value. The square of 1.755, the conjectured NOF-norm in the smaller case, is only about 3.08, leading to the conjecture that NOF-norm of the Kronecker product of two tensors with three or more dimensions is not the same as the product of the NOF-norms.

7.3.2 Weight and Kronecker Product

It is not unusual for a measure to be larger after taking the Kronecker product of two tensors than the product of the measure on the two tensors. For the weight measure this is true even for matrices. For example, the following weight values

were found via brute force computation:

$$W\left(\begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}\right) = 2$$

$$W\left(\begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}\right) = W\left(\begin{bmatrix} 1 & -1 & -1 & 1 \\ -1 & -1 & 1 & 1 \\ -1 & 1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}\right) = 8$$

$$W\left(\begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} -1 & 1 \\ 1 & 1 \end{bmatrix}\right) = 16$$

These examples show that it is possible for the weight to be greater in the Kronecker product of two matrices than the product of their original weights. It is easy to see that the weight cannot be less than the product of the original weights:

Theorem 7.7 *Let A and B be ± 1 matrices. Then*

$$W(A \otimes B) \geq W(A)W(B).$$

Proof: Consider a set of flips on A and B which maximize their respective excess. Flipping all lines in $A \otimes B$ that correspond to lines in these sets (flipping twice for those lines in both sets) gives the matrix $A' \otimes B'$ where A' is a matrix of maximal excess formed by flipping lines of A and B' is a matrix of maximal excess formed by flipping lines of B. This has excess equal to the product of the weights of A and B. ∎

Theorem 7.7 extends directly for tensors under Kronecker product:

Theorem 7.8 *Let A and B be ± 1 tensors. Then*

$$W(A \otimes B) \geq W(A)W(B).$$

Proof: Just as for matrices, the lines which generate maximum excess when flipped in A and B can be flipped in $A \otimes B$, giving a tensor whose excess is that of the product of the weights of A and B. ∎

7.3.3 General Tensors

The method of Lagrange multipliers gives an interesting insight into the computation of the NOF-norm that holds for all tensors. The definition of NOF-norm involves taking the sum of entries of an entrywise product of several tensors. It is helpful to give names to both the product tensor and the tensors used to generate it:

Definition 7.5 *Let A be a k-dimensional tensor. A set of $(k-1)$-dimensional tensors $\{T_1, \ldots, T_k\}$ is called a set of* NOF-norm multiplier tensors *for A if $||A||_{\text{NOF}} = |A \circ T_1' \circ \cdots \circ T_k'|$ where each T_i' is the cylindrical extension of T_i into the i-th dimension.*

Definition 7.6 *Let A be a k-dimensional tensor and let $\{T_1, \ldots, T_k\}$ be a set of NOF-norm multiplier tensors for A. The* NOF-norm tensor *for A and $T_1, \ldots, T_k$ is the entrywise product tensor $A \circ T_1' \circ \cdots \circ T_k'$ where each T_i' is the cylindrical extension of T_i into the i-th dimension.*

Theorem 7.9 *Let $f : X_1 \times \cdots \times X_k \to \{1, -1\}$ be represented by k-dimensional tensor A_f. Let A_f' be some tensor derived from A_f by flipping lines for which the NOF-norm of A_f can be achieved from A_f' using a set of NOF-norm multiplier tensors with non-negative entries. Let $\{T_1, \ldots, T_k\}$ be such a set of NOF-norm multiplier tensors for A_f' containing only non-negative entries. Then every entry in*

every tensor $T_i(\vec{x})$ is the k-th root of the ratio of the sum of the entries in the line of the NOF-norm tensor affected by $T_i(\vec{x})$ to the NOF-norm of A_f.

Proof: Consider some tensor entry $T_i(\vec{x})$. The Lagrange multiplier equation given by taking the partial derivative with respect to this entry gives the equation

$$\sum_{y_j \in X_j} A'_f(x_1, \ldots, x_{j-1}, y_j, x_{j+1}, \ldots, x_k) \prod_{\substack{m \in \{1,\ldots,k\} \\ m \neq i}} T_m(x_1, \ldots, x_{j-1}, y_j, x_{j+1}, \ldots, x_k) = \lambda k (T_i(\vec{x}))^{k-1}.$$

Multiplying both sides of this equation by the original entry $T_i(\vec{x})$ gives that the product $\lambda k T_i(\vec{x})^k$ is equal to the sum of the entries in the NOF-norm tensor $A'_f \circ T'_1 \circ \cdots \circ T'_k$ affected by $T_i(\vec{x})$. No entries in the NOF-norm tensor are affected by more than one entry from T_i. Thus, using the same process for all entries in T_i and adding the equations together gives that the sum of all entries in the NOF-norm tensor is the same as λk times the sum of the k-th powers of the entries in T_i. Since the k-th powers sum to 1, λk must be the NOF-norm of A_f. And thus $T_i(\vec{x})$ is the k-th root of the ratio of the sum of the entries in the NOF-norm tensor affected by $T_i(\vec{x})$ to the total NOF-norm. ∎

The proof above gives the following corollary:

Corollary 7.10 *Let $f : X_1 \times \cdots \times X_k \to \{1, -1\}$ be represented by k-dimensional tensor A_f. Let A'_f be some tensor derived from A_f by flipping lines for which the NOF-norm of A_f can be achieved from A'_f using a set of NOF-norm multiplier tensors with non-negative entries. Let $\{T_1, \ldots, T_k\}$ be such a set of NOF-norm multiplier tensors for A'_f containing only non-negative entries. For any entry $T_i(\vec{x})$, the Lagrange multiplier in the equation given by taking the partial derivative with respect to this entry is k times the NOF-norm of A_f in any solution of the Lagrange multiplier equations corresponding to the NOF-norm of A_f.*

Things are simpler when the number of players is even, since in computing the k-norm of the tensors from the (first) NOF-norm definition, the k-th powers are all positive. Thus when taking partial derivatives it is not necessary to assume the variables are positive or to complicate the derivatives with absolute values. This leads to the following corollary:

Corollary 7.11 *Let $f : X_1 \times \cdots \times X_k \to \{1, -1\}$ be represented by k-dimensional tensor A_f* with k even. *Let $\{T_1, \ldots, T_k\}$ be a set of NOF-norm multiplier tensors for A_f. Then the absolute value of any entry $T_i(\vec{x})$ is the k-th root of the ratio of the sum of the entries in the line of the NOF-norm product tensor affected by $T_i(\vec{x})$ to $||A_f||_{\text{NOF}}$.*

When considered specifically for matrices this gives a result relating the Lagrange multiplier to the spectral norm and the largest eigenvalue:

Corollary 7.12 *Let A be a matrix with order N. Then the spectral norm of A (and thus the square root of the largest eigenvalue of AA^T) is twice the value of the Lagrange multiplier from the calculation of the norm of A.*

Consider any tensor A and a set of NOF-norm multiplier tensors for A. Each entry in a NOF-norm multiplier tensor corresponds to a line of A. Fixing all but two coordinates of A gives a matrix which corresponds to a line in each of two NOF-norm multiplier tensors, one for the rows and one for the columns. Because of the relationship between the entries of an NOF-norm multiplier tensor and the corresponding norm tensor, the k-norm of these two lines is the same. By appropriate choosing of the coordinates, any two of the NOF-norm multiplier tensors can be seen to be composed entirely of these linked lines. Thus there is a way to partition any two NOF-norm multiplier tensors into lines such that the lines can be paired with the paired lines having the same k-norm.

Chapter 8

Tensor Rank

The rank method is one of the most commonly used methods for lower bounds in the two-party model, because it is easy to find matrices of high rank, giving good lower bounds on two-party communication complexity. However, the method has not previously been extended to the multiparty model.

8.1 Extending Matrix Rank to Tensors

Tensor rank has been previously studied [40]. In particular it has a close association with the matrix multiplication problem. In general, the definition of tensor rank that is commonly used is the following:

Definition 8.1 *A k-dimensional tensor has* rank 1 *if it can be expressed as the outer product of k vectors. That is, the entry in the tensor indexed by $(i_1, \ldots, i_k)$ is the product of the corresponding entries in each of the k vectors, taking the entry indexed by i_j from the j-th vector for all j. The* rank *of tensor A is the smallest number of rank 1 tensors whose entrywise sum is A.*

Under this definition the rank of a k-dimensional tensor with order N is at most N^{k-1} since each line can be separately specified with a rank 1 tensor. This definition

corresponds to the number in the hand model of communication complexity. To obtain lower bounds for the number on the forehead model, we propose the following definition:

Definition 8.2 *A k-dimensional tensor has* NOF-rank 1 *if it is the entrywise product of k k-dimensional tensors, each cylindrical in a different dimension. The* NOF-rank *of tensor A (denoted $\mathrm{rk}_{\mathrm{NOF}}(A)$) is the smallest number of NOF-rank 1 tensors whose entrywise sum is A.*

We use the notation NOF-rank to emphasize that the rank relates to the number on the forehead model of computation.

Lemma 8.1 *Let A and B be tensors of the same size and dimension. Then*

$$\mathrm{rk}_{\mathrm{NOF}}(A+B) \leq \mathrm{rk}_{\mathrm{NOF}}(A) + \mathrm{rk}_{\mathrm{NOF}}(B).$$

Proof: The tensor $A+B$ can be generated by adding together the $\mathrm{rk}_{\mathrm{NOF}}(A)$ tensors of NOF-rank 1 used to generate A and the $\mathrm{rk}_{\mathrm{NOF}}(B)$ tensors of NOF-rank 1 used to generate B. ∎

8.2 Estimating Communication Complexity with Rank

Just as in two-party communication complexity, NOF-rank provides a simple lower bound:

Theorem 8.2 *Let $f : X_1 \times \cdots \times X_k \to \{0,1\}$, and let A_f be the tensor representing f. Then*

$$D(f) \geq \log \mathrm{rk}_{\mathrm{NOF}}(A_f).$$

Proof: The proof mirrors the two dimensional case. A protocol tree has monochromatic cylinder intersections at its leaves. Each of these monochromatic cylinder

intersections can be represented by an NOF-rank 1 tensor. The sum of the tensors representing all of the 1-output leaves is A_f. Using Lemma 8.1, $\mathrm{rk}_{\mathrm{NOF}}(A_f)$ is no more than the number of leaves which is no more than $2^{D(f)}$. ∎

The previous proof uses a protocol to get a bound on the rank. It is also possible to use the rank to create a protocol, giving a bound on communication complexity. We first present the previously known results for matrices, and then give our extension for tensors and NOF-rank.

Theorem 8.3 *Let A_f be a matrix representing $f : \{0,1\}^n \to \{0,1\}$. Let r be the rank of A with respect to* GF(2)*. Then $D(f) \leq (r+1)$.*

Proof: Since r is the rank over GF(2) of A, there is a set of r rows such that every row of A is a linear combination (in GF(2)) of these rows. Fix such a set, and the following protocol can be used to compute f: Alice sends r bits to Bob, each bit telling whether the linear combination generating her row of input contains each of the r fixed rows. Now Bob knows the entries in the row (although possibly not her input as rows may be duplicated in the matrix), and uses that information to return the answer. The protocol takes $(r+1)$ bits of communication, giving $D(f) \leq (r+1)$. ∎

A similar bound holds for rank with respect to other finite fields:

Theorem 8.4 *Let A_f be a matrix representing $f : \{0,1\}^n \to \{0,1\}$. Let r be the rank of A with respect to finite field* GF(q) *for some prime power q. Then $D(f) \leq r \log(q) + 1$.*

Proof: Since A_f has rank r, it can be expressed as a sum of r matrices each of which can be expressed as an outer product of two length n vectors. Fix some such representation. The following protocol can be used to compute f: Alice sends $\log(q)$ bits for each of the r matrices in the representation giving the entries corresponding to her input in each of r vectors. Bob multiplies each of these entries by the entries

corresponding to his input in the other r vectors and sums the results to get the answer which he returns. The protocol requires $(r\log(q)+1)$ bits of communication, giving the desired result. ∎

Theorem 8.5 *Let A_f be a tensor representing $f : (\{0,1\}^n)^k \to \{0,1\}$. Let r be the NOF-rank of A with respect to finite field* GF(q) *for some prime power q. Then*

$$D(f) \leq (k-1)r\log(q) + 1.$$

Proof: The proof mirrors the two party case. Since A_f has NOF-rank r, there are a set of r NOF-rank 1 tensors whose sum is A_f. Each of these tensors is the entrywise product of a set of k tensors, each cylindrical in a different dimension. Fix such a set of tensors. Consider the following protocol for computing f. Each player but one sends the entries in each of the r cylindrical tensors for which the player knows the entry (because it does not depend on that player's input). This uses $(k-1)r\log(q)$ bits of communication. The k-th player multiplies the entries corresponding to each of the r tensors with the corresponding entry in the k-th tensor of each group and adds the results giving the entry in A_f that is the output of f. Thus the total amount of communication is $((k-1)r\log(q)+1)$. ∎

Another commonly studied model in communication complexity is simultaneous communication complexity. In this model all players simultaneously send a message to a referee who cannot see the input. (The players cannot see each other's message.) The referee must compute the output from the messages received. The simultaneous communication complexity is the total length of the messages sent to the referee. The protocol above extends to a simultaneous one giving the following related result: Instead of having the first $(k-1)$ players broadcast values and the last player compute the answer, all of the players could send the same values to a referee, giving the following corollary:

Corollary 8.6 *Let A_f be a tensor representing $f : (\{0,1\}^n)^k \to \{0,1\}$. Let r be the NOF-rank of A with respect to finite field* $\mathrm{GF}(q)$ *for some prime power q. Let s be the simultaneous communication complexity of f. Then $s \leq kr\log(q)$.*

Using this theorem with our bound on the multiparty communication complexity of Hadamard tensors gives the following result:

Theorem 8.7 *Let $f : (\{0,1\}^n)^k \to \{1,-1\}$ be a function represented by a k-dimensional average Hadamard tensor A_f with order $N = 2^n$. Then with respect to* $\mathrm{GF}(2)$*,*

$$\mathrm{rk}_{\mathrm{NOF}}(A_f) = \Omega(n/(k2^k)).$$

The bounds given by Theorems 8.2 and 8.5 are not close to each other. It is easy to see that the bound given by Theorem 8.3 for the two-party case is not close to tight by considering the equality function on n bits which has linear communication complexity but exponential (in n) rank. Nisan and Wigderson [30] gave an example of a function for which the bound given by Theorem 1.5 is not tight, but it may be close to tight. In fact, one of the most famous open problems in two-party communication complexity stated in [28] and commonly referred to as the log-rank conjecture, suggests that this is the case:

Conjecture 8.8 *Let $f : X \times Y \to \{0,1\}$. Then for matrix A_f representing f,*

$$D(f) = \log(\mathrm{rank}(A_f))^{O(1)}.$$

Even if the equivalent of the log-rank conjecture does not hold for the multiparty case it is still likely to be that the bound given by Theorem 8.5 is far from tight. However it is presently the only known means for estimating NOF-rank, and new methods are necessary to give more separation.

8.3 Rank and Kronecker Product

For matrices it is well known that the rank of the Kronecker product of two matrices is the product of the ranks of the matrices. We include a proof that is not the simplest known, but is more self-contained than the usual proof. This allows us to examine whether it can be extended to tensors.

Theorem 8.9 *Let A and B be matrices. Then*

$$\text{rank}(A \otimes B) = \text{rank}(A)\text{rank}(B).$$

Proof: It is easy to see that the rank of the Kronecker product cannot be larger than the product of the ranks since the Kronecker product of the rank(A) row and column vectors used to generate A and the rank(B) row and column vectors used to generate B give (rank(A)rank(B)) row and column vectors that generate $(A \otimes B)$.

Choose rank(A) linearly independent rows of A and rank(B) linearly independent rows of B. Consider the set of (rank(A)rank(B)) rows of $(A \otimes B)$ formed by the pairwise Kronecker products of this maximal set of linearly independent rows of A and B, respectively. Consider any nontrivial linear combination of these rows. Each row of $(A \otimes B)$ consists of several copies of a row of A, each copy multiplied by an element of B. Group the rows according to which row of A they correspond to, and consider the sum of each group separately. The linear combination of rows of $(A \otimes B)$ consists of the sum of these groups, and thus is several sequential linear combinations of rows of A. Since the rows of A being used are linearly independent, the only way that this linear combination can be zero is if each of these group sums is zero. Choose a group that has at least one row with a non-zero coefficient in the linear combination. Consider only the rows in this group that have non-zero coefficients in the linear combination. These rows correspond to a row of A multiplied by the elements from a nontrivial linear combination of rows of B. Since the rows of B

are linearly independent these multipliers cannot all be zero, and the group sum is non-zero. Thus any linear combination of these (rank(A)rank(B)) rows is non-zero, and the rank is at least (rank(A)rank(B)). ∎

For tensors this is not the case, although the product of the NOF-ranks of two tensors is a bound on the NOF-rank of their Kronecker product:

Theorem 8.10 *Let A and B be tensors with the same dimension. Then*

$$\mathrm{rk}_{\mathrm{NOF}}(A \otimes B) \leq \mathrm{rk}_{\mathrm{NOF}}(A)\mathrm{rk}_{\mathrm{NOF}}(B).$$

Proof: Consider the $\mathrm{rk}_{\mathrm{NOF}}(A)$ NOF-rank 1 tensors that sum to A and the $\mathrm{rk}_{\mathrm{NOF}}(b)$ NOF-rank 1 tensors that sum to B. Taking the Kronecker product of each pairwise combination from these sets give $\mathrm{rk}_{\mathrm{NOF}}(A)\mathrm{rk}_{\mathrm{NOF}}(B)$ tensors whose sum is $A \otimes B$. Each of these product tensors has NOF-rank 1 since it can be formed by taking the Kronecker products of the cylindrical tensors in each dimension used to form the original NOF-rank 1 tensors. ∎

Theorem 8.11 *There exist tensors A and B with the same dimension for which*

$$\mathrm{rk}_{\mathrm{NOF}}(A \otimes B) < \mathrm{rk}_{\mathrm{NOF}}(A)\mathrm{rk}_{\mathrm{NOF}}(B).$$

Proof: Consider the tensor representing AND for three players each missing one bit. This tensor has NOF-rank 2. However GIP is known to have multiparty communication complexity $O(n/(k2^k))$ [19], and is the result of taking the Kronecker product of AND with itself repeatedly. If the NOF-rank of a tensor product was the product of the NOF-ranks of the tensors, the NOF-rank of the GIP tensor would be 2^n leading to a lower bound of $\Omega(n)$. ∎

8.4 Rank and Probabilistic Communication Complexity

We can also use tensor rank to bound probabilistic communication complexity, extending the results of Krause [24] to more than two players. The unbounded error probabilistic communication complexity model was introduced by Paturi and Simon [31]. It is similar to the bounded error model, but only requires that a correct protocol (on every input) must output the correct answer more often than an incorrect one. Throughout this chapter we will use only the unbounded error model and refer to it as the probabilistic model. For two parties Alon, Frankl, and Rödl [1] showed that almost all functions on n-bit strings have probabilistic communication complexity at least $n - 5$ using a counting argument. Forster [18] showed that two-party communication problems represented by Hadamard matrices have linear probabilistic complexity.

To show the relation between multiparty probabilistic communication complexity and rank, we first introduce a variant of NOF-rank:

Definition 8.3 *For any tensor A restricted to ± 1 entries, the* sign pattern rank *of A (denoted $\mu(A)$) is*

$$\min\{\mathrm{rk}_{\mathrm{NOF}}(B) : \mathrm{sign}(B) = A\}.$$

Note that for any tensor A, $\mu(A)$ is bounded by the NOF-rank of A as obviously A has the same sign pattern as itself.

Using a proof analogous to Krause [24], we show that the sign pattern rank can be used to bound probabilistic communication complexity:

Theorem 8.12 *Suppose that $f : (\{0,1\}^n)^k \to \{1,-1\}$ can be computed by a probabilistic communication protocol P using at most d bits of communication. Let A_f be the tensor representing f. Then $\mu(A_f) \leq 2^d$.*

Proof: Consider all of the possible messages that can be output by P. Here we do not include the answer bit as part of the message. For each message, for each player, and for each input there is a probability that the player would output bits consistent with that message at the player's turn(s) to speak. Thus we can create for each message a tensor for each player which contains these probabilities. These tensors are cylindrical since each player cannot see a piece of the input along a particular dimension. For any given message, taking the entrywise product of the tensors associated with each player gives an NOF-rank 1 tensor that gives the probabilities of that message being generated on the different inputs.

Let p_m be the acceptance probability of message m, that is, the probability that when P generates m during its computation it outputs 1. Multiplying the tensor associated with message m by $(2p_m - 1)$ gives the weighted probability of acceptance for that message. Note that these tensors are still NOF-rank 1 since this effect can be achieved by multiplying all of the cylindrical tensors by $(2p_m - 1)^{(1/k)}$.

Summing these tensors over all messages gives the acceptance probability for each input. Since the protocol is correct, this tensor will have positive values when f has the value 1 and negative values when f has the value -1. This tensor has NOF-rank at most S_m where S_m is the number of messages. Since the number of messages possible with d bits of communication is at most 2^d, the result holds. ∎

In general it is quite difficult to estimate the sign pattern rank, even for matrices (see, for example, [1], [13]), and thus it is not surprising that it is even more difficult to do so for tensors.

8.5 Polynomial Decompositions

Any Boolean function with binary strings as input can be extended to a polynomial over the bits of the input with real output. This extension agrees with the original function whenever all of the inputs are set to 0 or 1. Buhrman and de Wolf [8] use

a relationship between this extension and matrix rank to prove two-party bounds. They begin with the following definition:

Definition 8.4 *Let $f : \{0,1\}^n \times \{0,1\}^n \to \mathbb{R}$. Then the* decomposition number *of f (denoted $m(f)$) is the minimum number m such that there exist functions $a_1(x), \ldots, a_m(x), b_1(y), \ldots, b_m(y)$, each from $\mathbb{R}^n$ to $\mathbb{R}$ with $f(x,y) = \sum_{i=1}^{m} a_i(x)b_i(y)$ for all binary strings x and y.*

It suffices to consider the a_i's and b_i's to be restricted to multi-linear polynomials. We only care about the outputs on inputs restricted to $\{0,1\}$, and on these inputs all powers of variables can be reduced to just the variable. They prove the following theorem, stated here without proof:

Theorem 8.13 *[8] Let $f : \{0,1\}^n \times \{0,1\}^n \to \mathbb{R}$. Let A_f be the matrix representing f. Then*

$$rank(A_f) = m(f).$$

We generalize this result to multiparty communication complexity. Not surprisingly, there are two ways to extend the decomposition number depending on whether the polynomials involved are viewed as depending on one variable or on all but one variable. The two extensions lead to results for the number in the hand and the number on the forehead models. As usual, we focus on the number on the forehead model, beginning with the appropriate definition of decomposition number:

Definition 8.5 *Let $f : (\{0,1\}^n)^k \to \mathbb{R}$. Then the* decomposition number *of f (denoted $m(f)$) is the minimum number m such that there exist mk functions $a_{1,1}, \ldots, a_{k,m}$, each from $\mathbb{R}^{n(k-1)}$ to $\mathbb{R}$ and with each $a_{i,j}$ depending only on the variables $\{x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_k\}$ and with $f(\vec{x}) = \sum_{i=1}^{m} a_{1,i}(\vec{x})a_{2,i}(\vec{x}) \cdots a_{k,i}(\vec{x})$ for all vectors of binary strings $\vec{x}$.*

As before, it is sufficient to consider only multi-linear polynomials. As in the two-party model, the decomposition number is the same as the rank:

Theorem 8.14 *Let $f : (\{0,1\}^n)^k \to \mathbb{R}$. Let A_f be the k-dimensional tensor with order 2^n representing f. Then*

$$\mathrm{rk}_{\mathrm{NOF}}(A_f) = m(f).$$

Proof: It suffices to show an equivalence between the set of tensors in the NOF-rank definition and the set of functions in the decomposition number definition. Clearly any set of k functions each on all but one of the inputs can be mapped to a set of tensors each cylindrical in a different dimension.

Suppose tensor T_i is cylindrical in the i-th dimension. It can be mapped to a multi-linear function that does not depend on $\vec{x_i}$ in the following way: We include a term for each line along the i-th dimension of T_i consisting of the product of the value of the tensor along that line and a selector for that line. The selector is a product which depends on every input bit of every input string x_j except where $j = i$. If bit m of x_j is 1 on the line we include the factor $(x_j)_m$ in the selector and otherwise the factor $(1 - (x_j)_m)$. (Since each player misses k bits, this function depends on $k(n-1)$ variables.) Sum the monomials corresponding to each line. For any $\vec{x}$ at most one of the monomials will be non-zero and it will give the entry associated with that line. The function so created is a multi-linear polynomial. Multiplying the functions for the tensors cylindrical along each dimension gives the function for an NOF-rank 1 tensor. ∎

These objects in the multiparty case, polynomials on all but one of the input variables are much more powerful than the corresponding objects in the two-party case. This is not surprising given the power of the number on the forehead model, but it is difficult to see how to achieve bounds on the decomposition number.

Chapter 9

Lower Bounds by Counting Arguments

Although the best lower bounds on multiparty communication complexity for explicit functions are of the form $\Omega(n/2^k)$, counting the number of functions and protocols gives several bounds for non-explicit functions. We show that most functions have communication complexity almost as large as the trivial bound $n + 1$.

9.1 Two-Party Bounds

We begin by counting protocols in the two-party model:

Theorem 9.1 *The number of protocols for two players each with n bits of information with Boolean output using d bits of communication is no more than $2^{[2^{n+d}-(2^n-2^{d+1})-1]}$.*

Proof: A protocol with d bits of communication is represented by a tree with $2^d - 1$ internal nodes and 2^d leaf nodes. Each internal node specifies a player (2 choices) and a function on that player's n input bits (2^{2^n} choices). Each leaf node is either

0 or 1. Thus the number of protocol trees is

$$(2 \cdot 2^{2^n})^{2^d - 1} 2^{2^d}.$$

This can be rearranged using simple algebra to

$$2^{[2^{n+d} - (2^n - 2^{d+1} - 1)]}. \quad \blacksquare$$

Corollary 9.2 *The number of protocols for two players each with n bits of information with Boolean output using $n-1$ bits of communication is no more than $2^{2^{2n-1}-1}$. The number of protocols using $d \leq (n-1)$ bits of communication is no more than $2^{2^{n+d}}$.*

Theorem 9.3 *The number of protocols for two players each with n bits of information using $(n-1)$ bits of communication is at most a $(1/2^{2^{2n-1}+1})$-fraction of the number of functions on $2n$ bits.*

Proof: There are $2^{2^{2n}}$ functions on $2n$ bits. The previous corollary gives that the number of protocols using $n-1$ bits of communication is at most $2^{2^{2n-1}-1}$. Thus the ratio is at most

$$(2^{2^{2n-1}-1})/(2^{2^{2n}}) = 1/(2^{2^{2n} - 2^{2n-1}+1}) = 1/(2^{2^{2n-1}+1}). \quad \blacksquare$$

We can also give bounds in the distributional model. In our computation we need the following definition of entropy:

Definition 9.1 *Let ϵ lie in the range $[0,1]$. The* entropy *of ϵ, denoted $H(\epsilon)$, is given by*

$$-\epsilon \log \epsilon - (1-\epsilon) \log(1-\epsilon).$$

Theorem 9.4 *Let ϵ be in the range $[0, 1/2)$ and let $\delta \geq (1 + \log H(\epsilon))$. Let*

$$d \leq ((n-1) + \log(2^{\delta} - 2H(\epsilon))).$$

Then at most a $(2^{2^{2n-1+\delta}}/2^{2^{2n}})$-fraction of functions on $2n$ bits are represented by two-party protocols that are correct on at least a $(1-\epsilon)$-fraction of the input and use at most d bits of communication.

Proof: Consider a fixed protocol and a fixed ϵ. The protocol completely specifies one function. It is correct on at least a $(1-\epsilon)$-fraction of the input of another function if the other function has different output on at most an ϵ-fraction of the input. Thus the number of functions for which a fixed protocol is $(1-\epsilon)$-correct is the number of ways to choose input sets of this size,

$$\sum_{i=0}^{\epsilon \cdot 2^{2n}} \binom{2^{2n}}{i}.$$

According to a well known estimate (see, for example, Corollary 23.6 of [21]), this is no more than $2^{2^{2n}H(\epsilon)}$.

Suppose ϵ, δ, and d are chosen as in the statement of the theorem. Then the number of protocols using at most d bits of communication is no more than $2^{2^{n+d}}$, and the number of functions for which at least one of these protocols is $(1-\epsilon)$ correct is at most $2^{2^{n+d}} \cdot 2^{2^{2n}H(\epsilon)}$.

Using simple algebraic manipulation,

$$\begin{aligned} & 2^{2^{n+d}} \cdot 2^{2^{2n}H(\epsilon)} \\ = \; & 2^{[2^{2n+\log(2^{\delta}-2H(\epsilon))-1}+2^{2n}H(\epsilon)]} \\ = \; & 2^{2^{2n-1}(2^{\delta}-2H(\epsilon)+2H(\epsilon))} \\ = \; & 2^{2^{2n-1+\delta}}. \end{aligned}$$

Since there are $2^{2^{2n}}$ functions on $2n$ bits, the desired fraction is achieved. ∎

Corollary 9.5 *At most a* $(2^{2^{2n-(1/4)}}/2^{2^{2n}})$*-fraction of functions on* $2n$ *bits are computed correctly on at least* $3/4$ *of the input with a two-party protocol using no more than* $n-6$ *bits of communication.*

Proof: This directly follows from the preceding theorem using $\epsilon = (1/4)$ and $\delta = (3/4)$. ∎

9.2 Multiparty Bounds

The techniques for two-party bounds generalize to the multiparty model. In this section we give bounds for deterministic, distributional, and randomized multiparty communication complexity.

Theorem 9.6 *The number of protocols for* $k \geq 3$ *players each missing* n *bits of information with binary output using* $d \leq n$ *bits of communication is no more than* $2^{2^{nk-(n-d)}}$.

Proof: A protocol with d bits of communication is represented by a tree with $2^d - 1$ internal nodes and 2^d leaf nodes. Each internal node specifies a player (k choices) and a function on the input bits that player can see ($2^{2^{n(k-1)}}$ choices). Each leaf node is either 0 or 1. Thus the number of protocol trees is

$$(k2^{2^{n(k-1)}})^{(2^d-1)}(2^{2^d}).$$

This can be rearranged using simple algebra to

$$2^{[2^{nk-(n-d)}-(2^{nk-n}+2^{\log\log k}-(2^d+2^{d+\log\log k}))]}.$$

For $k \geq 3$ and $d \leq n$, the large subtracted quantity is positive, giving the upper bound of $2^{2^{nk-(n-d)}}$. ∎

Theorem 9.7 *The number of multiparty protocols for k players each missing n bits of information using $(n-1)$ bits of communication is at most a $(1/2^{2^{nk-1}})$-fraction of the number of functions on nk bits.*

Proof: There are $2^{2^{nk}}$ functions on nk bits. The previous theorem gives that the number of protocols using $n-1$ bits of communication is at most $2^{2^{nk-1}}$. Thus the ratio is at most

$$(2^{2^{nk-1}})/(2^{2^{nk}}) = 1/(2^{2^{nk}-2^{nk-1}}) = 1/(2^{2^{nk-1}}). \quad ∎$$

Theorem 9.8 *Let ϵ be in the range $[0, 1/2)$, and let $\delta \geq (1 + \log H(\epsilon))$. Let*

$$d \leq ((n-1) + \log(2^{\delta} - 2H(\epsilon))).$$

Then at most a $(2^{2^{nk-1+\delta}}/2^{2^{nk}})$-fraction of functions on nk bits are represented by multiparty protocols that are correct on at least a $(1-\epsilon)$-fraction of the input and use at most d bits of communication.

Proof: As in the two-party case, the number of functions for which a fixed protocol is $(1-\epsilon)$ correct is the number of ways to choose a subset of the input that is at most a $(1-\epsilon)$-fraction of the size of the input. This is

$$\sum_{i=0}^{\epsilon \cdot 2^{nk}} \binom{2^{nk}}{i}.$$

Again, using the entropy bound, this is no more than $2^{2^{nk}H(\epsilon)}$.

Suppose ϵ, δ, and d are chosen as in the statement of the theorem. Then the number of protocols using at most d bits of communication is no more than

$2^{2^{n(k-1)+d}}$, and the number of functions for which at least one of these protocols is $(1-\epsilon)$ correct is at most $2^{2^{n(k-1)+d}} \cdot 2^{2^{nk}H(\epsilon)}$.

Using simple algebraic manipulation,

$$\begin{aligned} & 2^{2^{n(k-1)+d}} \cdot 2^{2^{nk}H(\epsilon)} \\ = \; & 2^{[2^{nk-1+\log(2^{\delta}-2H(\epsilon))-1}+2^{nk}H(\epsilon)]} \\ = \; & 2^{2^{nk-1}(2^{\delta}-2H(\epsilon)+2H(\epsilon))} \\ = \; & 2^{2^{nk-1+\delta}}. \end{aligned}$$

Since there are $2^{2^{nk}}$ functions on nk bits, the desired fraction is achieved. ∎

Corollary 9.9 *At most a* $(2^{2^{nk-(1/4)}}/2^{2^{nk}})$*-fraction of functions on* nk *bits are computed correctly on at least* $3/4$ *of the input with a multiparty protocol using no more than* $n-6$ *bits of communication.*

Proof: This directly follows from the preceding theorem using $\epsilon = (1/4)$ and $\delta = (3/4)$.

Similar results follow by choosing different values for ϵ and δ.

Bibliography

[1] Noga Alon, Peter Frankl, and Vojtěch Rödl. Geometrical realizations of set systems and probabilistic communication complexity. In *Proceedings of the 26th Annual IEEE Symposium on Foundations of Computer Science*, pages 277–280, 1985.

[2] Noga Alon and Joel Spencer. *The Probabilistic Method*. John Wiley, 2000.

[3] László Babai, Thomas Hayes, and Peter Kimmel. The cost of the missing bit: Communication complexity with help. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, pages 673–682, 1998.

[4] László Babai, Noam Nisan, and Márió Szegedy. Multiparty protocols, pseudo-random generators for logspace, and time–space trade-offs. *Journal of Computer and System Sciences*, 45:204–232, 1992.

[5] David A. Max Barrington. Bounded-width polynomial-size branching programs can recognize exactly those languages in NC_1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989.

[6] Richard Beigel and Jun Tarui. On ACC. In *Proceedings of the 32th Annual IEEE Symposium on Foundations of Computer Science*, pages 783–792, 1991.

[7] M. R. Best. The excess of a Hadamard matrix. *Indagationes Mathematicae*, 39(5):357–361, 1977.

[8] Harry Buhrman and Ronald de Wolf. Communication complexity lower bounds by polynomials. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity*, pages 120–130, 2001.

[9] Ashok Chandra, Merrick Furst, and Richard Lipton. Multi-party protocols. In *Proceedings of the 15th Annual ACM Symposium on Theory of Computing*, pages 94–99, 1983.

[10] Benny Chor and Oded Goldreich. Unbiased bits from sources of weak randomness and probabilistic communication complexity. *SIAM Journal on Computing*, 17(2):230–261, 1988.

[11] Fan Chung. Quasi-random classes of hypergraphs. *Random Structures and Algorithms*, 1(4):363–382, 1990.

[12] Fan Chung and Prasad Tetali. Communication complexity and quasi randomness. *SIAM Journal of Discrete Mathematics*, 6:110–123, 1993.

[13] Philippe Delsarte and Yves Kamp. Low rank matrices with a given sign pattern. *SIAM Journal on Discrete Mathematics*, 2(1):51–63, 1989.

[14] Danny Dolev and Tomás Feder. Multiparty communication complexity. In *Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science*, pages 428–433, 1989.

[15] Paul Erdős and Joel Spencer. *Probabilistic Methods in Combinatorics*. Academic Press, 1974.

[16] Jeff Ford and Anna Gál. Rank and norm lower bound methods for multiparty communication complexity. In preparation.

[17] Jeff Ford and Anna Gál. Hadamard tensors and lower bounds on multiparty

communication complexity. In *Proceedings of the 32nd International Colloquium on Automata, Languages and Programming*, pages 1163–1175, 2005.

[18] Jürgen Forster. A linear lower bound on the unbounded error probabilistic communication complexity. In *Proceedings of the 16th Annual IEEE Conference on Computational Complexity*, pages 100–106, 2001.

[19] Vince Grolmusz. The BNS lower bound for multi-party protocols is nearly optimal. *Information and Computation*, 112:51–54, 1994.

[20] Johan Haståd and Mikael Goldmann. On the power of small-depth threshold circuits. In *Proceedings of the 31th Annual IEEE Symposium on Foundations of Computer Science*, pages 610–618, 1990.

[21] Stasys Jukna. *Extremal Combinatorics: with applications in computer science.* Springer, 2001.

[22] Hadi Kharaghani and Behruz Tayfez-Rezaie. A Hadamard matrix of order 428. *Journal of Combinatorial Designs*, 13:435–440, 2005.

[23] Eike Kiltz and Hans Ulrich Simon. Complexity theoretic aspects of some cryptographic functions. In *Computing and Combinatorics*, pages 294–303, 2003.

[24] Matthias Krause. Geometric arguments yield better bounds for threshold circuits and distributed computing. *Theoretical Computer Science*, 156:99–117, 1996.

[25] Eyal Kushilevitz and Noam Nisan. *Communication Complexity.* Cambridge University Press, 1997.

[26] Rudolf Lidl and Harald Niederreiter. *Introduction to finite fields and their applications, revised edition.* Cambridge, 1994.

[27] Richard Lipton and Robert Sedgewick. Lower bounds for VLSI. In *Proceedings of the 13th Annual ACM Symposium on Theory of Computing*, pages 300–307, 1981.

[28] L'aszl'o Lovasz and Michael Saks. Communication complexity and combinatorial lattice theory. *Journal of Computer and System Sciences*, 47(2):322–349, 1993.

[29] Kurt Mehlhorn and Erik Schmidt. Las Vegas is better than determinism in VLSI and distributed computing. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing*, pages 330–337, 1982.

[30] Noam Nisan and Avi Wigderson. On rank vs. communication complexity. *Combinatorica*, 15(4):557–566, 1995.

[31] Ramamohan Paturi and Janos Simon. Probabilistic communication complexity. *Journal of Computer and System Sciences*, 33:106–123, 1986.

[32] Pavel Pudlák, Vojtěch Rödl, and Jiří Sgall. Boolean circuits, tensor ranks, and communication complexity. *SIAM Journal on Computing*, 26(3):605–633, 1997.

[33] Ran Raz. The BNS-Chung criterion for multi-party communication complexity. *Computational Complexity*, 9(2):113–122, 2000.

[34] Alexander Razborov. Lower bounds for the size of circuits of bounded depth with basis $\{\wedge, \oplus\}$. *Mathematical Notes of the Academy of Sciences of the USSR*, 41(4):333–338, 1987.

[35] Alexander Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55:24–35, 1997.

[36] Ronen Shaltiel. Towards proving strong direct prroduct theorems. In *Proceed-*

ings of the 16th Annual IEEE Conference on Computational Complexity, pages 107–119, 2001.

[37] Victor Shoup. New algorithms for finding irreducible polynomials over finite fields. *Mathematics of Computation*, 54:435–447, 1990.

[38] Roman Smolensky. Algebraic methods in the theory of lower bounds for Boolean circuit complexity. In *Proceedings of the 19th Annual ACM Symposium on Theory of Computing*, pages 77–82, 1987.

[39] Joel Spencer. *Ten Lectures on the Probabilistic Method*. Society for Industrial and Applied Mathematics, 1987.

[40] V. Strassen. Rank and optimal computation of generic tensors. *Linear Algebra and Its Applications*, 52/53:645–685, 1983.

[41] J. H. van Lint and R. M. Wilson. *A Course in Combinatorics*. Cambridge University Press, second edition, 2001.

[42] André Weil. On some exponential sums. *Proceedings of the National Academy of Sciences of the United States of America*, 34(5):204–207, 1948.

[43] Andrew Yao. Some complexity questions related to distributed computing. In *Proceedings of the 11th Annual ACM Symposium on Theory of Computing*, pages 209–213, 1979.

[44] Andrew Yao. Lower bounds by probabilistic arguments. In *Proceedings of the 24th Annual IEEE Symposium on Foundations of Computer Science*, pages 420–428, 1983.

[45] Andrew Yao. On ACC and threshold circuits. In *Proceedings of the 31th Annual IEEE Symposium on Foundations of Computer Science*, pages 619–627, 1990.

Vita

Jeff Ford was born in Lynchburg, Virginia on September 26, 1976, the son of Donald and Frances Ford. After completing his work at Craig County High School, New Castle, Virginia, in 1993, he entered the Virginia Polytechnic Institute and State University in Blacksburg, Virginia. He received a Bachelor of Science in Mathematics and a Bachelor of Science in Computer Science in May 1997. In August 1997 he entered the Graduate School of the University of Texas, and he received a Master of Science in Computer Sciences in August 1999.

Permanent Address: 12345 Lamplight Village Ave #1428
Austin, TX 78758

This dissertation was typeset with LaTeX 2_ε[1] by the author.

[1] LaTeX 2_ε is an extension of LaTeX. LaTeX is a collection of macros for TeX. TeX is a trademark of the American Mathematical Society. The macros used in formatting this dissertation were written by Dinesh Das, Department of Computer Sciences, The University of Texas at Austin, and extended by Bert Kay, James A. Bednar, and Ayman El-Khashab.