5-2016

Expander Graphs and Coding Theory

Michael C. Dowling Jr.
Clemson University

Expander Graphs and Coding Theory

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Mathematical Science

---

by
Michael C. Dowling, Jr.
May 2016

Accepted by:
Dr. Shuhong Gao, Committee Chair
Dr. Felice Manganiello
Dr. Gretchen Matthews
Dr. Michael Pursley

Abstract

Expander graphs are highly connected sparse graphs which lie at the interface of many different fields of study. For example, they play important roles in prime sieves, cryptography, compressive sensing, metric embedding, and coding theory to name a few. This thesis focuses on the connections between sparse graphs and coding theory.

It is a major challenge to explicitly construct sparse graphs with good expansion properties, for example Ramanujan graphs. Nevertheless, explicit constructions do exist, and in this thesis, we survey many of these constructions up to this point including a new construction which slightly improves on an earlier edge expansion bound.

The edge expansion of a graph is crucial in applications, and it is well-known that computing the edge expansion of an arbitrary graph is NP-hard. We present a simple algorithm for approximating the edge expansion of a graph using linear programming techniques. While Andersen and Lang (2008) proved similar results, our analysis attacks the problem from a different vantage point and was discovered independently.

The main contribution in the thesis is a new result in fast decoding for expander codes. Current algorithms in the literature can decode a constant fraction of errors in linear time but require that the underlying graphs have vertex expansion at least 1/2. We present a fast decoding algorithm that can decode a constant fraction of errors in linear time given any vertex expansion (even if it is much smaller than 1/2) by using a stronger local code, and the fraction of errors corrected almost doubles that of Viderman (2013).

Acknowledgments

I would like to thank the following people who have had a significant, positive impact in my graduate school experience.

My Lord and Savior Jesus Christ who loved me, died for me, and has given me salvation, life, and hope.

My mom, dad, sister, and grandfather for their unfailing and consistent love and support. Without them, I could not have completed my degree, and I am truly privileged to have them for my family.

Shuhong Gao, for his kindness, dedication, patience, and enthusiasm. I am very grateful to have had the opportunity to develop my research techniques under such an outstanding mentor.

My committee: Michael Pursley, Gretchen Matthews, and Felice Manganiello. I have greatly benefitted from their excellent instruction both inside and outside the classroom, and I have thoroughly enjoyed the opportunity of working with each of them.

My undergraduate professors who had an especially positive impact in my life: Gary Guthrie, Bob Taylor, Dave Brown, Richard Hand, and Ted Miller.

My friends and fellow graduate students for their stimulating discussions and for their encouragement and support: Nate Black, Jeff Beyerl, Hayato Ushijima-Mwesigwa, Thilo Strauss, Siddhartha Borkotoky, and Michael Juang.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this chapter, we introduce expander graphs and linear codes. First, we define expander graphs and provide some of the intuition behind these graphs as well. We then briefly survey several fundamental expansion bounds. Next, we provide a broad overview of coding theory, introduce linear codes, and state several key results. We also describe what we mean by asymptotically good codes and introduce the relationship between asymptotically good codes and expander graphs.

1.1 Expander Graphs

In this section, we first define the edge expansion and vertex expansion of a graph. As we will see, these expansion properties measure how well the vertices of a graph are inter-connected with each other. Next, we will present several well-known bounds for both edge expansion and vertex expansion. Finally, we will define Ramanujan graphs, which have very good expansion properties, followed by a brief description of a very recent result about Ramanujan graphs.

We now define edge expansion of a graph. Given a constant d, let G be a d-regular,

undirected graph (a graph for which exactly d edges are adjacent to each vertex). Let $V(G)$ denote the vertex set of G, and suppose that $S \subseteq V(G)$. Let $E(S, \overline{S})$ denote the set of edges between S and $\bar{S}$, where $\bar{S}$ denotes the vertices outside of S. Since we expect more edges between sets with similar sizes than between sets with vastly different sizes, we divide $|E(S, \overline{S})|$ by the size of the smaller of the two sets and scale by d. Since we want *every* small set to have many edges connected to vertices outside that set, we take the minimum of this ratio, and we denote the result as:

$$\Phi_E(G) := \min_{S \subseteq V(G)} \frac{|E(S, \overline{S})|}{d(\min(|S|, |\overline{S}|))}.$$

This definition is equivalent to

$$\Phi_E(G) := \min_{\substack{S \subseteq V(G) \\ |S| \leq |V(G)|/2}} \frac{|E(S, \overline{S})|}{d|S|}.$$

For any constant $\epsilon > 0$ and a fixed d, a d-regular graph G is called an ϵ-edge expander graph if $\Phi_E(G) \geq \epsilon$. Notice that for an ϵ-edge expander graph,

$$d|S| \geq |E(S, \overline{S})| \geq \epsilon d|S|$$

for every $S \subseteq V(G)$ with $|S| \leq |V(G)|/2$. For example, a 10-regular, 0.3-edge expander graph on a graph with 100 vertices is a graph for which every subset $S \subseteq V(G)$ with $|S| \leq 50$ has at least $3|S|$ edges to vertices outside S.

Intuitively, in a communication network represented by a graph in which the vertices correspond to communication nodes and the edges correspond to communication links between those nodes, an ϵ-edge expander graph gives a network in which there are many paths between any two sets of communication nodes. Larger values of ϵ imply more paths between the nodes. In other words, good ϵ-edge expanders ensure that there are no "bottlenecks" in

the communication network.

For a given $\epsilon > 0$ and fixed degree d, when we refer to ϵ-edge expander graphs, we typically have in mind an infinite family of d-regular ϵ-edge expander graphs.

Definition 1. Given a fixed $\epsilon > 0$ and a fixed degree d, a family $\{G_i\}_{i\in\mathbb{N}}$ of d-regular graphs with $|V(G_i)| < |V(G_{i+1})|$ is an infinite family of ϵ-edge expanders if $\Phi_E(G_i) \geq \epsilon$ for all $i \in \mathbb{N}$.

We now define expansion in terms of the vertices of a given d-regular graph $G = (V, E)$. We say that a vertex $v \in V$ is a neighbor of a set S if there is an edge $(u, v) \in E$ such that $u \in S$. For a set S of vertices, let $N(S)$ denote the set of all neighbors of vertices of S that are not in S. We can define a vertex expansion analogue to the edge expansion definition as follows:

$$\Phi_V(G) := \min_{\substack{S\subseteq G \\ |S|\leq|V|/2}} \frac{|N(S)|}{d|S|}.$$

Similarly to before, for an ϵ-vertex expander graph,

$$d|S| \geq |N(S)| \geq \epsilon d|S|$$

for every $S \subseteq V(G)$ with $|S| \leq |V(G)|/2$. Also, notice that

$$\Phi_V(G) \leq \Phi_E(G),$$

so vertex expansion is a stronger requirement than edge expansion. We say a graph G is an ϵ-vertex expander for some $\epsilon > 0$ if $\Phi_V(G) \geq \epsilon$, and we define a family of good vertex expanders analogously to before.

Definition 2. Given a fixed $\epsilon > 0$ and a fixed degree d, a family $\{G_i\}_{i\in\mathbb{N}}$ of d-regular graphs with $|V(G_i)| < |V(G_{i+1})|$ is an infinite family of ϵ-vertex expanders if $\Phi_V(G_i) \geq \epsilon$ for all $i \in \mathbb{N}$.

So far, we have concentrated on d-regular graphs. However, it is also possible to define expansion for non-regular graphs. In particular, we can characterize the vertex expansion of a (c, d)-biregular bipartite graph with c edges incident with each vertex on the left-hand side and d edges incident with each vertex on the right-hand side. This particular type of expansion will be very important to our results in Chapter 4.

Definition 3. A c-left regular bipartite graph $G = (L \cup R, E)$, where L denotes the set of vertices on the left-hand side of the graph and R denotes the set of vertices on the right-hand side of the graph, is a (c, γ, α) bipartite expander graph if for *every* $S \subseteq L$ with $|S| \leq \gamma|L|$,

$$|N(S)| \geq \alpha c|S|.$$

If G is (c, d)-biregular, we say that G is a (c, d, γ, α) bipartite expander graph.

We illustrate this definition pictorially below.

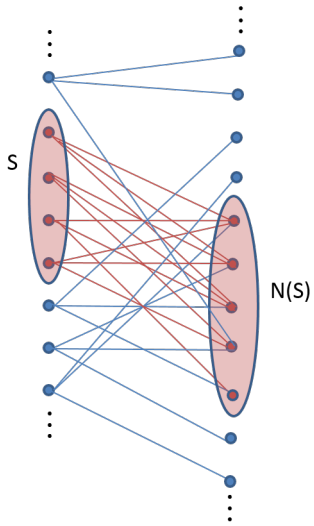


Figure 1.1: Bipartite Vertex Expansion

For example, in a $(36, 72, 0.04, 0.25)$ bipartite expander graph with 1000 vertices on the left-hand side and 500 vertices on the right-hand side, every set of 40 vertices on the

left-hand side has at least 360 neighbors on the right-hand side. The notion of bipartite vertex expansion is slightly different from that of vertex expansion since only sets of vertices on the left-hand side need to have many neighbors. There are no restrictions on the vertices on the right-hand side. Also, notice that it is not necessary to require that the graph is d-right regular, though we will use d-right regular graphs extensively in Chapter 4.

Pinsker [Pin73] showed that almost every d-regular graph has good edge expansion properties. Unfortunately, it is well-known that computing the exact edge expansion of a graph is an NP-hard problem [GJ79]. However, various methods exist for approximating the expansion of a graph. In Chapter 3, we will revisit one method for approximating $\Phi_E(G)$ using linear-programming techniques. Some of the most successful methods for bounding the expansion of a graph use the second-largest eigenvalue of the adjacency matrix of the graph, which we now define.

Definition 4. The edge-adjacency matrix of an undirected graph G with n vertices is the $n \times n$ symmetric matrix whose (i, j)-entry is 1 if there is an edge between vertex i and vertex j and 0 otherwise. Denote this matrix by $A(G)$.

Throughout this thesis, we denote the second-largest eigenvalue of $A(G)$ by $\lambda(G)$, and we call methods based on this second-largest eigenvalue "spectral methods." Notice that $\lambda(G)$ can be approximated in polynomial time.

Remark 1. The largest eigenvalue of the adjacency matrix of a d-regular graph is simply d and corresponds to the eigenvector consisting of all 1's.

By generalizing the Cheeger inequality (originally introduced in [Che70]), Dodziuk [Dod84] and Alon and Milman [AM85] used $\lambda(G)$ to approximate $\Phi_E(G)$.

Theorem 1 (Discrete Cheeger Inequality, [Dod84, AM85])**.** *Let G be a finite, connected,*

d-regular graph. Then,

$$\frac{1-\lambda(G)/d}{2} \leq \Phi_E(G) \leq \sqrt{2(1-\lambda(G)/d)}.$$

Note that the Discrete Cheeger Inequality guarantees that if $\lambda(G_n) < d$ for an infinite family of d-regular graphs $\{G_i\}_{i\in\mathbb{N}}$ with $|V(G_i)| < |V(G_{i+1})|$, then that family is an expander family.

Spectral methods can also be used to approximate $\Phi_V(G)$. One early result in this direction was given by Tanner [Tan84], who proved the following theorem:

Theorem 2 ([Tan84])**.** *Let G be a finite, connected, d-regular graph. Then,*

$$\frac{1}{d}\left(\frac{2}{1+\left(\frac{\lambda(G)}{d}\right)^2}\right) \leq \Phi_V(G).$$

Later, Kahale [Kah95] improved the spectral bound on vertex expansion.

Theorem 3 ([Kah95])**.** *Given a finite, connected, d-regular graph G, there is an absolute constant c such that*

$$\frac{1}{2}\left(1-\sqrt{1-4(d-1)/\lambda(G)^2}\right)(1-c\log d/\log 2) \leq \Phi_V(G).$$

Both Tanner and Kahale's results can also be applied to (c,d,γ,α) expander graphs. In this case, Kahale's approximation implies that spectral methods can only guarantee expansion up to $\alpha = c/2$ for a (c,d,γ,α) expander graph. Consequently, the eigenvalue bound for vertex expansion is not always tight. In fact, it is well-known that with high probability, random (c,d)-biregular graphs have expansion properties which surpass those which spectral expansion can guarantee. In Chapters 2 and 4, we will return to graphs whose vertex expansion exceeds the expansion bounds given by spectral methods. Finally, $\lambda(G)$ can also be used to approximate how rapidly a random walk on the graph converges to the uniform

distribution.

Theorem 4 ([HLW06])**.** *Let G denote a d-regular expander graph. Then, for any initial probability distribution $\mathbf{p}$, for all integers $t > 1$,*

$$\|\Psi^t \mathbf{p} - \mathbf{u}\|_1 \leq \sqrt{n} \left(\frac{\lambda(G)}{d} \right)^t,$$

where $\Psi = \frac{1}{d} A(G)$.

We will revisit the relationship between random walks and vertex expansion in the description of the modified zig-zag product given in Chapter 2.

In addition to providing insight into how quickly the distribution of a random walk converges to the uniform distribution, $\lambda(G)$ gives an approximation of how close the graph G is to being a random graph in the following sense. Given a graph G and two disjoint sets $S, T \subseteq G$, if we lay down an edge uniformly at random with one endpoint fixed in S, the probability that the other end will land in T is $\frac{|T|}{|V|}$. Summing over the $d|S|$ edges with one endpoint in S (and allowing the possibility of double-edges) we see that the expected number of edges between S and T in a random graph is $\frac{d|S||T|}{|V|}$. The Expander Mixing Lemma shows that in an expander graph G with small $\lambda(G)$, the number of edges between any two disjoint sets is close to the expected number of edges between those sets if the graph were constructed randomly.

Lemma 5 (Expander Mixing Lemma)**.** *Suppose we are given a d-regular graph $G = (V, E)$. Let $S, T \subseteq V$ be two disjoint sets. Then,*

$$\left| |E(S,T)| - \frac{d|S||T|}{|V|} \right| \leq \lambda(G) \sqrt{|S||T|}.$$

Given the fact that small values of $\lambda(G)$ imply good expansion, it is natural to ask how small $\lambda(G)$ can be made. The answer to this question (asymptotically) was given by Alon and Boppana in [Alo86] who presented the following theorem:

Theorem 6 (Alon-Boppana Bound, [Alo86])**.** *Let $\{G_i\}_{i\in\mathbb{N}}$ be any family of d-regular graphs with $|V(G_i)| \to \infty$ as $i \to \infty$. Then,*

$$\liminf_{i\to\infty} \lambda(G_i) \geq 2\sqrt{d-1}.$$

For four separate proofs of this theorem, see [LPS88, Nil91, Fri93, Nil04]. Graphs for which $\lambda(G) \leq 2\sqrt{d-1}$ (whose second-largest eigenvalues are asymptotically as small as possible) are called Ramanujan graphs. The Alon-Boppana bound also generalizes as follows:

Theorem 7 ([FL96])**.** *Let $\{G_i\}_{i\in\mathbb{N}}$ be any family of (c,d)-biregular bipartite graphs with $|V(G_i)| \to \infty$ as $i \to \infty$. Then,*

$$\liminf_{i\to\infty} \lambda(G_i) \geq \sqrt{c-1} + \sqrt{d-1}.$$

Analagously to before, we say that a (c,d)-biregular bipartite graph $G = (L \cup R, E)$ is Ramanujan if $\lambda(G) \leq \sqrt{c-1} + \sqrt{d-1}$.

In [Fri03], Friedman proved a conjecture by Alon that for any $d \geq 3$ and any $\epsilon > 0$, $\lambda(G) \leq 2\sqrt{d-1} + \epsilon$ for almost every graph on a sufficiently large number of vertices. The same paper mentioned that in numerical experiments, the average $\lambda(G)$ (given a graph G with a fixed degree and a fixed number of vertices) is actually smaller than $2\sqrt{d-1}$. Consequently, it is conjectured but not proved that almost all graphs are actually Ramanujan. Recently, Marcus et al. [MSS15] proved the following theorem:

Theorem 8 ([MSS15])**.** *For fixed $c, d \geq 3$, there exists an infinite family of (c,d)-biregular bipartite Ramanujan graphs.*

While this theorem proved the existence of such families of graphs, the only explicit constructions of Ramanujan graphs as of yet are d-regular graphs for special values of d. In Chapter 2, we will return to these constructions. Note that Ramanujan graphs are one of the few cases in which the parameters of explicitly constructed graphs surpass the parameters guaranteed by probabilistic arguments.

1.2 Linear Codes

In this section, we introduce linear codes and briefly mention applications of expander graphs to coding theory. We will significantly elaborate on these applications in Chapter 4.

1.2.1 Basic Definitions

A linear code $\mathcal{C}$ is a k-dimensional vector space in $\mathbb{F}^n$, where $\mathbb{F}$ is a finite field (for example $\mathbb{F}_{2^\ell}$ for some $\ell \geq 1$). We call such codes $[n, k]$ codes. Since $\mathcal{C}$ is a k-dimensional vector space, it can be described as the set of linear combinations of k linearly independent row vectors of length n. In particular, a $k \times n$ matrix G with rank k gives the k-dimensional code $\mathcal{C} = \{xG : x \in \mathbb{F}^k\}$, and we call G the generator matrix of the code $\mathcal{C}$. Alternatively, the code $\mathcal{C}$ can be defined by an $(n-k) \times n$ matrix H of rank $n-k$ (i.e. $\mathcal{C} = \{y \in \mathbb{F}^n : Hy = 0\}$). In this case, H is called the parity-check matrix of the code $\mathcal{C}$. (While G could be an $m \times n$ matrix with $m \geq k$, and H could be an $m' \times n$ matrix with $m' \geq n-k$, for sake of illustration, we consider only the case when $m = k$ and $m' = n - k$.) Notice that given a $k \times n$ generator matrix G of rank k, any vector $v \in \mathbb{F}^k$ can be encoded into a codeword $c \in \mathbb{F}^n$. In codeword c, there are $n - k$ redundant symbols and k information symbols. We define the rate of a k-dimensional code $\mathcal{C}$ as $r := k/n$. Intuitively, the rate measures the amount of information in the code. Codes with high rate have many information symbols and little redundancy.

The *Hamming distance* between two codewords is the number of coordinates in which

the two codewords differ. The *Hamming weight* of a codeword is the number of non-zero coordinates in that codeword. The *minimum distance* of a code $\mathcal{C}$ is the minimum Hamming distance between any two codewords. For a linear code, the minimum Hamming distance is equivalent to the minimum Hamming weight. We call a linear code of length n, dimension k, and minimum Hamming distance d an $[n, k, d]$ code. The *relative minimum distance* of an $[n, k, d]$ code is given by $\delta := d/n$.

It is fairly easy to see that a nearest-neighbor decoder which decodes an arbitrary vector in $\mathbb{F}^n$ to the nearest codeword can correct any combination of up to $\lfloor \frac{d-1}{2} \rfloor$ errors. Moreover, no decoder can guarantee correction of all patterns of e errors when $e > \lfloor \frac{d-1}{2} \rfloor$. Consequently, to obtain codes which can guarantee correction of as many errors as possible with as high of a rate as possible, it is of great interest to obtain codes with the best possible rate-minimum distance tradeoff. In particular, given a fixed minimum distance (rate), we would like to determine the highest possible rate (minimum distance) for a linear code.

1.2.2 Rate-distance Tradeoff

It is not too difficult to prove that the *Singleton bound* ($d \leq n - k + 1$) gives the best possible upper bound for the rate-distance tradeoff. In terms of the rate r and minimum distance δ of a code, the Singleton bound is given by $\delta \leq 1 - r + 1/n$. It is well-known that Reed-Solomon (RS) codes achieve the Singleton bound. However, they do so at the expense of a field size $q \approx n$. Consequently, it is a very important problem to find the best rate-distance tradeoff given a *fixed* field size. Despite many decades of research, provably tight bounds have eluded researchers even in the asymptotic case. However, several good bounds have been obtained. While we give these bounds for linear codes, analogous bounds hold for non-linear codes as well.

Let $\delta > 0$ be any constant less than 1. For any $n > 1$, let $d = \lfloor \delta n \rfloor$, and let $k_q(n, d)$ denote the maximum dimension of a linear code of length n with minimum distance d and

field size q. Define

$$r_q(\delta) = \limsup_{n \to \infty} \frac{k_q(n, \lfloor \delta n \rfloor)}{n}.$$

This parameter gives the maximum asymptotic rate possible for a code given a fixed field size and a fixed relative minimum distance. A very well-known lower bound is the (asymptotic) Gilbert-Varshamov (GV) bound:

$$r_q(\delta) \geq 1 - h_q(\delta) - o(1),$$

where

$$h_q(\delta) := \delta \log_q(q-1) - \delta \log_q \delta - (1-\delta) \log_q(1-\delta).$$

In a remarkable breakthrough, Tsfasman et al. [TVZ82] constructed algebraic geometry (AG) codes with a fixed field size $q \geq 49$ (when q is a square) which surpassed the GV bound. Unfortunately, to our knowledge, there is still no explicit construction of binary codes (linear or non-linear) which surpass or even reach the GV bound. However, random binary linear codes achieve the GV bound with high probability. It is conjectured that the GV bound is asymptotically tight for binary codes.

Note that if most of the codewords are far apart but only two codewords are close to each other, then a decoder which decodes to the nearest neighbor of the received word will typically perform well despite a possibly poor tradeoff between the rate and the minimum distance. In particular, the number of errors which the decoder is *guaranteed* to correct could be much smaller than the number of errors which the decoder is *likely* to correct, assuming the symbols in a codeword are altered independently and at random. Consequently, for many applications, it is of great interest to determine how well a code typically performs.

1.2.3 Channels and the Shannon Capacity

Before determining bounds on how well codes can typically perform, we must first specify a model for how the errors are introduced in the codewords. This model is called the *channel*. A *discrete channel* consists of an input alphabet $\mathcal{I}$ and an output alphabet $\mathcal{O}$ with a transition probability matrix

$$\mathbf{T} = (p(y|x))$$

where $p(y|x)$ denotes the probability of observing output symbol $y \in \mathcal{O}$ given that input symbol $x \in \mathcal{I}$ was sent. A *discrete memoryless channel* (DMC) is a discrete channel in which the probabilities $p(y|x)$ do not depend on previous channel input or output symbols. Given a received word $\hat{\mathbf{y}} = (\hat{y}_1, \ldots, \hat{y}_n) \in \mathcal{O}^n$, *maximum-likelihood (ML) decoding* decodes to the codeword $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \mathcal{I}^n$ which maximizes the probability $p(\hat{\mathbf{y}}|\mathbf{x}) = \prod_{i=1}^{n} p(\hat{y}_i|x_i)$.

The primary DMC channel in which we will be interested is the Binary Symmetric Channel (BSC). Given a binary codeword transmitted over a BSC, each bit is flipped with some probability p. It is easy to show that on the BSC, ML decoding is equivalent to nearest-neighbor decoding. Note that for field sizes greater than 2, one can easily generalize to a q-ary symmetric channel in which each symbol is altered with some probability p and takes one of the other possible symbol values with probability $p/(q-1)$. The BSC is often assumed for hard-decision decoders which decide that a bit is either a zero or a one and provide no additional reliability information. In this chapter, we will be primarily concerned with hard-decision decoders.

The fundamental limit on how well codes can perform on average over a given DMC channel was given by Shannon [Sha48] in his noisy channel coding theorem and in its converse. Shannon called this fundamental limit the *capacity* of the code. For additional details about the Shannon capacity, see [CT12]. Before giving the noisy channel coding theorem, we first introduce some notation. Given a code $\mathcal{C}$ and a DMC, let $\mathcal{I}$ denote the set of possible inputs

to the channel and $\mathcal{O}$ denote the set of possible outputs of the channel, where $\mathcal{C} \subseteq \mathcal{I}^n$. Let $\mathcal{D} : \mathcal{O}^n \to \mathcal{C}$ denote a decoding algorithm for $\mathcal{C}$. Suppose a codeword $\mathbf{x} \in \mathcal{C}$ is sent over a discrete memoryless channel where it may incur errors according to the channel's error model. Suppose the vector $\mathbf{y} \in \mathcal{O}^n$ is received. Then, let $P_e(\mathcal{C})$ denote the maximum probability of not decoding to the correct codeword. In particular, define

$$P_e(\mathcal{C}) := \max_{\mathbf{x} \in \mathcal{C}} \Pr(\mathcal{D}(\mathbf{y}) \neq \mathbf{x}).$$

Shannon's noisy channel coding theorem for DMC channels can now be stated as follows:

Theorem 9 (Shannon's Channel Coding Theorem)**.** *Let C denote the capacity of a given DMC channel with input set $\mathcal{I}$ and output set $\mathcal{O}$. Then, for every rate $R < C$, there exists an infinite sequence of codes $\{\mathcal{C}_n\}$ ($\mathcal{C}_n \subseteq \mathcal{I}^n$) of rate R with a decoding algorithm such that $P_e(\mathcal{C}_n) \to 0$ as $n \to \infty$. Conversely, for any infinite sequence of codes $\{\mathcal{C}_n\}$ ($\mathcal{C}_n \subseteq \mathcal{I}^n$) of rate R such that $R > C$, under any decoding algorithm, $P_e(\mathcal{C}_n) \to 1$ as $n \to \infty$.*

The capacity of quite a few channels can be explicitly computed. In particular, the capacity of a BSC is given by $1 - h_2(p)$, where $h_2(x)$ denotes the binary entropy function, and p is the probability of error. Similarly, the capacity of the q-ary symmetric channel is $1 - h_q(p)$.

1.2.4 Capacity-achieving Codes

Codes for which the properties of Shannon's Noisy Channel Coding Theorem hold are called *capacity-achieving codes*. Shannon's random-coding proof does not provide a method for constructing capacity-achieving codes having practical decoding algorithms. However, after decades of research, two classes of codes with practical decoding algorithms have emerged which come very close to achieving the capacity of the AWGN channel. The first class of codes coming close to achieving capacity are called low-density parity check (LDPC) codes and

were introduced by Gallager [Gal63]. The second class of codes coming close to achieving capacity are called Turbo codes and were introduced by Berrou, Glavieux, and Thitimajshima [BGT93].

There are a variety of codes which achieve the capacity of q-ary symmetric channels (see [Sho04]). One such code dating back to 1966 is the concatenated coding scheme which Forney [For66] introduced in his dissertation. In essence, concatenated codes encode a word using one encoder then provide the resulting codeword as the input to a second encoder. We will return briefly to concatenated codes in Chapter 4.

The analysis of both Turbo codes and LDPC codes are based on the study of graphs of which there are three major types. *Trellis graphs* provide the basis for much of the intuition underlying convolutional codes (including Turbo codes), *Tanner graphs* provide the basis for much of the intuition behind LDPC codes, and *factor graphs* unite Tanner graphs and trellis graphs under a single theoretical umbrella. In chapter 4, where we will be almost exclusively interested in LDPC codes, we will focus on Tanner graphs.

1.2.5 Asymptotically Good Codes

As mentioned previously, rate-distance tradeoff analysis and Shannon capacity analysis give two different types of bounds. In particular, rat-distance tradeoff analysis can guarantee correction of *any* pattern of m errors under ML-decoding whenever m is within a specified bound. In contrast, Shannon capacity analysis does not attempt to bound the number of errors which a decoder can correct. Instead (for the BSC), Shannon capacity analysis guarantees that there is a code and a decoding algorithm such that for any codeword, if errors are introduced independently and at random with some probability p, then as long as p is below $h_2^{-1}(1-r)$, where r is the rate of the code, the decoding algorithm can correct those errors with probability approaching 1 as the length of the code approaches infinity. Since the probability of correct decoding approaches 1, most error patterns occurring with

high probability can be corrected, though there could be a few error patterns with a fraction of errors much smaller than p which the decoder cannot correct.

To contrast rate-distance tradeoff analysis and Shannon capacity analysis, if we assume that the GV bound is indeed asymptotically tight for binary codes, no decoder for an arbitrarily long rate-1/2 binary code can guarantee correction of an error pattern having more than $(h_2^{-1}(1/2)/2)n \approx 0.055n$ errors (where n denotes the length of the code). In contrast, a capacity-achieving, rate-1/2 binary code over the BSC can correct with high probability an error pattern in which each bit is corrupted independently and at random with probability $h_2^{-1}(1/2) \approx 0.110$.

Both types of analysis provide valuable information about the performance of a code. However, in this thesis, we will focus on families of codes with (preferably fast) decoding algorithms which can correct *any* pattern of up to αn errors (even as $n \to \infty$), where n denotes the length of the code, and $\alpha > 0$ is a constant. Assuming the field size remains fixed, we will call such codes *asymptotically good* codes. The first family of asymptotically good codes with a linear-time decoding algorithm was constructed by Sipser and Spielman [SS96] from expander graphs. They called such codes "expander codes," and to the best of our knowledge, expander codes are the only asymptotically good codes with linear-time decoding algorithms.

1.3 Organization of Dissertation

This dissertation is organized as follows. In Chapter 2, we survey many explicit constructions of Ramanujan graphs and non-Ramanujan expander graphs. In Section 2.1.7, we also introduce a new construction of non-Ramanujan expander graphs based on affine linear transformations.

In Chapter 3, we present a simple algorithm for approximating the expansion of a

graph using linear programming techniques. In particular, we compute the set with minimum edge expansion within a given, fixed set of size at most half the size of the graph. We then show how to find a set with minimum edge expansion (up to an approximation guarantee) over all sets with "enough" overlap of a given, fixed set. Though similar results appear in [AL08], our results were discovered independently, and our analysis attacks the problem from a different point of view.

In Chapter 4, after briefly describing Tanner graphs, LP decoding, and message-passing decoding, we summarize the types of results obtained using girth-based analysis. While girth-based analysis can provide excellent typical-case bounds, we emphasize that girth-based analysis does not prove that these types of decoding algorithms can correct *any* pattern of of up to αn errors, where n denotes the length of the code, $\alpha > 0$ is fixed, and n is allowed to go to infinity.

Next, we focus on fast decoding algorithms for expander codes which can guarantee that any pattern of up to αn errors will be corrected, even as n goes to infinity. The analysis of these decoding algorithms breaks into two parts: spectral analysis based on the second-largest eigenvalue of the edge-adjacency matrix of the code's underlying Tanner graph and expansion analysis based only on the vertex expansion of the code's underlying Tanner graph.

After surveying results based on spectral analysis, we provide detailed descriptions of the algorithms whose analysis is based on vertex expansion, and we give full proofs of the guarantees for these algorithms. None of these guarantees allow vertex expansion less than 1/2. However, our new result in Section 4.4.2 shows that it is possible to produce asymptotically good families of codes with linear-time decoding complexity even when the vertex expansion is significantly smaller than 1/2. We present the details of our decoding algorithm and a full proof of our results. We also note that our decoding algorithm can correct approximately double the fraction of errors corrected by the decoding algorithm in [Vid13]. In Chapter 5, we make some concluding remarks and mention a few open problems.

Chapter 2

Explicit Expander Constructions

In this chapter, we provide a summary of many of the explicit constructions of expander graphs up to this point, describe several of these constructions in detail, and give a new construction of expander graphs using affine linear transformations.

2.1 Margulis-type Constructions

One of the original motivations for finding explicit constructions of expander graphs was to build superconcentrators. Intuitively, superconcentrators consist of a set of inputs and a set of outputs with paths in-between such that for any small set of inputs, there are many disjoint paths to the outputs. For more details about superconcentrators, see [Pip77, Chu79, AGM87]. Pinsker and Bassalygo [BP73] gave the first definition of expander graphs, and Pinsker [Pin73] provided a nonconstructive probabilistic argument for obtaining these graphs with high probability. He then used expander graphs to construct superconcentrators. The first *explicit* construction of expander graphs (hence also the first explicit construction of superconcentrators) was given by Margulis [Mar73] who constructed a bipartite 5-regular expander graph. A slight modification of Margulis's construction [Ang79] produced 3-regular

expander graphs as well. Another explicit construction was given in [Sch80], though that paper makes no explicit reference to expanders or superconcentrators. Unfortunately, the arguments in [Mar73, Ang79] were insufficient to verify how good their constructed expanders were. However, using a construction similar to that in [Mar73] coupled with techniques from Fourier analysis, Gabber and Galil [GG79] (full journal article in [GG81]) constructed expander graphs with guaranteed expansion properties. Recently, Linial and London [LL06] gave (vertex) expansion properties for a Margulis-like construction as well.

A few years later, several new constructions emerged. Tanner [Tan84] followed a different approach from that in [Mar73] and constructed expander graphs using finite geometries. Another way to build expander graphs geometrically was given by Alon [Alo85] (see [Alo86] for the full journal article version). While Tanner demonstrated that his method could not be used to construct an infinite family of expander graphs, his graphs had excellent expansion guarantees and were recently used to build codes with good properties [HJ06, BHPJ13]. Even more importantly, Tanner's paper was one of the first results linking the expansion of a graph with the second-largest eigenvalue of that graph's adjacency matrix.

In rapid succession, four other papers [Dod84, AM84, AM85, JM85] also studied the connection between the second-largest eigenvalue of the adjacency matrix (or almost equivalently, the graph Laplacian) and the graph's expansion properties. Using the correspondence between the expansion of a graph and its second-largest eigenvalue, Alon and Milman [AM84] and [AM85] introduced many new constructions of expander graphs. In addition, as described in Chapter 1, Dodziuk [Dod84] as well as Alon and Milman [AM85] explicitly linked the spectral properties of a graph with the Cheeger inequality [Che70] from analysis. This correspondence proved to be a major breakthrough in the construction and analysis of expander graphs. Soon afterwards, Buck [Buc86] explicitly linked good expanders with rapidly mixing random walks, and used techniques exploiting this relationship to provide a different construction for expander graphs. Jimbo and Maruoka [JM85] (see

[JM87] for the full journal article) used more elementary techniques from linear algebra and Fourier analysis to obtain a proof (which is still quite intricate) for the expansion guarantees of their construction. Finally, Alon, Galil, and Milman [AGM87] combined the ideas in [Tan84, AM84, AM85, JM85] to produce the construction surveyed in [HLW06]. We now give an overview of some of these constructions and introduce a new construction building on the analysis presented in [HLW06].

2.1.1 Original Margulis Construction [Mar73]

We will construct a bipartite graph $G = (L \cup R, E)$. Let $L = R = \mathbb{Z}_m^2$, and define

$$T_1 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, T_2 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}, e_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, e_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

For any $z \in L$, define the following edges in $L \times R$: (z, z), $(z, z + e_1)$, $(z, z + e_2)$, $(z, T_1 z)$, and $(z, T_2 z)$.

2.1.2 Angluin Construction [Ang79]

We will construct a bipartite graph $G = (L \cup R, E)$. Let $L = R = \mathbb{Z}_m^2$, and define

$$T_1 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, T_2 = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}.$$

For any $z \in L$, define the following edges: (z, z), $(z, T_1 z)$, and $(z, T_2 z + e_1)$.

2.1.3 Gabber Galil Constructions [GG81]

Construction 1: We will again construct a bipartite graph $G = (L \cup R, E)$. Let $L = R = \mathbb{Z}_m^2$, and define

$$T_1 = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, T_2 = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}.$$

For any $z \in L$, define the following edges: (z, z), $(z, T_1 z)$, $(z, T_1 z + e_2)$, $(z, T_2 z)$, and $(z, T_2 z + e_1)$.

Construction 2: It is possible to double the expansion factor with the next construction. Let

$$T_1 = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}, T_2 = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}.$$

For any $z \in \mathbb{Z}_m^2$, define the following edges: (z, z), $(z, T_1 z)$, $(z, T_1 z + e_2)$, $(z, T_1 z + 2e_2)$, $(z, T_2 z)$, $(z, T_2 z + e_1)$, and $(z, T_2 z + 2e_1)$.

2.1.4 Alon Milman Construction [AM84, AM85]

Before proceeding with this next construction, recall the definition of a Cayley graph:

Definition 5. Let G be a group and $S \subseteq G$ (S does not necessarily need to be a subgroup of G). Then, the Cayley graph $C(G, S)$ has vertex set G and edge set

$$E = \{(u, v) \in G \times G : u \cdot s = v \text{ for some } s \in S\}.$$

If for each $s \in S$ it is also true that $s^{-1} \in S$, the set S is called symmetric. In this case, the graph $C(G, S)$ is undirected and $|S|$-regular. Now, define

$$T_1 = \begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ (-1)^{n-1} & 0 & 0 & \cdots & 0 \end{pmatrix}, T_2 = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \\ (-1)^{n-1} & 0 & 0 & \cdots & 0 & 1 \end{pmatrix}.$$

We form an infinite family of Cayley graphs $C(G_i, S_i)$ with $G_i = SL_n(\mathbb{Z}_i)$ and $S_i = \left\{T_1, T_1^{-1}, T_2, T_2^{-1}\right\} \pmod{i}$.

2.1.5 Alon Galil and Milman Construction [AGM87]

Construction 1: We construct the graph $G = (V, E)$. Let $V = \mathbb{Z}_m^2$, and define T_1 and T_2 as in [GG81]:

$$T_1 = \begin{pmatrix} 1 & 0 \\ 2 & 1 \end{pmatrix}, T_2 = \begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix}.$$

For any $z \in \mathbb{Z}_m^2$, define the following edges: $(z, T_1 z)$, $(z, T_1 z + e_2)$, $(z, T_2 z)$, and $(z, T_2 z + e_1)$ along with their four inverses. This construction gives an 8-regular graph with expansion factor $8 - 5\sqrt{2}$. (This expansion factor was proved in [JM85].)

Construction 2: Same as above, but add the following edges with their inverses: $(z, T_1 z + 2e_2)$, $(z, T_2 z + 2e_1)$.

2.1.6 Analyzing the Discrete Margulis Construction

The goal of this section is to show that the undirected graph formed by the construction in [AGM87] is actually an expander and to provide an expansion guarantee. We closely

follow the analysis presented in [HLW06] but slightly enlarge on some of the explanations.

Theorem 10 ([JM87, AGM87]). *Let $\{G_n\}_{n\in\mathbb{N}}$ be a family of graphs constructed as in section 2.1.5, and let $\lambda(G_i)$ denote the second-largest eigenvalue of the adjacency matrix of G_i. Then, for any $i \in \mathbb{N}$, $\lambda(G_i) \leq 5\sqrt{2} \leq 8$.*

Note that since $\frac{1-\lambda/d}{2}$ is a lower bound for the expansion ratio $\Phi_E(G)$ by the Cheeger inequality, this theorem implies that $\{G_n\}$ is a family of expanders.

We will prove a slightly weaker result below. In particular, we will show that $\lambda(G_n) \leq 7.25 < 8$. However, Hoory et al. [HLW06] note that using similar analysis to that given below, it is possible to improve the bound to $5\sqrt{2} \approx 7.07$. To prove the slightly weaker result, we will use the Rayleigh quotient as is usual in arguments of this type. Let $\mathbb{R}^n$ be the vector space over which we are working. Recall the Courant-Fischer minimax principle (where $\lambda(G)$ denotes the second-largest eigenvalue in absolute value of the adjacency matrix of a graph G):

$$\lambda(G) = \min_{\dim\,(W)=n-1} \max_{\mathbf{x}\in W, \mathbf{x}\neq\mathbf{0}} \frac{\|\mathbf{x}^T A\mathbf{x}\|}{\|\mathbf{x}\|^2}.$$

Since the matrix A is symmetric, the Spectral Theorem guarantees that the eigenvectors of A are orthogonal and that they form a basis for $\mathbb{R}^n$. Since the vector of all ones, denoted $\mathbf{1}$, is an eigenvector of the adjacency matrix, the vectors $\{\mathbf{1}, \mathbf{x}_1, \ldots, \mathbf{x}_{n-1}\}$ form an orthogonal basis for $\mathbb{R}^n$, where the $\mathbf{x}_i$, $1 \leq i \leq n-1$ are eigenvectors of A. Let W' denote the subspace spanned by $\mathbf{x}_1, \ldots, \mathbf{x}_{n-1}$ (i.e. all vectors $\mathbf{y} \perp \mathbf{1}$). If for all $\mathbf{y} \in W'$,

$$\frac{\mathbf{y}A\mathbf{y}^T}{\|\mathbf{y}\|^2} \leq 7.25,$$

then $\lambda(G) \leq 7.25$ because by the Courant-Fischer minimax principle,

$$\lambda(G) = \min_{\dim(W)=n-1} \max_{\mathbf{y}\in W, \mathbf{y}\neq\mathbf{0}} \frac{\mathbf{y}A\mathbf{y}^T}{\|\mathbf{y}\|^2} \leq \max_{\mathbf{y}\in W'} \frac{\mathbf{y}A\mathbf{y}^T}{\|\mathbf{y}\|^2} \leq 7.25.$$

Recall that an n-dimensional real vector space can be represented as a collection of functions $f : [n] \to \mathbb{R}$ where $[n] := \{1, 2, \ldots, n\}$. Using this representation of our vector space, we wish to show that for any such function f where $\sum_x f(x) = 0$ (i.e. $\mathbf{f} \perp \mathbf{1}$),

$$\frac{2 \sum\limits_{(x,y)\in E} f(x)f(y)}{\sum_x f(x)^2} \leq 7.25. \tag{2.1}$$

Remark 2. The multiple of 2 results from the fact that the adjacency matrix contains an entry for the edge from vertex x to vertex y as well as an entry from the edge from vertex y to vertex x. Note that in the case of self-loops, the left-hand side of (2.1) provides an upper bound for the Rayleigh quotient, not the exact Rayleigh quotient.

Consequently, proving the theorem reduces to showing that

$$\sum_{(x,y)\in E} f(x)f(y) \leq 3.625 \sum_x f(x)^2.$$

As one can obtain all the edges by applying only forward transformations, showing (2.1) is equivalent to showing

$$\sum_{z\in\mathbb{Z}/n\mathbb{Z}\times\mathbb{Z}/n\mathbb{Z}} f(z)[f(T_1 z) + f(T_1 z + e_1) + f(T_2 z) + f(T_2 z + e_2)] \leq 3.625 \sum_z f(z)^2. \tag{2.2}$$

We are now ready to leverage tools from discrete Fourier analysis.

Definition 6. Let H be a group and F a field. A characteristic function is a homomorphism $\chi : H \to F^\times$ (i.e. $\chi(ab) = \chi(a)\chi(b)$).

For the construction in Section 2.1.5, $H = \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$, and $F = \mathbb{C}$.

Definition 7. Let ω denote the primitive root of unity $e^{2\pi i/n}$. Define $\chi_a : \mathbb{Z}/n\mathbb{Z}\times\mathbb{Z}/n\mathbb{Z} \to \mathbb{C}$ by $\chi_a(z) = \omega^{<a,z>}$.

Definition 8. Let $f : H \to \mathbb{C}$ be a function (not necessarily a homomorphism) from the group H to $\mathbb{C}$. The discrete Fourier transform of f is given by $\hat{f}(x) =< f, \chi_x >$, where $< f, g >= \sum_{x \in H} \overline{f(x)} g(x)$ denotes the complex inner product.

It is a well-known result that if H is a finite abelian group, the characteristic functions $\{\chi_a\}_{a \in H}$ form an orthonormal basis for the vector space $\mathcal{F}$ of all functions $f : H \to \mathbb{C}$. So, for any $f \in \mathcal{F}$, the function $f = \sum_{y \in H} \hat{f}(y) \chi_y$. We now list some properties of discrete Fourier transforms which we will soon need:

- Property 1: The inverse Fourier transform is given by $f(a) = \frac{1}{n^2} \sum_{b \in H} \hat{f}(b) \omega^{-<a,b>}$.
- Property 2: $\sum_{b \in H} f(b) = 0$ if and only if $\hat{f}(0) = 0$.
- Property 3: $\langle f, g \rangle = \frac{1}{n^2} \left\langle \hat{f}, \hat{g} \right\rangle$.
- Property 4 (Parseval's Identity): $\sum_{b \in H} |f(b)|^2 = \frac{1}{n^2} \sum_{b \in H} |\hat{f}(b)|^2$.
- Property 5 (Shift Property): If $g(x) = f(Ax + b)$, then $\hat{g}(x) = \omega^{-<A^{-1}b, x>} \hat{f}((A^{-1})^T x)$.

Note that because $f(x)$ is a real-valued function, we can re-write (2.2) as

$$\langle f(z), f(T_1 z) + f(T_1 z + e_1) + f(T_2 z) + f(T_2 z + e_2) \rangle \leq 3.625 \sum_z f(z)^2$$

which, after applying Property 3, the Shift Property, and Parseval's Identity, becomes

$$\frac{1}{n^2} \left| \left\langle \hat{f}(z), \hat{f}((T_1^{-1})^T z) + \omega^{-<T_1^{-1} e_1, z>} \hat{f}((T_1^{-1})^T z) + \hat{f}((T_2^{-1})^T z) + \omega^{-<T_2^{-1} e_2, z>} \hat{f}((T_2^{-1})^T z) \right\rangle \right| \tag{2.3}$$

$$\leq 3.625 \frac{1}{n^2} \sum_z |\hat{f}(z)|^2.$$

Note that

$$T_1^{-1} = \begin{pmatrix} 1 & n-2 \\ 0 & 1 \end{pmatrix} \equiv \begin{pmatrix} 1 & -2 \\ 0 & 1 \end{pmatrix} \mod n$$

and

$$T_2^{-1} = \begin{pmatrix} 1 & 0 \\ n-2 & 1 \end{pmatrix} \equiv \begin{pmatrix} 1 & 0 \\ -2 & 1 \end{pmatrix} \bmod n.$$

So, $T_1^{-1}e_1 = e_1$, $T_2^{-1}e_2 = e_2$, $(T_1^{-1})^T = T_2^{-1}$, and $(T_2^{-1})^T = T_1^{-1}$. Combining these facts and canceling the $1/n^2$ terms, we see that (2.3) is equivalent to

$$\left|\left\langle \hat{f}(z), \hat{f}(T_2^{-1}z)(1+\omega^{-z_1}) + \hat{f}(T_1^{-1}z)(1+\omega^{-z_2})\right\rangle\right| \leq 3.625 \sum_z |\hat{f}(z)|^2. \tag{2.4}$$

As an aside, using Cauchy-Schwarz and the triangle inequality gives

$$\left|\left\langle \hat{f}(z), \hat{f}(T_2^{-1}z)(1+\omega^{-z_1}) + \hat{f}(T_1^{-1}z)(1+\omega^{-z_2})\right\rangle\right| \leq$$

$$\|\hat{f}\|_2\|\hat{f}\|_2(|1| + |\omega^{-z_1}| + |1| + |\omega^{-z_2}|) = 4\sum_z |\hat{f}(z)|^2$$

which is not sufficient. We must show that $\lambda(G_n) < 8$ not that $\lambda(G_n) \leq 8$. Otherwise, we cannot conclusively state that this family of graphs is a family of expanders. Coming back to the argument, (2.4) gives the following form of the original theorem:

Theorem 11 ([HLW06])**.** *For every $F : \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z} \to \mathbb{C}$ with $F((0,0)) = 0$,*

$$\sum_{z \in \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}} \overline{F(z)}\left(F(T_2^{-1}z)(1+\omega^{-z_1}) + F(T_1^{-1}z)(1+\omega^{-z_2})\right) \leq 3.625 \sum_z |F(z)|^2.$$

Next, still following [HLW06], define $G : \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z} \to \mathbb{R}$ by $G := |F|$. Moreover, note that $|1+\omega^{-t}| = 2|\cos(\pi t/n)|$. This identity follows since

$$|1+\omega^{-t}| = \sqrt{(1+\cos(-2\pi t/n))^2 + \sin(-2\pi t/n)^2}.$$

(Recall that $\omega^t = \cos(2\pi t/n) + i\sin(2\pi t/n)$.) Then,

$$(1+\cos(-2\pi t/n))^2 + \sin^2(-2\pi t/n) = 2 + 2\cos(-2\pi t/n)$$
$$= 4\left(\frac{1+\cos(2\pi t/n)}{2}\right) = 4\cos^2(\pi t/n).$$

So, $|1+\omega^{-t}| = 2|\cos(\pi t/n)|$. Now, using the triangle inequality,

Theorem 12 ([HLW06])**.** *For every non-negative function $G : \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z} \to \mathbb{R}$ where $G(0,0) = 0$,*

$$\sum_{z \in \mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}} 2G(z)\left(G(T_2^{-1}z)|\cos(\pi z_1/n)|) + G(T_1^{-1}z)|\cos(\pi z_2/n)|\right)$$
$$\leq 3.625 \sum_z |G(z)|^2. \quad (2.5)$$

As noted previously, by using Cauchy-Schwarz and the triangle inequality, one cannot conclude that this family of graphs is a family of expanders. So, we must use a more flexible tool.

Lemma 13 (Young's Inequality)**.** *For any non-negative a, b, γ,*

$$2ab \leq \gamma a^2 + \gamma^{-1} b^2.$$

Proof. The proof is simply a consequence of the fact that $(\gamma a + \gamma^{-1} b)^2 \geq 0$. □

Note that by setting $\gamma = 1$, we are back to the triangle inequality which was almost enough to show that the graph was an expander. So, it is reasonable to expect that the ultimate value of γ will be fairly close to 1 as well. Applying Young's inequality to (2.5) gives the following theorem (where the subscript on γ emphasizes that different values for γ are chosen depending on the value of z):

Theorem 14 ([HLW06]). *For any $G : \mathbb{Z}_n^2 \to \mathbb{R}$ such that $G(0,0) = 0$,*

$$\sum_z |\cos(\pi z_1/n)|(\gamma_z G(z)^2 + \gamma_z^{-1} G(T_2^{-1}z)^2) + |\cos(\pi z_1/n)|(\gamma_z G(z)^2 + \gamma_z^{-1} G(T_1^{-1}z)^2) \leq 3.625 \sum_z G(z)^2$$

To better keep track of the γ_z terms, given a partial order on $\mathbb{Z}_n^2$, one can introduce the following function $\gamma(x, y)$, where $x, y \in \mathbb{Z}_n^2$:

$$\gamma(x,y) = \begin{cases} \epsilon & \text{if } x > y \\ \epsilon^{-1} & \text{if } x < y \\ 1 & \text{otherwise} \end{cases} \tag{2.6}$$

where ϵ corresponds to the γ in Young's inequality. Now, Theorem 14 can be re-written as

Theorem 15 ([HLW06]). *For any $G : \mathbb{Z}_n^2 \to \mathbb{R}$ such that $G(0,0) = 0$,*

$$\sum_z |\cos(\pi z_1/n)| \left(\gamma(z, T_2^{-1}z)G(z)^2 + \gamma(T_2^{-1}z, z)G(T_2^{-1}z)^2\right) + |\cos(\pi z_2/n)| \left(\gamma(z, T_1^{-1}z)G(z)^2 + \gamma(T_1^{-1}z, z)\right) G(T_1^{-1}z)^2 \leq 3.625 \sum_z G(z)^2.$$

Since T_1^{-1} and T_2^{-1} simply permute the elements in $\mathbb{Z}/n\mathbb{Z} \times \mathbb{Z}/n\mathbb{Z}$, we can replace $T_2^{-1}z$ and $T_1^{-1}z$ by z in the second and fourth terms in the sum respectively (simultaneously replacing z with $T_2 z$ and $T_1 z$ respectively in the second and fourth terms in the sum). We then obtain the following form of the inequality:

Theorem 16 ([HLW06]). *For any nonzero $G : \mathbb{Z}_n^2 \to \mathbb{R}$ such that $G(0,0) = 0$,*

$$\sum_z G(z)^2 (|\cos(\pi z_1/n)| \left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) + |\cos(\pi z_2/n)| \left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right)) \leq 3.625 \sum_z G(z)^2.$$

Cancelling the non-zero $\sum_z G(z)^2$ terms gives

$$(|\cos(\pi z_1/n)| \left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) + |\cos(\pi z_2/n)| \left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right)) \leq 3.625. \quad (2.7)$$

Next, define a partial order on $\mathbb{Z}_n^2$. Since a partial order on the elements of $\mathbb{Z}_n^2$ follows naturally from a partial order on $\mathbb{Z}_n$, first define a partial order on $\mathbb{Z}_n$. Also, due to the symmetry of the function $|\cos(\pi z_1/n)| + |\cos(\pi z_2/n)|$ about the origin, choose the representatives in $\mathbb{Z}_n$ from the interval $[-n/2, n/2)$. Attempting to define a partial order in the natural way (i.e. $x \geq y$ if $x - y \geq 0$) fails. In particular, if $8|n$, then $\frac{n}{4} \geq -\frac{n}{8}$ and $-\frac{n}{8} \geq -\frac{n}{4}$; however, $\frac{n}{4} \not\geq -\frac{n}{4}$ since $\frac{n}{4} + \frac{n}{4} \equiv -\frac{n}{2} < 0$, which violates transitivity. To avoid this problem, define a partial order on the given interval as follows: $x < y$ if and only if $|x| < |y|$ and $x = y$ if and only if $x = y$. (So, $x \leq y$ if and only if $|x| < |y|$ or $x = y$.) This definition satisfies all of the properties of a partial order.

Now, define a partial order on $\mathbb{Z}_n^2$ by letting $(z_1, z_2) > (z_1', z_2')$ if and only if $|z_1| \geq |z_1'|$ and $|z_2| \geq |z_2'|$ with the additional restriction that *one* of the inequalities is strict. This partial order will allow us to control the coefficients from Young's inequality. Recall that we are attempting to compare z with $T_1 z$, $T_2 z$, $T_1^{-1} z$, and $T_2^{-1} z$. But T_1 and T_1^{-1} fix z_2. Similarly, T_2 and T_2^{-1} fix z_1. So, we need only to make the following comparisons (respectively):

- z vs. $T_1 z$: z_1 vs. $z_1 + 2z_2$

- z vs. $T_1^{-1}z$: z_1 vs. $z_1 - 2z_2$
- z vs. $T_2 z$: z_2 vs. $z_2 + 2z_1$
- z vs. $T_2^{-1}z$: z_2 vs. $z_2 - 2z_1$,

where $z_1, z_2 \in [-n/2, n/2)$. Without loss of generality, suppose $|z_1| > |z_2|$. By symmetry, suppose $z_1, z_2 \geq 0$. (If $z_1 \leq 0$, replacing z_1 by $-z_1$ still yields the same four comparisons above.) At this step, impose the additional restriction that $z_1 + z_2 < n/2$. Then,

- $|z_1| < |z_1 + 2z_2|$
  - if $z_1 + 2z_2 < n/2$.
  - if $z_1 + 2z_2 > n/2$. Note that $z_1 + z_2 < n/2$ implies that $z_1 + 2z_2 - n < -z_1$. So, $|z_1 + 2z_2 - n| > |z_1|$, and consequently $|z_1| < |z_1 + 2z_2|$.
- $|z_1| > |z_1 - 2z_2|$ since $z_1 > z_2$ implies that $-z_1 < z_1 - 2z_2$.
- $|z_2| < |z_2 + 2z_1|$
  - if $z_2 + 2z_1 < n/2$.
  - if $z_1 + 2z_2 > n/2$. Note that $z_1 + z_2 < n/2$ implies that $z_2 + 2z_1 - n < -z_1$. So, $|z_2 + 2z_1 - n| > |z_2|$, and consequently $|z_2| < |z_2 + 2z_1|$.
- $|z_2| < |z_2 - 2z_1|$
  - if $z_2 - 2z_1 > -n/2$. Note that $-z_2 > -z_1$ implies that $-z_2 > z_2 - 2z_1$. So, $|z_2| < |z_2 - 2z_1|$.
  - if $z_2 - 2z_1 < -n/2$. Note that $z_1 < n/2$ implies that $z_2 < z_2 - 2z_1 + n$. So, $|z_2| < |z_2 - 2z_1|$.

In three cases out of the four, $z < Az$ when $A \in \left\{T_1, T_2, T_2^{-1}\right\}$, and $z > T_1^{-1}z$. Note that the reason for the choice of the matrices T_1 and T_2 is now clear. They were chosen so that the above properties would hold whenever $z_1 + z_2 < n/2$. We now address the case where $|z_1| = |z_2|$. In this case, if $z_1 + z_2 < n/2$ then either $z_1 = z_2$ or $z_1 = -z_2$. If $z_1 = z_2$

- $|z_1| = |z_1 - 2z_2| = |z_2 - 2z_1|$. So, $(z, T_1^{-1}z)$ and $(z, T_2^{-1}z)$ are either equal or incomparable under our partial order.
- $|z_1| < |z_1 + 2z_2|$ and $|z_2| < |z_2 + 2z_1|$. To see this, note that if $z_1 + 2z_2 < n/2$, the inequality is clear. If not, $-z_1 > z_1 + 2z_2 - n$ since $z_1 + z_2 < n/2$. So, $|z_1| < |z_1 + 2z_2|$. Clearly then, $|z_2| < |z_2 + 2z_1|$ as well. So, $z < T_1 z$ and $z < T_2 z$.

By symmetry, if $z_1 = -z_2$

- $|z_1| = |z_1 + 2z_2| = |z_2 + 2z_1|$. So, $(z, T_1 z)$ and $(z, T_2 z)$ are either equal or incomparable.
- $|z_1| < |z_1 - 2z_2|$ and $|z_2| < |z_2 - 2z_1|$. To see this, note that if $z_1 + 2z_2 < n/2$, the inequality is clear. If not, $-z_1 > z_1 + 2z_2 - n$ since $z_1 + z_2 < n/2$. So, $|z_1| < |z_1 + 2z_2|$. Clearly then, $|z_2| < |z_2 - 2z_1|$ as well. So, $z < T_1^{-1} z$ and $z < T_2^{-1} z$.

So, assuming $z_1 + z_2 < n/2$, equation (2.6) gives the following:

- When $|z_1| < |z_2|$
  - $\gamma(z, T_1 z) = \gamma(z, T_2 z) = \gamma(z, T_2^{-1} z) = \epsilon^{-1}$.
  - $\gamma(z, T_1^{-1} z) = \epsilon$.
- When $|z_1| = |z_2|$
  - If $z_1 = z_2$
    - $\gamma(z, T_1 z) = \gamma(z, T_2 z) = \epsilon^{-1}$

* $\gamma(z, T_1^{-1}z) = \gamma(z, T_2^{-1}z) = 1$

– If $z_1 = -z_2$

* $\gamma(z, T_1^{-1}z) = \gamma(z, T_2^{-1}z) = \epsilon^{-1}$

* $\gamma(z, T_1 z) = \gamma(z, T_2 z) = 1$.

So, from (2.7) to prove the theorem, it must be shown that

$$(|\cos(\pi z_1/n)|\left(\epsilon^{-1} + \epsilon^{-1}\right) + |\cos(\pi z_2/n)|\left(\epsilon + \epsilon^{-1}\right)) \leq 3.625$$

when $|z_1| < |z_2|$. When $|z_1| = |z_2|$, it must be shown that

$$(|\cos(\pi z_1/n)|\left(1 + \epsilon^{-1}\right) + |\cos(\pi z_2/n)|\left(1 + \epsilon^{-1}\right)) \leq 3.625.$$

From these inequalities, it is clearly advantageous to let $\epsilon > 1$ in definition (2.6). In particular, bounding the cosine terms by 1, choose ϵ so that $3\epsilon^{-1} + \epsilon < 3.625$ and $2 + 2\epsilon^{-1} < 3.625$. Note that if $\epsilon > 1$, then $3/\epsilon + \epsilon > 2 + 2/\epsilon$. So when $z_1 + z_2 < n/2$, we only need to ensure that $3\epsilon^{-1} + \epsilon < 3.625$.

Next, consider the case when $z_1 + z_2 \geq n/2$. Letting $f(x, y) = \left|\cos\left(\frac{\pi x}{n}\right)\right| + \left|\cos\left(\frac{\pi y}{n}\right)\right|$, note that when $z_1 + z_2 \geq n/2$, $f(z_1, z_2)$ is decreasing and is maximized along the line $z_1 = z_2$ when $z_1 + z_2 = n/2$. Consequently, $f(z_1, z_2)$ is maximized at $z_1 = z_2 = \frac{\pi}{4}$. (By symmetry, we only consider the first quadrant where $z_1, z_2 \geq 0$.) Bounding each γ term by ϵ, when $z_1 + z_2 \geq n/2$ gives:

$$(|\cos(\pi z_1/n)|\left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) + |\cos(\pi z_2/n)|\left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right)) \leq \sqrt{2}(2\epsilon).$$

Combining the inequalities for $z_1 + z_2 < n/2$ and $z_1 + z_2 \geq n/2$, we must ensure that for a given choice of ϵ, it is true that both $2\sqrt{2}\epsilon < 3.625$ and $3/\epsilon + \epsilon < 3.625$. The intersection

of $2\sqrt{2}\epsilon$ and $3/\epsilon + \epsilon$ occurs at $\sqrt{\frac{3}{2\sqrt{2}-1}} \approx 3.623 < 3.625$. So, setting $\epsilon = \sqrt{\frac{3}{2\sqrt{2}-1}}$ gives

$$(|\cos(\pi z_1/n)| \left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) + |\cos(\pi z_2/n)| \left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right)) < 3.625.$$

Consequently, the graph construction described in Section 2.1.5 gives an infinite family of expander graphs.

2.1.7 Improving the Discrete Margulis Construction

We next show that by carefully adding more edges, we can decrease the normalized spectral gap below $\frac{8-5\sqrt{2}}{8}$. We will introduce the following additional edges (several of which will be multi-edges) from vertex z along with their inverses:

$$T_1 z, T_1 z + \begin{pmatrix} 3 \\ 1 \end{pmatrix}, T_2 z, T_2 z + \begin{pmatrix} 1 \\ 3 \end{pmatrix}.$$

Note that this construction produces a 16-regular graph.

Theorem 17. *For the graph G given by the construction described above, $\lambda(G) < 13.941 < 16$, and $1 - \lambda(G_n)/d > 1 - 13.941/16$.*

As noted in the previous section, it is possible to bound the second-largest eigenvalue of the adjacency matrix of the Margulis 8-regular graph by $5\sqrt{2}$. Consequently, letting λ denote the second-largest eigenvalue of the adjacency matrix of the 8-regular graph, $1-\lambda/d > 1 - \frac{5\sqrt{2}}{8} \approx 0.116$. For the 16-regular graph described above, $1 - 13.941/16 \approx 0.1287$, and the Cheeger inequality is given by $\Phi_E(G) \geq \frac{1}{2}(1 - \lambda(G)/d)$. Consequently, by carefully introducing additional edges, we have improved the lower bound on the edge expansion. We now prove Theorem 17.

Proof. Following the reasoning in Section 2.1.6, to prove that our new graph is an expander graph, we must show that for all functions f with $\sum_z f(z) = 0$,

$$\sum_z f(z)[f(T_1 z) + f(T_1 z + e_1) + f(T_2 z) + f(T_2 z + e_2)$$
$$+ f(T_1 z) + f(T_1 z + \binom{3}{1}) + f(T_2 z) + f(T_2 z + \binom{1}{3})] < 8 \sum_z f(z)^2.$$

Equivalently, using the Shift Property and the fact that $|1 + \omega^a| = 2|\cos(\frac{\pi a}{n})|$, we must show that for all non-negative functions $G : \mathbb{Z}_n^2 \to \mathbb{R}$

$$\sum_z 2G(z)(G(T_2^{-1}z)|\cos(\pi z_1/n)|) + G(T_1^{-1}z)|\cos(\pi z_2/n)|$$
$$+ G(T_2^{-1}z)|\cos(\pi(z_1+z_2)/n)| + G(T_1^{-1}z)|\cos(\pi(z_1+z_2)/n)|) < 8\sum_z G(z)^2,$$

where $G = |F|$, and F denotes the Fourier transform of f. Next, applying Young's inequality as before, the above inequality reduces to showing that for any nonzero $G : \mathbb{Z}_n^2 \to \mathbb{R}$ such that $G(0,0) = 0$,

$$\sum_z G(z)^2(|\cos(\pi z_1/n)|\left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) +$$
$$|\cos(\pi(z_1+z_2)/n)|\left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) +$$
$$|\cos(\pi(z_1+z_2)/n)|\left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right) +$$
$$|\cos(\pi z_2/n)|\left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right)) < 8\sum_z G(z)^2. \tag{2.8}$$

Following the same process as in Section 2.1.6 when $z_1 + z_2 < n/2$, and bounding each cosine term by 1, this inequality reduces to showing that

$$6\epsilon^{-1} + 2\epsilon < 8 \text{ and } 4\epsilon^{-1} + 4 < 8.$$

Dividing through by 2, we see that these are exactly the same bounds as before. So, the

improvement in our approximation will come from the analysis when $z_1 + z_2 \geq n/2$. In this case, after bounding each γ term by ϵ as before, showing (2.8) reduces to showing that $2(|\cos(x)| + |\cos(y)| + 2|\cos(x+y)|)\epsilon < 8$, where we have let $x = \frac{\pi z_2}{n}$ and $y = \frac{\pi z_1}{n}$.

The global maximum of $|\cos(x)| + |\cos(y)| + 2|\cos(x+y)|$ for $x + y \geq \pi/2$ is $9/4$ which occurs when $x = y = \arccos(1/4)$. So,

$$2(|\cos(x)| + |\cos(y)| + 2|\cos(x+y)|)\epsilon < \frac{9\epsilon}{2}.$$

As before, for $\epsilon > 1$, it is the case that $6/\epsilon + 2\epsilon > 4/\epsilon + 4$. So, we need only require that $6/\epsilon + 2\epsilon < \frac{9\epsilon}{2}$, which is true when $\epsilon \leq 1.549$. By setting $\epsilon = 1.549$, we see that

$$\begin{aligned} |\cos(\pi z_1/n)| \left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) + |\cos(\pi(z_1+z_2)/n)| \left(\gamma(z, T_2^{-1}z) + \gamma(z, T_2 z)\right) + \\ |\cos(\pi(z_1+z_2)/n)| \left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right) + |\cos(\pi z_2/n)| \left(\gamma(z, T_1^{-1}z) + \gamma(z, T_1 z)\right) \\ < 4.5(1.549) = 6.9705 < 8. \end{aligned}$$

So, $\lambda(G_n) < 13.941$, and $1 - \lambda(G_n)/d > 1 - 13.941/16 \approx 0.1287$. □

2.2 Ramanujan Graphs

The discovery of the relationship between expansion and the second-largest eigenvalue of the adjacency matrix naturally raised the question of how much expansion the second-largest eigenvalue could guarantee. As noted in the previous chapter, the Alon-Boppana bound [Alo86, LPS86] answered this question asymptotically. Margulis [Mar84, Mar88] and Lubotzky et al. [LPS86, LPS88] independently produced the first explicit constructions of infinite families of graphs with fixed degree $p + 1$ (where $p > 2$ is a prime) which achieved the Alon-Boppana bound. Mestre [Mes86] also provided a construction of an infinite family of Ramanujan graphs with fixed degree $p + 1$. Since the proof in [LPS88] utilized the

Ramanujan conjecture, the resulting graphs were called Ramanujan graphs. While cubic Ramanujan graphs ($p = 2$) were mentioned in [LPS88], there had been no explicit construction of cubic Ramanujan graphs given up to that point. Chiu [Chi92] filled this gap by explicitly constructing an infinite family of Ramanujan graphs with degree 3. As in [Mar82], all of these constructions were built from Cayley graphs and motivated several other constructions of Ramanujan graphs. In particular, Pizer [Piz90] exhibited a different construction which also produced infinite families of fixed-degree Ramanujan graphs with degree $p + 1$. His construction also yielded "almost" Ramanujan graphs for other degrees as well.

All of the constructions of Ramanujan graphs mentioned so far had degree $p+1$ with p a prime. However, a few years after [LPS88], Morgenstern [Mor94] constructed infinite families of Ramanujan graphs with fixed degree $p^\ell + 1$ with p a prime and $\ell \geq 1$. A few years after Morganstern's paper, Jordan and Livne [JL97] generalized the construction presented in [LPS88] and showed that the construction in [LPS88] was a special case of their construction.

Other than these papers, very little progress was made on explicitly constructing infinite families of fixed-degree Ramanujan graphs. However, several other constructions of (infinitely many) finite families of Ramanujan graphs with fixed degree also emerged. In particular, using abelian groups, Chung [Chu89] and Li [Li92] built infinitely many finite families of Ramanujan graphs with fixed degree. These Ramanujan graphs allowed for a greater variety of degrees than $p + 1$ or even $p^\ell + 1$ with p a prime. Unfortunately, Klawe [Kla84] showed that any approach attempting to use abelian groups to construct infinite families of fixed-degree expander graphs, let alone Ramanujan graphs, was doomed to fail. Lubotzky and Weiss [LW93] pushed this result slightly farther to show that attempting to use even "almost" abelian groups was also doomed to failure. Terras et al. [ACP$^+$92, CPT$^+$93] also constructed infinitely many finite families of fixed-degree Ramanujan graphs, and the result implying that these graphs actually were Ramanujan was proved by Katz [Kat93]. Using simpler methods, de Reyna [AdR97] constructed p-regular Ramanujan graphs hav-

ing p^2 vertices for any prime p, and later, Chee and Ling [CL02] built on Alon's geometric construction in [Alo85] to produce finite families of highly symmetric Ramanujan graphs. Somewhat recently, Li and Meemark [LM05] gave another alternate construction of infinitely many finite families of fixed-degree Ramanujan graphs. While interesting, none of these constructions contributed to the problem of finding infinite families of fixed-degree Ramanujan graphs for degrees other than $p+1$ or $p^\ell+1$.

Lubotzky [Lub94] conjectured that there are infinite families of fixed-degree Ramanujan graphs for every degree $d \geq 3$. However, despite many years of intense research activity, the only explicit constructions of infinite families of Ramanujan graphs with fixed degree have had degree $p^\ell + 1$ for prime p and $\ell \geq 1$. Very recently, Marcus et al. [MSS13] (see [MSS15] for the full journal article) expanded on the techniques used to construct "almost" Ramanujan graphs in Bilu and Linial [BL04] (see [BL06] for the full journal article) to prove Lubotzky's conjecture in the bipartite case. In particular, they showed that there exist infinite families of fixed-degree bipartite Ramanujan graphs for every degree $d \geq 3$. Following a generalization of the definition of Ramanujan graphs for (c,d)-biregular bipartite graphs in [FL96], they also showed that there exist bipartite (c,d)-biregular Ramanujan graphs for every $c, d \geq 3$. In a follow-up work [MSS15], the same authors showed that there exist infinite families of fixed-degree bipartite Ramanujan graphs for every degree $d \geq 3$ and also every possible number of vertices. This breakthrough result has stimulated intense research effort toward finding explicit constructions of Ramanujan graphs. In [CV15], Chandrasekaran and Velingker took steps toward explicitly constructing d-regular Ramanujan graphs using k-lifts for $k = 2, 3$; however, their result does not yield infinite families of Ramanujan graphs. Finding alternate constructions of infinite families of fixed-degree bipartite Ramanujan graphs remains a very active area of research. We now briefly describe the classic construction in [LPS88].

Theorem 18 ([LPS88])**.** *For every pair of primes p, q congruent to 1 mod 4 such that p*

is a quadratic residue mod q, there is a $(p+1)$-regular Cayley graph of $PSL(2, \mathbb{Z}/q\mathbb{Z})$ with $q(q^2-1)/2$ vertices such that the second-largest eigenvalue of the graph is at most $2\sqrt{p}$. Moreover, this graph can be constructed in polynomial time. By varying q, we can construct an infinite family of Ramanujan graphs.

We now briefly describe the construction of Theorem 18. First, take $u \in \mathbb{Z}$ such that $u^2 \equiv -1 \bmod q$. By a well-known theorem of Jacobi, there are $8(p+1)$ solutions (a, b, c, d) such that $a^2 + b^2 + c^2 + d^2 = p$. There are $p+1$ solutions with $a > 0$ and b, c, d even. Each such solution corresponds to the matrix

$$\begin{pmatrix} a+ub & c+ud \\ -c+ud & a-ub \end{pmatrix} \in PGL_2(\mathbb{Z}/q\mathbb{Z}).$$

Let S be the set of these matrices, and let G be the group of all matrices in $PGL_2(\mathbb{Z}/q\mathbb{Z})$ (where we identify two matrices which are constant multiples of each other). The Cayley graph $C(G, S)$ is a $(p+1)$-regular Ramanujan graph. By varying p and q, we obtain an infinite family of Ramanujan graphs. The proof that this construction yields an infinite family of Ramanujan graphs is quite technical, and we refer the interested reader to [DSV03].

2.3 Zig-zag Product and Other Constructions

In a different direction, around the same time that the relationships between eigenvalues and expansion were being explored, Karp et al. [KPS85] showed that expander graphs held great promise for derandomization. Sipser [Sip86] took another step in this direction and was the first to use highly expanding multigraphs in derandomization. While he conjectured that the expander graphs necessary for derandomization existed, no explicit construction up to that time (including the Ramanujan graphs in [LPS86]) satisfied Sipser's requirements. It was not long, however, before Ajtai et al. [AKS87] gave an explicit construction satisfying

these requirements, and a few years later, Cohen et al. [CW89] improved this construction. Around the same time, Ajtai [Ajt87] introduced the first purely combinatorial construction of expander graphs with his construction of 3-regular expander graphs (see [Ajt94] for the full journal article). A few years later, several different combinatorial constructions with derandomization as the primary motivation appeared in [Zuc90, Zuc91] (see [Zuc96] for the full journal article). Wigderson and Zuckerman improved these constructions in [WZ93, WZ99] and explicitly pointed out that these constructions surpassed the eigenvalue bound given in [Kah92] (see [Kah95] for the full journal article). The next major combinatorial construction came with the introduction of the brand new zig-zag graph product in [RVW00, RVW02]. While this construction did not obtain as good of an eigenvalue bound as the Ramanujan graphs of [LPS88], it provided a new, entirely combinatorial method to construct infinite families of constant-degree expander graphs. Note that another proof of the properties of the zig-zag product was given by Reingold et al. in [RTV05], and in [RTV06] the same authors allude to an even simpler proof. Alon et al. [ASS08] presented another construction which directly analyzes the replacement product from which the zig-zag product was built.

Soon after [RVW00], Alon et al. [ALW01] derived a correspondence between the zig-zag product and the semi-direct product from group theory. Building on this correspondence, several new constructions merging group theoretic techniques with the zig-zag product appeared [MW02, MW04, RSW04, RSW06]. In particular, Meshulam and Wigderson [MW02, MW04] combined the zig-zag product and semi-direct product to construct infinite families of *almost* constant-degree families of expanders. Soon afterwards, Rozenman et al. [RSW04] showed that given an appropriate base graph, the semi-direct product version of the zig-zag product in [ALW01] could be used to yield infinite families of constant-degree expanders. The construction of Kassabov [Kas05b, Kas07] which produced bounded-degree Cayley expander graphs from A_d provided a base graph allowing Rozenman et al. to complete their construction [RSW06]. Kassabov [Kas05a] also constructed Cayley expander

graphs from $SL_n(p^m)$ where p is a prime, $n \geq 3$, and $m \geq 1$. In an unpublished result, Lubotzky obtained a similar result in the case when $n = 2$. For a summary of these results, see [KLN06]. Building on the work in [AC02], Capalbo et al. [CRVW02] refined the ideas from their earlier zig-zag product to construct lossless expander graphs with excellent vertex expansion. This result was a major breakthrough far surpassing Kahale's eigenvalue bound in [Kah95].

In the highly unbalanced bipartite graph setting in which $|R| << |L|$, it is well known that random constructions can with high probability produce highly unbalanced expander graphs with good parameters.

Proposition 19. *For any $\epsilon > 0$, a random c-left regular bipartite graph $G = (L \cup R, E)$ (where $|L| = n$ and $|R| = m$) with $c = O\left(\frac{\log(n/K)}{\epsilon}\right)$ and $m = O\left(\frac{cK}{\epsilon}\right)$ satisfies the following property with high probability: For any $S \subseteq L$ with $|S| \leq K$, $|N(S)| \geq (1-\epsilon)c|S|$.*

For a simple proof of this proposition, see [Ber09]. In contrast, Ta-Shma, Umans, and Zuckerman [TSUZ01] (see [TSUZ07] for the full journal article) *explicitly* constructed highly unbalanced expander graphs whose properties are given in the following theorem:

Theorem 20 ([TSUZ07])**.** *For any $\beta > 0$ and $\epsilon > 0$, one can explicitly construct an infinite family of c-left regular bipartite graphs $G = (L \cup R, E)$ with one of the following two pairs of parameters (where $|L| = n$ and $|R| = m$)*

1. *$c = \log^{\ell} n$ for some $\ell > 1$, and $m = 2^{(k/\epsilon)^{(1+\beta)}}$*
2. *$c = (\log n)^{O((\log\log n))}$ and $m = 2^{O(k/\epsilon)}$*

such that for any $S \subseteq L$ with $|S| < 2^k$, $|N(S)| \geq (1-\epsilon)c|S|$.

Later, by utilizing Parvaresh-Vardy codes, Guruswami and Umans [GUV09] improved the parameters in [TSUZ07].

Theorem 21 ([GUV09]). *For any $\beta > 0$ and $\epsilon > 0$, one can explicitly construct an infinite family of c-left regular bipartite graphs $G = (L \cup R, E)$ with the following parameters (where $|L| = n$ and $|R| = m$)*

1. $c = O((\log n)(\log K)/\epsilon)^{(1+1/\beta)}$
2. $m \leq c^2 \cdot K^{1+\beta}$

such that for any $S \subseteq L$ with $|S| < K$, $|N(S)| \geq (1-\epsilon)c|S|$.

Comparing this theorem with Proposition 19, it is clear that there is still some room for improvement in the highly unbalanced setting. We pause to mention that quantum expander graphs were also introduced quite recently in [BASTS08].

We now give an overview of the original zig-zag construction in [RVW00, RVW02] as well as the modified zig-zag construction in [CRVW02] which yields constant-degree lossless bipartite expander graphs. For the expander codes of Chapter 4, we are primarily interested in the "slightly unbalanced" setting in which the number of vertices on the right-hand side of the graph is a constant fraction of the number of vertices on the left-hand side of the graph.

2.3.1 Zig-zag Product Construction

With their original zig-zag construction, Reingold et al. [RVW00, RVW02] constructed infinite families of d-regular expander graphs with $\lambda(G)/d < 1$. In particular, Reingold et al. [RVW02] proved the following for d-regular graphs:

Theorem 22 ([RVW02]). *One can explicitly construct infinite families of expander graphs $\{G_i\}$ such that $\lambda(G_i) = O(1/d^{1/3})$.*

(Recall that for an infinite family $\{G_i\}$ of Ramanujan graphs, $\lambda(G_i) = O(1/d^{1/2})$.) By modifying the original zig-zag construction in [RVW00], Capalbo et al. [CRVW02] explicitly constructed lossless c-left regular bipartite graphs.

Theorem 23 ([CRVW02])**.** *For any $\epsilon > 0$ and $m \leq n$, one can explicitly construct a c-left regular $(c, \gamma, 1-\epsilon)$ bipartite vertex expander graph $G = (L \cup R, E)$ with $|L| = n$ and $|R| = m$, where $c = \Theta\left(\frac{\log(n/m)}{\epsilon}\right)$ and $\gamma = \Theta\left(\frac{\epsilon m}{cn}\right)$.*

For example, one can explicitly construct a (c, γ, α) expander graph with $\alpha = 1/2$, $m = n/8$, $c = 64$, $\gamma = 2^{-30}$. As we will see in Chapter 4, such a graph corresponds to a binary code with rate at least $7/8$. It is an interesting question whether or not the construction in [CRVW02] can be modified to construct (c, d)-regular bipartite graphs in addition to c-left regular bipartite graphs.

2.3.1.1 Replacement Product

We now explore the constructions in [RVW00] and [CRVW02] in more detail, beginning with the replacement product. Following the notation in [HLW06], we say that a regular graph G is an (n, d, α) graph if n is the number of vertices, d is the degree, and $\lambda(G)/d \leq \alpha$, where $\lambda(G)$ is the second-largest eigenvalue of $A(G)$. Before describing the zig-zag product, we must first describe the replacement product $G \textcircled{r} H$ where G is a (n, m, α) graph, and H is a (m, d, β) graph. Notice that the degree of G must match the number of vertices in H. The reason for this requirement will become clear in the description of the construction which we now provide.

- To obtain the vertices of $G \textcircled{r} H$, replace each vertex v in G with a copy of the graph H (which gives a graph with nm vertices). Denote this copy of H by H_v.
- To obtain the edges of $G \textcircled{r} H$, follow the procedure below:
  - All edges in each copy of H remain edges in $G \textcircled{r} H$.
  - Assign each edge adjacent to a given vertex $v \in V(G)$ to a vertex in H_v. (It is now clear why the number of edges in G must match the number of vertices in H.)

– Label the vertices of each H_v.

– Let e_v^i denote the edge assigned to the ith vertex of H_v. Given $u, v \in V(G)$ and $i, j \in V(H)$, if e_v^i and e_u^j correspond to the same edge (denoted by $e_u^i = e_v^j$), then there is an edge between vertices (u, i) and (v, j) in $V(G \textcircled{r} H)$.

Notice that $G \textcircled{r} H$ is an $(nm, d+1)$ graph since each vertex in a copy of H is associated with one additional edge. The replacement product is best illustrated by an example. Let $G = \mathbb{Z}_3^2$ with the edges as given in the graph below:

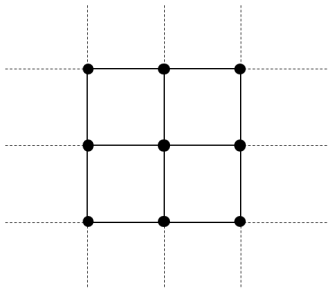


Figure 2.1: Replacement Product: $\mathbb{Z}_3^2$

Let $H = C_4$ be the graph given below:

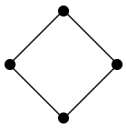


Figure 2.2: Replacement Product: C_4

We first replace the vertices in $V(G)$ with copies of H:

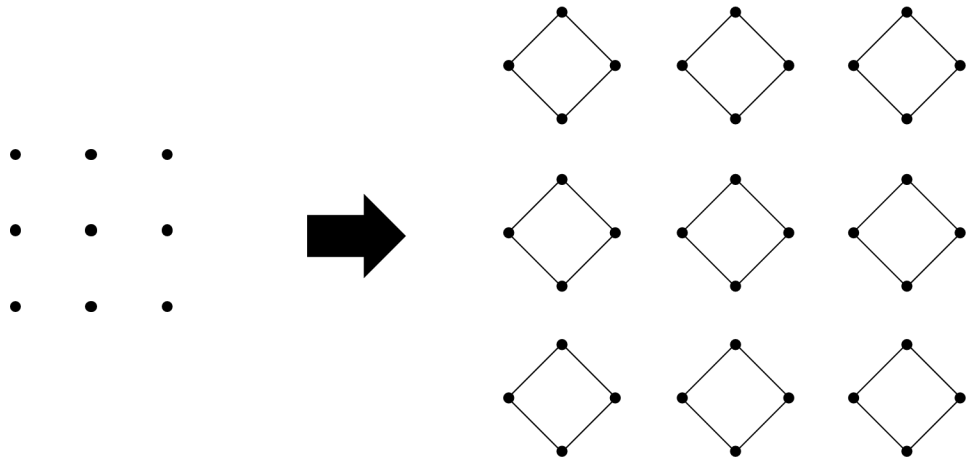


Figure 2.3: Replacement Product: Vertex Replacement

We now assign the edges adjacent to each vertex $v \in V(G)$ to the vertices in the copy of H which replaced vertex v.

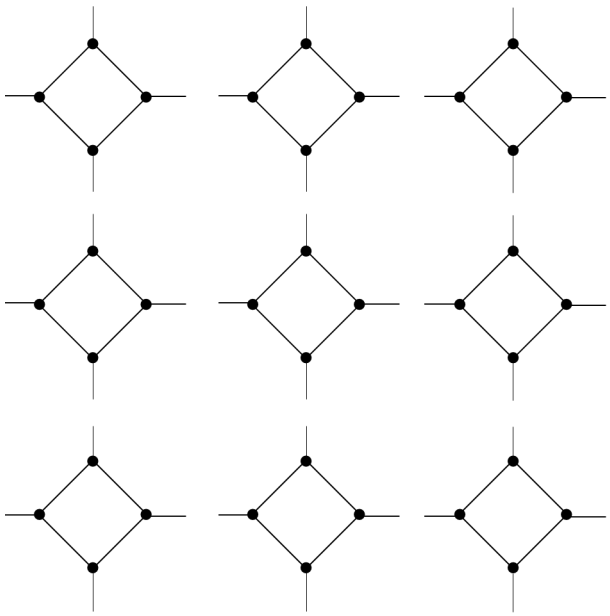


Figure 2.4: Replacement Product: Edge Assignment

We now order the vertices on each copy of H, which induces an ordering to the edges leaving H.

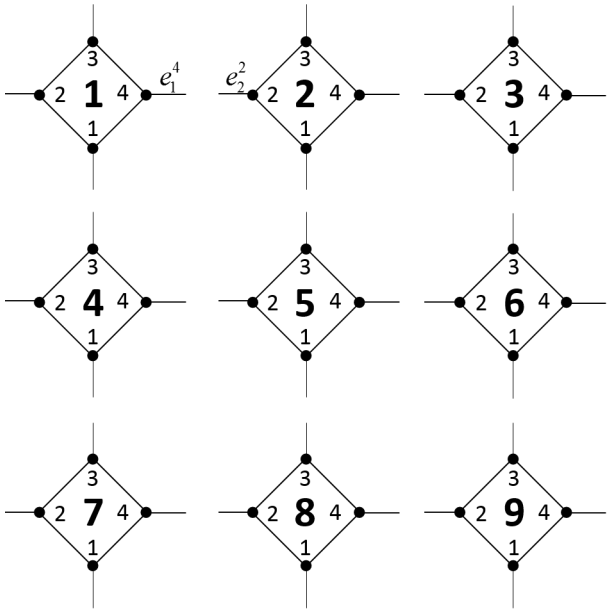


Figure 2.5: Replacement Product: Vertex Ordering

Next, when $e_u^i = e_v^j$ as is the case in the figure above for e_1^4 and e_2^2, we connect vertex (u, i) with vertex (v, j) where u and v specify the copy of H (or cloud of vertices) under consideration, and i and j specify the vertex in that particular cloud/copy of H.

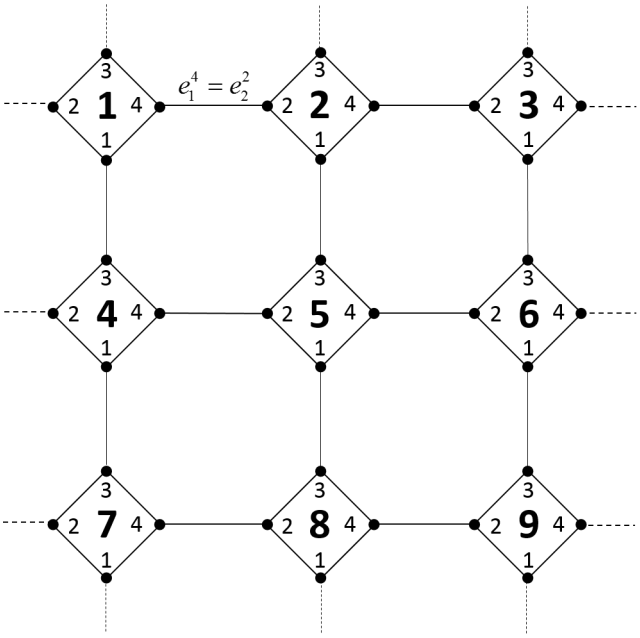


Figure 2.6: Replacement Product: Final Graph

2.3.1.2 Zig-zag Product

We now describe the construction of the zig-zag product $G ⓩ H$ of an (n, m, α)-graph G and an (m, d, β)-graph H.

- Begin with G ⓡ H.
- Include an edge between (u, i) and (v, j) if there is an edge between (u, i) and (u, k) for some $k \in H$ and an edge between (v, j) and (v, ℓ) for some $\ell \in H$ such that $e_u^k = e_v^\ell$.
- Remove the edges from G ⓡ H.

We again illustrate this procedure with an example. Consider the replacement graph given above. Suppose we begin with vertex $(5, 3)$. We first "zig" to the adjacent vertices $(5, 2)$ and $(5, 4)$ within copy 5 of graph H:

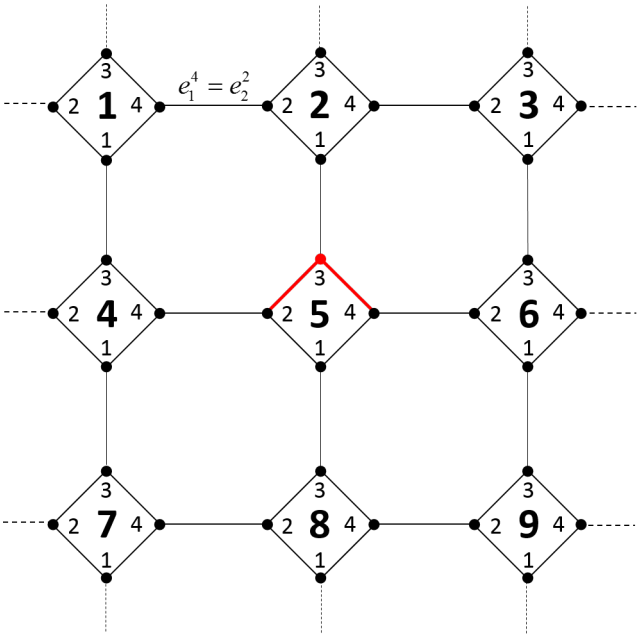


Figure 2.7: Zig-zag Product: "Zig" Step

We then follow the outgoing edges connected to vertices $(5, 2)$ and $(5, 4)$.

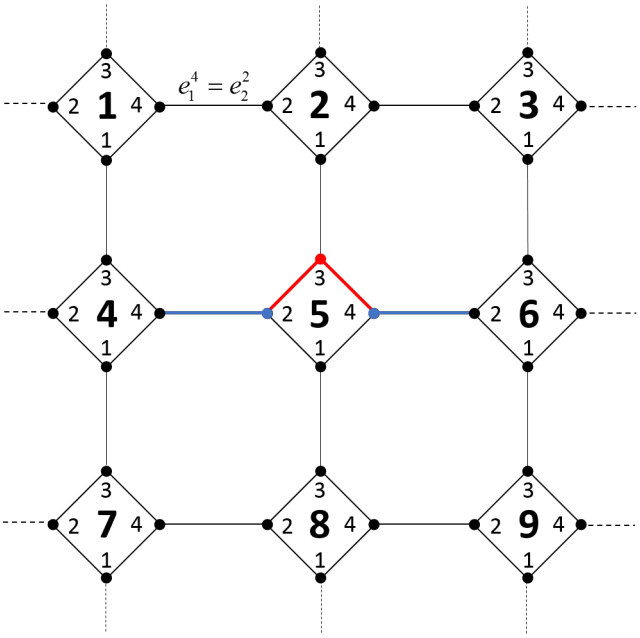


Figure 2.8: Zig-zag Product: Permutation Step

Finally, we "zag" to the adjacent vertices $(4, 3)$ and $(4, 1)$ within copy 4 of graph H and adjacent vertices $(6, 3)$ and $(6, 1)$ within copy 6 of graph H. In terms of the definition, note that there is an edge between $(4, 3)$ and $(4, 4)$. There is also an edge between $(5, 2)$ and $(5, 3)$. Since $e_4^4 = e_5^2$, there is an edge between $(4, 3)$ and $(5, 3)$.

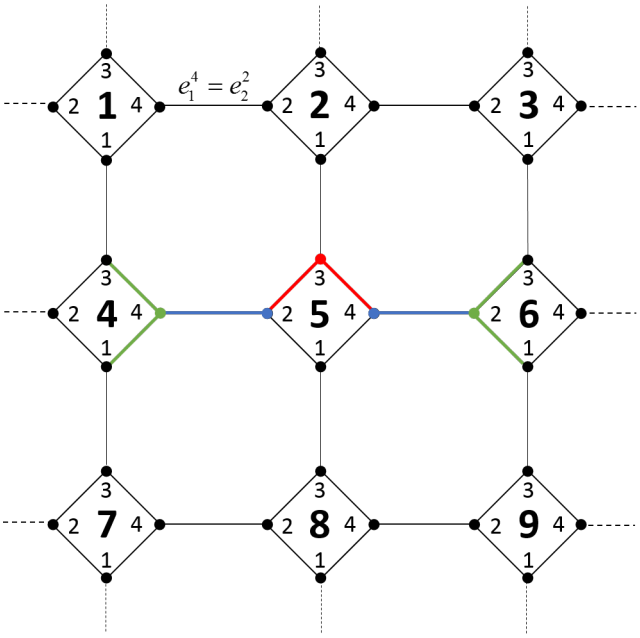


Figure 2.9: Zig-zag Product: "Zag" Step

We now remove the original edges and show the edges in the new zig-zag product which run horizontally.

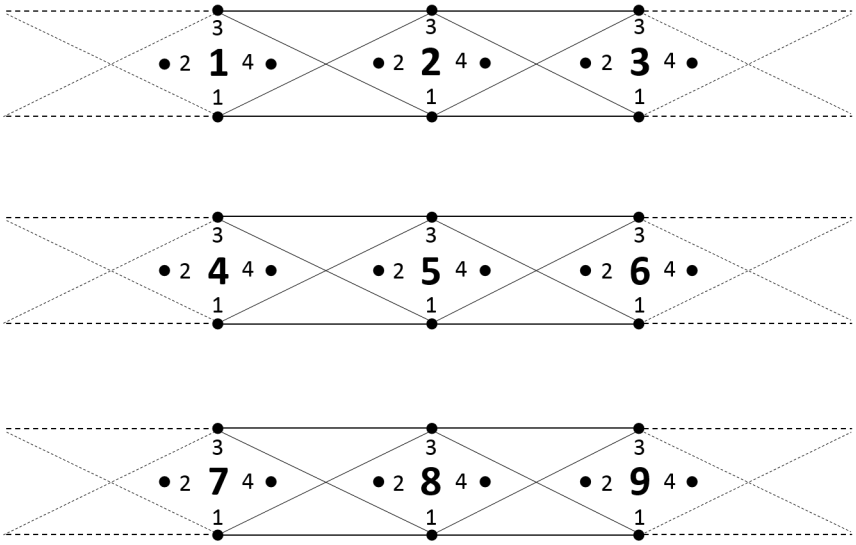


Figure 2.10: Zig-zag Product: Horizontal Edges

Next, we illustrate the edges in the zig-zag product which run vertically.

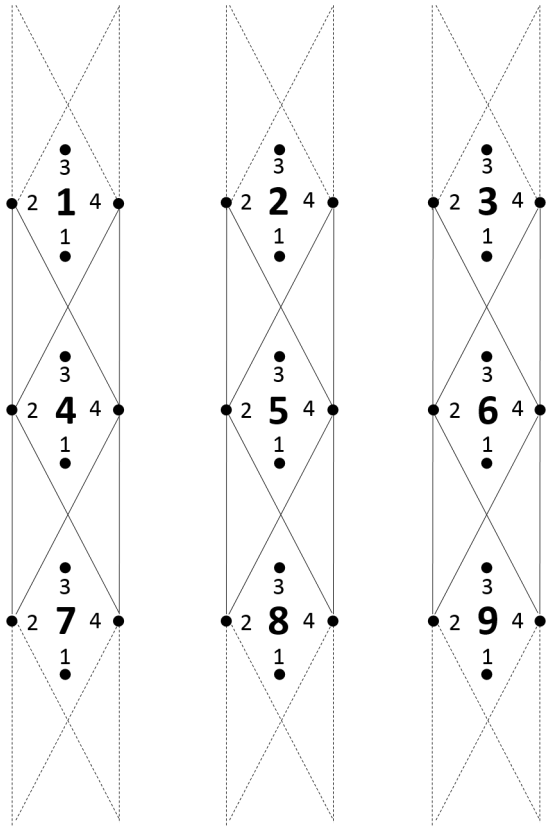


Figure 2.11: Zig-zag Product: Vertical Edges

Putting these figures together gives the full zig-zag product graph.

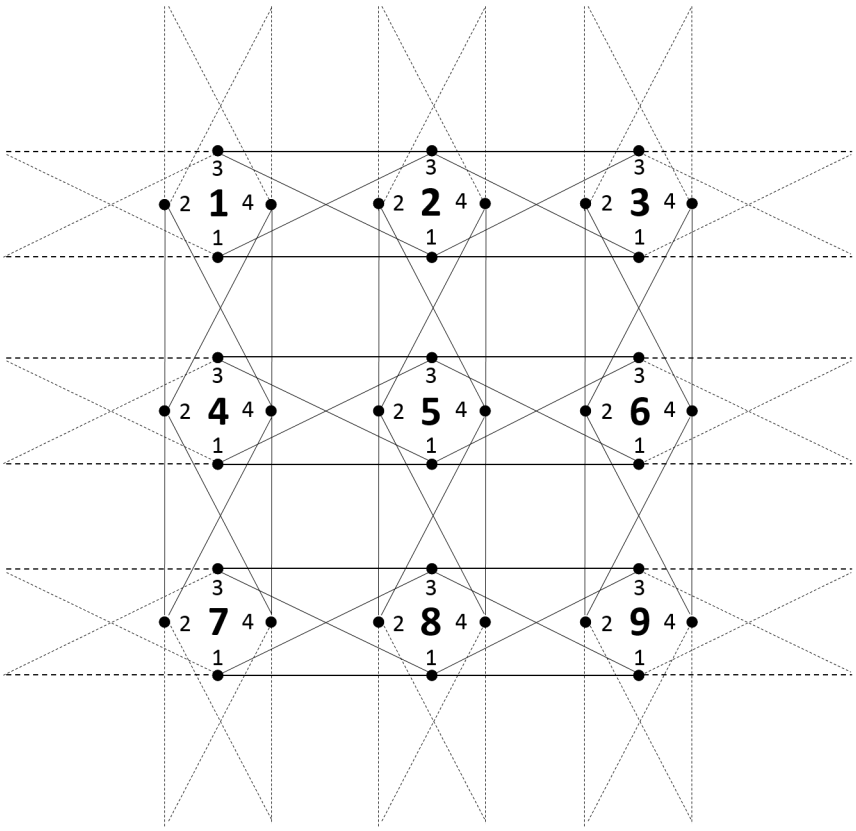


Figure 2.12: Zig-zag Product: Complete

Note that in this example, $G \textcircled{z} H$ is 2^2-regular. In general, the zig-zag product of an (n, m, α) graph with an (m, d, β) graph is d^2-regular.

2.3.2 Properties of the Zig-zag Product

Recall that given an $m \times n$ matrix

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$

and a $p \times q$ matrix B, the tensor product/Kronecker product of A and B is defined as

$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ a_{21}B & \cdots & a_{2n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}.$$

Define the $mn \times mn$ permutation matrix P as follows:

$$P_{(u,i),(v,j)} = \begin{cases} 1 & \text{if } e_u^i = e_v^j \\ 0 & \text{otherwise.} \end{cases}$$

Recall that $A(G)$ denotes the adjacency matrix of a graph G, and let I_n denote the $n \times n$ identity matrix. Letting $Z = G \circledz H$,

$$A(Z) = (I_n \otimes A(H))P(I_n \otimes A(H))$$

where $I_n \otimes A(H)$ represents the n copies of H in the replacement product. Define $M_H := I_n \otimes A(H)$. To see that the above equality is true, note that

$$\begin{aligned}(A(Z))_{(u,i),(v,j)} &= \sum_{(\omega,\ell)} \sum_{(w,k)} (M_H)_{(u,i),(\omega,\ell)} P_{(\omega,\ell),(w,k)} (M_H)_{(w,k),(v,j)} \\ &= \sum_{(u,\ell)} \sum_{(v,k)} (M_H)_{(u,i),(u,\ell)} P_{(u,\ell),(v,k)} (M_H)_{(v,k),(v,j)}.\end{aligned}$$

So, $(A(Z))_{(u,i),(v,j)} = 1$ if and only if there is an edge between (u,i) and (u,ℓ), $e_u^\ell = e_v^k$, and there is an edge between (v,k) and (v,j) for some ℓ, k. From the construction of the zig-zag product, this is precisely when there is an edge between (u,i) and (v,j) in $G \circledz H$.

Recall that a regular graph G is an (n,d,α) graph if n is the number of vertices, d is the degree, and $\lambda(G)/d \leq \alpha$, where $\lambda(G)$ is the second-largest eigenvalue of $A(G)$.

Theorem 24 ([RVW02]). *Let G be a (n, m, α) graph and H be a (m, d, β) graph. Then, $G \textcircled{z} H$ is an $(nm, d^2, \alpha + \beta + \beta^2)$ graph.*

We now describe a simple method given in [RVW02] of iterating the zig-zag product to obtain an infinite family of expander graphs. Given a graph $G = (V, E)$, let G^2 denote the graph defined on the same vertex set V as G with an edge $(u, v) \in G^2$ if and only if there is a path of length 2 between u and v in G. Now, starting with an arbitrary $(d^4, d, 1/5)$ graph H, let $G_1 = H^2$. Note that H^2 is a $(d^4, d^2, 1/25)$ graph since $A(G^2) = A(G)^2$, and the eigenvalues of $A(G)^2$ are the squares of the eigenvalues of $A(G)$. For all subsequent iterations $i > 1$, let

$$G_i = G_{i-1}^2 \textcircled{z} H.$$

From Theorem 24, G_2 is a $(d^8, d^2, (1/25)^2 + (1/5)^2 + 1/5)$ graph. So, since $(1/25)^2 + (1/5)^2 + 1/5 < 2/5$, G_2 is a $(d^8, d^2, 2/5)$ graph. Similarly, G_3 is a $(d^{12}, d^2, (2/5)^2 + (1/5)^2 + 1/5)$ graph. Since $(2/5)^2 + (1/5)^2 = 1/5$, G_3 is a $(d^{12}, d^2, 2/5)$ graph. By performing further iterations, it is now clear that for any $i \geq 1$, G_i is a $(d^{4i}, d^2, 2/5)$ expander graph. By iterating in this way, Reingold et al. [RVW02] obtained an infinite family of fixed-degree expander graphs.

This construction can be used to compute the neighborhood of a given vertex in a polynomial number of steps in the size of G_i; however, there is also a more intricate construction described in [RVW02] which requires only a polylogarithmic number of steps in the size of G_i to compute the neighborhood of a given vertex. In addition, by using a slightly more delicate argument, [RVW02] proved the following theorem:

Theorem 25 ([RVW02]). *Let G be a (n, m, α) graph and H be a (m, d, β) graph. Then, $G \textcircled{z} H$ is a $(nm, d^2, \frac{1}{2}(1 - \beta^2)\alpha + \frac{1}{2}\sqrt{(1 - \beta^2)^2\alpha^2 + 4\beta^2})$ graph.*

Note that $\frac{1}{2}(1 - \beta^2)\alpha + \frac{1}{2}\sqrt{(1 - \beta^2)^2\alpha^2 + 4\beta^2} \leq \alpha + \beta$, so the eigenvalue bound in Theorem 25 improves the eigenvalue bound in Theorem 24.

2.3.3 Improved Zig-Zag Product [CRVW02]

Recall that one characterization of expansion is that random walks on good expander graphs mix rapidly. More precisely, the probability distribution on the vertices converges exponentially quickly to the uniform distribution, and the rate of the convergence depends on how good of an expander graph we are given. In terms of entropy, we can view this mixing as an increase in entropy at each step of the random walk, and we can view the expansion as a measure of how quickly the entropy increases in the graph.

Following this intuition, the zig-zag product given above either increases the entropy on the uniform random "zig" step or on the uniform random "zag" step. Unfortunately, entropy is potentially lost at each step which results in an eigenvalue bound of $O(D^{1/3})$ which is sub-optimal to the $O(D^{1/2})$ bound obtained by Ramanujan graphs. To improve the zig-zag construction, [CRVW02] introduced entropy conductors designed to retain the lost entropy until the last step. Using these objects, they were able not only to improve the original zig-zag construction but to far surpass Kahale's bound [Kah95] on the best possible vertex expansion guarantee using spectral methods. We next give the formal (somewhat complicated) definition of the improved zig-zag product in [CRVW02]. In what follows, (n) denotes the set of all bit strings of length n; and $\langle E, C\rangle : (n) \times (d) \to (m) \times (b)$ denotes the concatenation of the two functions $E : (n) \times (d) \to (m)$ and $C : (n) \times (d) \to (b)$. Also, $x_1 \circ x_2$ denotes the concatenation of x_1 and x_2.

Definition 9 (Generalized Zig-zag Product)**.** Suppose we are given three functions

- $\langle E_1, C_1\rangle : (n_1) \times (d_1) \to (m_1) \times (b_1)$,
- $\langle E_2, C_2\rangle : (n_2) \times (d_2) \to (d_1) \times (b_2)$,
- $E_3 : (b_1 + b_2) \times (d_3) \to (m_3)$.

Given $x_1 \in (n_1)$, $x_2 \in (n_2)$, $r_2 \in (d_2)$, and $r_3 \in (d_3)$, let

- $\langle E_2, C_2 \rangle (x_2, r_2) = \langle r_1, z_1 \rangle$,
- $\langle E_1, C_1 \rangle (x_1, r_1) = \langle y_1, z_2 \rangle$,
- $E_3(z_1, z_2) = y_2$,

where $r_1 \in (m_1)$, $z_1 \in (b_1)$, $y_1 \in (d_1)$, $z_2 \in (b_2)$, and $y_2 \in (m_3)$. Then, define the zig-zag product $E : (n_1 + n_2) \times (d_2 + d_3) \to (m_1 + m_3)$ as follows:

$$E(x_1 \circ x_2, r_2 \circ r_3) = (y_1 \circ y_2)$$

where y_1 and y_2 are as defined above.

Before illustrating the generalized zig-zag product with an example, we first note that given a distribution X on (n) and the uniform distribution U_d on (d), a function $E : (n) \times (d) \to (m)$ induces a distribution on (m). Denote this induced distribution by $E(X, U_d)$. The min-entropy of a distribution X is defined as

$$H_\infty(X) := \min_{a \in \text{Supp}(X)} -\log(Pr[X = a]) = \log(1/(\max_{a \in \text{Supp}(X)} Pr[X = a]))$$

where $\text{Supp}(X)$ denotes the support set of the random variable X. Notice that requiring $H_\infty(X) \geq k$ is equivalent to requiring that $Pr[X = a] \leq 1/2^k$ for each $a \in Supp(X)$. So, by enforcing a min-entropy threshold, we are simultaneously "squeezing" the distribution down to a uniform distribution. In addition, we are forcing the size of the support set of X to be at least 2^k.

Definition 10 (k-source)**.** A distribution X is a k-source if $H_\infty(X) \geq k$.

Definition 11 (ϵ-close)**.** Two distributions X and Y are ϵ-close (in ℓ_1 norm) if

$$\frac{1}{2} \sum_{a \in S} |Pr[X = a] - Pr[Y = a]| \leq \epsilon.$$

Definition 12 ((k, ϵ)-source)**.** A distribution X is a (k, ϵ)-source if it is ϵ-close to a k-source.

We now review the types of functions from [CRVW02] which they included in the zig-zag product. Throughout the following discussion, let $N = 2^n$, $M = 2^m$, $K = 2^k$, $K_{max} = 2^{k_{max}}$, and $D = 2^d$. Also, define a bipartite graph $G = (L \cup R, E)$ whose N left vertices L are indexed by (n) and whose M right vertices R are indexed by (m). Finally, recall that U_d denotes the uniform distribution on (d).

Definition 13 (Lossless Conductor)**.** Given a function $E : (n) \times (d) \to (m)$, a $k_{max} \in \mathbb{R}^+$, and a k-source X (for some $0 \leq k \leq k_{max}$), E is a (k_{max}, ϵ) lossless conductor if $E(X, U_d)$ is a $(k + d, \epsilon)$ source.

The definition of a lossless conductor implies that for every subset $S \subseteq L$ with $|S| \leq K_{max}$, it holds that $|N(S)| \geq (1 - \epsilon)D|S|$. In particular, the condition that X is a k-source for $0 \leq k \leq k_{max}$ implies that $|S| \leq K_{max}$, and the condition that $E(X, U_d)$ is a $(k + d, \epsilon)$ source implies that $|N(S)| \geq (1 - \epsilon)D|S|$. In terms of entropy, this definition indicates that up to an ϵ factor, none of the k_{max} bits of entropy from X or the d bits of entropy from U_d are lost (hence the name "lossless conductor").

Definition 14 (Extracting Conductor)**.** Given a function $E : (n) \times (d) \to (m)$, an $a \in [0, m]$, and a k-source X over (n) with $k \in [0, m - a]$, E is an (ϵ, a) extracting conductor if $E(X, U_d)$ is a $(k + a, \epsilon)$ source.

Following previous notation, let $A = 2^a$. In the context of vertex expansion, the definition of an extracting conductor implies that for $S \subseteq L$ with $|S| \geq K$, it holds that $|N(S)| \geq (1 - \epsilon)K \cdot A$. In terms of entropy, at least a bits of entropy are extracted from the d random bits of entropy available in U_d while the k bits of entropy in X are preserved.

Definition 15 (Permutation Conductor)**.** A function $\langle E, C \rangle : (n) \times (d) \to (m) \times (b)$ is an (ϵ, a) permutation conductor if E is an (ϵ, a) extracting conductor and $\langle E, C \rangle$ is a permutation of the elements in $(n + d)$.

A permutation conductor simply re-arranges the vertices while maintaining the original distribution (and hence the original entropy). However, during the re-arrangement, the permutation conductor ensures that there are at least $k+a$ bits of entropy in the distribution on (m) (assuming X is a k-source).

Definition 16 (Buffer Conductor)**.** A function $\langle E, C\rangle : (n) \times (d) \to (m) \times (b)$ is a buffer conductor if E is an (ϵ, a) extracting conductor and $\langle E, C\rangle$ is a (k_{max}, ϵ) lossless conductor.

Assuming X is a k-source ($0 \leq k \leq k_{max}$), a buffer conductor is a lossless conductor which extracts at least $k + a$ bits of entropy into the distribution on (m) and transfers $k + d$ overall bits of entropy to the distribution on $(m) \times (b)$.

We are now ready to describe the generalized zig-zag product which produces a lossless conductor. The goal at each step will be to preserve all of the entropy introduced in both the "zig" and the "zag" steps. As noted after the definition of lossless conductors, a distribution with large min-entropy implies that the size of the distribution's support set is large, and large sets of neighbors of a given set S (as close to $D|S|$ as possible) are exactly the goal.

The "zig" step ($\langle E_2, C_2\rangle : (n_2) \times (d_2) \to (d_1) \times (b_2)$ in Definition 9) will be a buffer conductor. Consequently, all of the entropy initially present in the distribution on (n_2) as well as the new entropy introduced by U_{d_2} on (d_2) will be preserved (up to a $1 - \epsilon$ factor). Part of the entropy will be extracted and stored in the distribution on (d_1). The rest of the entropy will be stored in the distribution on (b_2).

The intermediate permutation step ($\langle E_1, C_1\rangle : (n_1) \times (d_1) \to (m_1) \times (b_1)$ in Definition 9) will preserve the entropy from the previous step since it is simply a permutation (the underlying distribution has been re-ordered but not altered). Part of this entropy will be extracted into the distribution on (m_1). The rest of the entropy will be stored in the distribution on (b_1).

The "zag" step ($E_3 : (b_1 + b_2) \times (d_3) \to (m_3)$ in Definition 9) will be a lossless conductor. Consequently, it will preserve all the entropy (up to a $1 - \epsilon$ factor) from the

distributions on (b_1) and (b_2) as well as the additional entropy introduced by the distribution U_{d_3} on (d_3). It will then transfer as much of this entropy as possible to the distribution on (m_3). Note that enough entropy must be extracted into the distributions on (d_1) and (m_1) from E_2 and E_1 respectively in order for E_3 to be able to transmit the desired number of bits of entropy to (m_3). This is the reason for requiring that E_2 and E_1 be extractors. Overall, the new zig-zag product will be a lossless expander graph. The following theorem formally states the result which we have just summarized intuitively:

Theorem 26 (Theorem 6.2 in [CRVW02])**.** *Suppose we are given the following functions:*

- *An* (ϵ, a_1) *permutation conductor:* $\langle E_1, C_1 \rangle : (n_1) \times (d_1) \to (m_1) \times (b_1)$
- *An* (n_2, ϵ, a_2) *buffer conductor:* $\langle E_2, C_2 \rangle : (n_2) \times (d_2) \to (d_1) \times (b_2)$
- *An* $(m_3 - a_3, \epsilon)$ *lossless conductor:* $E_3 : (b_1 + b_2) \times (d_3) \to (m_3)$.

Moreover, suppose that the parameters satisfy the following relationships:

- $a_1 \geq d_2 + a_3 + (n_2 - m_3) + \log 1/\epsilon$
- $m_3 \geq d_1 + (n_1 - m_1) + (d_2 - a_2) + a_3 + \log 1/\epsilon$.

Then, the zig-zag product $E : (n_1 + n_2) \times (d_2 + d_3) \to (m_1 + m_3)$ *of* $\langle E_1, C_1 \rangle$, $\langle E_2, C_2 \rangle$, *and* E_3 *is a* $(k'_{max}, 5\epsilon)$ *lossless conductor, where* $k'_{max} = m_1 + m_3 - a_3 - d_2$.

Loosely, the two conditions

$$a_1 \geq d_2 + a_3 + (n_2 - m_3) + \log 1/\epsilon$$

and

$$m_3 \geq d_1 + (n_1 - m_1) + (d_2 - a_2) + a_3 + \log 1/\epsilon$$

respectively require that (b_1) does not have too many bits of entropy to transfer and that (m_3) has enough bits to receive the required number of bits of entropy via the E_3 map. Next, set the parameters as follows (where $0 \leq t \leq n$):

- $a_3 = d_3 + \log(1/\epsilon) + O(1)$ (which implies that all d_3 bits of entropy are transmitted to (m_3) via E_3)
- $a_1 = t + c \cdot (\log(t+1) + \log(1/\epsilon)$
- $n_2 = c \cdot a_1$ (for a sufficiently large constant c)
- $d_2 = \log n_2 + 2\log(1/\epsilon)$
- $a_2 = d_2 - 2\log(1/\epsilon) - O(1)$
- $b_2 = n_2 + d_2 - d_1$
- $d_3 = \log(d_1 + b_2) + \log(1/\epsilon) + O(1)$.

With this selection of parameters, the two conditions of Theorem 26 are satisfied, and $E : (n) \times (d) \to (m)$ is a (k_{max}, ϵ) lossless conductor for $k_{max} = (n-t) - d - \log(1/\epsilon) - O(1)$, $n = n_1 + n_2$, $d = d_2 + d_3$, and $m = n - t$. In summary, for any $\epsilon > 0$, this theorem gives an explicit construction of a D-left regular (γ, α) vertex expander with $\gamma = \Omega(\frac{\epsilon M}{N \cdot D})$ such that when $|S| < \gamma n$, it holds that $|N(S)| > (1-\epsilon)D|S|$.

Unlike the previous zig-zag product, this new zig-zag product is not composed iteratively. Instead, the (ϵ, a_1) permutation conductor is explicitly constructed from a large d-regular expander graph for which many explicit constructions (including the original zig-zag product) are known. The (n_2, ϵ, a_2) buffer conductor and $(m_3 - a_3, \epsilon)$ lossless conductors can be constructed by brute force since they are of constant size for a D-regular graph.

This general zig-zag product is best illustrated by an example. In an attempt to follow the example for the original zig-zag product as closely as possible, we set $n_2 = d_1 = m_3$ and

$m_1 = n_1$. Also, we let $G_1 = \mathbb{Z}_3^2$ ($n_1 = \log_2(9)$, $d_1 = \log_2(4)$), $d_3 = \log_2(2)$, and $G_2 = C_4$ ($n_2 = 4$, $d_2 = \log_2(2)$). Note that 9 is not in the allowable vertex range, but we allow it here it for consistency with the previous example. We let $b_1 = \log_2(4)$ and leave b_2 unspecified. The initial graph is represented pictorially as follows:

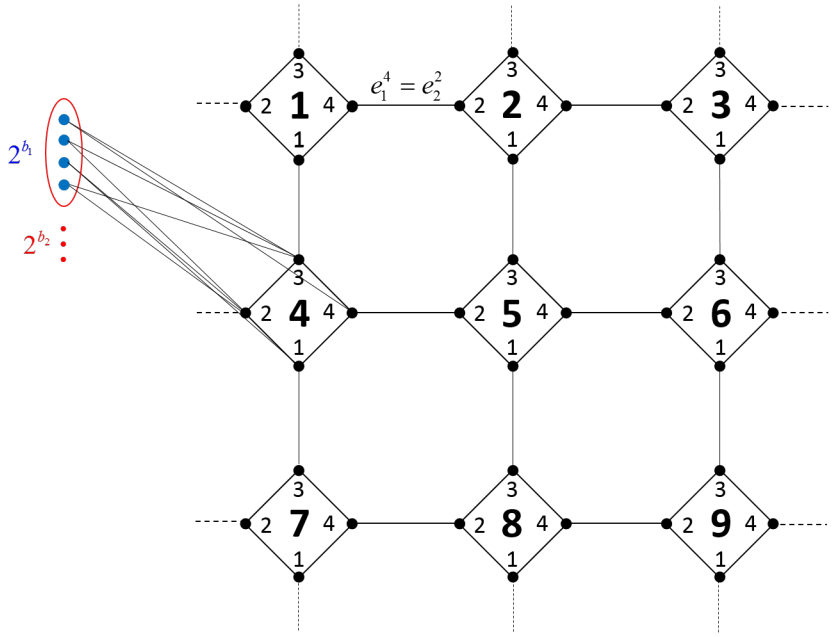


Figure 2.13: Generalized Zig-zag Product: Initial Graph

First select vertex 5 in G_1 and vertex 3 in G_2.

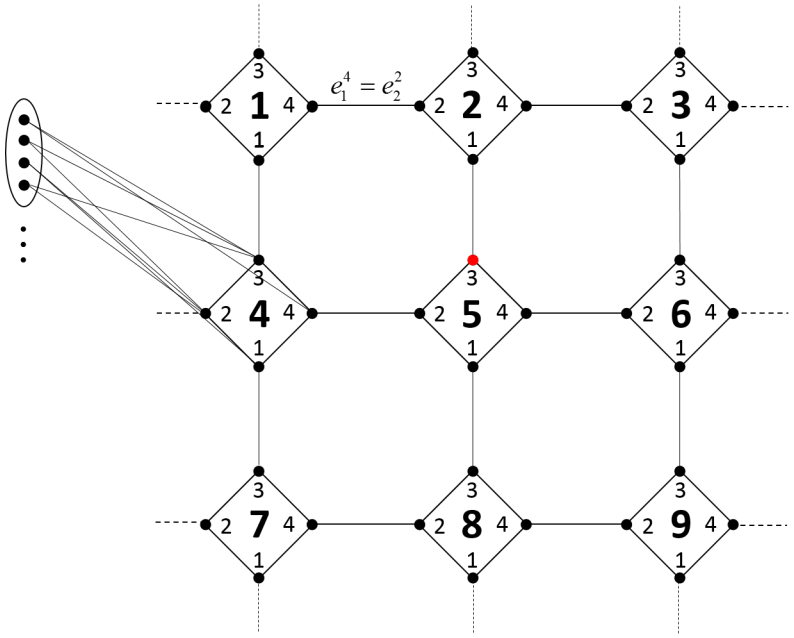


Figure 2.14: Generalized Zig-zag Product: Select Initial Vertex

Next, take a "zig" step on the smaller graph from $(5, 3)$ to $(5, 2)$ using the E_2 function and simultaneously select one of the 2^{b_2} "clumps" of vertices using the C_2 function.

- $E_2 : (d_1) \times (d_2) \to (d_1)$: Given by the edge set on the 2^{d_2}-regular 2^{d_1}-vertex graph $G_2 = C_4$ (denoted by the solid red line in the figure below). E_2 must extract at least a given amount of entropy to the distribution on (d_1).

- $C_2 : (d_1) \times (d_2) \to (b_2)$: Given by the edge set from the vertices of $G_2 = C_4$ to the 2^{b_2} vertex clusters (the selected edge is denoted by the dashed red line in the figure below). The remaining entropy in the distributions on (d_1) and (d_2) is stored in the distribution on (b_2).

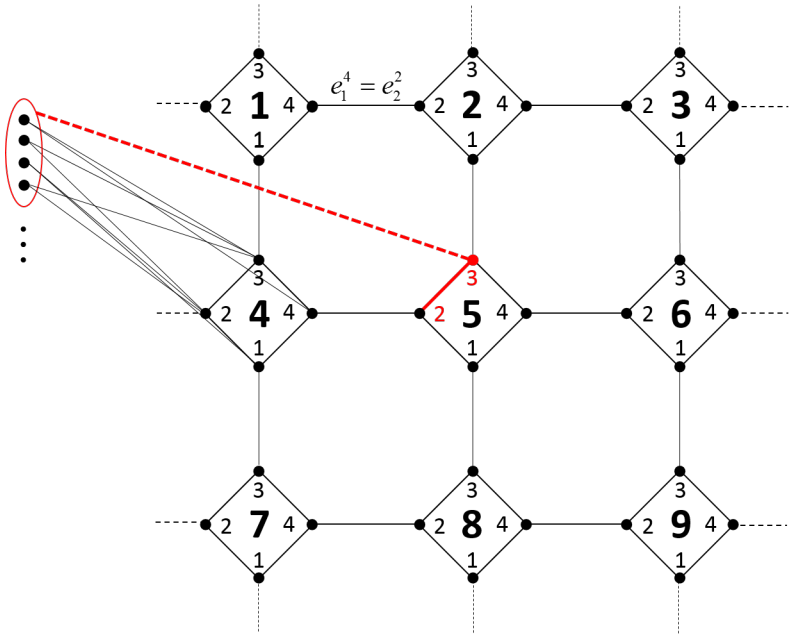


Figure 2.15: Generalized Zig-zag Product: "Zig" Step

We next take a step on the larger graph from $(5, 2)$ to $(4, 4)$ using the E_1 function. Using the C_1 function, we simultaneously select one of the 2^{b_1} vertices in the vertex cluster which we selected during the previous step. Since $\langle E_1, C_1 \rangle$ is a permutation, none of the entropy from the previous step is lost.

- $E_1 : (n_1) \times (d_1) \to (n_1)$: Given by the edge set on the 2^{d_1}-regular 2^{n_1}-vertex graph G_1 (denoted by the solid blue line in the figure below). E_1 must extract at least a fixed amount of entropy into the distribution on (n_1).

- $C_1 : (n_1) \times (d_1) \to (b_1)$: Given by the edge set from the vertices of G_1 to the 2^{b_1} vertices in the vertex cluster selected during the previous step (the selected edge is denoted by the dashed blue line in the figure below). The remaining entropy is stored in the distribution on (b_1)

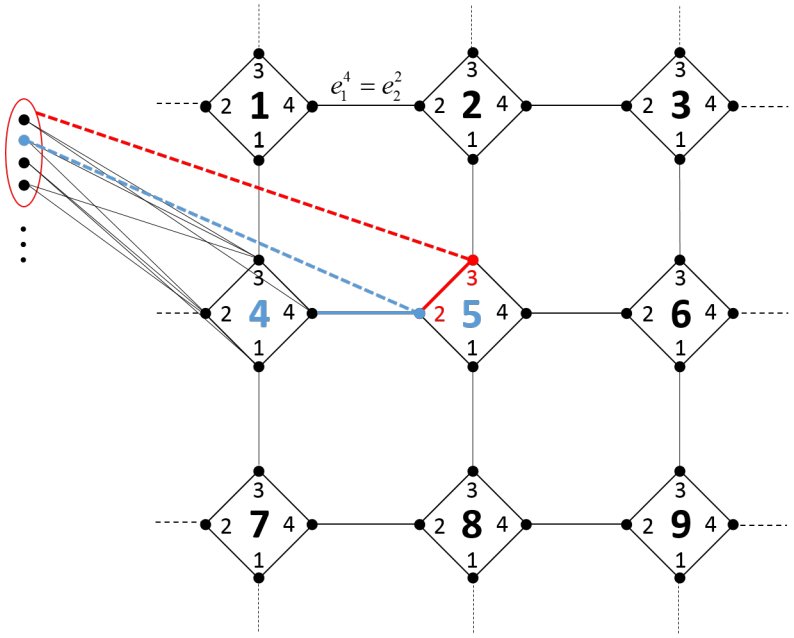


Figure 2.16: Generalized Zig-zag Product: Permutation Step

Finally, using the E_3 function, take a step from the vertex which was chosen in the previous two steps. In our example, the two possibilities are vertices $(4, 2)$ and $(4, 3)$.

- $E_3 : (b_1 + b_2) \to (d_1)$: Given by the edge set from the $2^{b_1+b_2}$ left vertices to the 2^{d_1} vertices in a given copy of G_2 (the selected edges are denoted by the solid green lines in the figure below). E_3 must transfer all of the entropy in (b_1) and (b_2) as well as the d_3 additional bits of entropy to the distribution on (d_1).

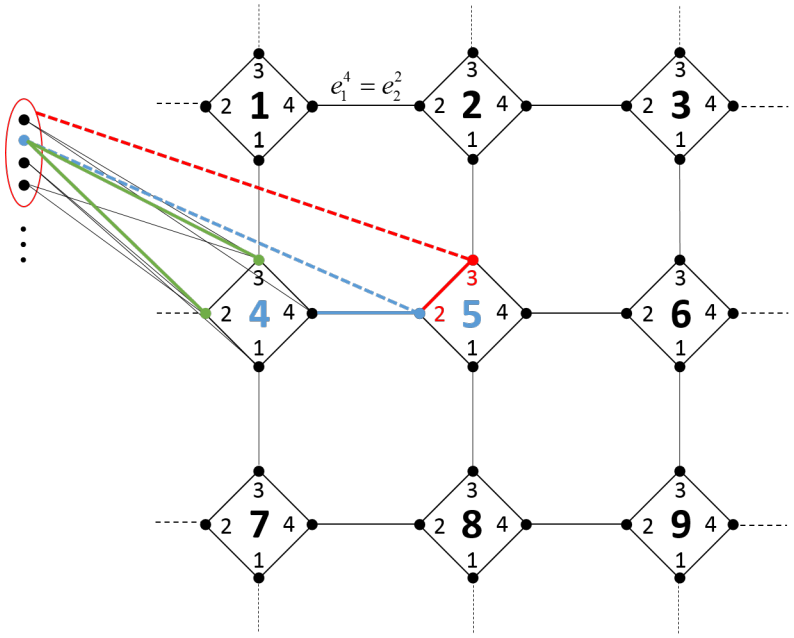


Figure 2.17: Generalized Zig-zag Product: "Zag" Step

In total, $d_2 + d_3$ additional bits of entropy have been introduced to the distribution on $(n_1) \times (d_1)$ (up to a $1 - \epsilon$ factor), and the expander graph is lossless as desired. Of course, in this example, we have completely ignored all restrictions in Theorem 26, and we already noted that we have introduced degrees which are not powers of 2. Nevertheless, at a high level, this example serves to illustrate the general zig-zag product construction and to provide a conceptual bridge between the original zig-zag product and the generalized zig-zag product.

Chapter 3

Computing Edge Expansion

In this chapter, we present a linear-programming formulation which allows us to compute the set with minimal edge expansion within a given, fixed set of size at most half the size of the graph. We then show how to find a set with minimal edge expansion (up to an approximation guarantee) over all sets with "enough" overlap of a given, fixed set. Finally, we formulate the problem of finding the minimum edge expansion over the whole graph as a program which is linear except in one constraint.

3.1 Background

3.1.1 Introduction

Given an arbitrary graph G, we define the edge expansion of a subset $T \subseteq V(G)$ to be $e(T) := |E(T, \overline{T})|/|T|$, where we recall that $E(T, \overline{T})$ denotes the set of edges from T to its complement. We say that $(T, \overline{T})$ forms a *cut* over the set of vertices V, and for notational convenience, we let $e(T, \overline{T}) := |E(T, \overline{T})|$. Also, we recall that the edge expansion ratio $\Phi_E(G)$ is defined as the minimum $e(T)$ over all $T \subseteq V(G)$ with $|T| \leq \frac{n}{2}$.

A fundamental question in algorithmic design is to find a set $S \subseteq V(G)$ with $|S| \leq n/2$

such that $e(S) = \Phi_E(G)$. Since exactly computing such a set is NP-hard, many researchers have invested considerable effort in finding good approximations to $\Phi_E(G)$. In particular, for very large networks such as web graphs, social networks, neural networks, etc., there has been a growing interest in approximating $\Phi_E(G)$ (and related quantities) locally [LS93, ST04, ACL06, AC07, Chu07].

For any graph G and any $T \subseteq V(G)$, define the local minimal edge expansion ratio as

$$\tilde{\Phi}_E(T) := \min_{S \subseteq T} e(S).$$

In what follows, we are interested in computing $\tilde{\Phi}_E(T)$ exactly for any fixed subset T of vertices in any given graph. Since finding $\Phi_E(G)$ is an NP-hard problem, we might intuitively expect finding $\tilde{\Phi}_E(T)$ to be very difficult as well. Surprisingly, this is not the case, and there is a well-known algorithm presented by Lang and Rao [LR04] which finds $\tilde{\Phi}_E(T)$ exactly in polynomial time. Andersen and Lang [AL08] showed that it is also possible to extend the search for small cuts slightly beyond T for a given $T \subseteq V(G)$. In addition, Goemans [Goe97] provided an integer programming formulation for finding $\Phi_E(G)$ itself. In what follows, we present a unified, independently derived framework for these three results in the context of a single linear programming formulation which we slightly modify at each step. Using this framework, we first provide an alternate algorithm and proof that it is possible to find $\tilde{\Phi}_E(T)$ in polynomial time. Building on this result, we then show that we can approximate small cuts which are "close" to the original given set T. We note that the main result in [AL08] follows from our result in the case of a d-regular graph. Building on these first two results, we finally provide a formulation for computing the minimal cut over the whole graph in terms of an optimization problem which is linear in everything except for a single 1-norm constraint.

3.1.2 Setup

In what follows, for any integer n, let $[n]$ denote the level set $\{1, 2, \ldots, n\}$. Let G be any undirected graph with vertex set $[n]$ and edge set $E(G) \subset [n] \times [n]$. Our primary objective will be to minimize the function

$$B(x_1, \ldots, x_n) = \sum_{(i,j) \in E(G)} |x_i - x_j|$$

subject to

$$A(x) \leq b,$$

where $A(x) \leq b$ denotes a set of linear constraints. The B-function $B(x_1, x_2, \ldots, x_n)$ is used in the formulation of the Cheeger constant in [Chu97, Section 2.5] and is also the 1-norm analogue of the Dirichlet sum in [Chu97, Section 1.2]. It is also closely related to the B_f function used in [HLW06] to prove the discrete Cheeger inequality in the d-regular case. We can convert B into a linear function by introducing a new variable $y_{i,j}$ for each edge $(i, j) \in E(G)$ which gives us the following problem:

$$\begin{aligned} &\min \sum_{(i,j) \in E(G)} y_{i,j} \\ \text{subject to} \quad & \\ &y_{i,j} \geq x_i - x_j, \qquad y_{i,j} \geq -(x_i - x_j), \quad \text{for each edge } (i, j) \in E(G), \\ &A(x) \leq b, \end{aligned}$$

where the variables $x_1, \ldots, x_n$ correspond to the vertices of the graph and the variables $y_{i,j}$ for $(i, j) \in E(G)$ correspond to the edges of the graph. One can see that the solution of this problem is equal to the minimum value of $B(x_1, \ldots, x_n)$ subject to $A(x) \leq b$.

Lemma 27. *Let* $\mathbf{x} = (x_1, \ldots, x_n)$ *be any vector in* $\mathbb{R}^n$*. Suppose*

$$x_{t_1} \geq x_{t_2} \geq \cdots \geq x_{t_n}$$

where $\{t_1, t_2, \ldots, t_n\}$ *is a permutation of* $[n]$*. Let* $T_i = \{t_1, t_2, \ldots, t_i\}$ *for* $1 \leq i \leq n$*. Then*

$$B(x_1, \ldots, x_n) = \sum_{i=1}^{n-1} (x_{t_i} - x_{t_{i+1}}) e(T_i, \overline{T_i}).$$

Proof. This proof is motivated by the proof of Lemma 4.13 in [HLW06]. Note that

$$B(x_1, x_2, \ldots, x_n) = \sum_{(t_i, t_j) \in E(G)} |x_{t_i} - x_{t_j}| = \sum_{\substack{(t_i, t_j) \in E(G) \\ i<j}} \sum_{k=i}^{j-1} x_{t_k} - x_{t_{k+1}}$$

by telescoping sums. Then,

$$\sum_{\substack{(t_i, t_j) \in E(G) \\ i<j}} \sum_{k=i}^{j-1} x_{t_k} - x_{t_{k+1}} = \sum_{i=1}^{n-1} (x_{t_i} - x_{t_{i+1}}) e(T_i, \overline{T_i})$$

which follows by noting that in the double sum, the term $x_{t_i} - x_{t_{i+1}}$ appears only for edges from t_ℓ to t_m where $\ell \leq i < m$. But $e(T_i, \overline{T_i})$ gives exactly the number of edges from $t_1, \ldots, t_i$ to $t_{i+1}, \ldots, t_n$ by definition, so equality holds. □

We next describe a method for finding the set with minimal edge expansion within a given, fixed set of size at most half the size of the graph.

3.2 Searching for Sets Inside a Neighborhood

Given a set $T \subseteq V(G)$, we show that there is a polynomial-time algorithm for finding $\tilde{\Phi}_E(T)$. This proof will motivate the main result of this chapter which occurs in the next

section. Our primary tool is the following optimization problem:

$$\begin{aligned} & \min_{\mathbf{x}\in\mathbb{R}^n} B(x_1,\ldots,x_n) \\ \text{subject to} \quad & \\ & \sum_{i=1}^{n} x_i = 1, \\ & x_i \geq 0 \text{ for } i \in T, \quad \text{and } x_i = 0 \text{ for } i \in \overline{T}, \end{aligned}$$

Notice that in this formulation, we restrict ourselves to the set T by only allowing positive values of x_i for $i \in T$.

Theorem 28. *For any subset $T \subseteq [n]$, an optimal solution to the above linear programming problem provides a cut $(S, \overline{S})$ with $S \subseteq T$ for which $\tilde{\Phi}_E(T) = e(S)$.*

Note that it follows as an immediate result of this theorem that $\tilde{\Phi}_E(T)$ can be computed in polynomial time.

Proof. Let $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ denote an optimal solution to the given linear program, and define

$$M := B(y_1, \ldots, y_n).$$

Order the entries so that $y_{t_1} \geq y_{t_2} \geq \cdots \geq y_{t_n}$ where $\{t_1, t_2, \ldots, t_n\}$ is a permutation of $[n]$. Let $T_i = \{t_1, t_2, \ldots, t_i\}, 1 \leq i \leq n$. Then, by Lemma 1,

$$B(y_1, y_2, \ldots, y_n) = \sum_{i=1}^{n-1} (y_{t_i} - y_{t_{i+1}}) e(T_i, \overline{T_i}) = \sum_{i=1}^{n-1} i(y_{t_i} - y_{t_{i+1}}) e(T_i)$$

since $|T_i| = i$. For $T \subsetneq [n]$, suppose that there are $k > 0$ non-zero entries in the solution vector. Also, note that for $|T| < n$, there is at least one zero entry in the optimal solution vector by definition of the linear-programming formulation. For $T = [n]$, the optimal solution is trivially $(\frac{1}{n}, \cdots, \frac{1}{n})$ which represents the cut $(V(G), \emptyset)$ with $e(V(G)) = 0$.

We now show that $e(T_i) \geq e(T_k)$ for all $1 \leq i < k$, proceeding by contradiction. Suppose

$$e(T_m) < e(T_k)$$

for some $1 \leq m < k$. Then, we define a new vector $\mathbf{y}'$ with its entries given by

$$y'_{t_j} = \begin{cases} y_{t_j} - y_{t_k} + \frac{k y_{t_k}}{m} & \text{if } 1 \leq j \leq m \\ y_{t_j} - y_{t_k} & \text{if } m < j \leq k \\ 0 & \text{otherwise.} \end{cases}$$

Note that $\mathbf{y}'$ satisfies the constraints of our linear program and the order of the entries is preserved. So, by Lemma 1

$$\begin{aligned} B(\mathbf{y}') &= \sum_{i=1}^{n-1} i(y'_{t_i} - y'_{t_{i+1}})e(T_i) \\ &= M + \frac{k y_{t_k}}{m} m e(T_m) - k y_{t_k} e(T_k) = M + k y_{t_k} e(T_m) - k y_{t_k} e(T_k). \end{aligned}$$

But we assumed that $e(T_m) < e(T_k)$ which contradicts the minimality of M. Thus, $e(T_m) \geq e(T_k)$ for all $1 \leq m < k$.

Using this fact, we show that given an optimal solution vector with k non-zero entries, there is an optimal solution vector whose ordered entries have the form $\left\{\frac{1}{k}, \frac{1}{k}, \ldots, \frac{1}{k}, 0, \ldots, 0\right\}$. We simultaneously provide an efficient method for transforming any optimal solution of the given linear program into this form. To begin, re-consider the optimal solution $\mathbf{y}$ with ordered entries

$$\left\{y_{t_1}, y_{t_2}, \ldots, y_{t_m}, y_{t_{m+1}}, y_{t_{m+2}}, \ldots, y_{t_k}, 0, \ldots, 0\right\}$$

where

$$y_{t_1} = \cdots = y_{t_m} > y_{t_{m+1}} \geq y_{t_{m+2}} \geq \cdots \geq y_{t_k} > 0$$

and $1 \leq m \leq k$. If $m = k$, we are done. Otherwise, let $\varepsilon := y_{t_m} - y_{t_{m+1}}$. Then, define $\mathbf{y}''$ such that

$$y''_{t_j} = \begin{cases} y_{t_j} - \varepsilon + \frac{\varepsilon m}{k} & \text{if } 1 \leq j \leq m \\ y_{t_j} + \frac{\varepsilon m}{k} & \text{if } m < j \leq k \\ 0 & \text{otherwise.} \end{cases}$$

Observe that this vector also satisfies the constraints of the linear program and preserves the original order. So, again using Lemma 1,

$$B(\mathbf{y}'') = \sum_{i=1}^{n-1} i(y''_{t_i} - y''_{t_{i+1}})e(T_i) = M - \varepsilon m e(T_m) + \frac{\varepsilon m}{k} k e(T_k)$$

which is at most M since $e(T_k) \leq e(T_m)$. Thus, $B(\mathbf{y}'') = B(\mathbf{y})$ since $B(\mathbf{y})$ is minimal. Note that in $\mathbf{y}''$,

$$y''_{t_1} = y''_{t_2} = \cdots = y''_{t_m} = y''_{t_{m+1}}$$

and $\mathbf{y}''$ is still an optimal solution. By repeating this process until all non-zero terms are equal, we see that there is an optimal solution whose ordered entries have the form $\left\{\frac{1}{k}, \frac{1}{k}, \ldots, \frac{1}{k}, 0, \ldots, 0\right\}$. Denoting this optimal solution by $\mathbf{y}'''$, we have

$$B(\mathbf{y}''') = \sum_{i=1}^{n-1} i(y'''_{t_i} - y'''_{t_{i+1}})e(T_i) = \frac{1}{k} k e(T_k) = e(T_k).$$

Finally, we show that $e(T_k)$ is minimal among all $e(S)$ with $S \subseteq T$. For any $S \subseteq T$, consider the vector $\mathbf{y_S}$ with the i^{th} entry given by

$$y^i_S = \begin{cases} \frac{1}{|S|} & \text{if } i \in S \\ 0 & \text{if } i \in \overline{S}. \end{cases}$$

Then, from Lemma 1, $B(\mathbf{y_S}) = e(S)$. So, since $B(\mathbf{y}) = e(T_k)$ is minimal, $e(S) \geq e(T_k)$ for

all $S \subseteq T$, and T_k is given by the set of vertices corresponding to the non-zero entries of $\mathbf{y}'''$. $\square$

Note that when $|T| \leq n/2$, we have located the set $S \subseteq T$ having minimum edge expansion. Next, we show how to find a set with minimal edge expansion (up to an approximation guarantee) over all sets with "enough" overlap of a given, fixed set of size at most half the size of the graph.

3.3 Searching for Sets Overlapping a Neighborhood

In the previous section, we searched for an optimal set within a given subset $T \subseteq [n]$. In this section, we extend the arguments of the previous section to allow us to search for an optimal set which overlaps a given subset $T \subseteq [n]$. Define $m := |T|$, and without loss of generality, assign the vertices in T to the integer indices in $[1, m]$. We can partition the vertex set $[n]$ into two disjoint sets:

$$[n] = [1, m] \cup (m, n].$$

Consider the following optimization problem

$$\min_{\mathbf{x} \in \mathbb{R}^n} B(x_1, \ldots, x_n)$$

subject to

$$(a) \quad \sum_{1 \leq i \leq m} x_i \geq 1,$$
$$(b) \quad \sum_{m < i \leq n} x_i \leq 0.$$

Intuitively, we are "encouraging" vertices to lie in T but are allowing vertices to lie outside T if they significantly contribute to a decrease in $B(x_1, \ldots, x_n)$. Let B_0 denote the optimal value of this optimization problem, which can be computed in polynomial time via a linear program. We are interested in characterizing the optimal solutions in terms of edge expansion, that is, how the optimal value B_0 is related to $e(S)$ for certain subsets $S \subseteq [n]$. For any subset $S \subseteq [n]$, let

$$S_1 = S \cap [1, m], \quad S_2 = S \cap (m, n].$$

Let $\delta_1, \delta_2 \in \mathbb{R}$ be such that $|\delta_1 - \delta_2|$ is minimum subject to

$$\delta_1|S_1| + \delta_2(m - |S_1|) \geq 1 \quad \text{and} \quad \delta_1|S_2| + \delta_2(n - m - |S_2|) \leq 0.$$

The feasible region defined by the two given inequalities must not cross the line $\delta_1 = \delta_2$. If $\delta_1 = \delta_2 = \delta$, then $\frac{1}{m} \leq \delta$ and $\delta \leq 0$ which gives a contradiction. It is then not difficult to see that at the optimal solution,

$$\delta_1|S_1| + \delta_2(m - |S_1|) = 1, \quad \text{and} \quad \delta_1|S_2| + \delta_2(n - m - |S_2|) = 0.$$

So, if $(n - m)|S_1| \neq m|S_2|$,

$$\delta_1 = \frac{n - m - |S_2|}{(n - m)|S_1| - m|S_2|} \quad \text{and} \quad \delta_2 = \frac{-|S_2|}{(n - m)|S_1| - m|S_2|}. \tag{3.1}$$

Define x_S as follows:

$$x_S[i] = \begin{cases} \delta_1, & i \in S, \\ \delta_2, & i \notin S. \end{cases}$$

Note that x_S satisfies conditions (a) and (b) in the linear-programming formulation. If

$\frac{|S_1|}{|S|} > \frac{m}{n}$, then $(n-m)|S_1| - m|S_2| > 0$. So, if $\frac{|S_1|}{|S|} > \frac{m}{n}$, then $\delta_1 > \delta_2$ (since $m < n$). Consequently by Lemma 1,

$$B(x_S) = (\delta_1 - \delta_2)|S| \cdot e(S) = \frac{1 - \frac{m}{n}}{\frac{|S_1|}{|S|} - \frac{m}{n}} e(S), \tag{3.2}$$

where δ_1 and δ_2 are given in (3.1). We are now ready to state the theorem.

Theorem 29. *Consider any $T \subseteq [n]$ and assign the vertices of T to [1,m] in the above optimization problem. Then,*

1. *Any optimal solution gives a subset $T' \subseteq [n]$ so that $B_0 = B(x_{T'})$ and $\frac{|T'_1|}{|T'|} > \frac{m}{n}$.*
2. *For every $S \subseteq V(G)$ with $\frac{|S_1|}{|S|} > \frac{m}{n}$, $e(T') \leq \left(\frac{\frac{|T'_1|}{|T'|} - \frac{m}{n}}{\frac{|S_1|}{|S|} - \frac{m}{n}} \right) e(S)$.*

The relationship between the approximation and overlap will become clearer in the next two corollaries.

Corollary 30. *$e(T') \leq e(S)$ for every $S \subseteq V(G)$ with $\frac{|S_1|}{|S|} \geq \frac{|T'_1|}{|T'|}$.*

Proof. Follows immediately from (ii) in the theorem since, in this case, $\left(\frac{\frac{|T'_1|}{|T'|} - \frac{m}{n}}{\frac{|S_1|}{|S|} - \frac{m}{n}} \right) \cdot e(S) \leq e(S)$. □

Corollary 31. *Given T' and T as in Theorem 29,*

1. *For any $\epsilon > 0, e(T') \leq \frac{1}{\epsilon} e(S)$ for all $S \subseteq V(G)$ satisfying $\frac{|S_1|}{|S|} \geq \frac{m}{n} + \epsilon \left(\frac{|T'_1|}{|T'|} - \frac{m}{n} \right)$.*
2. *For any $\epsilon > 0, e(\overline{T'}) \leq \frac{1}{\epsilon} e(S)$ for all $S \subseteq V(G)$ satisfying $\frac{|S_1|}{|S|} \geq \frac{m}{n} + \epsilon \left(\frac{m}{n} - \frac{|\overline{T'}_1|}{|\overline{T'}|} \right)$.*

It is in this precise sense that for a given T, up to an approximation factor, we can find a set S for which $e(S)$ is minimum over all sets S having large overlap with T. Note that ϵ determines the tightness of the inequalities $e(T') \leq \frac{1}{\epsilon} e(S)$ and $e(\overline{T'}) \leq \frac{1}{\epsilon} e(S)$. Larger values of ϵ imply a tighter inequality. Also, notice that ϵ and the ratio $\frac{|T'_1|}{|T'|}$ govern the amount

of overlap required between S and T. In particular, large values of ϵ and $\frac{|T_1'|}{|T'|}$ imply large required overlap between S and T. Small values of ϵ and $\frac{|T_1'|}{|T'|}$ imply small required overlap between S and T. Given Theorem 29, we now prove the corollary.

Proof. Let

$$\left(\frac{\frac{|T_1'|}{|T'|} - \frac{m}{n}}{\frac{|S_1|}{|S|} - \frac{m}{n}} \right) \leq \frac{1}{\epsilon}.$$

Then,

$$\epsilon \left(\frac{|T_1'|}{|T'|} - \frac{m}{n} \right) \leq \left(\frac{|S_1|}{|S|} - \frac{m}{n} \right),$$

which can be re-written as

$$\frac{|S_1|}{|S|} \geq \frac{m}{n} + \epsilon \left(\frac{|T_1'|}{|T'|} - \frac{m}{n} \right).$$

To show (ii), we follow a similar procedure. From (3.1),

$$\begin{aligned}
B(x_S) = (\delta_1 - \delta_2)|\overline{S}| \cdot e(\overline{S}) &= \left(\frac{n-m}{(n-m)|S_1| - m|S_2|} \right) |\overline{S}| e(\overline{S}) \\
= \left(\frac{n-m}{n|S_1| - m|S|} \right) |\overline{S}| e(\overline{S}) &= \left(\frac{n-m}{n(m - |\overline{S}_1|) - m(n - |\overline{S}|)} \right) |\overline{S}| e(\overline{S}) \\
= \left(\frac{n-m}{m|\overline{S}| - n|\overline{S}_1|} \right) |\overline{S}| e(\overline{S}) &= \left(\frac{1 - \frac{m}{n}}{\frac{m}{n} - \frac{|\overline{S}_1|}{|\overline{S}|}} \right) e(\overline{S}).
\end{aligned}$$

Note that

$$\begin{aligned}
\frac{m}{n} - \frac{|\overline{T'}_1|}{|\overline{T'}|} &= \frac{mn - m|T'| - mn + n|T_1'|}{n(n - |T'|)} \\
&= \frac{|T_1'| - \frac{m}{n}|T'|}{n - |T'|} = \frac{\frac{|T_1'|}{|T'|} - \frac{m}{n}}{\frac{n}{|T'|} - 1}.
\end{aligned}$$

Since $\frac{|T_1'|}{|T'|} > \frac{m}{n}$ and $\frac{n}{|T'|} > 1$, we see that $\frac{|\overline{T'}_1|}{|\overline{T'}|} < \frac{m}{n}$. So,

$$\left(\frac{1-\frac{m}{n}}{\frac{m}{n}-\frac{|\overline{T'}_1|}{|\overline{T'}|}}\right)e(\overline{T'}) \leq \left(\frac{1-\frac{m}{n}}{\frac{|S_1|}{|S|}-\frac{m}{n}}\right)e(S)$$

and

$$e(\overline{T'}) \leq \left(\frac{\frac{m}{n}-\frac{|\overline{T'}_1|}{|\overline{T'}|}}{\frac{|S_1|}{|S|}-\frac{m}{n}}\right)e(S).$$

Now, requiring that

$$\left(\frac{\frac{m}{n}-\frac{|\overline{T'}_1|}{|\overline{T'}|}}{\frac{|S_1|}{|S|}-\frac{m}{n}}\right) \leq \frac{1}{\epsilon},$$

we see that

$$\epsilon\left(\frac{m}{n}-\frac{|\overline{T'}_1|}{|\overline{T'}|}\right) \leq \frac{|S_1|}{|S|}-\frac{m}{n}.$$

So, the condition for the inequality to hold becomes

$$\frac{|S_1|}{|S|} \geq \frac{m}{n}+\epsilon\left(\frac{m}{n}-\frac{|\overline{T'}_1|}{|\overline{T'}|}\right)$$

as desired. □

Note that in the d-regular case, the main theorem in [AL08] can be written as

Theorem 32 ([AL08])**.** *Given $T \subseteq V(G)$ with $|T| := m \leq \frac{n}{2}$, we can find a subset $\tilde{T} \subseteq V(G)$ with $|\tilde{T}| \leq \frac{n}{2}$ such that*

- $e(\tilde{T}) \leq e(S) \quad \forall\ S \subseteq T$
- $e(\tilde{T}) \leq \frac{1}{\epsilon}e(S) \quad \forall\ S \subseteq V(G)\ s.t.\ \frac{|S_1|}{|S|} \geq \frac{m}{n}+\epsilon\left(1-\frac{m}{n}\right).$

We now show that Theorem 32 is a corollary of Theorem 29.

Proof. First, it follows immediately from Corollary 30 that $e(\tilde{T}) \leq e(S)$ for all $S \subseteq T$. Second, if $|T'| \leq \frac{n}{2}$, it is easy to see that $\epsilon\left(\frac{|T'_1|}{|T'|}-\frac{m}{n}\right) \leq \epsilon\left(1-\frac{m}{n}\right)$. Consequently, in this case,

(i) in Corollary 31 immediately implies that $e(\tilde{T}) \leq \frac{1}{\epsilon} e(S)$ for all $S \subseteq V(G)$ such that $\frac{|S_1|}{|S|} \geq \frac{m}{n} + \epsilon\left(1 - \frac{m}{n}\right)$.

If $|\overline{T'}| \leq \frac{n}{2}$, we must make a slightly more complicated argument. Note that $2\frac{m}{n} \leq 1$ since we are assuming in Theorem 32 that $m \leq \frac{n}{2}$. So, $2\frac{m}{n} \leq 1 + \frac{|\overline{T'}_1|}{|\overline{T'}|}$, which implies that $\frac{m}{n} - \frac{|\overline{T'}_1|}{|\overline{T'}|} \leq 1 - \frac{m}{n}$. Consequently, in this case, by setting $\tilde{T} = \overline{T'}$, part (ii) of Corollary 31 implies that $e(\tilde{T}) \leq \frac{1}{\epsilon} e(S)$ for all $S \subseteq V(G)$ such that $\frac{|S_1|}{|S|} \geq \frac{m}{n} + \epsilon\left(1 - \frac{m}{n}\right)$.

□

We now proceed to the proof of Theorem 29.

Proof. We first prove (ii). Assume for the moment that (i) holds so that there exists a $T' \subseteq [n]$ with $\frac{|T'_1|}{|T'|} > \frac{m}{n}$ and $B_0 = B(x_{T'})$. Then for every $S \subseteq V(G)$,

$$B(x_{T'}) \leq B(x_S) \tag{3.3}$$

since $B(x_{T'})$ is optimal. From (3.2), inequality (3.3) is equivalent to

$$\frac{1 - \frac{m}{n} - \frac{m}{n}\gamma}{\frac{|T'_1|}{|S|} - \frac{m}{n}} e(T') \leq \frac{1 - \frac{m}{n} - \frac{m}{n}\gamma}{\frac{|S_1|}{|S|} - \frac{m}{n}} e(S). \tag{3.4}$$

Finally, since $\frac{|T'_1|}{|T'|} > \frac{m}{n}$, we can easily manipulate (3.4) to obtain

$$e(T') \leq \left(\frac{\frac{|T'_1|}{|T'|} - \frac{m}{n}}{\frac{|S_1|}{|S|} - \frac{m}{n}} \right) e(S).$$

We now give the proof of (i). Our strategy will be to show that for the given optimization problem, there is an optimal solution vector taking at most two distinct values. Using this fact, Theorem 29 will follow as an easy consequence of Lemma 1. Suppose we take an optimal vector $\mathbf{x} = (x_1, \ldots, x_n)$ and order the vertices so that $x_{t_1} \geq x_{t_2} \geq \cdots \geq x_{t_n}$, where $\{t_1, t_2, \ldots, t_n\}$ is a permutation of $[n]$. Moreover, suppose that the elements in $\mathbf{x}$ take

at least three distinct values. Let $a > b > c$ denote the three largest values of $\mathbf{x}$. Then, define

$$L_1 := \{i \in [1,m] : x_i = a\}, L_2 := \{x_i \in [1,m] : x_i = b\}, L_3 := [1,m] \cap (L_1 \cup L_2)^c$$

and

$$R_1 := \{i \in (m,n] : x_i = a\}, R_2 := \{i \in (m,n] : x_i = b\}, R_3 := (m,n] \cap (R_1 \cup R_2)^c.$$

Further, define

$$\Delta_1 := a - b, \quad \Delta_2 := b - c, \quad e_1 := e(L_1 \cup R_1), \quad e_2 := e(L_1 \cup R_1 \cup L_2 \cup R_2).$$

Following similar reasoning to the proof of Theorem 28, define a new vector $\mathbf{x}'$ from $\mathbf{x}$ by setting

$$x_i' = \begin{cases} x_i + \delta_1', & i \in L_1 \cup R_1, \\ x_i + \Delta_1 + \delta_1', & i \in L_2 \cup R_2, \\ x_i + \delta_2', & i \in L_3 \cup R_3 \end{cases}$$

where $\delta_1', \delta_2' \in \mathbb{R}$ and satisfy

$$\delta_1'|L_1| + \Delta_1|L_2| + \delta_1'|L_2| + \delta_2'|L_3| = 0, \quad \delta_1'|R_1| + \Delta_1|R_2| + \delta_1'|R_2| + \delta_2'|R_3| = 0.$$

Then, assuming $\delta_1' + \Delta_1 + \Delta_2 \geq \delta_2'$, we see that $\mathbf{x}'$ still satisfies the constraints to the optimization problem and preserves the original ordering of the vertices as well. Solving this system of equations yields

$$\delta_1' = \frac{1}{d_1}\left(-\Delta_1|L_3||R_2| + \Delta_1|R_3||L_2|\right), \quad \delta_2' = \frac{1}{d_1}\left(\Delta_1|R_2|(|L_1| + |L_2|) - \Delta_1|L_2|(|R_1| + |R_2|)\right)$$

where

$$d_1 := |L_3|(|R_1| + |R_2|) - |R_3|(|L_1| + |L_2|),$$

and for now, we assume that $d_1 \neq 0$. Notice that

$$\begin{aligned} d_1 &= |L_3|(|R_1| + |R_2|) - |R_3|(|L_1| + |L_2|) \\ &= |L_3|(|R_1| + |R_2| + |R_3|) - |R_3|(|L_1| + |L_2| + |L_3|) = |L_3|(n-m) - |R_3|m. \end{aligned}$$

By Lemma 1,

$$B(\mathbf{x}) = (|L_1| + |R_1|)\Delta_1 e_1 + (|L_1| + |R_1| + |L_2| + |R_2|)\Delta_2 e_2 + C$$

where C denotes the contribution to $B(\mathbf{x})$ from the elements in $\{x_1, x_2, \ldots, x_n\}$ which do not take values in $\{a, b, c\}$. Moreover,

$$B(\mathbf{x}') = (|L_1| + |R_1| + |R_2| + |L_2|)(\Delta_1 + \Delta_2 + \delta_1' - \delta_2')e_2 + C. \tag{3.5}$$

So,

$$B(\mathbf{x}') - B(\mathbf{x}) = (|L_1| + |R_1| + |R_2| + |L_2|)(\Delta_1 + \delta_1' - \delta_2')e_2 - (|L_1| + |R_1|)\Delta_1 e_1.$$

Next, note that

$$\delta_1' - \delta_2' = \frac{1}{d_1}(\Delta_1|L_2|(|R_1|+|R_2|+|R_3|) - \Delta_1|R_2|(|L_1|+|L_2|+|L_3|)) = \frac{\Delta_1}{d_1}(|L_2|(n-m) - |R_2|m).$$

So,

$$\delta_1' - \delta_2' = \Delta_1 \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m}.$$

We now consider the option of subtracting Δ_2 from each $x \in L_2 \cup R_2$. Similarly to

before, define a new vector $\mathbf{x}''$ from $\mathbf{x}$ by setting

$$x_i'' = \begin{cases} x_i + \delta_1'', & i \in L_1 \cup R_1, \\ x_i - \Delta_2 + \delta_2'', & i \in L_2 \cup R_2, \\ x_i + \delta_2'', & i \in L_3 \cup R_3. \end{cases}$$

where $\delta_1'', \delta_2'' \in \mathbb{R}$ and satisfy

$$\delta_1''|L_1| - \Delta_2|L_2| + \delta_2''|L_2| + \delta_2''|L_3| = 0, \quad \delta_1''|R_1| - \Delta_2|R_2| + \delta_2''|R_2| + \delta_2''|R_3| = 0.$$

Then, $\mathbf{x}''$ still satisfies the constraints to the optimization problem and preserves the original ordering of the vertices if we assume that $\delta_1'' + \Delta_1 + \Delta_2 \geq \delta_2''$. Solving this system of equations yields

$$\delta_1'' = \frac{1}{d_2}(\Delta_2|L_2|(|R_2| + |R_3|) - \Delta_2|R_2|(|L_2| + |L_3|)), \quad \delta_2'' = \frac{1}{d_2}(-\Delta_2|L_2||R_1| + \Delta_2|L_1||R_2|)$$

where

$$d_2 := |L_1|(|R_2| + |R_3|) - |R_1|(|L_2| + |L_3|),$$

and for now, we assume $d_2 \neq 0$. We will return to prove that both d_1 and d_2 must be non-zero. Note that $d_2 = |L_1|(|R_2| + |R_3|) - |R_1|(|L_2| + |L_3|) = |L_1|(|R_1| + |R_2| + |R_3|) - |R_1|(|L_1| + |L_2| + |L_3|) = |L_1|(n - m) - |R_1|m$ as well. Now,

$$B(\mathbf{x}'') = (|L_1| + |R_1|)(\Delta_1 + \Delta_2 + \delta_1'' - \delta_2'')e_1 + C.$$

and consequently,

$$B(\mathbf{x}'') - B(\mathbf{x}) = (|L_1| + |R_1|)(\Delta_2 + \delta_1'' - \delta_2'')e_1 - (|L_1| + |R_1| + |L_2| + |R_2|)\Delta_2 e_2.$$

Similarly to the case where we computed $\delta_1' - \delta_2'$, we have that

$$\delta_1'' - \delta_2'' = \Delta_2 \frac{|L_2|(n-m) - |R_2|m}{|L_1|(n-m) - |R_1|m}.$$

We next derive an important identity. Given that $d_1, d_2 \neq 0$,

$$\begin{aligned}
& \left(1 + \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m}\right)\left(1 + \frac{|L_2|(n-m) - |R_2|m}{|L_1|(n-m) - |R_1|m}\right) \\
= 1 \quad + & \quad \frac{|L_2|(n-m)^2(|L_1| + |L_3|) + m^2|R_2|(|R_1| + |R_3|)}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
= 1 \quad + & \quad \frac{|L_2|(n-m)^2(|L_1| + |L_3|) + m^2|R_2|(|R_1| + |R_3|)}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
& - \quad \frac{m|R_2|(n-m)(|L_1| + |L_3|) + m(n-m)|L_2|(|R_1| + |R_3|)}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
& + \quad \frac{(|L_2|(n-m) - |R_2|m)^2}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
= 1 \quad + & \quad \frac{(n-m)^2|L_2|(m - |L_2|) + m^2|R_2|((n-m) - |R_2|)}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
& - \quad \frac{m|R_2|(n-m)(m - |L_2|) + m(n-m)|L_2|((n-m) - |R_2|)}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
& + \quad \frac{(|L_2|(n-m) - |R_2|m)^2}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} = \\
= 1 \quad + & \quad \frac{-(n-m)^2|L_2|^2 + 2m(n-m)|L_2||R_2| - m^2|R_2|^2}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
& + \quad \frac{(|L_2|(n-m) - |R_2|m)^2}{(|L_1|(n-m) - |R_1|m)(|L_3|(n-m) - |R_3|m)} \\
= 1.
\end{aligned}$$

Consequently,

$$1 + \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m} = \frac{1}{1 + \frac{|L_2|(n-m) - |R_2|m}{|L_1|(n-m) - |R_1|m}}. \tag{3.6}$$

We now prove the result. Suppose for sake of contradiction that

$$B(\mathbf{x}'') - B(\mathbf{x}) > 0 \quad \text{and} \quad B(\mathbf{x}') - B(\mathbf{x}) > 0.$$

Specifically, suppose that

$$(|L_1| + |R_1| + |R_2| + |L_2|)(\Delta_1 + \delta_1' - \delta_2')e_2 - (|L_1| + |R_1|)\Delta_1 e_1 > 0$$

and

$$(|L_1| + |R_1|)(\Delta_2 + \delta_1'' - \delta_2'')e_1 - (|L_1| + |R_1| + |L_2| + |R_2|)\Delta_2 e_2 > 0.$$

Let $D := 1 + \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m}$. Substituting for $\delta_1' - \delta_2'$ and $\delta_1'' - \delta_2''$ respectively, dividing by Δ_1 and Δ_2 respectively (which are guaranteed to be nonzero by assumption), and applying (3.6), we see that the above inequalities are equivalent to the inequalities

$$(|L_1| + |R_1| + |R_2| + |L_2|)De_2 - (|L_1| + |R_1|)e_1 > 0$$

$$(|L_1| + |R_1|)\frac{1}{D}e_1 - (|L_1| + |R_1| + |L_2| + |R_2|)e_2 > 0.$$

Multiplying the second equation by D and re-writing, we see that this pair of inequalities reduces to

$$(|L_1| + |R_1| + |R_2| + |L_2|)De_2 - (|L_1| + |R_1|)e_1 > 0$$

and

$$(|L_1| + |R_1| + |R_2| + |L_2|)De_2 - (|L_1| + |R_1|)e_1 < 0$$

both of which cannot be satisfied simultaneously. So, we have the desired contradiction. We

implicitly assumed $D > 0$, but if $D \leq 0$, the first inequality already does not hold, so we have a contradiction in that case as well. So, either $B(\mathbf{x}') \leq B(\mathbf{x})$ or $B(\mathbf{x}'') \leq B(\mathbf{x})$. Since we assumed $\mathbf{x}$ was an optimal solution, either $B(\mathbf{x}') = B(\mathbf{x})$ or $B(\mathbf{x}'') = B(\mathbf{x})$. So, we can reduce the number of distinct values in the optimal solution by at least one. We can continue this process until only two values remain.

Recall that we assumed throughout the proof that $d_1 \neq 0, d_2 \neq 0$. We now show that these assumptions are valid. To begin, suppose for sake of contradiction that $d_1 = |L_3|(n-m) - |R_3|m = 0$ and $d_2 = |L_1|(n-m) - |R_1|m = 0$. Note that

$$|R_1| + |R_2| + |R_3| = |L_1| + |L_2| + |L_3| + n - 2m.$$

So, from our assumption that $|R_3| = \frac{|L_3|(n-m)}{m}$, we find after algebraic manipulation that $|R_1| + |R_2| = (|L_1| + |L_2|)\left(\frac{n-m}{m}\right)$. However, we also assumed that $|R_1| = \frac{|L_1|(n-m)}{m}$, so we see that $|R_2| = \frac{|L_2|(n-m)}{m}$ as well.

Consequently, in the following system of equations:

$$\delta_1'|L_1| + \Delta_1|L_2| + \delta_1'|L_2| + \delta_2'|L_3| = 0, \quad \delta_1'|R_1| + \Delta_1|R_2| + \delta_1'|R_2| + \delta_2'|R_3| = 0,$$

the second equation is entirely redundant. Consequently, we have one free variable. Setting

$$\delta_1' = -\Delta_1 \left(1 + \frac{|L_2|}{|L_3|}\right) \frac{|L_3|}{m},$$

we see that $\delta_1' - \delta_2' = -\Delta_1$ in which case $B(\mathbf{x}') - B(\mathbf{x}) = -(|L_1| + |R_1|)\Delta_1 e_1 < 0$ contradicting minimality of $B(\mathbf{x})$.

Next, suppose $d_1 = 0, d_2 \neq 0$. Then, as we have seen, $|R_1| + |R_2| = (|L_1| + |L_2|)\left(\frac{n-m}{m}\right)$ which implies that $(n-m)|L_1| - m|R_1| = -\left((n-m)|L_2| - m|R_2|\right)$. So, $\delta_1'' - \delta_2'' = -\Delta_2$ and $B(\mathbf{x}'') - B(\mathbf{x}) = -(|L_1| + |R_1| + |L_2| + |R_2|)\Delta_2 e_2 < 0$ which again contradicts minimality

of $B(\mathbf{x})$. Similarly, when $d_1 \neq 0, d_2 = 0$, we see that $\delta_1' - \delta_2' = -\Delta_1$ which implies that $B(\mathbf{x}') - B(\mathbf{x}) < 0$. Thus, $d_1, d_2 \neq 0$.

Recall that we also assumed that $\delta_1' + \Delta_1 + \Delta_2 \geq \delta_2'$ and $\delta_1'' + \Delta_1 + \Delta_2 \geq \delta_2''$. We now show that these assumptions are valid. For sake of contradiction, begin by supposing both $\delta_1' - \delta_2' < -\Delta_1 - \Delta_2$ and $\delta_1'' - \delta_2'' < -\Delta_1 - \Delta_2$. Note that

$$\delta_1' - \delta_2' < -\Delta_1 - \Delta_2 \text{ if and only if } \Delta_1 \left(1 + \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m}\right) < -\Delta_2$$

and

$$\delta_1'' - \delta_2'' < -\Delta_1 - \Delta_2 \text{ if and only if } \Delta_2 \left(1 + \frac{|L_2|(n-m) - |R_2|m}{|L_1|(n-m) - |R_1|m}\right) < -\Delta_1.$$

Observing that $1 + \frac{|L_2|(n-m)-|R_2|m}{|L_1|(n-m)-|R_1|m} < 0$ if the second inequality holds and applying (3.6) to the second inequality yields that the given system of inequalities is equivalent to

$$\Delta_1 \left(1 + \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m}\right) < -\Delta_2, \quad \Delta_1 \left(1 + \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m}\right) > -\Delta_2,$$

which gives the desired contradiction. Thus, $\delta_1' - \delta_2' \geq -\Delta_1 - \Delta_2$ or $\delta_1'' - \delta_2'' \geq -\Delta_1 - \Delta_2$. Now, suppose that $\delta_1' - \delta_2' \geq -\Delta_1 - \Delta_2$ and $\delta_1'' - \delta_2'' < -\Delta_1 - \Delta_2$. Then,

$$1 + \frac{|L_2|(n-m) - |R_2|m}{|L_1|(n-m) - |R_1|m} < 0 \text{ and (3.6) imply that } 1 + \frac{|L_2|(n-m) - |R_2|m}{|L_3|(n-m) - |R_3|m} < 0.$$

Multiplying this last inequality by Δ_1, we see that $\Delta_1 + \delta_1' - \delta_2' < 0$.
Hence,

$$B(\mathbf{x}') - B(\mathbf{x}) = (|L_1| + |R_1| + |R_2| + |L_2|)(\Delta_1 + \delta_1' - \delta_2')e_2 - (|L_1| + |R_1|)\Delta_1 e_1 < 0$$

which is impossible since we assumed $B(\mathbf{x})$ was optimal. The case where $\delta_1' - \delta_2' < -\Delta_1 - \Delta_2$

and $\delta_1'' - \delta_2'' \geq -\Delta_1 - \Delta_2$ is handled similarly. Thus, our assumptions are valid, and we have established the fact that there is an optimal solution to the given optimization problem with at most two distinct values. Note that no solution can have only one value. Suppose for sake of contradiction that there were an optimal solution having the form $(\alpha, \alpha, \ldots, \alpha)$. Then, any solution in this form which satisfies the given constraints must satisfy

$$m\alpha \geq 1, \quad (n-m)\alpha \leq 0$$

which gives a contradiction. So, there is an optimal solution taking exactly two values, say δ_1, δ_2 with $\delta_1 > \delta_2$. Define T' to be those vertices corresponding to the entries in the optimal solution with value δ_1. Then, $B_0 = B(x_{T'})$. Moreover, from (3.1),

$$\delta_1 - \delta_2 = \frac{n-m}{n|T_1'| - m|T'|}.$$

Thus, since $\delta_1 - \delta_2 > 0$, $\frac{|T_1'|}{|T'|} > \frac{m}{n}$ proving (i). □

Next, we formulate the problem of exactly computing the minimum edge expansion of a graph.

3.4 Computing Exactly

In this section, we extend our arguments to formulate the problem of finding $\Phi_E(G)$ as an optimization problem which is linear except for a single 1-norm constraint. Consider the following optimization problem:

$$\min_{\mathbf{x} \in \mathbb{R}^n} B(x_1, \ldots, x_n)$$

subject to

$$(i) \quad \sum_{i=1}^{n} x_i = 1, \quad x_i \geq 0 \text{ for } 1 \leq i \leq n,$$
$$(ii) \quad \sum_{i=1}^{n} \left| x_i - \frac{1}{n^4} \right| = 1.$$

Proposition 33. *For any $x \in \mathbb{R}^n$ satisfying (i) and (ii), define*

$$S = \left\{ 1 \leq i \leq n : x_i > \frac{1}{n^4} \right\}, \quad \alpha = \sum_{i \in S} x_i.$$

Then $k = |S| \leq n/2$ and $\alpha = 1 - \frac{n-2k}{2n^4}$.

Proof. Define $\beta = \sum_{i \notin S} x_i$. Then (i) and (ii) imply that

$$\alpha + \beta = 1, \quad \left(\alpha - \frac{k}{n^4} \right) + \left(\frac{n-k}{n^4} - \beta \right) = 1.$$

Hence

$$\alpha = 1 - \frac{n-2k}{2n^4}, \quad \beta = \frac{n-2k}{2n^4}.$$

Since $\beta \geq 0$, we have $k \leq n/2$. □

Now, for any nonempty subset $T \subset [n]$ of size $k \leq n/2$, we define

$$x_T[i] = \begin{cases} \frac{1}{k} \left(1 - \frac{n-2k}{2n^4} \right), & i \in T, \\ \frac{n-2k}{(n-k)(2n^4)}, & i \notin T. \end{cases}$$

Note that x_T satisfies conditions (i) and (ii), and

$$B(x_T) = \frac{1}{k} \left(1 - \frac{n-2k}{2(n-k)n^3} \right) \cdot e(T, \bar{T}) = \left(1 - \frac{n-2k}{2(n-k)n^3} \right) \cdot e(T) = e(T) - \epsilon. \qquad (3.7)$$

Also, since $e(T)$ is at most the largest vertex degree of G (which is less than n),

$$0 \leq \epsilon = \frac{n-2k}{2(n-k)n^3} \cdot e(T) < \frac{1}{2n^2}. \tag{3.8}$$

We now show that the set S computed in the previous proposition is optimum.

Theorem 34. *Let* $\mathbf{x}$ *be any optimal solution with* S *being defined as in Proposition 33. Then* $\Phi_E(G) = e(S)$.

Proof. We follow a similar approach to the proof in the previous section. Suppose $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ is an optimal solution vector and $\{t_1, t_2, \ldots, t_n\}$ is a permutation of $[1, n]$ such that $x_{t_1} \geq x_{t_2} \geq x_{t_k} \geq x_{t_{k+1}} \geq \ldots \geq x_{t_n}$. As before, define $M := B(\mathbf{x})$. Notice that the first k terms in $\{t_1, t_2, \ldots, t_n\}$ are in S since for each element $x \in S$, we have $x > \frac{1}{n^4}$. In contrast, for each element $y \in \overline{S}$, we have $y \leq \frac{1}{n^4}$. As before, define $T_i := \{t_1, t_2, \ldots, t_i\}$. We first wish to show that $e(T_i) \geq e(T_k)$ for all $1 \leq i < k$. For sake of contradiction, suppose there is an integer m with $1 \leq m < k$ such that $e(T_m) < e(T_k)$. Define $\Delta := x_{t_k} - \frac{1}{n^4}$, then define a new vector $\mathbf{x}'$ where

$$x'_{t_j} = \begin{cases} x_{t_j} - \Delta + \frac{k\Delta}{m} & \text{if } 1 \leq j \leq m \\ x_{t_j} - \Delta & \text{if } m < j \leq k \\ x_{t_j} & \text{otherwise.} \end{cases}$$

Since $\mathbf{x}'$ still satisfies the constraints and the original ordering, by Lemma 27,

$$B(\mathbf{x}') = M + \frac{k\Delta}{m} m e(T_m) - k\Delta e(T_k)$$

which contradicts the minimality of M since we assumed that $e(T_m) < e(T_k)$. So, $e(T_i) \geq e(T_k)$ for all $1 \leq i < k$.

Next, let $\varepsilon = x_{t_1} - x_{t_2}$. Then, define a new vector $\mathbf{x}''$ such that

$$x''_{t_j} = \begin{cases} x_{t_j} - \varepsilon + \frac{\varepsilon}{k} & \text{if } j = 1 \\ x_{t_j} + \frac{\varepsilon}{k} & \text{if } 1 < j \leq k \\ x_{t_j} & \text{otherwise.} \end{cases}$$

Note that, again by Lemma 27,

$$B(\mathbf{x}'') = M - \varepsilon e(T_1) + \varepsilon e(T_k).$$

But, since $e(T_1) \geq e(T_k)$, $B(\mathbf{x}'') \leq B(\mathbf{x})$. So, by optimality, $B(\mathbf{x}'') = B(\mathbf{x})$ and we have decreased the number of distinct terms in $\mathbf{x}$ by at least one. Also, note that $\sum_{i=1}^{k} x'_i = \alpha$ as well, and (i) and (ii) are still satisfied in the linear-programming formulation given at the beginning of this section. Following the same pattern until we reach the $k-1$ term, we see that there is an optimal solution for which $x_{t_1} = x_{t_2} = \cdots = x_{t_k}$. Moreover, since $\sum_{i=1}^{k} x_i = \alpha$, it follows that $x_i = \frac{\alpha}{k} = \frac{1}{k}\left(1 - \frac{n-2k}{2n^4}\right)$.

Now, suppose the distinct values among x_i, for $i > k$ ($i \notin S$), are $\delta_1, \ldots, \delta_m$, which we may assume satisfy

$$\frac{1}{n^4} > \delta_1 > \cdots > \delta_m \geq 0$$

by definition of S. Then, by Lemma 27, there are subsets $S_1 \subset \cdots \subset S_m \subset [n]$ so that

$$B(\mathbf{x}) = \left(\frac{1}{k}\left(1 - \frac{n-2k}{2n^4}\right) - \delta_1\right) e(S, \bar{S}) + \sum_{i=1}^{m-1} (\delta_i - \delta_{i+1}) e(S_i, \bar{S}_i) = e(S) - \delta, \tag{3.9}$$

where

$$\delta = \left(\frac{n-2k}{2n^4} + k\delta_1\right) \cdot e(S) - \sum_{i=1}^{m-1} (\delta_i - \delta_{i+1}) e(S_i, \bar{S}_i).$$

Note that since $e(S)$ is at most the largest vertex degree of G, we see that $e(S) < n$. So,

$$\left(\frac{n-2k}{2n^4} + k\delta_1\right) \cdot e(S) < \left(\frac{n-2k}{2n^4} + \frac{k}{n^4}\right) \cdot n = \frac{1}{2n^2}$$

and

$$\begin{aligned}
\sum_{i=1}^{m-1}(\delta_i - \delta_{i+1})e(S_i, \bar{S}_i) &= \sum_{i=1}^{m-1}(\delta_i - \delta_{i+1}) \cdot \frac{n}{2} \cdot \frac{e(S_i, \bar{S}_i)}{n/2} \\
&\leq \sum_{i=1}^{m-1}(\delta_i - \delta_{i+1}) \cdot \frac{n}{2} \cdot n \\
&= (\delta_1 - \delta_m) \cdot \frac{n^2}{2} < \frac{1}{n^4} \cdot \frac{n^2}{2} < \frac{1}{2n^2}.
\end{aligned}$$

It follows that

$$|\delta| < \frac{1}{2n^2}. \tag{3.10}$$

Let $T \subset [n]$ be any subset with $|T| \leq n/2$. By (3.7) and (3.8), we have

$$B(x_T) = e(T) - \epsilon$$

where $0 \leq \epsilon < 1/(2n^2)$. Moreover, the expressions in (3.9) and (3.10) imply that

$$B(x_T) = B(\mathbf{x}) - B(\mathbf{x}) + e(T) - \epsilon = B(\mathbf{x}) + (\delta - \epsilon) - (e(S) - e(T)),$$

where $|\delta - \epsilon| < 1/n^2$. Suppose for sake of contradiction that $e(T) < e(S)$. Then, by definition,

$$e(S) - e(T) = \frac{N}{|T| \cdot |S|}$$

for some integer $N \geq 1$. Since $|T|$ and $|S|$ are at most $n/2$, we see that

$$e(S) - e(T) \geq \frac{1}{(n/2)^2} = \frac{4}{n^2}.$$

Hence $e(S) - e(T) > |\delta - \epsilon|$, which implies that $B(x_T) < B(\mathbf{x})$. However, this is impossible, since $B(\mathbf{x})$ is the minimum value. Therefore, for every subset $T \subset [n]$ of size at most $n/2$, we have $e(T) \geq e(S)$, which implies that $\Phi_E(G) = e(S)$ as desired. $\square$

Chapter 4

Expander Codes

In this chapter, we first introduce codes on graphs. In particular, we discuss LDPC and GLDPC codes based on Tanner graphs. Next, after giving an overview of linear programming and message-passing decoding, we briefly describe girth-based approaches used to analyze the performance of these decoding algorithms. We also discuss the advantages and limitations of girth-based analysis. In particular, while girth-based approaches give excellent results with high probability, they do not provide asymptotically good bounds (i.e. bounds which guarantee the correction of any pattern of at most αn errors even as $n \to \infty$, where $\alpha > 0$ is fixed and n denotes the length of the code). In contrast, arguments based on the expansion of the underlying graph do give asymptotically good bounds on fast decoding algorithms. Such arguments either rely on the second-largest eigenvalue of the underlying adjacency matrix (spectral arguments), or they rely directly on the vertex expansion of the underlying graph. After providing a synopsis of results obtained using spectral expansion arguments, we give a detailed summary of results obtained by arguing directly from the vertex expansion of the underlying graph. While all of these results require vertex expansion greater than 1/2, we describe our new contribution which gives asymptotically good codes with fast decoding algorithms even when the vertex expansion is much smaller than 1/2.

4.1 Background

4.1.1 Tanner Graphs

As mentioned in Chapter 1, one can describe a linear code as the set of solutions to the linear system $Hx = 0$, where $H = (h_{ij})$ is called the *parity-check matrix*. Each row of H gives a parity check constraint for the code. Gallager [Gal63] provided a graphical representation of the relationship between the parity check constraints and the variable bits of a codeword by forming a tree in which paths represented parity checks and nodes represented variable bits. The nodes lying along a given path denoted the variable nodes associated with a given parity check equation (see Figure 2.5 in [Gal63]). However, it was Tanner [Tan81] who first noticed that any binary linear code has a very simple graphical description as a bipartite graph $G = (L \cup R, E)$, where L denotes the set of vertices on the left-hand side and R denotes the set of vertices on the right-hand side. In particular, each row of the parity-check matrix corresponds to a *constraint node* in R, and each column of the parity-check matrix corresponds to a *variable node* in L. Given $i \in L$ and $j \in R$, $(i, j) \in E$ if and only if $h_{ij} = 1$.

Alternately, given a bipartite graph $G = (L \cup R, E)$, we can define a binary linear code. Given an ordering on the nodes in L, index the individual bits in a vector $\mathbf{x} \in \mathbb{F}_2^n$ by the nodes in L. Denote this indexing by $\mathbf{x} = (x_i)_{i \in L}$. In what follows, given a node $k \in L \cup R$, let $N(k)$ denote the neighbors of the node k (not including k itself). We can now define a binary linear code from the graph $G = (L \cup R, E)$.

$$\mathcal{C} := \left\{ \mathbf{c} = (c_i)_{i \in L} \in \mathbb{F}_2^n \text{ such that for each } j \in R, \sum_{i \in N(j)} c_i = 0 \bmod 2 \right\}. \tag{4.1}$$

For example, the [7,4,3] Hamming code can be represented as follows:

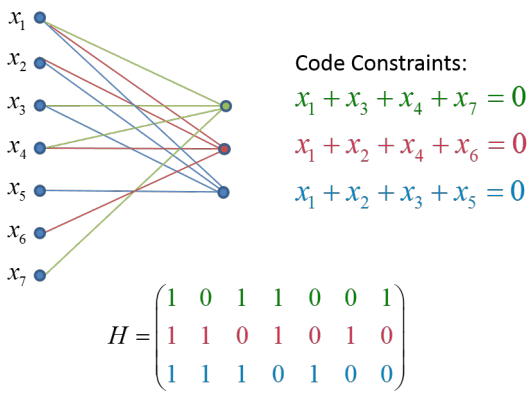


Figure 4.1: Tanner Graph for the [7,4,3] Hamming Code

Codes defined as above for which the number of non-zero entries in the columns and rows of the parity-check matrix are bounded by constants c and d (respectively) which are independent of the size of the matrix are called low-density parity-check (LDPC) codes.

It is possible to generalize this graphical approach to codes over any field. Suppose we are given a d right-regular bipartite graph $G = (L \cup R, E)$ and an $\mathbb{F}_q$-linear code C_d of length d (called the *inner code*). Given a a node $j \in R$ and a vector $\mathbf{x} = (x_i)_{i \in L}$ indexed by the nodes in L, let $\mathbf{x}|_{N(j)}$ denote $(x_i)_{i \in N(j)}$. We can now define an $\mathbb{F}_q$-linear *Tanner code* $T(G, C_d)$. A vector $\mathbf{c} = (c_i)_{i \in L} \in \mathbb{F}_q^n$, where n is the number of vertices in L, is a codeword of $T(G, C_d)$ if and only if

$$\mathbf{c}|_{N(j)} \in C_d \text{ for each } j \in R.$$

We also call the code $T(G, C_d)$ a generalized LDPC (GLDPC) code. Recall that the parity-check code is defined as the set of all codewords

$$\mathcal{C}_{pc} := \left\{ \mathbf{c} = (c_i)_{i \in [1,d]} \in \mathbb{F}_2^d \text{ such that } \sum_{i=1}^{d} c_i = 0 \bmod 2 \right\}.$$

Consequently, in the case of d-right regular bipartite graphs, the codes described in (4.1)

can alternately be described as Tanner codes $T(G, C_d)$, where C_d is a parity-check code of length d.

Theorem 35 ([Tan81])**.** *Given a d-right regular graph $G = (L \cup R, E)$ with $|L| = n$ and $|R| = m$, the rate of $T(G, C_d)$ is at least $1 - \frac{md}{n}(1 - r_d)$, where r_d denotes the rate of C_d. If G is (c, d)-biregular, then the rate of $T(G, C_d)$ is at least $1 - c(1 - r_d)$.*

Proof. Given a code C_d with rate r_d and length d, the dimension of C_d is $d \cdot r_d$. So, C_d is defined by $d - d \cdot r_d = d(1 - r_d)$ linear equations over $\mathbb{F}_q$. Consequently, $T(G, C_d)$ is defined by at most $md(1 - r_d)$ linear equations over $\mathbb{F}_q$. This implies that $T(G, C_d)$ has dimension at least $n - md(1 - r_d)$ (since $n = |L|$ gives the length of the codewords in $T(G, C_d)$). So, the rate of $T(G, C_d)$ is at least

$$\frac{n - md(1 - r_d)}{n} = 1 - \frac{md}{n}(1 - r_d).$$

Note that when G is a (c, d)-biregular graph, $nc = dm$, so the rate of $T(G, C_d)$ is $1 - c(1 - r_d)$ in this case. □

As an aside, note that using similar arguments, it is possible to show that the rate of $T(G, C_d)$ is upper bounded by $c \cdot r_0$.

Corollary 36. *Let G be a (c, d)-biregular graph, and let C_d be the parity-check code. Then, the rate of $T(G, C_d)$ is at least $1 - \frac{c}{d}$.*

Proof. Substitute $r_0 = 1 - 1/d$ in Theorem 35. □

We pause to note that both LDPC codes and GLDPC codes can be defined over irregular graphs. The use of irregular graphs has been shown to enhance the properties of LDPC codes [LMSS01]; however, due to the increased difficulty in analyzing such codes, we will primarily focus on (c, d)-biregular Tanner graphs in this chapter. Also, though we

do not discuss it further here, GLDPC codes have been used effectively over the AWGN channel using a simple soft-decision decoder [BPZ99], coming within 0.72dB of the capacity of the channel. We next describe two general decoding methods which will be relevant in the discussion that follows.

4.1.2 Linear Programming (LP) Decoding

Using linear programming techniques, Feldman et al. [FWK05] introduced a different method for decoding a binary linear code over a DMC. Given a code $\mathcal{C}$, the idea is to solve a linear optimization problem over the convex hull of the codewords, where we now view $\mathcal{C}$ as a subset of $\mathbb{R}^n$ instead of a subset of $\mathbb{F}_2^n$. The convex hull of $\mathcal{C}$ is given as follows:

$$conv(\mathcal{C}) = \left\{ \sum_{c \in \mathcal{C}} \lambda_c c : \lambda_c \geq 0 \text{ and } \sum_{c \in \mathcal{C}} \lambda_c = 1 \right\}.$$

Since the vertices of this convex hull are the codewords in $\mathcal{C}$, and since the optimal solution of a linear optimization problem over a convex hull is always a vertex of the convex hull, the solution to a linear optimization problem over $conv(\mathcal{C})$ will always be a codeword.

The objective function can be chosen to ensure that the optimal codeword is the maximum-likelihood codeword. Let $\hat{\mathbf{y}}$ denote the input to the decoder, and let $\hat{y}_i$ denote the ith bit of $\hat{\mathbf{y}}$. Similarly, let $\mathbf{y}$ denote the output of the decoder, and let y_i denote the ith bit of $\mathbf{y}$. Now, for a code of length n, define the linear program as follows:

$$\min \sum_{i=1}^{n} \gamma_i y_i, \tag{4.2}$$

subject to

$$\mathbf{y} \in conv(\mathcal{C}),$$

where

$$\gamma_i = \log\left(\frac{Pr[\hat{y}_i|y_i = 0]}{Pr[\hat{y}_i|y_i = 1]}\right).$$

Assuming that $Pr[i|i] > Pr[i|j]$ for $i \neq j$ with $i, j \in \{0, 1\}$, notice that if $\hat{y}_i = 1$, $\gamma_i < 0$. If $\hat{y}_i = 0$, $\gamma_i > 0$. Consequently, minimizing (4.2) over $conv(\mathcal{C})$ penalizes the decoder for flipping bits. The decoder is penalized more for flipping bits whose value is quite certain than for flipping bits whose value is quite uncertain. Intuitively, it is clear that minimizing (4.2) over $conv(\mathcal{C})$ is an alternate method for performing ML-decoding.

In the case of the BSC, we can set $\gamma_i = -1$ if $\hat{y}_i = 1$ and $\gamma_i = 1$ if $\hat{y}_i = 0$. Note that if we minimize over this cost function, each time the decoder flips a value in $\hat{y}_i$, the decoding cost increases by 1. Consequently, it is not hard to see that the codeword which minimizes this cost function over $conv(\mathcal{C})$ will be the closest codeword to the received word in Hamming distance. (Recall that for the BSC, the closest codeword is also the maximum-likelihood codeword.)

While it is possible to describe $conv(\mathcal{C})$ with a finite number of constraints, the number of these constraints grows exponentially quickly in the length of the code. Consequently, Feldman et al. [FWK05] defined a relaxation of an *integer* programming problem with a manageable number of fixed-length constraints. Given an LDPC code defined on a bipartite graph $(L \cup R, E)$, denote the parity-check code on constraint node $j \in R$ by $\mathcal{C}_j$, and define the code

$$\mathcal{C} := \left\{\mathbf{c} = (c_1, c_2, \ldots, c_n) \in \mathbb{F}_2^n \text{ such that } \mathbf{c}|_{N(j)} \in \mathcal{C}_j \text{ for each } j \in R\right\}.$$

In many cases, the codes on the constraint nodes will be the same, so $\mathcal{C}_{j'} = \mathcal{C}_{j''}$ for $j' \neq j''$. However, to provide as much flexibility as possible, we allow a different code on each constraint. Note that the formulation in [FWK05] exclusively considers the parity-check code, while the formulation we present below generalizes to any binary code. For each

$j \in R$, let d_j denote the degree of node j. To make the notation $\mathbf{c}|_{N(j)}$ precise, we need a bijection

$$\tau_j : N(j) \to \{1, 2, \cdots, d_j\}$$

for each $j \in R$. For notational convenience, we will drop the subscript j when it is clear from the context. Then, we can write

$$\mathbf{c}|_{N(j)} = (c_{\tau^{-1}(1)}, c_{\tau^{-1}(2)}, \ldots, c_{\tau^{-1}(d_j)}).$$

For each $j \in R$ and each codeword $v \in \mathcal{C}_j$, introduce a new variable $I(v, j)$. Consider the following linear programming problem:

$$\min \sum_{i=1}^{n} \gamma_i y_i$$

such that for each $j \in R$,

$$\sum_{v \in \mathcal{C}_j} I(v, j) = 1 \tag{4.3}$$

and for each $j \in R$ and $i \in N(j)$,

$$y_i = \sum_{v \in \mathcal{C}_j} v_{\tau(i)} I(v, j) \tag{4.4}$$

with

$$I(v, j) \in \{0, 1\} \text{ for each } j \in R \text{ and each } v \in \mathcal{C}_j. \tag{4.5}$$

The constraints in (4.3) and (4.5) imply that $I(v, j) = 1$ for exactly one codeword $v \in \mathcal{C}_j$ and $I(v, j) = 0$ for all other codewords in $\mathcal{C}_j$. Thus, $I(v, j)$ serves as an indicator variable for the local codeword selected in $\mathcal{C}_j$. The constraints in (4.4) ensure that $y_i = v_{\tau(i)}$ for each selected codeword $v \in \mathcal{C}_j$ such that $j \in N(i)$. Consequently, every $y_i \in \{0, 1\}$, and each feasible

solution $(y_1, y_2, \ldots, y_n)$ is a codeword in $\mathcal{C}$. For an LDPC code defined over a (c, d)-biregular graph, there are a total of $|R|(1 + d + 2^{d-1})$ linear constraints and $2^{d-1}|R| + |L|$ variables. Consequently, both the number of constraints and number of variables depend only linearly on the length of the code. Next, we relax the integrality constraints and obtain the linear program in [FWK05] (in a slightly more general form):

$$\min \sum_{i=1}^{n} \gamma_i y_i$$

such that for each $j \in R$,

$$\sum_{v \in \mathcal{C}_j} I(v, j) = 1 \tag{4.6}$$

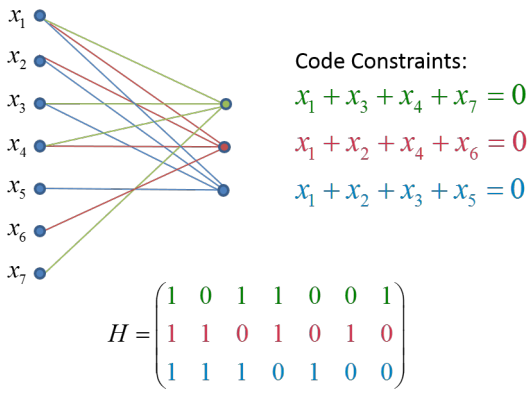
and for each $j \in R$ and $i \in N(j)$,

$$y_i = \sum_{v \in \mathcal{C}_j} v_{\tau(i)} I(v, j) \tag{4.7}$$

with

$$0 \leq I(v, j) \leq 1 \text{ for each } j \in R \text{ and each } v \in \mathcal{C}_j. \tag{4.8}$$

For example, the bijection for the first constraint node in the [7,4,3] Hamming code in Figure 4.1 (reproduced below for convenience)



is given by

$$\tau_1 : 1 \mapsto 1$$

$$\tau_1 : 3 \mapsto 2$$

$$\tau_1 : 4 \mapsto 3$$

$$\tau_1 : 7 \mapsto 4$$

The bijections for the two other constraint nodes are given similarly. Assume that the word $(0, 0, 0, 0, 0, 0, 1)$ is received over a BSC. Notice that $\mathcal{C}_1 = \mathcal{C}_2 = \mathcal{C}_3 = \mathcal{C}_{pc}$, where $\mathcal{C}_{pc}$ has length 4. Since $\mathcal{C}_{pc}$ in this case has dimension 3, there are 8 codewords in each local code. Denote these codewords by $v^{(1)}, v^{(2)}, \ldots, v^{(8)}$. Then, the linear program for the [7,4,3] Hamming code in Figure 4.1 is given by

$$\min y_1 + y_2 + y_3 + y_4 + y_5 + y_6 - y_7$$

such that

$$I(v^{(1)}, 1) + I(v^{(2)}, 1) + \cdots + I(v^{(8)}, 1) = 1$$

$$I(v^{(1)}, 2) + I(v^{(2)}, 2) + \cdots + I(v^{(8)}, 2) = 1$$

$$I(v^{(1)}, 3) + I(v^{(2)}, 3) + \cdots + I(v^{(8)}, 3) = 1$$

and

$$y_1 = v_1^{(1)} I(v^{(1)}, 1) + v_1^{(2)} I(v^{(2)}, 1) + \cdots + v_1^{(8)} I(v^{(8)}, 1)$$

$$y_3 = v_2^{(1)} I(v^{(1)}, 1) + v_2^{(2)} I(v^{(2)}, 1) + \cdots + v_2^{(8)} I(v^{(8)}, 1)$$

$$y_4 = v_3^{(1)} I(v^{(1)}, 1) + v_3^{(2)} I(v^{(2)}, 1) + \cdots + v_3^{(8)} I(v^{(8)}, 1)$$

$$y_7 = v_4^{(1)} I(v^{(1)}, 1) + v_4^{(2)} I(v^{(2)}, 1) + \cdots + v_4^{(8)} I(v^{(8)}, 1)$$

$$y_1 = v_1^{(1)} I(v^{(1)}, 2) + v_1^{(2)} I(v^{(2)}, 2) + \cdots + v_1^{(8)} I(v^{(8)}, 2)$$

$$y_2 = v_2^{(1)} I(v^{(1)}, 2) + v_2^{(2)} I(v^{(2)}, 2) + \cdots + v_2^{(8)} I(v^{(8)}, 2)$$

$$y_4 = v_3^{(1)} I(v^{(1)}, 2) + v_3^{(2)} I(v^{(2)}, 2) + \cdots + v_3^{(8)} I(v^{(8)}, 2)$$

$$y_6 = v_4^{(1)} I(v^{(1)}, 2) + v_4^{(2)} I(v^{(2)}, 2) + \cdots + v_4^{(8)} I(v^{(8)}, 2)$$

$$y_1 = v_1^{(1)} I(v^{(1)}, 3) + v_1^{(2)} I(v^{(2)}, 3) + \cdots + v_1^{(8)} I(v^{(8)}, 3)$$

$$y_2 = v_2^{(1)} I(v^{(1)}, 3) + v_2^{(2)} I(v^{(2)}, 3) + \cdots + v_2^{(8)} I(v^{(8)}, 3)$$

$$y_3 = v_3^{(1)} I(v^{(1)}, 3) + v_3^{(2)} I(v^{(2)}, 3) + \cdots + v_3^{(8)} I(v^{(8)}, 3)$$

$$y_5 = v_4^{(1)} I(v^{(1)}, 3) + v_4^{(2)} I(v^{(2)}, 3) + \cdots + v_4^{(8)} I(v^{(8)}, 3)$$

where

$$0 \leq I(v, j) \leq 1 \text{ for each } j \in R \text{ and each } v \in \mathcal{C}_j.$$

Any solver for this linear program gives an LP decoder. However, note that the solution to the relaxed linear program could have fractional solutions. In particular, constraints (4.6), (4.7), and (4.8) imply that $0 \leq y_i \leq 1$. If at least one y_i $(1 \leq i \leq n)$ takes a fractional value, then the decoder returns "failure." However, if the solution to the relaxed linear program is integral, then the resulting codeword is guaranteed to be the maximum-likelihood solution. Feldman et al. [FWK05] called this property the "ML Certificate Property" and proved that the probability of decoding error is independent of the transmitted codeword.

As mentioned, the formulation given above is slightly more general than the formulation given in [FWK05] since it can accommodate any binary code on the constraints, not just the parity-check code.

4.1.3 Message-passing Decoding

Given a DMC, let $\mathcal{O}$ denote the set of possible channel outputs (corresponding to the set of possible inputs to the decoder), and let $\mathcal{M}$ denote the set of possible messages which the variable and check nodes of a Tanner graph can pass to each other. Message-passing decoders typically assume $\mathcal{O} \subseteq \mathcal{M}$. In the description that follows, assume that the underlying Tanner graph is (c, d)-biregular.

We make use of the following diagram in which the node on the left represents a variable node, and the node on the right represents a check node. This particular diagram represents a $(3, 6)$-regular graph, but the concepts generalize to any (c, d)-biregular graph. Initially, the representative variable node receives message r, and the probability that this message is in error is p_0.

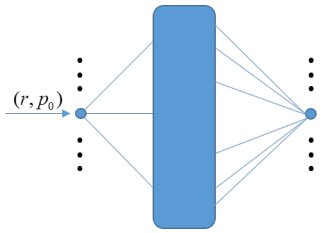


Figure 4.2: Message-passing: Initialization

Typically, in the first step (round 0) of a message-passing decoder, the variable nodes forward their received values to the neighboring constraint nodes. Denote this step by $\Phi_v^{(0)} : \mathcal{O} \to \mathcal{M}$. This stage is represented pictorially in the following figure:

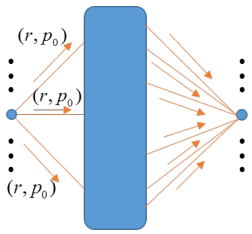


Figure 4.3: Message-passing: Initial Forwarding

In each subsequent round $\ell > 0$, every constraint node processes the messages received from the variable nodes and sends the processed messages back to the neighboring variable nodes. For the message-passing algorithms considered in [RU01], there is an important restriction. The message sent from constraint node c_i to variable node v_j cannot depend on the message sent from node v_j to c_i in the previous round. Denote the processing at the constraint nodes of the incoming messages from the variable nodes at round ℓ by $\Phi_c^{(\ell)} : \mathcal{M}^{d-1} \to \mathcal{M}$. This stage is represented pictorially below for the first edge from the representative constraint node. The blue arrows denote messages received at the previous step, and the orange arrows denote messages sent at the current step. In addition, q_ℓ denotes the probability that the sent message y is in error.

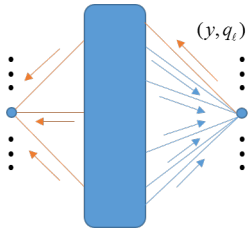


Figure 4.4: Message-passing: Constraint Nodes to Variable Nodes

Similarly, every variable node processes the messages received from the constraint nodes (as well as the message received initially) and sends the resulting messages back to the neighboring constraint nodes. Denote the processing of the incoming nodes at round ℓ

by $\Phi_v^{(\ell)} : \mathcal{O} \times \mathcal{M}^{c-1} \to \mathcal{M}$. Following the same color convention as before, this stage is represented pictorially below for the first edge from the variable node. The probability p_ℓ denotes the probability that the sent message x is in error.

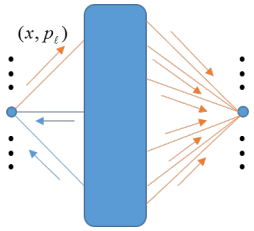


Figure 4.5: Message-passing: Variable Nodes to Constraint Nodes

Density evolution is the process of determining the maximum p_0 (threshold probability of error) so that $p_\ell \to 0$ as $\ell \to \infty$ (preferably, this convergence is quite fast). While we will consider several thresholds for message passing algorithms, we do not explore this direction in detail in this chapter.

4.2 Girth-based Analysis

Though Gallager's thesis did not mention the explicit correspondence between the parity-check code and the corresponding graph, his arguments implicitly relied on the girth of the underlying Tanner graph (see, for example, Theorem 2.5 in [Gal63]). The importance of girth lies in the fact that locally, up to the girth of the graph, the neighborhood of each vertex is a tree. More precisely, given a vertex v in the graph, let N_v^ℓ denote the induced subgraph of depth ℓ obtained from all paths of length ℓ starting from (and including) vertex v. Given an edge $e = (u, v)$ of the graph, let $N_e^\ell := N_u^\ell \cup N_v^\ell$. (Similarly, let $N_{\vec{e}}^\ell$ denote the induced subgraph from directed paths $\vec{e_1}, \ldots, \vec{e_\ell}$, where $\vec{e_1} \neq \vec{e}$.) If $\ell \leq girth(G)$, then N_e^ℓ is a tree. Based on this observation, Gallager's arguments (and subsequent arguments based on girth) used a tree-based analysis to obtain bounds on the number of random errors

the code could correct with high probability over various channels. In particular, Gallager provided an algorithm for which a rate 1/2 code over a BSC could correct a fraction of approximately 0.07 errors with high probability. He also introduced a simple hard-decision decoder but gave no guarantees for how many errors his hard-decision decoder could correct. The main limitation of his analysis lay in the short cycles of the underlying Tanner graph. Consequently, he implicitly attempted to find codes whose Tanner graphs had the largest possible girth. Margulis [Mar82] explicitly constructed Cayley graphs with large girth and then substituted these graphs into Gallager's analysis. His results were similar to Gallager's, but he achieved lower computational complexity costs for determining entries in the parity-check matrix.

In addition to introducing Tanner graphs, Tanner [Tan81] provided lower bounds on the rate and minimum distance of codes on graphs. In addition, he introduced a hard-decision decoder as well as a soft-decision decoder having parallel computational complexity $O(\log n)$. Unfortunately, as in Gallager's analysis, Tanner's analysis was limited by the girth of the underlying graph. In fact, as in [CNVM10], girth-based analysis generally shows that the error-correction capabilities grow exponentially in the girth of the graph. However, because the girth of the graph is only logarithmic, such results do not guarantee the correction of a constant fraction of errors in the code length. A natural question then is to investigate the performance of codes with no cycles. Unfortunately, Etzion et al. [ETV99] showed that such codes have very small minimum distance. In fact, for codes with rate greater than or equal to 1/2, the minimum distance of codes with cycle-free Tanner graphs can be at most 2. For rate $r < 1/2$, the minimum distance of codes with cycle-free Tanner graphs can be at most $2/r$.

Instead of selecting a graph with a specified girth, Richardson and Urbanke [RU01] considered graphs chosen uniformly at random from the set of all (c, d)-biregular graphs. In this way, they were able to simplify Gallager's analysis [Gal63] and give an improved bound

on the fraction of errors that their belief-propagation decoder was likely to correct. The key tool in their simplification was a proof that on a random graph with a random channel output, after a fixed number of iterations ℓ, the number of incorrect messages passed during round ℓ of a message-passing decoder converges in probability to the average number of incorrect messages passed during round ℓ of a message-passing decoder on a cycle-free graph. Consequently, for a fixed number of iterations ℓ, they were able to perform their analysis assuming no cycles of length 2ℓ. The errors are assumed to be made uniformly at random.

Theorem 37 (Theorem 2, [RU01])**.** *Suppose the underlying (c, d)-biregular Tanner graph of an LDPC code is chosen uniformly at random, and suppose the errors are introduced uniformly at random. Let Z be a random variable which denotes the number of incorrect variable-to-check node messages passed at iteration ℓ of a message-passing decoder, and let p denote the average number of errors passed during iteration ℓ along a single edge $\vec{e}$ of a graph for which $\mathcal{N}_{\vec{e}}^{2\ell}$ is a tree. Then, there exist positive constants $\beta = \beta(c, d, \ell)$ and $\gamma = \gamma(c, d, \ell)$ such that the following hold:*

- *Concentration around the expected value: For any $\epsilon > 0$,*

$$Pr\left\{|Z - E[Z]| > cn\epsilon/2\right\} \leq 2e^{-\beta\epsilon^2 n}.$$

- *Convergence to the cycle-free case: For any $\epsilon > 0$ and $n > \frac{2\gamma}{\epsilon}$,*

$$|E[Z] - ncp| < cn\epsilon/2.$$

- *Concentration around the cycle-free case: For any $\epsilon > 0$ and $n > \frac{2\gamma}{\epsilon}$,*

$$Pr\left\{|Z - ncp| > cn\epsilon\right\} \leq 2e^{-\beta\epsilon^2 n}.$$

Concentration around the expected value guarantees that as $n \to \infty$, the actual number of error messages passed at round ℓ of a message-passing decoder converges in probability to the average number of error messages passed at round ℓ. Convergence to the cycle-free case guarantees that the average number of incorrect messages passed at a particular round ℓ of a message-passing decoder on a randomly chosen graph is close to the average number of incorrect messages passed at round ℓ of a message-passing decoder on a cycle-free graph. Concentration around the cycle-free case guarantees that the actual number of incorrect messages passed at a particular round ℓ of a message-passing decoder on a randomly chosen graph converges in probability to the average number of incorrect messages passed at iteration ℓ on a cycle-free graph. Note that in the proof, concentration around the cycle-free case results from combining concentration around the expected value and convergence to the cycle-free case.

By performing their analysis on cycle-free graphs for a given number of ℓ iterations, then invoking Theorem 37 to show that their cycle-free assumption was asymptotically valid with high probability, Richardson and Urbanke [RU01] improved on Gallager's bounds for various channels. As a particular example, for a rate 1/2 code over the BSC channel, they showed that their belief-propagation decoder could correct a fraction of 0.084 errors with high probability. By using a similar analysis over irregular graphs (instead of (c, d)-biregular graphs), Richardson, Shokrollahi, and Urbanke [RSU01] demonstrated a degree distribution and decoding algorithm which could correct a fraction of 0.106 errors with high probability. Note that due to the analysis techniques used, all of these results are asymptotic in the length of the code.

Recently, Arora et al. [ADS12] used a girth-based analysis for LP decoding to prove the following theorem:

Theorem 38 (Theorem 1 in [ADS12])**.** *Consider the BSC channel with transition probability p and an LDPC code of length n defined over a (c, d)-biregular graph with girth $\Omega(\log n)$. Let*

$x \in \{0,1\}^n$ be the transmitted codeword and $y \in \{0,1\}^n$ be the received word. Suppose c, d, and p satisfy the following conditions:

- $p < 1 - 2^{-\frac{1}{d-1}}$
- $\sqrt{p}(1-(1-p)^{d-1})^{\frac{c-2}{2}}(1-p)^{\frac{(d-1)(c-2)}{2}+\frac{1}{2}} < \frac{1}{(d-1)\cdot 2^{c-1}}$.

Then, for some constant $\gamma > 0$, the following two properties hold with probability at least $1 - \exp(-n^\gamma)$:

- *The codeword x is the unique optimal solution to the LP described in* (4.6)-(4.8)*; and*
- *A simple message-passing (dynamic programming) algorithm running in time $O(n \cdot \log^2 n)$ can find x and certify that it is the nearest codeword to y.*

Recall from Corollary 36 that the rate of an LDPC defined on a $(3, 6)$-regular graph has rate at least $1/2$. Also, by Theorem 38, if we choose $c = 3$ and $d = 6$, we can take $p \leq 0.05$.

While this bound is numerically weaker than the bound given by Richardson and Urbanke [RU01], it assumes a fixed graph with a girth requirement which can be checked in polynomial time. In contrast, Richardson and Urbanke's bound assumes a graph chosen uniformly at random with no requirements made on the graph. Consequently, in this sense, the bound in Arora et. al [ADS12] is stronger than Richardson and Urbanke's bound in [RU01]. We summarize the results discussed so far for a rate-1/2 code over the BSC channel in the following table, where p denotes the channel error probability:

	Decoding Strategy	Error Type	Graph Type	p
Capacity limit	n/a	random	n/a	0.11
Richardson et al. [RSU01]	BP Decoding	random	random, irregular	0.106
Richardson and Urbanke [RU01]	BP Decoding	random	random, regular	0.084
Gallager [Gal63]	BP Decoding	random	fixed, regular	0.07
Arora et al. [ADS12]	LP/Min-Sum Decoding	random	fixed, regular	0.05

Table 4.1: Errors Corrected w.h.p. by Rate-1/2 Codes Over BSC

Despite the success of girth-based analysis, it leaves several questions unanswered. In particular, in the presence of cycles, it cannot guarantee the correction of *all* error patterns with no more than αn errors (where n denotes the length of the code and $\alpha > 0$). It can only guarantee with high probability the correction of errors chosen independently and at random with probability p, where p is less than a specified threshold. In fact, there could exist patterns of only a few errors which such analysis cannot guarantee will be corrected. However, such patterns will occur with very small probability. Moreover, girth-based analysis is primarily geared toward asymptotic analysis, not finite-length analysis (though the authors in [RU01] note that the actual convergence is faster than that suggested by the analysis). To partially address these points, there is a second type of analysis of LDPC codes based on the expansion of the underlying graph. In particular, expansion-based arguments will be able to guarantee the correction of a linear fraction of errors in the length of the code in linear time regardless of the error pattern (though this fraction is typically quite small).

4.3 Spectral Analysis

4.3.1 Sipser and Spielman [SS96] Decoding

Recall that one of the primary tools used to study the expansion of a graph G is the graph's second-largest eigenvalue, $\lambda(G)$. We will call analysis of expander codes based on $\lambda(G)$ the "spectral approach." By basing the generator matrix of a code on an expander graph with good spectral properties, Alon et al. [ABN$^+$92] obtained codes with good minimum distance properties. However, they were unable to produce a polynomial-time algorithm for the codes which they constructed.

By instead basing the parity-check matrix on graphs with good spectral properties, Sipser and Spielman [SS94, SS96] provided a simple algorithm which could correct any pattern of up to αn errors (where n denotes the length of the code and $\alpha > 0$) in $O(n)$ time (or $O(\log n)$ time on n parallel processors). Spielman [Spi96] also provided a linear-time encoder for codes whose parity-check matrix was based on a good expander graph; however, the rate was lower than in [SS96]. As noted before, guaranteeing the correction of up to a constant fraction of errors in linear time by a code which has fixed rate and is allowed to grow arbitrarily long has not been accomplished using girth-based arguments alone. We pause to note that empirically, expander codes with Hamming codes on the constraints perform quite well over the AWGN channel (coming within 0.47 dB of capacity) [DMT04]; however, we will not discuss these empirical studies further here.

Though the new result which we present later in this chapter is not based on spectral arguments, the proof of our result will follow somewhat closely to the spectral arguments presented in [SS96]. Consequently, we present them in some detail. Given a d-regular graph $G = (V, E)$, Sipser and Spielman [SS96] placed the codewords on the edges of the graph and placed the constraints on the vertices. Given an edge $e = (u, v)$ in the graph G, let $N(e) := \{u, v\}$. Similarly, let $E(v) := \{(u, v) : u \in V\}$. Order the edges. Then, given a

vector $\mathbf{c} = (c_e)_{e \in E}$, define $\mathbf{c}|_{E(v)} := (c_e)_{e \in E(v)}$. Let C_d be a linear code of length d over $\mathbb{F}_q$. A vector $\mathbf{c} = (c_e)_{e \in E}$ in $\mathbb{F}_q^n$, where n is the number of edges in E, is a codeword if and only if

$$\mathbf{c}|_{E(v)} \in C_d \text{ for each } v \in V. \tag{4.9}$$

We now describe a generalized bit-flipping strategy for decoding such a code. In the discussion that follows, given a vector $\mathbf{w}$, we will say that constraint vertex $v \in V$ is *unsatisfied* if $\mathbf{w}|_{E(v)} \notin C_d$.

Generalized Bit-flipping

Input:

A d-regular graph $G = (V, E)$ with $|E| = n$.

A binary $(d, r_0 d, \delta_0 d)$ code C_d with relative minimum distance δ_0 and rate r_0.

A received word $\mathbf{w} = (w_e)_{e \in E} \in \mathbb{F}_q^n$.

A constant $0 < \theta < 1/3$.

τ, τ': Constants to be determined.

Initialize:

Set $h := \theta \delta_0 d$.

Set $\mathbf{z} := \mathbf{w}$.

Repeat $\log_\tau(\tau' n)$ times:

Local decoding: For each $v \in V$, find a codeword $\mathbf{c}^{(v)} \in C_d$ with

$d(\mathbf{z}|_{E(v)}, \mathbf{c}^{(v)}) \leq h$ if it exists.

If $\mathbf{z}|_{E(v)} = \mathbf{c}^{(v)}$ for each $v \in V$, then output $\mathbf{z}$.

Update: For each $e \in E$, if there is a $v \in N(e)$ such that $\mathbf{c}^{(v)}$ exists,

set $\mathbf{z}|_{E(v)} = \mathbf{c}^{(v)}$.

Output "failure."

Note that the algorithm presented above allows slightly more flexibility than the algorithm

described in the original paper [SS96] in which θ was set to the fixed constant $1/4$. In addition, the above algorithm extends the algorithm in [SS96] to codes over $\mathbb{F}_q$.

Remark 3. Note that because $\theta < 1/3$, a codeword with $d(N(j), C_d) \leq h$ will be the nearest codeword.

Theorem 39 ([SS96])**.** *Let G be a d-regular graph, and let $C_d \subset \mathbb{F}_q^d$ be a code with rate r_0 and relative minimum distance δ_0. Then, the code described in* (4.9) *has the following properties:*

- *Rate:* $r \geq 2r_0 - 1$
- *Relative minimum distance:*
$$\delta \geq \left(\frac{\delta_0 - \frac{\lambda(G)}{d}}{1 - \frac{\lambda(G)}{d}} \right)^2.$$

Proof. We have already given a proof that $r \geq 2r_0 - 1$ in Theorem 35. We now present the proof of the bound on the relative minimum distance. Let m be the number of vertices in the graph G. For convenience, define

$$\phi(x) = \frac{d \cdot x^2}{2m} + \lambda(G) \left(\frac{x}{2} - \frac{x^2}{2m} \right).$$

Let $\mathbf{c}$ be any non-zero codeword $\mathbf{c} = (c_e)_{e \in E} \in \mathbb{F}_q^n$ with support $S \subseteq E$, where

$$S := \{e \in E : c_e \neq 0\}.$$

Suppose $|S| = \phi(t)$, where t is a positive real number, and let V denote the set of vertices in G incident with the edges in S. Note that S is certainly contained in $E(V, V)$, and by the Expander Mixing Lemma in its original form (Lemma 2.3 in [AC88]),

$$|E(V, V)| \leq \phi(|V|). \tag{4.10}$$

Now, combining (4.10), and the fact that $\phi(x)$ is strictly increasing for $x \geq 0$ when $\lambda(G) < d$, we see that if $|V| < t$, then

$$|S| \leq |E(V,V)| \leq \phi(|V|) < \phi(t) = |S|,$$

which gives a clear contradiction. So, $|V| \geq t$. Since each edge in S is incident with exactly two vertices, for a support set S of size $\phi(t)$, the average number of edges per vertex is at most $2\phi(t)/t$. Consequently, there is at least one vertex which is incident with at most $2\phi(t)/t$ edges in S. Since the minimum distance of C_d is $\delta_0 d$, it must be the case that

$$\frac{2\phi(t)}{t} \geq \delta_0 d. \tag{4.11}$$

Let $t = \gamma m$ for some $\gamma \leq 1$. Then, (4.11) can be written as

$$\left(\gamma + \frac{\lambda(G)}{d}(1-\gamma)\right) \geq \delta_0.$$

This inequality is satisfied when

$$\gamma \geq (\delta_0 - \lambda(G)/d)/(1 - \lambda(G)/d).$$

Note that

$$|S| = \phi(t) = \frac{md}{2}\left(\gamma^2 + \frac{\lambda(G)}{d}(\gamma - \gamma^2)\right) \geq \frac{md}{2}\gamma^2$$

since $\gamma \leq 1$. So, the relative weight $|S|/(md/2)$ of the codeword $\mathbf{c}$ must be at least

$$((\delta_0 - \lambda(G)/d)/(1 - \lambda(G)/d))^2$$

as desired. □

Theorem 40 ([SS96])**.** *The fraction of errors which the Generalized Bit-flipping algorithm can correct approaches* $\theta^2\delta_0^2\left(\frac{1-3\theta}{1-\theta}\right)$ *as* $d \to \infty$.

Proof. Let $\mathcal{E}$ denote the set of errors before a decoding round, and let $\mathcal{E}'$ denote the set of errors after a decoding round. Suppose there are n edges and m vertices in the graph G. (Note that $n = md/2$.) At each decoding round, each constraint on a vertex which is incident with at least one error symbol on an edge could be in one of three categories:

- The constraint vertex is incident with fewer than $\theta\delta_0 d$ errors. Such constraints will decode correctly. Following the notation in [SS96], we call these constraints helpful. Denote the set of helpful constraints by H.
- The constraint vertex is incident with more than $\theta\delta_0 d$ errors but fewer than $(1-\theta)\delta_0 d$ errors. Such constraints will not decode. We call these constraints unhelpful. Denote the set of unhelpful constraints by U.
- The constraint vertex is incident with more than $(1-\theta)\delta_0 d$ errors. Such constraints could decode incorrectly. Call these constraints confused. Denote the set of confused constraints by C.

The error edge set is incident with $2|\mathcal{E}|$ constraint vertices (which may not be distinct), so since each constraint node in U is incident with at least $\theta\delta_0 d$ error edges,

$$|U| \leq \frac{2|\mathcal{E}|}{\theta\delta_0 d}. \tag{4.12}$$

Similarly,

$$|C| \leq \frac{2|\mathcal{E}|}{(1-\theta)\delta_0 d}. \tag{4.13}$$

Note that if the edge corresponding to an error symbol is incident with two unhelpful constraint vertices, then the symbol will remain in error. The Expander Mixing Lemma

(truncated from in its original form in Lemma 2.3 in [AC88]) combined with (4.12) and (4.13) gives an upper bound on the number of errors after one decoding round:

$$|\mathcal{E}'| \leq \theta\delta_0 d|C| + \frac{2|\mathcal{E}|^2}{\theta^2\delta_0^2 dm} + \frac{\lambda(G)|\mathcal{E}|}{\theta\delta_0 d} \leq \frac{2\theta|\mathcal{E}|}{1-\theta} + \frac{2|\mathcal{E}|^2}{\theta^2\delta_0^2 dm} + \frac{\lambda(G)|\mathcal{E}|}{\theta\delta_0 d}. \tag{4.14}$$

Let $|\mathcal{E}| = \alpha n$. Then, write (4.14) as

$$|\mathcal{E}'| \leq \alpha n \left(\frac{2\theta}{1-\theta} + \frac{\alpha}{\theta^2\delta_0^2} + \frac{\lambda(G)}{\theta\delta_0 d} \right).$$

If

$$\left(\frac{2\theta}{1-\theta} + \frac{\alpha}{\theta^2\delta_0^2} + \frac{\lambda(G)}{\theta\delta_0 d} \right) < 1,$$

then the fraction of bits which are in error decreases after one decoding round. It is possible to argue similarly for each subsequent decoding round. Note that

$$\left(\frac{2\theta}{1-\theta} + \frac{\alpha}{\theta^2\delta_0^2} + \frac{\lambda(G)}{\theta\delta_0 d} \right) < 1$$

if and only if

$$\alpha < \theta^2\delta_0^2 \left(\frac{1-3\theta}{1-\theta} - \frac{\lambda(G)}{\theta\delta_0 d} \right).$$

If G is an expander graph, then $\theta^2\delta_0^2 \left(\frac{1-3\theta}{1-\theta} - \frac{\lambda(G)}{\theta\delta_0 d} \right) \to \theta^2\delta_0^2 \left(\frac{1-3\theta}{1-\theta} \right)$ as $d \to \infty$. Consequently, asymptotically in d, if the fraction of errors is smaller than $\theta^2\delta_0^2 \left(\frac{1-3\theta}{1-\theta} \right)$, the Generalized Bit-flipping algorithm will correct all of the errors. □

From this proof, we see that in the Generalized Bit-flipping Algorithm,

$$\tau = \left(\frac{2\theta}{1-\theta} + \frac{\alpha}{\theta^2\delta_0^2} + \frac{\lambda(G)}{\theta\delta_0 d} \right)^{-1}, \tau' = \theta^2\delta_0^2 \left(\frac{1-3\theta}{1-\theta} - \frac{\lambda(G)}{\theta\delta_0 d} \right).$$

Lemma 41. *The Generalized Bit-flipping algorithm runs in linear time.*

Proof. Let $\mathcal{E}_k$ denote the set of errors after decoding round k, where $k \geq 1$ and $\mathcal{E}_0 = \mathcal{E}$. From the proof of Theorem 40,

$$|\mathcal{E}_k| \leq \epsilon |\mathcal{E}_{k+1}|,$$

where

$$\epsilon = \left(\frac{2\theta}{1-\theta} + \frac{\alpha}{\theta^2 \delta_0^2} + \frac{\lambda(G)}{\theta \delta_0 d} \right) < 1.$$

During the first step of the decoding algorithm, the decoder must check the code on each of the m constraint vertices. However, at each subsequent step the decoder only needs to decode the codes on the constraint vertices which are incident with at least one edge whose symbol was altered. Because the number of errors at each step decreases, the maximum number of bits which have changed after decoding round k is $2|\mathcal{E}_{k-1}|$. It follows that at most $2c|\mathcal{E}_{k-1}|$ decoding operations are required at each step with the exception of the first step in which m decoding operations are required. Consequently, the maximum number of decoding operations during the course of the algorithm is

$$m + 2c \sum_{k=0}^{\infty} |\mathcal{E}_k| = m + 2c \sum_{k=0}^{\infty} \epsilon^k |\mathcal{E}_0| \leq m + 2c \frac{1}{1-\epsilon} |\mathcal{E}|.$$

Since $|\mathcal{E}| = \alpha n$ for some constant $\alpha < 1$, and since each decoding operation takes constant time, the decoding algorithm runs in time linear in the length of the code. □

The fraction $\theta^2 \delta_0^2 \left(\frac{1-3\theta}{1-\theta} \right)$ is maximized at approximately $0.02129 \delta_0^2$ when $\theta \approx 0.2324$. This is an extremely negligible improvement over Sipser and Spielman's guarantee of $\delta_0^2/48 \approx 0.02083 \delta_0^2$ when $\theta = 1/4$. Consequently, it is clear that a different method of analysis or a different decoding algorithm is necessary to improve on the result given above. Note that as $d \to \infty$, the lower bound on the minimum distance is δ_0^2, so the fraction of errors corrected is only 1/24 of what is possible with minimum-distance decoding. Moreover, it is not hard to show that the right-degree of the underlying codes must be at least 9,200

to guarantee correction of even a positive fraction of error. Despite these drawbacks, the theorem guarantees the correction of a linear fraction of errors in the length of the code in linear time, and the underlying graph can be given explicitly. However, there is clearly room for substantial improvement.

4.3.2 Zémor [Zem01] Decoding

Note that a code defined on a d-regular graph G can be naturally extended to a bipartite graph $(L \cup R, E)$ in two ways. First, one can associate each vertex in L with an edge of G and each vertex in R with a vertex of G. There is an edge (e, v) between L and R if and only if the edge e is incident with vertex v. The resulting bipartite graph is called the edge-vertex incidence graph. All of the results given in the theorems of the previous section can be described in terms of an edge-vertex incidence graph. Second, one can associate L with the set of vertices, and associate R with a copy of the set of vertices. There is an edge (u, v) between L and R if and only if there is an edge (u, v) in the original graph. The resulting graph is called the double-cover of the original d-regular graph and has twice as many edges and vertices as the original d-regular graph.

Instead of using the edge-vertex incidence graph, Zémor [Zem01] used the double-cover of a graph to define his codes. Because each edge in the double-cover is incident with only one vertex in L (and R respectively), he could perform minimum-distance decoding at each of the constraint nodes without conflict. In addition, he modified Sipser and Spielman's analysis and showed that asymptotically in d, his algorithm could correct a $\delta_0^2/4$ fraction of errors in linear time. For his algorithm and analysis, the length of the underlying codes must be at least 574 to guarantee the correction of a positive fraction of error.

Let $G' = (L \cup R, E)$ denote the double-cover of a d-regular graph $G = (V, E)$. Then, given a code C_d of length d, define the code $H(G', C_d) := \left\{ \mathbf{z} : \mathbf{z}|_{E(v)} \in C_d \text{ for all } v \in L \cup R \right\}$. The decoding algorithm for $H(G', C_d)$ given in [Zem01] is outlined below. (Note that while

Zémor defined binary codes, we can easily extend the construction to define q-ary codes.)

Zémor's Decoding Algorithm

Input:

The double-cover $H = (L \cup R, E')$ of a d-regular graph $G = (V, E)$ with

$|E| = n/2$ and $|E'| = n$.

A binary $(d, r_0 d, \delta_0 d)$ code C_d with relative minimum distance δ_0 and rate r_0.

A received word $\mathbf{w} = (w_e)_{e \in E'} \in \mathbb{F}_q^n$.

τ, τ': Constants to be determined.

Initialize:

Set $\mathbf{z} := \mathbf{w}$.

Repeat $\log_\tau(\tau' n)$ times:

Left-hand side decoding:

Local decoding: For each $v \in L$, find a codeword $\mathbf{c}^{(v)} \in C_d$ closest to $\mathbf{z}|_{E(v)}$.

Update: Set $\mathbf{z}|_{E(v)} = \mathbf{c}^{(v)}$ for each $v \in L$.

Right-hand side decoding:

Local decoding: For each $v \in R$, find a codeword $\mathbf{c}^{(v)} \in C_d$ closest to $\mathbf{z}|_{E(v)}$.

Check: If $\mathbf{z}|_{E(v)} = \mathbf{c}^{(v)}$ for each $v \in R$, then output $\mathbf{z}$.

Update: Set $\mathbf{z}|_{E(v)} = \mathbf{c}^{(v)}$ for each $v \in R$.

Output "failure."

The algorithm fails if some of the constraint nodes $j \in R$ are still unsatisfied after $\log_\tau(\tau' n)$ steps of the algorithm. If the algorithm succeeds, all constraint nodes in $L \cup R$ will be satisfied.

Theorem 42 ([Zem01])**.** *Let $\mathcal{E}$ denote the initial set of errors and let $\lambda(G)$ denote the second-largest eigenvalue of the underlying Tanner graph. Suppose $|\mathcal{E}| < \frac{\delta_0}{2}\left(\frac{\delta_0}{2} - \frac{\lambda(G)}{d}\right)n$. Then, if $\delta_0 d \geq 3\lambda(G)$, Zémor's Decoding Algorithm will decode to the correct codeword.*

Like the Generalized Bit-flipping algorithm, Zémor's Decoding algorithm clearly runs in parallel in logarithmic time, and a more careful analysis reveals that it can be implemented in linear time. Moreover, from Zémor's analysis, it is easy to deduce that $\tau = \frac{1}{2-\theta}$ and $\tau' = \theta\frac{\delta_0}{2}\left(\frac{\delta_0}{2} - \frac{\lambda(G)}{d}\right)$ for a given, arbitrary $\theta < 1$. (The results given in the theorem are for $\theta \approx 1$.) Finally, Zémor's argument only held for (d, d)-regular double-cover graphs. However, Janwa and Lal [JL03] generalized Zémor's decoding algorithm and analysis to (c, d)-biregular graphs.

4.3.3 Additional Results using Spectral Analysis

By exploiting the close relationship between concatenated codes and GLDPC codes, Roth and Skachek [SR03] increased the fraction of correctable errors to $\delta_0^2/2$ asymptotically in d. They used a decoding algorithm which was a variant of Forney's generalized minimum distance (GMD) decoding algorithm for concatenated codes, and their analysis requires the right degree to be greater than 254 to correct a positive fraction of error. Roth and Skachek's arguments in [SR03] also hold for (d, d)-regular graphs. Just as Janwa and Lal [JL03] generalized Zémor's argument for (c, d)-biregular graphs, Kim [Kim05] generalized Roth and Skachek's arguments to (c, d)-biregular graphs. The results in [SS96], [Zem01], and [SR03] are summarized in the following table:

	Errors Corrected	Required Degree
Sipser and Spielman 1996	$(\delta_0^2/48 - (\delta_0\sqrt{d-1})/2d)n$	$d > 9,200$
Zémor 2001	$(\delta_0^2/4 - \delta_0\sqrt{d-1}/d)n$	$d > 574$
Roth and Skachek 2006	$(\delta_0^2/2 - 2\delta_0\sqrt{d-1}/d)n$	$d > 254$

In a series of papers, Barg and Zémor [BZ02, BZ04, BZ05a, BZ06] explored the

relationships between expander codes and concatenated codes. They showed that GLDPC codes whose underlying graphs are expander graphs can essentially match the performance of concatenated codes with linear-time decoding instead of the quadratic-time decoding inherent in Forney's original concatenated codes. In particular, they showed that their codes attain the Zyablov bound (which is asymptotically the best rate-distance tradeoff for concatenated codes with a singly-extended GRS code as the outer code and a code meeting the GV bound as the inner code). For more details, see [Rot06]. Moreover, they showed that their codes achieve capacity for the BSC, and the error exponent matches that of Forney's concatenated codes. Following a construction of Blokh and Zyablov [BZ82], Barg and Zémor also constructed multilevel expander codes which achieved the Blokh-Zyablov bound (which surpasses the Zyablov bound) [BZ05b]. Justesen and Høholdt [JH04] also studied the relationship between concatenated codes and expander codes, and they showed that for their construction, the codes must be very long (on the order of 10^7 to 10^9) to have an advantage over concatenated codes. Later, Frolov and Zyablov [FZ11] computed an upper bound on the rate-minimum distance trade-off of the types of codes described in [BZ05a] which was slightly below the GV bound. While the decoding complexity of these types of codes is linear in the length of the code, Skachek and Ashikhmin [AS05, AS06] noted that it depends exponentially on $1/\epsilon^2$, where the rate $R = (1 - \epsilon)C$, and where C denotes the capacity of the given BSC channel. They introduced a scheme for constructing capacity-achieving codes with decoding complexity polynomial in $1/\epsilon$, though they were limited by the codes available to them in explicitly constructing such codes.

Independently, by combining ideas from [ABN$^+$92], [AEL95], and [AL96], Guruswami and Indyk [GI02, GI05] also gave codes which achieved the Zyablov bound and Forney exponent. In particular, they constructed codes which can be made arbitrarily close to the Singleton bound. They then noted that it is possible to concatenate these codes with good inner codes to attain the Zyablov bound (or Blokh-Zyablov bound if multilevel concate-

nation is used). They also achieved Forney's exponent for concatenated codes. Note that their codes are also linear-time encodable while Barg and Zémor's codes do not have this additional feature. Given a code whose rate-distance tradeoff is $R + \delta - \epsilon \leq 1$ (where R is the rate and δ is the relative minimum distance), their alphabet size depended exponentially on $O(\log(1/\epsilon)/\epsilon^4)$. Roth and Skachek [RS06] improved this alphabet size to depend exponentially on $O(\log(1/\epsilon)/\epsilon^3)$.

In a different direction, Høholdt and Justesen [HJ06] studied codes derived from expander graphs constructed from finite geometries such as those given by Tanner [Tan84]. While these arguments do not scale as the length of the code goes to infinity, they were used to construct a code with length 4641 with rate slightly greater than 0.5 which can correct 52 errors (approximately 0.011 fraction of errors). The authors note that due to the very small probability that a RS codeword in error will be confused with a RS codeword, the code can typically correct a much higher fraction of errors (approximately 0.05). Later Høholdt, Justesen, and Beelen [BHPJ13] exploited the specific geometric structure of the underlying expander graphs to obtain an improved lower bound on the dimension of expander codes. Recently, Høholdt and Justesen [HJ14] studied the problem of finding the largest graph having a particular eigenvalue. In this way, they were able to generalize the minimum distance bounds given by Barg and Zémor in [BZ02] in the non-asymptotic case.

We pause to mention that Kelley et al. [KSR08] implemented a sum-product algorithm on codes defined on the zig-zag product presented in [RVW00]. However, they gave mainly experimental results. Also, though we do not study them here, we mention in passing that quantum expander codes have very recently been explored in [LTZ15].

4.4 Expansion Analysis

Recall that while expander codes based on spectral methods have relatively good error correction guarantees for large degrees and alphabet sizes, for relatively small degrees and alphabet sizes, there are no good (if any) error correction guarantees. However, the following theorem shows that with high probability, a randomly generated (c, d)-biregular graph with relatively small c and d has very good expansion properties (surpassing Kahale's eigenvalue bound given in [Kah95]).

Theorem 43 ([RU08])**.** *Suppose* (γ, α) *satisfies*

$$\frac{c-1}{c} \cdot h_2(\alpha) - \frac{1}{d} \cdot h_2(\alpha\gamma d) - \alpha \cdot \gamma \cdot d \cdot h_2(\frac{1}{\alpha d}) < 0.$$

Then, with high probability, a (c, d) *regular graph chosen at random from the set of all* (c, d)*-biregular graphs will be a* (γ, α) *expander.*

This theorem is not at all intuitive. However, the table below compares the expansion properties for several parameters and illustrates that for a (c, d, γ, α) expander, as α decreases, γ quickly increases (which, as we will see, leads to greater error-correction capabilities of expander codes). In addition, it illustrates that for a fixed α, as the ratio c/d decreases, γ decreases.

(c, d)	$(\gamma, \frac{3}{4})$	$(\gamma, 0.71)$	$(\gamma, \frac{1}{2})$	$(\gamma, \frac{1}{4})$
$(36, 72)$	$(0.000366, \frac{3}{4})$	$(0.000851, 0.71)$	$(0.00915, \frac{1}{2})$	$(0.0410, \frac{1}{4})$
$(5, 70)$	$(1.7 \times 10^{-17}, \frac{3}{4})$	$(4.812 \times 10^{-11}, 0.71)$	$(0.000274, \frac{1}{2})$	$(0.00971, \frac{1}{4})$

Table 4.2: Expansion of Random Graphs

Because of the inverse relationship between γ and α, it is clearly desirable to construct codes

with the smallest possible requirement on α. The same inverse relationship between γ and α holds with the explicitly constructed lossless expanders of [CRVW02] as well. Unlike the results using spectral analysis, very large degrees are not required to guarantee correction of a constant fraction of errors. Moreover, the guarantee on the fraction of errors which can be corrected depends entirely on the expansion of the underlying graph, not on the relative minimum distance of the inner code.

4.4.1 LDPC Codes

4.4.1.1 Bit-flipping Algorithm

In the 1970s, Zyablov and Pinsker [ZP75] showed that with high probability over the choice of a random (c, d)-biregular graph, Gallager's bit-flipping algorithm could correct a constant fraction of errors in the length of the code. While their method shared some similarities with expansion arguments used later, the first explicit analysis of LDPC codes directly in terms of the expansion of the underlying graph without the use of spectral methods was given in the analysis of the bit-flipping strategy described in Sipser and Spielman's paper [SS96]. Unlike the strategies in the previous section, the bit-flipping strategy does not decode an inner code. Instead, each variable node counts the number of neighboring unsatisfied constraints, and variable nodes with more than h unsatisfied constraints (where h is a fixed threshold) are flipped. The bit-flipping algorithm described below allows more flexibility than the strategy in the original paper [SS96] which set $\alpha = 3/4$ and the threshold $h = c/2$. However, the analysis proceeds almost identically. The added flexibility will allow an easier comparison with Viderman's decoding algorithm [Vid13] which is presented later.

Bit-flipping

Input:

A (c, d, γ, α) expander graph $(L \cup R, E)$ with $|L| = n$.

The binary parity-check code C_d.

A received word $\mathbf{w} = (w_i)_{i \in L} \in \{0,1\}^n$.

Constants τ and τ' to be determined later.

Initialize:

Set $\mathbf{z} := \mathbf{w}$.

Set $h := \lceil (2\alpha - 1)c \rceil$.

Repeat $\log_\tau (\tau' n)$ times:

Find the set of unsatisfied constraints: $R_0 := \left\{ j \in R \text{ s.t. } \mathbf{z}|_{N(j)} \notin C_d \right\}$.

If $R_0 = \emptyset$, then output $\mathbf{z}$.

Update: For each $i \in L$

If $|N(i) \cap R_0| \geq h$, flip z_i (i.e. set $z_i := z_i + 1 \bmod 2$).

Output "failure."

Theorem 44 ([SS96])**.** *Let $\mathcal{E}$ denote the initial set of errors. Given a code whose Tanner graph is a (c, d, γ, α) expander graph with $\alpha > 3/4$, if $|\mathcal{E}| < \frac{\alpha c + h - c}{h} \lfloor \gamma n \rfloor$ with $h = \lceil (2\alpha - 1)c \rceil$, then the Bit-flipping algorithm will correct all of the errors after $\log |\mathcal{E}|$ decoding rounds.*

Proof. After one decoding round, let $\mathcal{E}_1$ denote the set of correct bits which become error bits, and let $\mathcal{E}_2$ denote the set of error bits which remain in error. The proof proceeds in two steps. First, to guarantee that expansion arguments can be used at each step of the algorithm, we show that $|\mathcal{E}_1 \cup \mathcal{E}_2| < \lfloor \gamma n \rfloor$. Second, to prove that the algorithm terminates in logarithmic time, we show that $|\mathcal{E}_1 \cup \mathcal{E}_2| \leq \theta |\mathcal{E}|$ for some $\theta < 1$.

Proving that $|\mathcal{E}_1 \cup \mathcal{E}_2| < \lfloor \gamma n \rfloor$ follows by proving the stronger result that $|\mathcal{E}_1 \cup \mathcal{E}| < \lfloor \gamma n \rfloor$. Suppose to the contrary that $|\mathcal{E}_1 \cup \mathcal{E}| > \lfloor \gamma n \rfloor$. Since $|\mathcal{E}| < \lfloor \gamma n \rfloor$, it is possible to select

a set $\mathcal{E}_1' \subseteq \mathcal{E}_1$ such that $|\mathcal{E}_1' \cup \mathcal{E}| = \lfloor \gamma n \rfloor$. Then, by expansion,

$$\alpha c(|\mathcal{E}_1'| + |\mathcal{E}|) \leq |N(\mathcal{E}_1' \cup \mathcal{E})|. \tag{4.15}$$

Each element in $\mathcal{E}_1'$ must have at least h unsatisfied neighbors. Consequently, each vertex in $\mathcal{E}_1'$ has at most $c - h$ satisfied neighbors. Since $R_0 \subseteq N(\mathcal{E})$,

$$|N(\mathcal{E}_1' \cup \mathcal{E})| = |N(\mathcal{E})| + |N(\mathcal{E}_1') \backslash N(\mathcal{E})| \leq c|\mathcal{E}| + (c-h)|\mathcal{E}_1'|. \tag{4.16}$$

Because $|\mathcal{E}| + |\mathcal{E}_1'| = \lfloor \gamma n \rfloor$, combining inequalities (4.15) and (4.16) implies that

$$\alpha c \lfloor \gamma n \rfloor \leq |N(\mathcal{E}_1' \cup \mathcal{E})| \leq c|\mathcal{E}| + (c-h)(\lfloor \gamma n \rfloor - |\mathcal{E}|). \tag{4.17}$$

Rearranging gives

$$\frac{\alpha c + h - c}{h} \lfloor \gamma n \rfloor \leq |\mathcal{E}|$$

which contradicts our original assumption that $|\mathcal{E}| < \frac{\alpha c + h - c}{h} \lfloor \gamma n \rfloor$. So, $|\mathcal{E}_1 \cup \mathcal{E}_2| < \lfloor \gamma n \rfloor$.

We now prove that $|\mathcal{E}_1 \cup \mathcal{E}_2| \leq \theta |\mathcal{E}|$ for some $\theta < 1$. Because each error in $\mathcal{E}_2$ is not corrected, it must have at most h neighbor constraints which are unsatisfied. The remaining satisfied neighbors of $\mathcal{E}_2$ must have at least two edges to $\mathcal{E}_2$ since otherwise, they could not be satisfied. Consequently,

$$|N(\mathcal{E}_2)| \leq \left(\frac{c-h}{2} + h \right) |\mathcal{E}_2|.$$

So,

$$\begin{aligned} |N(\mathcal{E})| = |N(\mathcal{E}_2)| + |N(\mathcal{E}) \backslash N(\mathcal{E}_2)| &\leq \left(\frac{c-h}{2} + h \right) |\mathcal{E}_2| + c(|\mathcal{E}| - |\mathcal{E}_2|) \\ &= c|\mathcal{E}| - \left(\frac{c-h}{2} \right) |\mathcal{E}_2|. \end{aligned} \tag{4.18}$$

Using expansion and the fact that each vertex in $\mathcal{E}_1$ has at most $c-h$ neighbors which are not in $N(\mathcal{E})$,

$$\alpha c(|\mathcal{E}| + |\mathcal{E}_1|) \leq |N(\mathcal{E}_1 \cup \mathcal{E})| \leq |N(\mathcal{E})| + (c-h)|\mathcal{E}_1|. \tag{4.19}$$

Substituting (4.18) into (4.19) gives

$$\alpha c(|\mathcal{E}| + |\mathcal{E}_1|) \leq c|\mathcal{E}| - \left(\frac{c-h}{2}\right)|\mathcal{E}_2| + (c-h)|\mathcal{E}_1| \tag{4.20}$$

which after re-arranging gives

$$(\alpha c + h - c)|\mathcal{E}_1| + \frac{c-h}{2}|\mathcal{E}_2| \leq (1-\alpha)c|\mathcal{E}|. \tag{4.21}$$

Multiplying both sides of (4.21) by $\frac{2}{c-h}$ yields

$$\frac{2(\alpha c + h - c)}{c-h}|\mathcal{E}_1| + |\mathcal{E}_2| \leq \frac{2(1-\alpha)c}{c-h}|\mathcal{E}|. \tag{4.22}$$

So, if $\frac{2(\alpha c+h-c)}{c-h} \geq 1$ and $\frac{2(1-\alpha)c}{c-h} < 1$, then $|\mathcal{E}_1 \cup \mathcal{E}_2| = |\mathcal{E}_1| + |\mathcal{E}_2| < \frac{2(1-\alpha)c}{c-h}|\mathcal{E}|$ which implies that the algorithm corrects all of the errors after $\log_{(c-h)/(2(1-\alpha)c)}(|\mathcal{E}|)$ decoding rounds. We can re-write the two conditions $\frac{2(\alpha c+h-c)}{c-h} \geq 1$ and $\frac{2(1-\alpha)c}{c-h} < 1$ as $\left(1 - \frac{2}{3}\alpha\right)c \leq h < (2\alpha - 1)c$. Note that this inequality is non-empty only when $\alpha > 3/4$. So, since $|N(i) \cap R_0|$ takes integral values, we can set $h := \lceil (2\alpha - 1)c \rceil$. □

(Notice from this analysis that $\tau = (c-h)/(2(1-\alpha)c)$ and $\tau' = \frac{\alpha c+h-c}{h}\gamma$.)

4.4.1.2 Sequential Bit-flipping

Sipser and Spielman also introduced a sequential version of the bit-flipping algorithm.

Sequential Bit-flipping

Input:

A (c, d)-biregular graph $(L \cup R, E)$ with $|L| = n$.

The binary parity-check code C_d.

A received word $\mathbf{w} = (w_i)_{i \in L} \in \{0, 1\}^n$.

Initialize:

Set $\mathbf{z} := \mathbf{w}$.

Set $R_0 := \left\{ j \in R \text{ s.t. } \mathbf{z}|_{N(j)} \notin C_d \right\}$.

For each $i \in L$, set $S_i = N(i) \cap R_0$.

Repeat cn/d times:

Count unsatisfied neighbors: Identify index k such that $|S_k|$ is maximum.

Update: If $|S_k| > c/2$, flip z_k. Otherwise, proceed to Output.

For each $j \in N(k)$, update as follows:

If $j \notin R_0$, set $R_0 := R_0 \cup \{j\}$, and for each $\ell \in N(j)$

set $S_\ell := S_\ell \cup \{j\}$.

Else if $j \in R_0$, set $R_0 := R_0 \backslash \{j\}$, and for each $\ell \in N(j)$

set $S_\ell := S_\ell \backslash \{j\}$.

If $R_0 = \emptyset$, proceed to Output.

Output:

If $R_0 \neq \emptyset$, output "failure."

Otherwise, output z.

Note that unlike the previous Bit-flipping algorithm, the Sequential Bit-flipping algorithm updates the constraints every time a single bit is flipped.

Theorem 45 ([SS96])**.** *Let $\mathcal{E}$ denote the set of errors present before decoding begins. Given a code whose Tanner graph is a (c, d, γ, α) expander graph with $\alpha > 3/4$, if $|\mathcal{E}| < (2\alpha - 1)\lfloor \gamma n \rfloor$,*

the Sequential Bit-flipping algorithm is guaranteed to correct all of the errors in $\mathcal{E}$ in time linear in the length of the code.

Notice that when $\alpha \approx 3/4$, the error correction bound in this theorem is identical to that in Theorem 44. However, for larger α, the bound in Theorem 44 is superior to this bound. To prove the bound in Theorem 45, it must be shown that the algorithm can begin and that once it has begun, it can correct all errors up to the specified threshold. Proving that there is a variable node j for which $|N(j) \cap R_0| > c/2$ will show that the algorithm can begin.

Because the number of unsatisfied constraints decreases at each step, the algorithm will correct all errors as long as it is allowed to finish. By using the expansion property of the graph, proving that the algorithm never encounters more than $\lfloor \gamma n \rfloor$ errors will imply that the algorithm successfully finishes decoding. The intuition behind these proofs depends on the fact that for a bipartite expander graph $(L \cup R, E)$, small sets of nodes $S \subseteq L$ have many unique neighbors, which we now define.

Definition 17. Given a bipartite graph $(L \cup R, G)$ and a set $S \subseteq L$, an element $i \in R$ is a unique neighbor of S if $|N(i) \cap S| = 1$, that is, there is only one edge between S and i. Let $N_1(S)$ denote the set of unique neighbors of a set S.

Following similar notation, define $N_k(S) := \{v \in N(S) \text{ s.t. } |N(v) \cap S| = k\}$.

Lemma 46. *Given a (c, d, γ, α) bipartite expander graph $G = (L \cup R, E)$, for any $t > 0$ and any subset $S \subseteq L$ such that $|S| \leq \lfloor \gamma n \rfloor$, $(t\alpha - 1)c|S| \leq \sum_{i=1}^{d} (t - i)|N_i(S)|$.*

Proof. Because G is a (c, d, γ, α) expander, $|S| \leq \lfloor \gamma n \rfloor$ implies that $\alpha c|S| \leq |N(S)|$. So, of course, $t\alpha c|S| \leq t|N(S)|$. By definition, $|N(S)| = \sum_{i=1}^{d} |N_i(S)|$. So, $t\alpha c|S| \leq t \sum_{i=1}^{d} |N_i(S)|$. Also, $c|S| = \sum_{i=1}^{d} i|N_i(S)|$ by definition. Combining these facts shows that $(t\alpha - 1)c|S| \leq t|N(S)| - c|S| = \sum_{i=1}^{d} (t - i)|N_i(S)|$. □

Corollary 47. *Given a set $S \subseteq L$ with $|S| \leq \lfloor \gamma n \rfloor$, it follows that $(2\alpha - 1)c|S| \leq |N_1(S)|$.*

Proof. Set $t = 2$ in the previous lemma. □

We now show that the algorithm can begin.

Lemma 48. *Let $\mathcal{E}$ denote the initial number of errors, and let R_0 denote the initial set of unsatisfied constraints. If $|\mathcal{E}| < \lfloor \gamma n \rfloor$ and $\alpha > 3/4$, then there exists an $i \in \mathcal{E}$ such that $|N(i) \cap R_0| > c/2$.*

Proof. Since $|\mathcal{E}| < \lfloor \gamma n \rfloor$, Corollary 47 guarantees that $|N_1(\mathcal{E})| \geq (2\alpha - 1)c|\mathcal{E}|$. Consequently, there must be at least one error with at least $(2\alpha - 1)c$ unique neighbors. So, since $\alpha > 3/4$, there exists an error with more than $c/2$ unique neighbors. Because each unique neighbor of the set of error bits is unsatisfied, there must be an $i \in \mathcal{E}$ such that $|N(i) \cap R_0| > c/2$. □

This lemma shows that if there are at most $\lfloor \gamma n \rfloor$ errors, the decoder can begin. After each iteration of the algorithm, the number of unsatisfied *constraints* decreases by at least one. However, the number of *errors* could increase or decrease by one. If the number of errors ever exceeds $\lfloor \gamma n \rfloor$, then we can no longer use the expansion property to ensure that the algorithm can continue. The next lemma verifies that the number of errors remains below $\lfloor \gamma n \rfloor$ at *each* stage of the algorithm.

Lemma 49. *Let $\mathcal{E}$ denote the initial set of errors, and let $\mathcal{E}_i$ denote the set of errors at step i of the algorithm. If $|\mathcal{E}| \leq (2\alpha - 1)\lfloor \gamma n \rfloor$, then $|\mathcal{E}_i| < \lfloor \gamma n \rfloor$ for all $i \geq 0$.*

Proof. Because $\alpha < 1$, $|\mathcal{E}| = |\mathcal{E}_0| \leq (2\alpha - 1)\lfloor \gamma n \rfloor < \lfloor \gamma n \rfloor$. Since the decoder flips only one bit at a time, if $|\mathcal{E}_j| > \lfloor \gamma n \rfloor$ for some $j > 0$, then $|\mathcal{E}_k| = \lfloor \gamma n \rfloor$ errors for some $0 < k < j$. By Corollary 47, $|N_1(\mathcal{E}_k)| \geq (2\alpha - 1)c\lfloor \gamma n \rfloor$. All of these unique neighbor constraints must be unsatisfied. However, because $|\mathcal{E}| \leq (2\alpha - 1)\lfloor \gamma n \rfloor$, there can initially be at most $(2\alpha - 1)c\lfloor \gamma n \rfloor$ unsatisfied constraints which contradicts the fact that the number of unsatisfied constraints decreases by at least one at each step in the algorithm. □

Together, Lemma 48 and Lemma 49 show that the bit-flipping algorithm will terminate after correcting all errors in $\mathcal{E}$. Because the number of unsatisfied constraints decreases by at least one after each decoding round, the algorithm must terminate after a linear number of steps, each of which takes constant time. So, Theorem 45 is proved. Notice that for $\alpha \approx 3/4$, the bound in Theorem 45 is as strong as the bound in Theorem 44. However, for $\alpha > 3/4$, the bound in Theorem 44 is stronger.

4.4.1.3 LP Decoding and Additional Results

Luby et al. [LMSS98] (see [LMSS01] for the full journal article) noted that these expander-based arguments could be used in conjunction with girth-based arguments. In particular, girth-based arguments on a hard-decision message passing algorithm could be used to show that most of the errors were corrected, and the expansion-based analysis of the bit-flipping strategy could then guarantee that the remaining errors were corrected. Burshtein and Miller [BM01] showed that changing the algorithm from message-passing to bit-flipping was unnecessary, and they directly analyzed various message-passing algorithms using expansion arguments. We mention in passing that by using probabilistic arguments, Burshtein in [Bur08] was able to guarantee correction of a larger fraction of errors than were Sipser and Spielman in [SS96] using expansion arguments. However, as the emphasis of this dissertation is on expander codes (whose arguments also apply to explicit code constructions, not only probabilistic constructions), we do not pursue this direction further here.

Feldman et al. [FMS$^+$07] used direct non-spectral expansion arguments to compute the number of errors that the LP decoder described in (4.6)-(4.8) could correct.

Theorem 50. *Let $\mathcal{E}$ denote the initial error set. Given an LDPC code whose Tanner graph is a c-left regular (c, γ, α) bipartite vertex expander graph with $\alpha > 2/3 + 1/(3c)$, if $|\mathcal{E}| < \left(\frac{3\alpha-2}{2\alpha-1}\right) \lfloor \gamma n \rfloor$, then the LP decoder described in* (4.6)-(4.8) *is guaranteed to correct all of the errors in $\mathcal{E}$.*

While the analysis of Sipser and Spielman [SS96] required expansion $\alpha > 3/4$, the analysis of Feldman et al. [FMS$^+$07] required expansion of only $\alpha > 2/3 + 1/(3c)$. However, though the expansion requirement for an LP decoding algorithm is less than the expansion requirement of Sipser and Spielman's bit-flipping algorithm, an LP decoding algorithms runs in polynomial time while the bit-flipping algorithm runs in linear time. We pause to mention that Skachek [Ska11] extended the result in [FMS$^+$07] to the case of nonbinary codes. We also mention that using very similar (spectral) expansions arguments to those of Barg and Zémor [BZ02], Feldman et al. showed in a different paper [FS05] that an LP decoding algorithm can achieve the capacity of the BSC.

4.4.1.4 Viderman [Vid13] Decoding

Recently, Viderman [Vid13] introduced a new *linear-time* decoding algorithm with a slightly smaller expansion requirement than in [FMS$^+$07]. Viderman's decoding strategy shares some similarities with the Sequential Bit-flipping algorithm; however, it introduces a new decoding aspect. Instead of flipping bits connected to many unsatisfied constraints, Viderman's algorithm erases them and then applies a peeling decoder to recover the erased bits. Also, instead of only considering unsatisfied constraints, Viderman utilizes "suspicious" constraints which include the unsatisfied constraints as well as all of the other neighboring constraints of erased bits.

Let R_0 denote the set of unsatisfied constraints, let $L'(\ell)$ denote the set of erased bits at step ℓ of the algorithm, and let $R'(\ell)$ denote the set of "suspicious" constraints at step ℓ of the algorithm. Given a bipartite Tanner graph $G = (L \cup R, E)$ which is a (c, d, γ, α) expander graph, the corrected algorithm which erases the error bits is given as follows:

Find Erasures

Input:

A (c, d, γ, α) bipartite expander graph $(L \cup R, E)$ with $|L| = n$.

The binary parity-check code C_d.

A received word $\mathbf{w} = (w_i)_{i \in L} \in \{0,1\}^n$.

Initialize:

Set $\mathbf{z} := \mathbf{w}$.

Set $h := \lceil (2\alpha - 1)c \rceil$.

Set $R_0 := \left\{ j \in R \text{ s.t. } \mathbf{z}|_{N(j)} \notin C_d \right\}$.

Set $L'(0) := \{ i \in L : |N(i) \cap R_0| \geq h \}$.

Set $R'(0) = N(L'(0)) \cup R_0$.

Set $\ell := 0$.

Repeat n times:

Count potentially corrupt neighbors: Identify an index $k \in L \backslash L'(\ell)$

such that $|N(k) \cap R'(\ell)| \geq h$.

If no such index exists, proceed to Output.

Set $L'(\ell + 1) := L'(\ell) \cup \{k\}$.

Set $R'(\ell + 1) := R'(\ell) \cup N(k)$.

Set $\ell := \ell + 1$.

Output:

For each $i \in L'(\ell)$, set $z_i := ?$.

Return $\mathbf{z}$.

Note that in Viderman's original decoding strategy, $R'(0) = R_0$ in which case the algorithm will never begin. However, setting $R'(0) := N(L'(0)) \cup R_0$ easily fixes this difficulty, and the proof proceeds almost identically to the proof in [Vid13].

Theorem 51 ([Vid13])**.** *Let $\mathcal{E}$ denote the initial error set, and let T denote the step at*

which this algorithm terminates. Given an LDPC code whose Tanner graph is a (c, d, γ, α) *expander graph with* $\alpha > 2/3$*, if* $|\mathcal{E}| < \frac{\alpha c+h-c}{h}\lfloor \gamma n \rfloor$ *with* $h = \lceil (2\alpha - 1)c \rceil$*, then*

- $\mathcal{E} \subseteq L'(T)$
- $|L'(T)| < \lfloor \gamma n \rfloor$.

The error bound in this theorem is the same as the error bound in Theorem 44 for $\alpha > 3/4$, the error bound in Theorem 45 for $\alpha \approx 3/4$, and the error bound in Theorem 50 for $\alpha > 2/3 + 1/(3c)$. However, the expansion requirement in Theorem 51 is smaller than the expansion requirement in any of these other theorems.

The first result of the theorem guarantees that the algorithm actually erases all of the errors. The second result guarantees that the size of erasures does not grow beyond $\lfloor \gamma n \rfloor$, which allows the proof to leverage expansion. Since the Find Erasures algorithm terminates after at most n steps, each of which takes constant time, the Find Erasures algorithm is a linear-time algorithm. We now present a proof of the theorem.

Proof. Notice that the algorithm is guaranteed to terminate in at most n steps. So, assuming the algorithm terminates at step T, we first prove that $\mathcal{E} \subseteq L'(T)$. For sake of contradiction, suppose $\mathcal{E} \backslash L'(T) \neq \emptyset$. Since $|\mathcal{E}| < \frac{\alpha c+h-c}{h}\lfloor \gamma n \rfloor < \lfloor \gamma n \rfloor$, it follows that $|\mathcal{E} \backslash L'(T)| < \lfloor \gamma n \rfloor$. Consequently, by Lemma 47 at least one error bit $i \in \mathcal{E} \backslash L'(T)$ must have at least $\lceil (2\alpha - 1)c \rceil$ unique neighbors which are either unsatisfied constraints or are satisfied constraints neighboring error bits in $L'(T)$. Letting $h = \lceil (2\alpha - 1)c \rceil$, it must be the case that $|N(i) \cap R'(T)| \geq h$ which contradicts the fact that i is not in $L'(T)$.

We next show that $|L'(T)| < \lfloor \gamma n \rfloor$. First, we demonstrate that $|L'(0)| < \lfloor \gamma n \rfloor$. Let $\mathcal{E}' := L'(0) \backslash \mathcal{E}$. Suppose $|\mathcal{E} \cup \mathcal{E}'| \geq \lfloor \gamma n \rfloor$. Then, take a subset $\mathcal{E}'' \subseteq \mathcal{E}'$ such that $|\mathcal{E} \cup \mathcal{E}''| = \lfloor \gamma n \rfloor$. By the expansion of the graph,

$$\alpha c \lfloor \gamma n \rfloor \leq |N(\mathcal{E} \cup \mathcal{E}'')| = |N(\mathcal{E})| + |N(\mathcal{E}'') \backslash N(\mathcal{E})| \leq |N(\mathcal{E})| + (c - h)|\mathcal{E}''|$$

since for each $i \in \mathcal{E}''$ we know that $|N(i) \cap R_0| \geq h$ and $R_0 \subseteq N(\mathcal{E})$. So,

$$\alpha c \lfloor \gamma n \rfloor \leq c|\mathcal{E}| + (c-h)(\lfloor \gamma n \rfloor - |\mathcal{E}|) = h|\mathcal{E}| + (c-h)\lfloor \gamma n \rfloor$$

which implies that $\frac{\alpha c+h-c}{h}\lfloor \gamma n \rfloor \leq |\mathcal{E}|$, contradicting the assumption that $|\mathcal{E}| < \frac{\alpha c+h-c}{h}\lfloor \gamma n \rfloor$. So, $|L'(0)| < \lfloor \gamma n \rfloor$.

We now show that $|L'(\ell)| < \lfloor \gamma n \rfloor$. First group the "suspicious" constraints $R'(\ell)$ at some step ℓ of the algorithm into the three following disjoint sets:

- Unsatisfied constraints: R_0
- Satisfied neighbors of the error bits in $L'(\ell)$: $R_1(\ell)$
- Remaining satisfied neighbors of $L'(\ell)$: $R_2(\ell)$.

For convenience, let $\mathcal{E}_0(\ell) = \mathcal{E} \cap L'(\ell)$ denote the set of error bits in $L'(\ell)$, and let $\mathcal{E}_1(\ell) = \mathcal{E} \backslash L'(\ell)$ denote the set of error bits which are not in $L'(\ell)$. Next, bound $R_1(\ell)$. Notice that

$$|R_1(\ell)| = |N(\mathcal{E}_0(\ell))| - |R_0 \cap N(\mathcal{E}_0(\ell))|. \tag{4.23}$$

Also, due to possible overlap between $N(\mathcal{E}_0(\ell))$ and $N(\mathcal{E}_1(\ell))$,

$$|R_0| - |R_0 \cap N(\mathcal{E}_1(\ell))| \leq |R_0 \cap N(\mathcal{E}_0(\ell))|. \tag{4.24}$$

Substituting 4.24 into 4.23, it follows that

$$|R_1(\ell)| \leq |N(\mathcal{E}_0(\ell))| - |R_0| + |R_0 \cap N(\mathcal{E}_1(\ell))|.$$

For each $i \in \mathcal{E}_1(\ell)$ and $\ell \geq 0$, $|N(i) \cap R_0| < h$. So, $|R_0 \cap N(\mathcal{E}_1(\ell))| < h|\mathcal{E}_1(\ell)|$. Also,

$|N(\mathcal{E}_0(\ell))| \leq c|\mathcal{E}_0(\ell)|$. Consequently,

$$|R_1(\ell)| \leq c|\mathcal{E}_0(\ell)| - |R_0| + h|\mathcal{E}_1(\ell)|.$$

To obtain an upper bound on $|R_2(\ell)|$, argue by induction. For the base case, notice that at the first step of the algorithm, $|N(i) \cap R_0| \geq h$ for each $i \in L'(0)$. Consequently, each $i \in L'(0) \backslash \mathcal{E}_0(0)$ can have at most $c - h$ neighbors in $R_2(0)$ (since R_0 and $R_2(0)$ are disjoint), so

$$|R_2(0)| \leq (c - h)(|L'(0)| - |\mathcal{E}_0(0)|).$$

By the inductive hypothesis, suppose that

$$|R_2(\ell - 1)| \leq (c - h)(|L'(\ell - 1)| - |\mathcal{E}_0(\ell - 1)|), \tag{4.25}$$

where $\ell \geq 1$. Let $\{i\} = L'(\ell) \backslash L'(\ell - 1)$. If $i \in \mathcal{E}$, then

$$R_2(\ell) = R_2(\ell - 1).$$

So, because $|L'(\ell - 1)| - |\mathcal{E}_0(\ell - 1)| = |L'(\ell)| - |\mathcal{E}_0(\ell)|$ (since $i \in \mathcal{E}_0(\ell)$), the following bound trivially holds:

$$|R_2(\ell)| \leq (c - h)(|L'(\ell)| - |\mathcal{E}_0(\ell)|)$$

Next, suppose $i \notin \mathcal{E}$. Since $i \in L'(\ell)$,

$$|N(i) \cap R'(\ell - 1)| \geq h$$

which implies that

$$|N(i) \backslash R'(\ell - 1)| = c - |N(i) \cap R'(\ell - 1)| \leq c - h.$$

So,

$$|R'(\ell)\backslash R'(\ell-1)| \leq c-h. \tag{4.26}$$

Because no more constraints can be added to $R_2(\ell-1)$ than can be added to $R'(\ell-1)$,

$$|R_2(\ell)\backslash R_2(\ell-1)| \leq |R'(\ell)\backslash R'(\ell-1)|,$$

$$|R_2(\ell)\backslash R_2(\ell-1)| \leq c-h. \tag{4.27}$$

Combining 4.25 and 4.27 gives the desired bound on $R_2(\ell)$:

$$|R_2(\ell)| = |R_2(\ell)\backslash R_2(\ell-1)| + |R_2(\ell-1)| \leq (c-h) + (c-h)(|L'(\ell-1)| - |\mathcal{E}_0(\ell-1)|).$$

Since $|L'(\ell-1)|+1 = |L'(\ell)|$,

$$|R_2(\ell)| \leq (c-h)(|L'(\ell)| - |\mathcal{E}_0(\ell)|)$$

which gives the same bound on $R_2(\ell)$ as in the case when $i \in \mathcal{E}$. In summary, for $\ell \geq 0$,

- $|R_0| \leq |R_0|$
- $|R_1(\ell)| \leq c|\mathcal{E}_0(\ell)| - |R_0| + h|\mathcal{E}_1(\ell)|$
- $|R_2(\ell)| \leq (c-h)(|L'(\ell)| - |\mathcal{E}_0(\ell)|)$.

Since $|R_0| + |R_1(\ell)| + |R_2(\ell)| = |N(L'(\ell))|$, summing these inequalities gives

$$|N(L'(\ell))| \leq h|\mathcal{E}_1(\ell)| + (c-h)|L'(\ell)| + h|\mathcal{E}_0(\ell)|. \tag{4.28}$$

Because $|\mathcal{E}_1(\ell)| + |\mathcal{E}_0(\ell)| = |\mathcal{E}|$,

$$|N(L'(\ell))| \leq h|\mathcal{E}| + (c-h)|L'(\ell)|. \tag{4.29}$$

Notice that $L'(\ell)$ increases by one at each step of the algorithm. Consequently, because $|L'(0)| < \lfloor \gamma n \rfloor$, if $|L'(\ell)| > \lfloor \gamma n \rfloor$ for some $\ell > 0$, it must have been the case that $|L'(\ell')| = \lfloor \gamma n \rfloor$ for some $\ell' < \ell$. At this point, $|N(L'(\ell')| \geq \alpha c \lfloor \gamma n \rfloor$ by the expansion property of the graph. So,

$$|\mathcal{E}| \geq \left(\frac{\alpha c + h - c}{h} \right) \lfloor \gamma n \rfloor$$

by (4.29) which contradicts the assumption that $|\mathcal{E}| < \left(\frac{\alpha c + h - c}{h}\right) \lfloor \gamma n \rfloor$. □

Assuming that the code on the right-hand side constraint nodes is the parity-check code, the decoding algorithm is the simple peeling algorithm from LDPC codes. In what follows, let ? denote an erasure.

Peeling Decoder

Input:

A (c, d) bipartite graph $(L \cup R, E)$ with $|L| = n$.

The binary parity-check code C_d.

A received word $\mathbf{w} = (w_i)_{i \in L} \in \{0, 1, ?\}^n$.

Set of erasure positions E_r.

Initialize:

Set $\mathbf{z} := \mathbf{w}$.

Repeat n times:

Local decoding: For each constraint $j \in N_1(E_r)$, find the codeword $\mathbf{c}^{(j)}$ closest to $\mathbf{z}|_{N(j)}$.

Update: Set $\mathbf{z}|_{N(j)} := \mathbf{c}^{(j)}$, and remove the corrected bit's index from E_r.

If $N_1(E_r) = \emptyset$, then proceed to Output.

Output:

If $E_r = \emptyset$, return $\mathbf{z}$.

Otherwise, return "failure."

The Peeling Decoder identifies constraints adjacent to only one erased bit. Each constraint can easily correct one erasure by setting the erased bit equal to the sum of its other adjacent un-erased bits. The constraint is then satisfied. By Theorem 51, if $|\mathcal{E}| < \frac{\alpha c + h - c}{h} \lfloor \gamma n \rfloor$, then $\mathcal{E} \subseteq L'(T)$ and $|L'(T)| < \lfloor \gamma n \rfloor$. Set $E_r = L'(T)$. Then, Lemma 47 guarantees that there exists an $i \in E_r$ with a unique neighbor constraint while E_r is not empty. Consequently, Find Erasures followed by the Peeling Decoder will successfully decode any set $\mathcal{E}$ of errors with $|\mathcal{E}| < \frac{\alpha c + h - c}{h} \lfloor \gamma n \rfloor$. Here again, the Peeling Decoder terminates after a linear number of steps each of which requires constant time. Since erasing the bits also took linear time, the overall decoding algorithm is a linear-time algorithm in the length of the code.

Notice that $\frac{\alpha c + h - c}{h} > 0$ for $\alpha > \frac{2}{3}$. So, this result slightly improved on the expansion

of $\alpha > 2/3 + 1/(3c)$ required by Feldman's LP decoding [FMS+07] and the expansion of $\alpha > 3/4$ required by Sipser and Spielman [SS96].

By slightly modifying the FindErasures algorithm, Viderman showed that it is only necessary that $\alpha > 2/3 - 1/(6c)$, where the underlying graph is a (c, d, γ, α) bipartite expander graph. Furthermore, he showed that if $\alpha \leq 1/2$ and if the binary parity-check code is used on the constraint nodes, no decoding algorithm can guarantee correction of a constant fraction of errors. This result is intuitively satisfying since for $\alpha \leq 1/2$, Lemma 47 cannot even guarantee the existence of a single unique neighbor.

These conclusions led Viderman to raise the question of whether or not it was possible to guarantee the correction of a linear number of errors of an LDPC code while only requiring that $\alpha > 1/2$. While this question is still open, we will see that by considering more general codes on the constraints, there are simple decoding algorithms which only require $\alpha > 1/2$ asymptotically. In fact, in our main result of this section, we will show that there is a simple, linear-time decoding algorithm which only requires that $\alpha > 1/t$ for an arbitrary $t > 1$.

4.4.2 GLDPC Codes

4.4.2.1 Chilappagari et al. [CNVM10] Decoding

In the context of girth-based expansion guarantees, Chilappagari et al. [CNVM10] analyzed the error-correction capabilities of GLDPC codes based directly on the expansion of the underlying Tanner graph. The authors introduced a decoding algorithm which is very similar to the parallel bit flipping algorithm in [SS96] for GLDPC codes. This time, however, the threshold is set to half the minimum distance of the inner code instead of a quarter of the minimum distance of the inner code. In addition, the decoding algorithm is generalized to (c, d)-biregular graphs instead of just d-regular graphs (or the corresponding $(2, d)$-regular edge-vertex incidence graphs).

Generalized Bit-flipping 2

Input:

A (c, d, γ, α) expander graph $(L \cup R, E)$ with $|L| = n$.

A binary code C_d of length d with relative minimum distance at least $2t + 1$.

A received word $\mathbf{w} = (w_i)_{i \in L} \in \{0, 1\}^n$.

Initialize:

Set $\mathbf{z} := \mathbf{w}$.

Repeat γn times:

Local decoding: For each $j \in R$, find a codeword $\mathbf{c}^{(j)} \in C_d$ with

$d(\mathbf{z}|_{N(j)}, \mathbf{c}^{(j)}) \leq t$ if it exists.

Update: For each $i \in L$, if there are more than $c/2$ constraints $j \in N(i)$

such that $\mathbf{c}^{(j)}$ exists and $c_i^{(j)} \neq z_i$, then flip z_i

(i.e. set $z_i := z_i + 1 \bmod 2$).

If $\mathbf{z}$ remains unchanged, proceed to Output.

Output:

If there exists a $j \in R$ such that $\mathbf{z}|_{N(j)} \notin C_d$, output "failure."

Otherwise, output $\mathbf{z}$.

Theorem 52 ([CNVM10])**.** *Given a (c, d, γ, α) bipartite expander graph G and a code C_d of length d and minimum distance at least $2t + 1$, if $\alpha > \frac{t+2}{2(t+1)}$, then Generalized Bit-flipping 2 will correct up to $\lfloor \gamma n \rfloor$ errors.*

Notice that as $t \to \infty$, this theorem only requires that $\alpha > 1/2$. Also, unlike Theorems 44, 45, and 51, there is no fraction in front of the $\lfloor \gamma n \rfloor$ term.

Proof. We first introduce the following terminology from [CNVM10] which is similar to the terminology used in previous proofs. Notice that "unhelpful" and "confused" constraints from the proof of Theorem 40 have both been merged into "confused" constraints.

- Call the set of constraints in $N(\mathcal{E})$ which either send a "flip" message to a correct bit or do not send a "flip" message to an incorrect bit "confused," and denote the set of such constraints by C.

- Call the constraints in $N(\mathcal{E})$ which send a "flip" message to incorrect bits "helpful" constraints, and denote the set of such constraints by H.

Notice that $H \cup C = N(\mathcal{E})$. To prove the theorem, it is enough to show that the number of errors decreases after each decoding round. Following the same notation used in the proof of Theorem 44, after one decoding round let $\mathcal{E}_1$ denote the set of error bits resulting from flipping correct code bits, and let $\mathcal{E}_2$ denote the set of error bits resulting from not flipping incorrect code bits. Suppose that $|\mathcal{E}_1| + |\mathcal{E}_2| > |\mathcal{E}|$. Then, there is a subset $\mathcal{E}_2' \subseteq \mathcal{E}_2$ such that $|\mathcal{E}_1| + |\mathcal{E}_2'| = |\mathcal{E}|$. The proof works toward showing that the number of neighbors of $\mathcal{E}_1$ and $\mathcal{E}_2$ is smaller than that guaranteed by the expansion property of the graph, thus yielding a contradiction. Notice that each code bit in $\mathcal{E}_1$ must be connected to at least $\lfloor \frac{c}{2} \rfloor + 1$ confused constraints (hence at most $\lfloor \frac{c}{2} \rfloor - 1$ non-confused constraints), and each code bit in $\mathcal{E}_2$ must be connected to at least $\lfloor \frac{c}{2} \rfloor$ confused constraints (hence at most $\lfloor \frac{c}{2} \rfloor$ non-confused constraints). So,

$$|N(\mathcal{E}_1 \cup \mathcal{E}_2')| \leq |C| + |\mathcal{E}_1|(\lfloor \frac{c}{2} \rfloor - 1) + |\mathcal{E}_2'| \lfloor \frac{c}{2} \rfloor < |C| + |\mathcal{E}| \frac{c}{2}. \tag{4.30}$$

Next, bound the number of confused constraints. Because C_d can correct at least t errors, confused constraints must have at least $t + 1$ neighbors in $\mathcal{E}$. Of course, helpful constraints must have at least one neighbor in $\mathcal{E}$. Exactly $c|\mathcal{E}|$ edges are adjacent to $\mathcal{E}$. So,

$$(t + 1)|C| + |H| \leq c|\mathcal{E}|.$$

Since $|C| + |H| = |N(\mathcal{E})|$,

$$(t+1)|C| + |N(\mathcal{E})| - |C| \leq c|\mathcal{E}|. \tag{4.31}$$

Rearranging,

$$|C| \leq \frac{c|\mathcal{E}| - |N(\mathcal{E})|}{t}.$$

By expansion, $\alpha c|\mathcal{E}| \leq |N(\mathcal{E})|$. So,

$$|C| \leq \frac{c|\mathcal{E}| - \alpha c|\mathcal{E}|}{t}. \tag{4.32}$$

Combining 4.30 and 4.32 gives

$$|N(\mathcal{E}_1 \cup \mathcal{E}_2')| < \left(\frac{1-\alpha}{t} + \frac{1}{2}\right) c|\mathcal{E}|. \tag{4.33}$$

Recall that $\alpha > \frac{t+2}{2(t+1)}$. So,

$$\frac{1-\alpha}{t} < \frac{2\alpha - 1}{2}.$$

Substituting this inequality into 4.33 gives

$$|N(\mathcal{E}_1 \cup \mathcal{E}_2')| < \left(\frac{1-\alpha}{t} + \frac{1}{2}\right) c|\mathcal{E}| < \left(\frac{2\alpha-1}{2} + \frac{1}{2}\right) c|\mathcal{E}|.$$

So,

$$|N(\mathcal{E}_1 \cup \mathcal{E}_2')| < \alpha c|\mathcal{E}|. \tag{4.34}$$

Since $|\mathcal{E}_1 \cup \mathcal{E}_2'| = |\mathcal{E}| < \lfloor \gamma n \rfloor$, (4.34) contradicts the assumption that the underlying Tanner graph was a (c, d, γ, α) bipartite expander graph. □

Notice that the update step takes $O(n)$ time, and the algorithm terminates after at most γn iterations. Consequently, the algorithm is an $O(n^2)$ algorithm (or an $O(n)$ parallel-

time algorithm). The authors in [CNVM10] note that by requiring slightly larger expansion, it can be shown that their algorithm runs in $O(\log n)$ parallel time.

4.4.2.2 Decoding with Small Expansion

We now present our recent contribution which shows that by using a more refined analysis, we can obtain simple GLDPC codes for which the expansion of the underlying Tanner graphs can be made arbitrarily small. In addition, these codes can be decoded in linear time. We also show that by employing stronger codes on the constraints, we can approximately double the error correction capabilities of Viderman's expander decoding algorithm when the underlying expander graphs are constructed probabilistically.

As before, let $G = (L \cup R, E)$ be a (c, d, γ, α) bipartite vertex expander graph. Recall that we denoted the number of constraints which have k edges adjacent to the set S as $N_k(S) := \{v \in N(S) \text{ s.t. } |N(v) \cap S| = k\}$, where $S \subseteq L$ represents the set of errors and $N(S)$ denotes the neighbors of S. We also called the constraints in $N_1(S)$ *unique neighbor* constraints. The key observation in analyzing more powerful codes using bipartite vertex expansion arguments is in Lemma 46 which shows that for *any* $k \in [1, d)$, we can bound the number of constraints in $N_k(S)$. In particular, Lemma 46 showed that given $S \subseteq L$ with $|S| \leq \gamma n$, $(t\alpha - 1)c|S| \leq \sum_{i=1}^{d}(t - i)|N_i(S)|$.

If $t = 2$ and if we employ a code on the constraints with minimum distance 3 which can correct one error, all of the constraints in $N_1(S)$ will decode their neighboring bits to the correct value. However, it is also possible for a constraint in $\cup_{i\geq 2}N_i(S)$ to decode its neighboring bits incorrectly since the decoder could perceive a constraint in $\cup_{i\geq 2}N_i(S)$ as a unique neighbor. As we increase the minimum distance of the code, we also increase the number of errors which must be introduced in order for a non-unique neighbor to appear to be a unique neighbor. For example, if the minimum distance of the code is 5, then only constraints in $\cup_{i\geq 4}N_i(S)$ could be misinterpreted as unique neighbors.

For a given $t > 1$, if $\alpha > 1/t$ and the minimum distance of the code is large enough, we will harness the bound in Lemma 46 to guarantee that the total number of errors corrected in a decoding round is a positive fraction of the total number of errors in the previous decoding round. By adjusting t, we will be able to make α arbitrarily small. Naturally, as t increases, we will need a more powerful code with a higher minimum distance, and for a given value of t, we will use an inner code C_d with minimum distance at least $2t+c(t-1)^2-1$. Although using such a powerful inner code would enable us to correct more than $t-1$ errors, we will restrict ourselves to only decoding a constraint when it detects $t-1$ or fewer errors. Moreover, we will say that a constraint "suggests" values to its neighboring variable nodes if the decoder at that constraint would have decoded to those values if allowed. Our decoding algorithm is very similar to Sipser and Spielman's Bit-flipping strategy except that our threshold is $t-1$, and our algorithm is defined on (c, d)-biregular graphs instead of d-regular graphs.

Generalized Bit-flipping 3

Input:

A (c, d)-biregular graph $(L \cup R, E)$ with $|L| = n$.

A code C_d of length d with relative minimum distance at least $2t + c(t-1)^2 + 1$, where $t > 1$.

A received word $\mathbf{w} = (w_i)_{i \in L} \in \mathbb{F}_q^n$.

Constants τ and τ' to be determined later.

Initialize:

Set $\mathbf{z} := \mathbf{w}$.

Repeat $\log_\tau(\tau' n)$ times:

Local decoding: For each $j \in R$, find a codeword $\mathbf{c}^{(j)} \in C_d$ with $d(\mathbf{z}|_{N(j)}, \mathbf{c}^{(j)}) \leq t - 1$ if it exists.

Update: For each $i \in L$, if there is a $j \in N(i)$ such that $\mathbf{c}^{(j)}$ exists and $c_i^{(j)} \neq z_i$, then set $z_i := c_i^{(j)}$.

If $\mathbf{z}$ remains unchanged, proceed to Output.

Output:

If there exists a $j \in R$ such that $\mathbf{z}|_{N(j)} \notin C_d$, output "failure."

Otherwise, output $\mathbf{z}$.

Theorem 53. *Let G be a (c, d, γ, α) expander graph, and suppose C_d has minimum distance at least $2t + c(t-1)^2 - 1$. If $\alpha > 1/t$, any error pattern of size at most $\lfloor \gamma n \rfloor$ in $T(G, C_d)$ can be corrected in time linear in the length of the code.*

Notice that t can be made arbitrarily large. Consequently, at the cost of a higher minimum distance on the inner code C_d, our code's expansion requirements can be made arbitrarily small. In contrast, recall that the expansion for the codes in Chilappagari et al. [CNVM10] could be made no smaller than $1/2$. We prove Theorem 53 in a sequence of lemmas. First, we

show that the number of errors decreases by a positive fraction at each step of the decoding algorithm.

Lemma 54. *Let $\mathcal{E}_k$ denote the number of errors remaining after round k of the algorithm has completed. Then, $|\mathcal{E}_{k+1}| \leq \left(1 - \frac{t\alpha - 1}{t-1}\right) |\mathcal{E}_k|$.*

Proof. Similarly to the proof of Theorem 40, we first partition the constraints after round k into three categories:

- Helpful constraints: $\bigcup_{i=1}^{t-1} N_i(\mathcal{E}_k)$
- Unhelpful constraints: $\bigcup_{i=t}^{t+c(t-1)^2-1} N_i(\mathcal{E}_k)$
- Potentially confused constraints: $\bigcup_{i=t+c(t-1)^2}^{d} N_i(\mathcal{E}_k)$

Because the minimum distance of the code is at least $2t + c(t-1)^2 - 1$, unhelpful constraints do not make any suggestions during a round of the decoding algorithm. In contrast, a helpful constraint in $N_i(\mathcal{E}_k)$ sends i correct suggestions to its neighboring variables. Let $\mathcal{E}_0(k)$ denote the set of errors in $\mathcal{E}$ which are corrected after a given decoding round k, and let $\mathcal{E}_1(k)$ denote the new errors introduced after decoding round k. Since the graph is c-left regular, each variable node can be connected to at most c helpful constraints. Consequently,

$$|\mathcal{E}_0(k)| \geq \frac{\sum_{i=1}^{t-1} i|N_i(\mathcal{E}_k)|}{c}.$$

Moreover, a potentially confused constraint sends at most $t-1$ erroneous suggestions to its neighboring variables. So,

$$|\mathcal{E}_1(k)| \leq (t-1) \sum_{i=t+c(t-1)^2}^{d} |N_i(\mathcal{E}_k)|.$$

Combining these two facts, the number of errors corrected during each round of the decoding algorithm is at least

$$|\mathcal{E}_k| - |\mathcal{E}_{k+1}| = \mathcal{E}_0(k) - \mathcal{E}_1(k) \geq \frac{\sum_{i=1}^{t-1} i|N_i(\mathcal{E}_k)|}{c} - (t-1) \sum_{i=t+c(t-1)^2}^{d} |N_i(\mathcal{E}_k)|. \tag{4.35}$$

From Lemma 46, given $\mathcal{E} \subseteq L$ with $|\mathcal{E}| \leq \gamma n$,

$$\sum_{i=1}^{t-1}(t-i)|N_i(\mathcal{E})| - \sum_{i=t}^{t+c(t-1)^2-1} (i-t)|N_i(\mathcal{E})| - \sum_{i=t+c(t-1)^2}^{d} (i-t)|N_i(\mathcal{E})| \geq (t\alpha - 1)c|\mathcal{E}|.$$

Removing the unhelpful constraints, it immediately follows that

$$\sum_{i=1}^{t-1}(t-i)|N_i(\mathcal{E}_k)| - \sum_{i=t+c(t-1)^2}^{d} (i-t)|N_i(\mathcal{E}_k)| \geq (t\alpha - 1)c|\mathcal{E}_k|. \tag{4.36}$$

Note that because $i \geq (t-i)/(t-1)$ for $1 \leq i \leq t-1$ and $(i-t)/(c(t-1)) \geq t-1$ for $i \geq t + c(t-1)^2$,

$$\frac{\sum_{i=1}^{t-1} i|N_i(\mathcal{E}_k)|}{c} - (t-1) \sum_{i=t+c(t-1)^2}^{d} |N_i(\mathcal{E}_k)| \geq \frac{\sum_{i=1}^{t-1}(t-i)|N_i(\mathcal{E}_k)|}{c(t-1)} - \sum_{i=t+c(t-1)^2}^{d} \frac{i-t}{c(t-1)}|N_i(\mathcal{E}_k)|. \tag{4.37}$$

Finally, combining 4.35, 4.36, and 4.37 gives

$$|\mathcal{E}_k| - |\mathcal{E}_{k+1}| \geq \frac{t\alpha - 1}{t-1}|\mathcal{E}_k|.$$

Consequently,

$$|\mathcal{E}_{k+1}| \leq \left(1 - \frac{t\alpha - 1}{t-1}\right)|\mathcal{E}_k|.$$

If $\alpha > 1/t$, then

$$1 - \frac{t\alpha - 1}{t - 1} < 1.$$

Consequently, for any $\alpha > 1/t$, if $|\mathcal{E}| < \lfloor \gamma n \rfloor$, the number of errors decreases by a positive fraction at each step of decoding. □

From this proof, we see that $\tau = \left(1 - \frac{t\alpha - 1}{t-1}\right)^{-1}$, and $\tau' = \gamma$.

Lemma 55. *Generalized Bit-flipping 3 runs in time linear in the length of the code.*

Proof. To see that the algorithm terminates in linear time, notice that at the first step of the algorithm, the decoder must decode each constraint. This initial step requires cn/d decoding operations. After this initial step, the decoder only needs to check the neighbor constraints of each adjusted variable node since the other constraints cannot have changed. Since the number of errors decreases at each step, the number of adjusted variable nodes after round k of decoding is at most $2|\mathcal{E}_{k-1}|$ (which would occur if $|\mathcal{E}_{k-1}|$ errors were corrected and $|\mathcal{E}_{k-1}|$ errors were introduced). But from the previous lemma, $\{|\mathcal{E}_{k-1}|\}_{k=1}^{\infty}$ forms a geometric sequence. So, the number of decoding operations (each of which takes constant time) is bounded by

$$cn/d + 2c\sum_{k=0}^{\infty} |\mathcal{E}_k| = cn/d + 2c\sum_{k=0}^{\infty} \left(1 - \frac{t\alpha - 1}{t - 1}\right)^k |\mathcal{E}_0| \leq cn/d + 2c\left(\frac{t-1}{t\alpha - 1}\right)\gamma n$$

where $\mathcal{E}_0 = \mathcal{E}$. Consequently, the algorithm runs in time linear in the length of the code.

□

Theorem 53 now follows as an immediate corollary of Lemmas 54 and 55. Note that in the previous lemma, $\frac{t-1}{t\alpha-1}$ could be an arbitrarily large constant. If we desire to enforce that the constant be no greater than $2(t-1)$, we can do so by requiring $\alpha > 2/t$ instead of $\alpha > 1/t$. Alternately, given a fixed $\alpha > 1/t$, we can translate this condition into a stricter minimum

distance requirement. In particular, we can require the minimum distance to be at least $4t + c(2t-1)^2 - 1$ to maintain the same value of $\alpha > 1/t$.

We now compare our result to those of Sipser and Spielman [SS96], Feldman et al. [FMS+07], Chilappagari et al. [CNVM10], and Viderman [Vid13]. We let $\beta(c, d, \gamma, \alpha)$ denote a function of the parameters c, d, γ, and α. In the table below, G is a (c, d, γ, α) bipartite expander graph, C_d is a code of length d, and the parameters listed are for $T(G, C_d)$:

C_d is the parity-check code			
	Required Expansion	Run Time	Number of Errors
Sipser and Spielman 1996	$\alpha > \frac{3}{4}$	linear	$(2\alpha - 1)\gamma n$
Feldman et al. 2007	$\alpha > \frac{2}{3} + \frac{1}{3c}$	polynomial	$\frac{3\alpha-2}{2\alpha-1}\gamma n$
Viderman 2013	$\alpha > \frac{2}{3} - \frac{1}{6c}$	linear	$\frac{3\alpha-2}{2\alpha-1}\gamma n$
Open	$\alpha > \frac{1}{2}$	polynomial	$\beta(c, d, \gamma, \alpha)\gamma n$
Open	$\alpha > \frac{1}{2}$	linear	$\beta(c, d, \gamma, \alpha)\gamma n$

C_d has minimum distance $w \geq 2t - 1$, $t > 1$			
	Required Expansion	Run Time	Number of Errors
Chilappagari et al. 2010	$\alpha > \frac{t+2}{2(t+1)}$	linear	γn

C_d has minimum distance $w \geq 2t + c(t-1)^2 - 1$, $t > 1$			
	Required Expansion	Run Time	Number of Errors
Our result	$\alpha > \frac{1}{t}$	linear	γn

Table 4.3: Comparisons of Required Expansion

Note that when using a more powerful code on the constraints, the rate of $T(G, C_d)$ decreases while the minimum distance of $T(G, C_d)$ increases. The exact relationship between the

minimum distance of C_d and the rate and minimum distance of $T(G, C_d)$ when C_d is an MDS code is made precise in the tables below:

Previous setting: C_d is a parity-check code with minimum distance 2 and rate $1 - \frac{1}{d}$:

rate of $T(G, C_d)$	min dist. of $T(G, \mathcal{C}_d)$	# of errors corrected	required expansion
$\geq 1 - \frac{c}{d}$	$\geq 2\alpha\gamma n$	$\frac{3\alpha-2}{2\alpha-1}\gamma n$	$\alpha > \frac{2}{3}$

Our setting: $\mathcal{C}_d$ is an MDS code with minimum distance w and rate $1 - \frac{w-1}{d}$:

rate of $T(G, C_d)$	min dist. of $T(G, \mathcal{C}_d)$	# of errors corrected	required expansion
$\geq 1 - \frac{c}{d}(w-1)$	$\geq w\alpha\gamma n$	γn	$\alpha > \frac{1}{t}$, $t > 1$

Table 4.4: Comparison of LDPC and GLDPC Parameters

Recall that in our construction, $w \geq 2t + c(t-1)^2 - 1$. Consequently, for large values of t (corresponding to very small expansion requirements), there is a large gap between $\frac{w\alpha\gamma n}{2}$ and γn. We believe this gap can be significantly reduced. Also, as t increases, $\frac{c}{d}$ must decrease to maintain a comparable rate to the rate when C_d is a parity-check code. We pause to point out that, as noted by McEliece and Swanson [MS86], the probability that a RS code decodes to an incorrect codeword is $1/\ell!$, where ℓ is the number of errors the RS code can correct. Consequently, for small expansion requirements (corresponding to large minimum distance requirements), we need only require $w \geq 2t - 1$ to guarantee correction of most patterns of up to γn errors. Høholdt and Justesen [HJ06] noted a similar phenomenon for their codes.

Finally, we numerically compare our construction to Viderman's using probabilistically constructed expander graphs whose expansion is bounded with high probability by the expression in Theorem 43. In the comparison, we fix the rate at 1/2. Empirically, expansion of 0.71 gave the best results for Viderman's decoding algorithm which is why we selected it.

Previous setting: C_d is a parity-check code with minimum distance 2 and rate $1 - \frac{1}{d}$:

(c, d)	rate of $T(G, C_d)$	(α, γ)	min dist. of $T(G, \mathcal{C}_d)$	# of errors
$(24, 48)$	$\geq \frac{1}{2}$	$(0.71, 0.000870)$	$\geq 0.00124n$	$\geq 0.000269n$
$(36, 72)$	$\geq \frac{1}{2}$	$(0.71, 0.000851)$	$\geq 0.00121n$	$\geq 0.000263n$
$(48, 96)$	$\geq \frac{1}{2}$	$(0.71, 0.000771)$	$\geq 0.00109n$	$\geq 0.000235n$

Our setting: C_d is an MDS code with minimum distance $3 + c$ and rate $1 - \frac{2+c}{d}$:

(c, d)	rate of $T(G, \mathcal{C}_d)$	(α, γ)	min dist. of $T(G, \mathcal{C}_d)$	# of errors
$(4, 48)$	$\geq \frac{1}{2}$	$(\frac{1}{2}, 0.000094)$	$\geq 0.00033n$	$\geq 0.000094n$
$(5, 70)$	$\geq \frac{1}{2}$	$(\frac{1}{2}, 0.000274)$	$\geq 0.0011n$	$\geq 0.000274n$
$(6, 96)$	$\geq \frac{1}{2}$	$(\frac{1}{2}, 0.000426)$	$\geq 0.0019n$	$\geq 0.000426n$

Table 4.5: Viderman Decoding vs. Our Result

These tables show that for $d \approx 100$, our decoding algorithm corrects approximately twice the number of errors as Viderman's decoding algorithm. (Recall that $d = 96$ is a smaller degree than the degree of 256 required by [SR03] to guarantee correction of even a positive fraction of errors.) We believe that the decoding algorithm and analysis can be modified to improve on Viderman's decoding capabilities for all values of d.

Chapter 5

Conclusions and Future Work

In this thesis, we studied explicit constructions of expander graphs, methods for computing both approximate and exact edge expansion using linear programming techniques, and methods for analyzing fast decoding algorithms for expander codes using vertex expansion analysis.

In Chapter 2, we modified the original construction of Margulis [Mar73] to construct a family of 16-regular expander graphs with a larger spectral gap than that of the family of 8-regular graphs in [AGM87]. We also surveyed many of the existing constructions of expander graphs including the modified zig-zag product construction of the d-left regular lossless expander graphs presented in [CRVW02]. Unfortunately, our expander code constructions in Chapter 4 rely on (c, d)-biregular graphs with good vertex expansion properties. It is an interesting question whether the construction in [CRVW02] can be modified to produce such (c, d)-biregular bipartite expander graphs.

In Chapter 3, we provided a linear program for finding the smallest ratio $|E(S, \overline{S})|/|S|$ with $S \subseteq T$ for some pre-determined set T. We then modified this linear program to find the smallest ratio $|E(S, \overline{S})|/|S|$ up to an approximation factor while allowing some overlap between S and $\overline{T}$, where again T is a pre-determined set. Finally, we added a non-linear

constraint to give an integer program which exactly computes the minimum $|E(S, \overline{S})|/|S|$ over all subsets S in the graph having size at most half the number of vertices in the graph. It would be interesting to see whether or not it is possible to modify our linear programming formulation to reduce the amount of overlap required for the approximation.

In Chapter 4, we gave a fast decoding algorithm for GLDPC codes which is guaranteed to correct a constant fraction of errors in linear time even when the expansion is much smaller than 1/2, thus improving on a result by Chilappagari et al. [CNVM10]. This result allowed us to approximately double the fraction of errors corrected in [Vid13]. Unfortunately, for very small expansion requirements and large left-degree, the gap between the guaranteed minimum distance and the bound on the number of errors which our algorithm is guaranteed to correct is quite large. We believe it is possible to drastically reduce (or even eliminate) this gap, possibly by incorporating ideas from [Vid13] into our generalized setting.

We note that Viderman gave an example of an LDPC code with minimum distance 2 which was constructed from a bipartite expander graph with vertex expansion 1/2. Consequently, no decoding algorithm for this code could guarantee the correction of even one error. Viderman also introduced a linear-time decoding algorithm which could correct a constant fraction of errors on any LDPC code constructed from a bipartite expander graph with vertex expansion greater than $2/3 - 1/(6c)$, where c is the left degree of the graph. However, it is not known whether there exist even polynomial-time decoding algorithms for LDPC codes constructed from bipartite expander graphs with vertex expansion between 1/2 and $2/3 - 1/(6c)$. Viderman conjectured that such decoding algorithms do exist, and our original motivation for studying the expansion requirements of expander codes was to answer this conjecture. While we have shown that GLDPC codes constructed from bipartite expander graphs with vertex expansion greater than $1/t$ (where t is an arbitrarily large constant) have linear-time decoding algorithms, Viderman's conjecture for LDPC codes is still open.

Bibliography

[ABN+92] N. Alon, J. Bruck, J. Naor, M. Naor, and R.M. Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38(2):509–516, March 1992.

[AC88] N. Alon and F. R. K. Chung. Explicit construction of linear sized tolerant networks. *Discrete Mathematics*, 72(13):15–19, December 1988.

[AC02] N. Alon and M. Capalbo. Explicit unique-neighbor expanders. In *The 43rd Annual IEEE Symposium on Foundations of Computer Science, 2002. Proceedings*, pages 73–79, 2002.

[AC07] Reid Andersen and Fan Chung. Detecting sharp drops in PageRank and a simplified local partitioning algorithm. In *Proceedings of the 4th international conference on Theory and applications of models of computation*, TAMC'07, pages 1–12, Berlin, Heidelberg, 2007. Springer-Verlag.

[ACL06] R. Andersen, Fan Chung, and K. Lang. Local graph partitioning using PageRank vectors. In *47th Annual IEEE Symposium on Foundations of Computer Science, 2006. FOCS '06*, pages 475–486, 2006.

[ACP+92] Jeff Angel, Nancy Celniker, Steve Poulos, Audrey Terras, Cindy Trimble, and Elinor Velasquez. Special functions on finite upper half planes. In *Hypergeometric functions on domains of positivity, Jack polynomials, and applications (Tampa, FL, 1991)*, volume 138 of *Contemp. Math.*, pages 1–26. Amer. Math. Soc., Providence, RI, 1992.

[AdR97] J Arias de Reyna. Finite Fields and Ramanujan Graphs. *Journal of Combinatorial Theory, Series B*, 70(2):259–264, July 1997.

[ADS12] S. Arora, C. Daskalakis, and D. Steurer. Message-Passing Algorithms and Improved LP Decoding. *IEEE Transactions on Information Theory*, 58(12):7260–7271, December 2012.

[AEL95] N. Alon, J. Edmonds, and M. Luby. Linear time erasure codes with nearly optimal recovery. In *, 36th Annual Symposium on Foundations of Computer Science, 1995. Proceedings*, pages 512–519, October 1995.

[AGM87] N Alon, Z Galil, and V. D Milman. Better expanders and superconcentrators. *Journal of Algorithms*, 8(3):337–347, September 1987.

[Ajt87] M. Ajtai. Recursive construction for 3-regular expanders. In *, 28th Annual Symposium on Foundations of Computer Science, 1987*, pages 295–304, October 1987.

[Ajt94] M. Ajtai. Recursive construction for 3-regular expanders. *Combinatorica*, 14(4):379–416, December 1994.

[AKS87] M. Ajtai, J. Komlos, and E. Szemeredi. Deterministic Simulation in LOGSPACE. In *Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing*, STOC '87, pages 132–140, New York, NY, USA, 1987. ACM.

[AL96] N. Alon and M. Luby. A linear time erasure-resilient code with nearly optimal recovery. *IEEE Transactions on Information Theory*, 42(6):1732–1736, November 1996.

[AL08] Reid Andersen and Kevin J. Lang. An algorithm for improving graph partitions. SODA '08, pages 651–660, Philadelphia, PA, USA, 2008. Society for Industrial and Applied Mathematics.

[Alo85] N Alon. Expanders, Sorting in Rounds and Superconcentrators of Limited Depth. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 98–102, New York, NY, USA, 1985. ACM.

[Alo86] Noga Alon. Eigenvalues and expanders. *Combinatorica*, 6(2):83–96, June 1986.

[ALW01] N. Alon, A Lubotzky, and A Wigderson. Semi-direct product in groups and zig-zag product in graphs: connections and applications. In *42nd IEEE Symposium on Foundations of Computer Science, 2001. Proceedings*, pages 630–637, October 2001.

[AM84] N. Alon and V.D. Milman. Eigenvalues, Expanders And Superconcentrators. In *25th Annual Symposium on Foundations of Computer Science, 1984*, pages 320–322, October 1984.

[AM85] N Alon and V.D Milman. $\lambda 1$, Isoperimetric inequalities for graphs, and superconcentrators. *Journal of Combinatorial Theory, Series B*, 38(1):73–88, February 1985.

[Ang79] Dana Angluin. A note on a construction of Margulis. *Information Processing Letters*, 8(1):17–19, January 1979.

[AS05] A. Ashikhmin and V. Skachek. Decoding of expander codes at rates close to capacity. In *International Symposium on Information Theory, 2005. ISIT 2005. Proceedings*, pages 317–321, September 2005.

[AS06] A. Ashikhmin and V. Skachek. Decoding of Expander Codes at Rates Close to Capacity. *IEEE Transactions on Information Theory*, 52(12):5475–5485, December 2006.

[ASS08] Noga Alon, Oded Schwartz, and Asaf Shapira. An Elementary Construction of Constant-degree Expanders. *Comb. Probab. Comput.*, 17(3):319–327, May 2008.

[BASTS08] A. Ben-Aroya, O. Schwartz, and A. Ta-Shma. Quantum Expanders: Motivation and Constructions. In *23rd Annual IEEE Conference on Computational Complexity, 2008. CCC '08*, pages 292–303, June 2008.

[Ber09] Radu Berinde. *Advances in sparse signal recovery methods*. Thesis, Massachusetts Institute of Technology, 2009.

[BGT93] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Technical Program, Conference Record, IEEE International Conference on Communications, 1993. ICC '93 Geneva*, volume 2, pages 1064–1070 vol.2, May 1993.

[BHPJ13] P. Beelen, T. Hoholdt, F. Pinero, and J. Justesen. On the dimension of graph codes with Reed-Solomon component codes. In *2013 IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 1227–1231, July 2013.

[BKV+81] M. Blum, R. M. Karp, O. Vornberger, C. H. Papadimitriu, and M. Yannakakis. The complexity of testing whether a graph is a superconcentrator. *Information Processing Letters*, 13(45):164–167, 1981.

[BL04] Y. Bilu and Nathan Linial. Constructing expander graphs by 2-lifts and discrepancy vs. spectral gap. In *45th Annual IEEE Symposium on Foundations of Computer Science, 2004. Proceedings*, pages 404–412, October 2004.

[BL06] Yonatan Bilu and Nathan Linial. Lifts, Discrepancy and Nearly Optimal Spectral Gap. *Combinatorica*, 26(5):495–519, October 2006.

[BM01] D. Burshtein and G. Miller. Expander graph arguments for message-passing algorithms. *IEEE Transactions on Information Theory*, 47(2):782–790, February 2001.

[BP73] LA Bassalygo and MS Pinsker. The complexity of an optimal non-blocking commutation scheme without reorganization. *Problemy Peredaci Informacii*, 9(1):84–87, 1973.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Computer Networks and ISDN Systems*, 30(17):107–117, April 1998.

[BPZ99] J. Boutros, O. Pothier, and G. Zemor. Generalized low density (Tanner) codes. In *1999 IEEE International Conference on Communications, 1999. ICC '99*, volume 1, pages 441–445 vol.1, 1999.

[Buc86] M. Buck. Expanders and Diffusers. *SIAM Journal on Algebraic Discrete Methods*, 7(2):282–304, April 1986.

[Bur08] D. Burshtein. On the Error Correction of Regular LDPC Codes Using the Flipping Algorithm. *IEEE Transactions on Information Theory*, 54(2):517–530, February 2008.

[BZ82] EL Blokh and Victor V Zyablov. Linear concatenated codes. *Moscow, USSR: Nauka*, 1982.

[BZ02] A. Barg and G. Zemor. Error exponents of expander codes. *IEEE Transactions on Information Theory*, 48(6):1725–1729, June 2002.

[BZ04] A. Barg and G. Zemor. Error Exponents of Expander Codes under Linear-Complexity Decoding. *SIAM Journal on Discrete Mathematics*, 17(3):426–445, January 2004.

[BZ05a] A. Barg and G. Zemor. Concatenated codes: serial and parallel. *IEEE Transactions on Information Theory*, 51(5):1625–1634, May 2005.

[BZ05b] A. Barg and G. Zemor. Multilevel expander codes. In *International Symposium on Information Theory, 2005. ISIT 2005. Proceedings*, pages 1315–1319, September 2005.

[BZ06] A. Barg and G. Zemor. Distance properties of expander codes. *IEEE Transactions on Information Theory*, 52(1):78–90, January 2006.

[Che70] Jeff Cheeger. A lower bound for the smallest eigenvalue of the Laplacian. In *Problems in analysis (Papers dedicated to Salomon Bochner, 1969)*, pages 195–199. Princeton Univ. Press, Princeton, N. J., 1970.

[Chi92] Patrick Chiu. Cubic Ramanujan graphs. *Combinatorica*, 12(3):275–285, September 1992.

[Chu79] F.R.K. Chung. On concentrators, superconcentrators, generalizers, and nonblocking networks. *Bell System Technical Journal, The*, 58(8):1765–1777, October 1979.

[Chu89] F. R. K. Chung. Diameters and eigenvalues. *Journal of the American Mathematical Society*, 2(2):187–196, 1989.

[Chu97] Fan Chung. *Spectral graph theory*. Number 92 in Regional Conference Series in Mathematics. American Mathematical Soc., 1997.

[Chu07] Fan Chung. The heat kernel as the pagerank of a graph. *Proceedings of the National Academy of Sciences*, 104(50):19735–19740, December 2007.

[CL02] Yeow Meng Chee and San Ling. Highly Symmetric Expanders. *Finite Fields and Their Applications*, 8(3):294–310, July 2002.

[CMS08] Sebastian M. Cioab, M. Ram Murty, and Peter Sarnak. Expander graphs and gaps between primes. *Forum Mathematicum*, 20(4):745–756, July 2008.

[CNVM10] S.K. Chilappagari, D.V. Nguyen, B. Vasic, and M.W. Marcellin. On Trapping Sets and Guaranteed Error Correction Capability of LDPC Codes and GLDPC Codes. *IEEE Transactions on Information Theory*, 56(4):1600–1611, April 2010.

[CPT+93] Nancy Celniker, Steven Poulos, Audrey Terras, Cindy Trimble, and Elinor Velasquez. Is there life on finite upper half planes? In *A tribute to Emil Grosswald: number theory and related analysis*, volume 143 of *Contemp. Math.*, pages 65–88. Amer. Math. Soc., Providence, RI, 1993.

[CRVW02] Michael Capalbo, Omer Reingold, Salil Vadhan, and Avi Wigderson. Randomness Conductors and Constant-degree Lossless Expanders. In *Proceedings of the Thirty-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 659–668, New York, NY, USA, 2002. ACM.

[CT12] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. John Wiley & Sons, November 2012.

[CV15] Karthekeyan Chandrasekaran and Ameya Velingker. Constructing Ramanujan Graphs Using Shift Lifts. *arXiv:1502.07410 [cs, math]*, February 2015. arXiv: 1502.07410.

[CW89] A. Cohen and A. Wigderson. Dispersers, deterministic amplification, and weak random sources. In *, 30th Annual Symposium on Foundations of Computer Science, 1989*, pages 14–19, October 1989.

[DMT04] S. Dihidar, S.W. McLaughlin, and P. Tetali. On the trade-off between rate and performance of expander codes on AWGN channels. In *International Symposium on Information Theory, 2004. ISIT 2004. Proceedings*, pages 6–, June 2004.

[Dod84] Jozef Dodziuk. Difference equations, isoperimetric inequality and transience of certain random walks. *Transactions of the American Mathematical Society*, 284(2):787–794, 1984.

[DSV03] Giuliana Davidoff, Peter Sarnak, and Alain Valette. *Elementary Number Theory, Group Theory and Ramanujan Graphs*. Cambridge University Press, January 2003.

[ETV99] T. Etzion, A. Trachtenberg, and A. Vardy. Which codes have cycle-free Tanner graphs? *IEEE Transactions on Information Theory*, 45(6):2173–2181, September 1999.

[FL96] Keqin Feng and Wen-Ch'ing Winnie Li. Spectra of Hypergraphs and Applications. *Journal of Number Theory*, 60(1):1–22, September 1996.

[FMS+07] J. Feldman, T. Malkin, R.A. Servedio, C. Stein, and M.J. Wainwright. LP Decoding Corrects a Constant Fraction of Errors. *IEEE Transactions on Information Theory*, 53(1):82–89, January 2007.

[For66] G. David Forney. *Concatenated Codes*. The MIT Press, Cambridge, Mass., MIT Press, December 1966.

[Fri93] Joel Friedman. Some geometric aspects of graphs and their eigenfunctions. *Duke Mathematical Journal*, 69(3):487–525, 1993.

[Fri03] Joel Friedman. A Proof of Alon's Second Eigenvalue Conjecture. In *Proceedings of the Thirty-fifth Annual ACM Symposium on Theory of Computing*, STOC '03, pages 720–724, New York, NY, USA, 2003. ACM.

[FS05] Jon Feldman and Cliff Stein. LP Decoding Achieves Capacity. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '05, pages 460–469, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[FWK05] J. Feldman, M.J. Wainwright, and D.R. Karger. Using linear programming to Decode Binary linear codes. *IEEE Transactions on Information Theory*, 51(3):954–972, March 2005.

[FZ11] A. Frolov and V. Zyablov. Upper and lower bounds on the minimum distance of expander codes. In *2011 IEEE International Symposium on Information Theory Proceedings (ISIT)*, pages 1397–1401, July 2011.

[Gal63] Robert G. Gallager. *Low-Density Parity-Check Codes*. 1963.

[GG79] Ofer Gabber and Zvi Galil. Explicit constructions of linear size superconcentrators. In *, 20th Annual Symposium on Foundations of Computer Science, 1979*, pages 364–370, October 1979.

[GG81] Ofer Gabber and Zvi Galil. Explicit constructions of linear-sized superconcentrators. *Journal of Computer and System Sciences*, 22(3):407–420, June 1981.

[GI02] Venkatesan Guruswami and Piotr Indyk. Near-optimal Linear-time Codes for Unique Decoding and New List-decodable Codes over Smaller Alphabets. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 812–821, New York, NY, USA, 2002. ACM.

[GI05] V. Guruswami and P. Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 51(10):3393–3400, October 2005.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, first edition edition, January 1979.

[Goe97] Michel X. Goemans. Semidefinite programming in combinatorial optimization. *Mathematical Programming*, 79(1-3):143–161, October 1997.

[GUV09] Venkatesan Guruswami, Christopher Umans, and Salil Vadhan. Unbalanced Expanders and Randomness Extractors from Parvaresh-Vardy Codes. *J. ACM*, 56(4):20:1–20:34, July 2009.

[HJ06] T. Hoholdt and J. Justesen. Graph Codes with Reed-Solomon Component Codes. In *2006 IEEE International Symposium on Information Theory*, pages 2022–2026, July 2006.

[HJ11] Tom Høholdt and Jørn Justesen. The Minimum Distance of Graph Codes. In Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology*, number 6639 in Lecture Notes in Computer Science, pages 201–212. Springer Berlin Heidelberg, January 2011.

[HJ14] Tom Høholdt and Jørn Justesen. On the sizes of expander graphs and minimum distances of graph codes. *Discrete Mathematics*, 325:38–46, June 2014.

[HLW06] Shlomo Hoory, Nathan Linial, and Avi Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006.

[JH04] J. Justesen and T. Hoholdt. From concatenated codes to graph codes. In *IEEE Information Theory Workshop, 2004*, pages 13–16, October 2004.

[JL97] Bruce W. Jordan and Ron Livne. Ramanujan local systems on graphs. *Topology*, 36(5):1007–1024, September 1997.

[JL03] H. Janwa and A. K. Lal. On Tanner Codes: Minimum Distance and Decoding. *Applicable Algebra in Engineering, Communication and Computing*, 13(5):335–347, February 2003.

[JM85] S Jimbo and A Maruoka. Expanders Obtained from Affine Transformations. In *Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing*, STOC '85, pages 88–97, New York, NY, USA, 1985. ACM.

[JM87] Shuji Jimbo and Akira Maruoka. Expanders obtained from affine transformations. *Combinatorica*, 7(4):343–355, December 1987.

[Kah92] N. Kahale. On the second eigenvalue and linear expansion of regular graphs. In *33rd Annual Symposium on Foundations of Computer Science, 1992. Proceedings*, pages 296–303, October 1992.

[Kah95] Nabil Kahale. Eigenvalues and Expansion of Regular Graphs. *J. ACM*, 42(5):1091–1106, September 1995.

[Kas05a] Martin Kassabov. KAZHDAN CONSTANTS FOR SLn(). *International Journal of Algebra and Computation*, 15(05n06):971–995, October 2005.

[Kas05b] Martin Kassabov. Symmetric Groups and Expander Graphs. *arXiv:math/0505624*, May 2005. arXiv: math/0505624.

[Kas07] Martin Kassabov. Symmetric groups and expander graphs. *Inventiones mathematicae*, 170(2):327–354, August 2007.

[Kat93] Nicholas M. Katz. Estimates for Soto-Andrade sums. *Journal fr die reine und angewandte Mathematik*, 438:143–162, 1993.

[Kim05] Saejoon Kim. Generalized minimum distance iterative decoding of Tanner codes. *IEEE Communications Letters*, 9(8):738–740, August 2005.

[Kla84] M. Klawe. Limitations on Explicit Constructions of Expanding Graphs. *SIAM Journal on Computing*, 13(1):156–166, February 1984.

[KLN06] Martin Kassabov, Alexander Lubotzky, and Nikolay Nikolov. Finite simple groups as expanders. *Proceedings of the National Academy of Sciences*, 103(16):6116–6119, April 2006.

[KPS85] Richard Karp, Nicholas Pippenger, and Michael Sipser. A time-randomness tradeoff. In *AMS Conference on Probabilistic Computational Complexity*, 1985.

[KSR08] Christine Kelley, Deepak Sridhara, and Joachim Rosenthal. Zig-zag and replacement product graphs and LDPC codes. *Advances in Mathematics of Communications*, 2(4):347–372, November 2008.

[Li92] Wen-Ch'ing Winnie Li. Character sums and abelian Ramanujan graphs. *Journal of Number Theory*, 41(2):199–217, June 1992.

[LL06] Nathan Linial and Eran London. On the expansion rate of Margulis expanders. *Journal of Combinatorial Theory, Series B*, 96(3):436–442, May 2006.

[LM05] Wen-Ching Winnie Li and Yotsanan Meemark. Ramanujan graphs on cosets of PGL2(Fq). *Finite Fields and Their Applications*, 11(3):511–543, August 2005.

[LMSS98] M. Luby, M. Mitzenmacher, A. Shokrollahi, and D. Spielman. Analysis of Low Density Codes and Improved Designs Using Irregular Graphs. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, pages 249–258, New York, NY, USA, 1998. ACM.

[LMSS01] M.G. Luby, M. Mitzenmacher, M.A. Shokrollahi, and D.A. Spielman. Improved low-density parity-check codes using irregular graphs. *IEEE Transactions on Information Theory*, 47(2):585–598, February 2001.

[LPS86] A Lubotzky, R Phillips, and P Sarnak. Explicit Expanders and the Ramanujan Conjectures. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, STOC '86, pages 240–246, New York, NY, USA, 1986. ACM.

[LPS88] A. Lubotzky, R. Phillips, and P. Sarnak. Ramanujan graphs. *Combinatorica*, 8(3):261–277, September 1988.

[LR04] Kevin Lang and Satish Rao. A Flow-Based Method for Improving the Expansion or Conductance of Graph Cuts. In Daniel Bienstock and George Nemhauser, editors, *Integer Programming and Combinatorial Optimization*, pages 325–337. Springer Berlin Heidelberg, January 2004.

[LS93] L. Lovasz and M. Simonovits. Random walks in a convex body and an improved volume algorithm. *Random Structures & Algorithms*, 4(4):359–412, 1993.

[LTZ15] Anthony Leverrier, Jean-Pierre Tillich, and Gilles Zemor. Quantum Expander Codes. *arXiv:1504.00822 [quant-ph]*, April 2015. arXiv: 1504.00822.

[Lub94] Alex Lubotzky. *Discrete Groups, Expanding Graphs and Invariant Measures*. Springer Science & Business Media, August 1994.

[Lub12] Alexander Lubotzky. Expander graphs in pure and applied mathematics. *Bulletin of the American Mathematical Society*, 49(1):113–162, 2012.

[LW93] A. Lubotzky and B. Weiss. Groups and expanders. In Joel Friedman, editor, *Expanding Graphs: Proceedings of a DIMACS Workshop, May 11-14, 1992*, pages 95–109. American Mathematical Soc., January 1993.

[Mar73] G. A. Margulis. Explicit Construction of Concentrators. *Problemy Peredachi Informatsii*, 9(4):71–80, 1973. (English translation Problems of Information Transmission, Plenum, New York (1975), p. 325-332).

[Mar82] G. A. Margulis. Explicit constructions of graphs without short cycles and low density codes. *Combinatorica*, 2(1):71–78, March 1982.

[Mar84] GA Margulis. Arithmetic groups and graphs without short cycles. In *6th Intern. Symp. on Information Theory, Tashkent, abstracts*, volume 1, pages 123–125, 1984.

[Mar88] G. A. Margulis. Explicit group-theoretic constructions of combinatorial schemes and their applications in the construction of expanders and concentrators. *Akademiya Nauk SSSR. Institut Problem Peredachi Informatsii Akademii Nauk SSSR. Problemy Peredachi Informatsii*, 24(1):51–60, 1988. (English translation Problems of Information Transmission, Plenum, New York (1988), p. 39-46).

[Mes86] J.-F. Mestre. La methode des graphes. Exemples et applications. pages 217–242, Katata, Japan, June 1986.

[Mor94] M. Morgenstern. Existence and Explicit Constructions of q + 1 Regular Ramanujan Graphs for Every Prime Power q. *Journal of Combinatorial Theory, Series B*, 62(1):44–62, September 1994.

[MS86] R.J. McEliece and L. Swanson. On the decoder error probability for Reed - Solomon codes (Corresp.). *IEEE Transactions on Information Theory*, 32(5):701–703, September 1986.

[MSS13] Adam Marcus, Daniel A. Spielman, and Nikhil Srivastava. Interlacing Families I: Bipartite Ramanujan Graphs of All Degrees. *arXiv:1304.4132 [math]*, April 2013. arXiv: 1304.4132.

[MSS15] Adam Marcus, Daniel Spielman, and Nikhil Srivastava. Interlacing families I: Bipartite Ramanujan graphs of all degrees. *Annals of Mathematics*, 182(1):307–325, July 2015.

[MW02] Roy Meshulam and Avi Wigderson. Expanders from Symmetric Codes. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 669–677, New York, NY, USA, 2002. ACM.

[MW04] Roy Meshulam and Avi Wigderson. Expanders In Group Algebras. *Combinatorica*, 24(4):659–680, September 2004.

[Nil91] A. Nilli. On the second eigenvalue of a graph. *Discrete Mathematics*, 91(2):207–210, August 1991.

[Nil04] A. Nilli. Tight estimates for eigenvalues of regular graphs. *The Electronic Journal of Combinatorics [electronic only]*, 11(1):Research paper N9, 4 p.–Research paper N9, 4 p., 2004.

[Pin73] Mark S. Pinsker. On the complexity of a concentrator. In *7th International Teletraffic Conference*, 1973.

[Pip77] N. Pippenger. Superconcentrators. *SIAM Journal on Computing*, 6(2):298–304, June 1977.

[Piz90] Arnold K. Pizer. Ramanujan graphs and Hecke operators. *Bulletin (New Series) of the American Mathematical Society*, 23(1):127–137, July 1990.

[Rot06] Ron Roth. *Introduction to Coding Theory*. Cambridge University Press, February 2006.

[RS06] R.M. Roth and V. Skachek. Improved Nearly-MDS Expander Codes. *IEEE Transactions on Information Theory*, 52(8):3650–3661, August 2006.

[RSU01] T.J. Richardson, M.A. Shokrollahi, and R.L. Urbanke. Design of capacity-approaching irregular low-density parity-check codes. *IEEE Transactions on Information Theory*, 47(2):619–637, February 2001.

[RSW04] Eyal Rozenman, Aner Shalev, and Avi Wigderson. A New Family of Cayley Expanders (?). In *Proceedings of the Thirty-sixth Annual ACM Symposium on Theory of Computing*, STOC '04, pages 445–454, New York, NY, USA, 2004. ACM.

[RSW06] Eyal Rozenman, Aner Shalev, and Avi Wigderson. Iterative Construction of Cayley Expander Graphs. *Theory OF Computing*, 2(1):91–120, 2006.

[RTS06] E. Rom and A. Ta-Shma. Improving the Alphabet-Size in Expander-Based Code Constructions. *IEEE Transactions on Information Theory*, 52(8):3695–3700, August 2006.

[RTV05] Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom Walks in Biregular Graphs and the RL vs. L Problem. Technical Report TR05-022, 2005.

[RTV06] Omer Reingold, Luca Trevisan, and Salil Vadhan. Pseudorandom Walks on Regular Digraphs and the RL vs. L Problem. In *Proceedings of the Thirty-eighth Annual ACM Symposium on Theory of Computing*, STOC '06, pages 457–466, New York, NY, USA, 2006. ACM.

[RU01] T.J. Richardson and R.L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Transactions on Information Theory*, 47(2):599–618, February 2001.

[RU08] Tom Richardson and Rüdiger Urbanke. *Modern Coding Theory*. Cambridge University Press, March 2008.

[RVW00] O. Reingold, S. Vadhan, and A. Wigderson. Entropy waves, the zig-zag graph product, and new constant-degree expanders and extractors. In *41st Annual Symposium on Foundations of Computer Science, 2000. Proceedings*, pages 3–13, 2000.

[RVW02] Omer Reingold, Salil Vadhan, and Avi Wigderson. Entropy Waves, the Zig-Zag Graph Product, and New Constant-Degree Expanders. *Annals of Mathematics*, 155(1):157–187, January 2002.

[Sch80] Klaus Schmidt. Asymptotically invariant sequences and an action of SL (2,Z) on the 2-sphere. *Israel Journal of Mathematics*, 37(3):193–208, September 1980.

[Sha48] C. E. Shannon. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423, July 1948.

[Sho04] A. Shokrollahi. Capacity-approaching codes on the q-ary symmetric channel for large q. In *IEEE Information Theory Workshop, 2004*, pages 204–208, October 2004.

[Sip86] Michael Sipser. Expanders, randomness, or time versus space. In Alan L. Selman, editor, *Structure in Complexity Theory*, number 223 in Lecture Notes in Computer Science, pages 325–329. Springer Berlin Heidelberg, 1986. DOI: 10.1007/3-540-16486-3_108.

[Ska11] V. Skachek. Correcting a Fraction of Errors in Nonbinary Expander Codes With Linear Programming. *IEEE Transactions on Information Theory*, 57(6):3698–3706, June 2011.

[Spi96] D.A. Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1731, November 1996.

[SR03] V. Skachek and R.M. Roth. Generalized minimum distance iterative decoding of expander codes. In *2003 IEEE Information Theory Workshop, 2003. Proceedings*, pages 245–248, March 2003.

[SS94] M. Sipser and D.A. Spielman. Expander codes. In *, 35th Annual Symposium on Foundations of Computer Science, 1994 Proceedings*, pages 566–576, November 1994.

[SS96] M. Sipser and D.A. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 42(6):1710–1722, November 1996.

[ST04] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, STOC '04, pages 81–90, New York, NY, USA, 2004. ACM.

[Tan81] R.M. Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27(5):533–547, September 1981.

[Tan84] R. Tanner. Explicit Concentrators from Generalized N-Gons. *SIAM Journal on Algebraic Discrete Methods*, 5(3):287–293, September 1984.

[TSU12] A. Ta-Shma and C. Umans. Better Condensers and New Extractors from Parvaresh-Vardy Codes. In *2012 IEEE 27th Annual Conference on Computational Complexity (CCC)*, pages 309–315, June 2012.

[TSUZ01] Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Loss-less Condensers, Unbalanced Expanders, and Extractors. In *Proceedings of the Thirty-third Annual ACM Symposium on Theory of Computing*, STOC '01, pages 143–152, New York, NY, USA, 2001. ACM.

[TSUZ07] Amnon Ta-Shma, Christopher Umans, and David Zuckerman. Lossless Condensers, Unbalanced Expanders, And Extractors. *Combinatorica*, 27(2):213–240, March 2007.

[TVZ82] M. A. Tsfasman, S. G. Vldutx, and Th. Zink. Modular curves, Shimura curves, and Goppa codes, better than Varshamov-Gilbert bound. *Mathematische Nachrichten*, 109(1):21–28, January 1982.

[Vid13] Michael Viderman. Linear-time Decoding of Regular Expander Codes. *ACM Trans. Comput. Theory*, 5(3):10:1–10:25, August 2013.

[WZ93] Avi Wigderson and David Zuckerman. Expanders That Beat the Eigenvalue Bound: Explicit Construction and Applications. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Theory of Computing*, STOC '93, pages 245–251, New York, NY, USA, 1993. ACM.

[WZ99] Avi Wigderson and David Zuckerman. Expanders That Beat the Eigenvalue Bound: Explicit Construction and Applications. *Combinatorica*, 19(1):125–138, January 1999.

[Zem01] G. Zemor. On expander codes. *IEEE Transactions on Information Theory*, 47(2):835–837, February 2001.

[ZP75] Zyablov, V. V. and M. S. Pinsker. Estimation of the error-correction complexity for Gallager low-density codes. *Problems of Information Transmission*, 11(1):23–36, 1975.

[Zuc90] D. Zuckerman. General weak random sources. In *, 31st Annual Symposium on Foundations of Computer Science, 1990. Proceedings*, pages 534–543 vol.2, October 1990.

[Zuc91] D. Zuckerman. Simulating BPP using a general weak random source. In *, 32nd Annual Symposium on Foundations of Computer Science, 1991. Proceedings*, pages 79–89, October 1991.

[Zuc96] D. Zuckerman. Simulating BPP using a general weak random source. *Algorithmica*, 16(4-5):367–391, October 1996.