Lecture 6: Expander Codes

Lecturer: Prahladh Harsha *Scribe: Hovav Shacham*

In today's lecture, we will discuss the application of expander graphs to error-correcting codes. More specifically, we will describe the construction of linear-time decodable expander codes due to Sipser and Spielman. We begin with some preliminaries on error-correcting codes.

(Several of the proofs presented in this lecture are adapted from the lecture notes of Venkatesan Guruswami's course on Codes and Pseudo-random objects [**?**]).

6.1 Error Correcting Codes – Preliminaries

We first recall the definition of error-correcting codes. For more information, see, e.g., the excellent survey by Guruswami [**?**]. Suppose Alice wants to send a k-bit message to Bob over a noisy channel (i.e., the channel flips some bits of the message). In order for Bob to recover (decode) the correct message even after the channel corrupts the transmitted word, Alice instead of sending the k-bit message, encodes the message by adding several redundancy bits and instead sends an n-bit encoding of it across the channel. The encoding is chosen in such a way that a decoding algorithm exists to recover the message from a codeword that has not been corrupted too badly by the channel. (What this means depends on the specific application.)

More formally, a code $\mathcal{C}$ is specified by a injective map $E : \Sigma^k \to \Sigma^n$ that maps k-symbol messages to n-symbol codewords where Σ is the underlying set of symbols called the alphabet. For the most most of today's lecture, we will only consider the binary alphabet (i.e., $\Sigma = \{0, 1\}$). The map E is called the *encoding*. The image of E is the set of codewords of the code $\mathcal{C}$. Some times, we abuse notation and refer to the set of codewords $\{E(x)|x \in \{0,1\}^k\}$ as the code. k is refered to as the *message-length* of the code $\mathcal{C}$ while n is called the *block-length*.

The *rate* of the code, (denoted by r), is the ratio of the logarithm of number of codewords to the block-length n, i.e, $r(C) = \log(\#\text{codewords})/n = k/n \leq 1$. Informally, a rate is the amount of information (about the message) contained in each bit of the codeword.

The (Hamming) distance $\Delta(x, y)$ between any two strings $x, y \in \{0, 1\}^n$ is the number of bits in which they differ. The *distance* of the code, denoted by d, is the minimum Hamming distance of any two of its codewords, i.e., $d(C) = \min_{x,y\in\mathcal{C}} \Delta(x, y)$. The *relative distance*, denoted by δ, is the ratio of the distance to the block-length, i.e. $\delta = d/n$.

We will refer to a code $\mathcal{C}$ that maps k message-bits to n codewords with distance d as a (n, k, d)-code.

If the distance of codeword is large and if not too many codeword bits are corrupted by the channel (more precisely if not more than $d/2$ bits are flipped), then we can uniquely decode the corrupted codeword by picking the codeword with the smallest Hamming distance from it. Note that for this unique decoding to work, it must be the case that there are no more than $d/2$ errors caused by the channel. Furthermore, clearly the above algorithm is not efficient as we need to search over the entire space $\{0,1\}^n$ to find the nearest codeword. In today's lecture, we will describe a code (based on expanders) for which this decoding can be done efficiently (more precisely in time linear in the length of the codeword).

Linear Code A code $\mathcal{C}$ is *linear* if 0^n is (a codeword) in $\mathcal{C}$ and if, whenever x and y are in $\mathcal{C}$, so is $x \oplus y$[1]. Linear codes are usually defined in the more general setting when the underlying alphabet for the codeword is some finite field, however for the purpose of this lecture we will restrict ourselves to the binary alphabet $\{0,1\}$. We refer to a linear code that maps k message bits to n codeword bits with distance d as a $[n,k,d]$-code.

It is an easy observation that , in a linear code, d equals the smallest Hamming weight of a non-zero codeword. A linear code can be described as an $n \times k$ generator matrix C (such that $Cm \in \{0,1\}^n$ is the codeword corresponding to a message $m \in \{0,1\}^k$), or by an $(n-k) \times n$ parity-check matrix H (such that $x \in \{0,1\}^n$ is a codeword whenever Hx equals 0^{n-k}). The existence of a generator matrix C immediately implies that the encoding time for linear codes is at most quadratic. The parity check-matrix H implies that a $[n,k,d]$-code can be described by the $n-k = n(1-r)$ linear constraints (i.e., the columns of H) imposed on the codewords bits. Conversely, if a linear code is described by a set of t (consistent) linear equations, then the rate of the code is at least $r \geq 1 - t/n$.

A random $[n,k,d]$-linear code is formed by choosing a random $n \times k$ generator matrix C of zeros and ones (where each entry is chosen to be either 0 or 1 with probability 1/2) and defining the code $\mathcal{C}$ accordingly (i.e, setting $E(x) = Cx, \forall x \in \{0,1\}^k$). The Gilbert-Varshamov bound states that such a random linear code has (with high probability) distance δ if the block-length is at least $n \geq k/(1 - H(\delta))$ where $H(\cdot)$ is the binary entropy function $H(p) = p \log\left(\frac{1}{p}\right) + (1-p)\log\left(\frac{1}{1-p}\right)$.

Theorem 6.1 (Gilbert-Varshamov)**.** *Let $\delta < \frac{1}{2}$. If $\mathcal{C}$ is a random linear code with rate at most $1 - H(\delta)$, then*

$$\text{Prob}\left[d(\mathcal{C}) \geq \delta\right] = 1 - o(1).$$

6.2 Expander based codes

We now present the expander codes of Sipser and Spielman [**?**]. These expander codes have the advantage that the decoding process is very efficient, can be performed in linear time on a single processor or in $\log n$ time by a parallel processor with n-machines. We describe the construction due to Zémor [**?**] which is a slight modification of the original construction of Sipser and Spielman [**?**]

6.2.1 Zémor Codes – construction

The family of expander codes is parametrized by a fixed-size code $\mathcal{C}$ with some small block-length d and a family expander graph $\mathcal{G}_n$ on n vertices with constant degree d. The construction due to Zémor is specified by a process that converts a fixed-size code $\mathcal{C}$ of block-length d and an expander graph $\mathcal{G}$ on n vertices and degree d into a new code $\mathcal{Z} = \mathcal{Z}(\mathcal{C}, \mathcal{G})$ with

[1] where $\oplus$ refers to the bit-wise xor operation

block-length nd. The rate and distance of the new code $\mathcal{Z}$ depend on the rate and distance r and δ of $\mathcal{C}$, and on the spectral expansion λ of $\mathcal{G}$.

The construction proceeds as follows. Take the graph $\mathcal{G} = (V, E)$, and duplicate its vertex set V into left and right vertex sets L and R. For an edge $e \in E$ with endpoints u and v in V, connect both u_{L} in L to v_{R} in R and v_{L} in L to u_{R} in R. This creates a d-regular $2n$-vertex bipartite graph $\mathcal{G}'$. Since the graph G' is constructed from an expander G with spectral expansion λ, the expander mixing lemma can be applied to this graph. In other words, for all sets $S \subset L$ and $T \subset R$, we have that

$$\left| e(S,t) - d\frac{|S||T|}{n} \right| \leq \lambda d \sqrt{|S||T|},$$

where $e(S,T)$ represents the number of edges between the sets S and T.

Now we will use the new graph $\mathcal{G}'$ to describe codewords in $\mathcal{Z}$. These codewords are dn bits long, and $\mathcal{G}'$ has dn edges. We will associate each bit position in the codeword with an edge in $\mathcal{G}'$. It is thus possible to consider a codeword x as an assignment of ones and zeroes to the edges in $\mathcal{G}'$. Moreover, for each vertex v (on either side) we can consider the d-bit restriction x_v of x to edges incident on v. For this purpose, we assume some canonical ordering among the edges incident on any vertex. If $x \in \{0,1\}^{dn}$ and $e_1, \ldots, e_d$ are the edges incident on vertex v, then $x_v = (x_{e_1}, \ldots, x_{e_d}) \in \{0,1\}^d$. Observe that this association also works if x is not a proper codeword: for example, if it has been corrupted by the channel.

Given these mappings, the code itself is actually quite simple. A bit string $x \in \{0,1\}^{dn}$ is a codeword in $\mathcal{Z}$ if, for each vertex $v \in L \cup R$, x_v is a codeword in $\mathcal{C}$. Note that each edge label must satisfy constraints imposed by both its left and right endpoint.

If $\mathcal{C}$ is a linear code, then so is $\mathcal{Z}$. The following theorem characterizes $\mathcal{Z}$ in terms of $\mathcal{C}$ and $\mathcal{G}$.

Theorem 6.2. *Suppose $\mathcal{C}$is a $[d, rd, \delta d]$-code with rate $r < 1/2$ and $\mathcal{G}$ is a d-regular expander on n vertices with spectral expansion $\lambda < \delta$. Then $\mathcal{Z}(\mathcal{C}, \mathcal{G})$ is a $[dn, (2r-1)dn, \delta(\delta - \lambda)dn]$-code.*

Proof. It is clear that block length of $\mathcal{Z}$ is dn.

The codes $\mathcal{C}$ and $\mathcal{Z}$ are linear, and so we can consider their rates in terms of constraints in their parity-check matrices. Since $\mathcal{C}$ has block-length d and rate r, its parity-check matrix imposes at most $d - rd = (1-r)d$ constraints on codewords. These constraints are imposed in $\mathcal{Z}$ at each of the $2n$ vertices, so the total number of constraints in $\mathcal{Z}$ is at most $2n(1-r)d = \big(1 - (2r-1)\big)nd$, and $\mathcal{Z}$'s rate is at least $(2r-1)$.

Since $\mathcal{Z}$ is linear, its distance equals the minimum Hamming weight of a non-zero codeword. Consider such a codeword x. Let X be the edges labeled 1 in x: $X = \{e \mid x_e = 1\}$, and let S and T be the sets of left- and right-hand vertices, respectively, on which edges in X are incident.

The degree of X with respect to vertices in S and T must be at least δd, since otherwise x would not locally be a $\mathcal{C}$-codeword at these vertices. With a factor of 2 to allow for the double-counting of edges, we have that

$$|X| \geq \frac{\delta d}{2}(|S| + |T|) \ . \tag{1}$$

However, by the Expander Mixing Lemma, we have

$$|X| \le e(S,T) \le \frac{d}{n}|S||T| + \lambda d\sqrt{|S||T|} \ .$$

Combining the inequalities and dividing out d yields

$$\frac{\delta}{2}(|S|+|T|) < \frac{1}{n}|S||T| + \lambda\sqrt{|S||T|} \ .$$

We can simplify this inequality by means of the following AM-GM inequality

$$|S||T| \le \frac{(|S|+|T|)^2}{4} \ , \tag{2}$$

obtaining

$$\frac{\delta}{2}(|S|+|T|) < \frac{1}{4n}(|S|+|T|)^2 + \frac{\lambda}{2}(|S|+|T|) \ ,$$

or, after some algebra,

$$|S|+|T| > 2n(\delta-\lambda) \ ; \tag{3}$$

Substituting **??** into (**??**) shows that $\mathcal{Z}$'s distance is at least $\delta(\delta-\lambda)dn$, as required. □

6.2.2 Decoding Algorithm

We now consider how one might decode corrupted codewords in the code $\mathcal{Z}(C,G)$. The algorithm we give has the considerable advantage of being *local*: at each round, algorithm choices at a node depend only values local to that node. In a distributed or multi-processing environment, these local choices can all be made in parallel. It will be evident from the algorithm below, how the decoding can be implemented in $O(\log n)$ time on a parallel machine with n machines. To obtain a linear-time decoding algorithm (on a single machine), a little more book-keeping is necessary. However, for today's lecture, we will ignore this book-keeping (the details of which can be found in [**?**]).

Algorithm 6.1 (Local Decoding for $\mathcal{Z}(\mathcal{C},\mathcal{G})$)**.** *The algorithm is given as input a corrupted codeword x, interpreted as consisting of corrupted local codewords x_v at vertices v in the left or right vertex sets L and R. It corrects these codewords locally as follows.*

1. *Set $V_0 \leftarrow L$*
2. *Set $i \leftarrow 0$*
3. *while there exists a vertex v such that $x_v \notin \mathcal{C}$ do:*
   - *(a) For each $v \in V_i$ such that $x_v \notin \mathcal{C}$, decode x_v to nearest codeword in $\mathcal{C}$*
   - *(b) If $V_i = L$, set $V_{i+1} \leftarrow R$, otherwise set $V_{i+1} \leftarrow L$.*
   - *(c) set $i \leftarrow i+1$*
4. *Return x*

Since all the choices in any single round of the algorithm can be carried out in parallel, we will analyze the algorithm in terms of the number of rounds before it converges on a codeword. In particular, we obtain the following theorem, parametrized on α, which can take values between 0 (faster convergence, but corrects less noise) and 1 (slower, but corrects more).

Theorem 6.3. *Suppose $\mathcal{C}$ is a $[d, rd, \delta d]$-code and $\mathcal{G}$ is a d-regular expander on n vertices with spectral expansion $\lambda < \delta/3$. Then for all α, $0 \leq \alpha \leq 1$, the decoding algorithm given in Algorithm* **??** *corrects $\alpha\frac{\delta}{2}(\frac{\delta}{2} - \lambda)dn$ errors in $O\left(\frac{\lg n}{\lg(2-\alpha)}\right)$ rounds.*

Proof. Because $\mathcal{Z}$ is linear, we can, wlog, assume that the closet codeword to the corrupted codeword x is the all-zeros codeword. Denote the working word after i rounds by $x^{(i)}$. Thus, $x^{(0)}$ represents the initial corrupted codeword. Let $E^{(i)}$ be the edges labeled 1 after i rounds:

$$E^{(i)} = \left\{ e \mid x_e^{(i)} = 1 \right\} \quad ,$$

Thus, $E^{(i)}$ represents the codeword bits that haven't been decoded correctly after i rounds. Let $S^{(i)}$ be the vertices in the set V_{i-1} (i.e., left or right side depending on i) with edges in $E^{(i)}$, i.e., the vertices that have not yet correctly decoded at the end of the i rounds.

$$S^{(i)} = \left\{ v \in V_{i-1} \mid E_v \cap E^{(i)} \neq \emptyset \right\} \quad .$$

where E_v represents the set of edges incident on v.

By a series of claims, we will show a geometric decline in the size of $S^{(i)}$,

$$|S^{(i+1)}| < \frac{|S^{(i)}|}{2 - \alpha} \quad ,$$

proving the theorem.

We first make some observations:

- Every edge in $E^{(i)}$ has at least one endpoint in $S^{(i)}$.
- With respect to $E^{(i)}$, every vertex in $S^{(i+1)}$ has degree at least $\delta d/2$, since otherwise the vertex would have been locally decoded to the zero codeword in the $(i+1)^{\text{th}}$ round.

Claim 6.4. *$|S^{(1)}| \leq \alpha(\frac{\delta}{2} - \lambda)n$.*

Proof. By assumption, $|E^{(0)}| \leq \alpha\frac{\delta}{2}(\frac{\delta}{2} - \lambda)dn$. By the observations above, $|E^{(0)}| \geq |S^{(1)}|\frac{\delta d}{2}$; adjoining the two inequalities proves the claim. □

Claim 6.5. *If $\delta > 3\lambda$ and $|S^{(i)}| < \alpha(\frac{\delta}{2} - \lambda)n$, then $|S^{(i+1)}| < \frac{|S^{(i)}|}{2-\alpha}$.*

Proof. Consider the edge set $E(S^{(i)}, S^{(i+1)})$. This set certainly includes all edges in $E^{(i)}$ between $S^{(i)}$ and $S^{(i+1)}$, and these number (by the observation) at least $\frac{\delta d}{2}|S^{(i+1)}|$. Thus we have

$$e(S^{(i)}, S^{(i+1)}) \geq \frac{\delta d}{2}|S^{(i+1)}| \quad .$$

By the Expander Mixing Lemma, however, we have

$$e(S^{(i)}, S^{(i+1)}) \leq \frac{d|S^{(i)}||S^{(i+1)}|}{n} + \lambda d\sqrt{|S^{(i)}||S^{(i+1)}|} \ .$$

Combining these inequalities, we obtain

$$\frac{\delta d}{2}|S^{(i+1)}| \leq \frac{d|S^{(i)}||S^{(i+1)}|}{n} + \lambda d\sqrt{|S^{(i)}||S^{(i+1)}|} \ ,$$

and, using the AM-GM inequality (**??**) with $S = S^{(i)}$ and $T = S^{(i+1)}$,

$$\frac{\delta d}{2}|S^{(i+1)}| < \frac{d|S^{(i)}||S^{(i+1)}|}{n} + \frac{\lambda d}{2}(|S^{(i)}| + |S^{(i+1)}|) \ .$$

Applying the claim precondition $|S^{(i)}| < \alpha(\frac{\delta}{2} - \lambda)n$ to the first summand gives

$$\frac{\delta d}{2}|S^{(i+1)}| < \alpha d(\frac{\delta}{2} - \lambda)|S^{(i+1)}| + \frac{\lambda d}{2}(|S^{(i)}| + |S^{(i+1)}|) \ ,$$

which we can rearrange as

$$\big(\delta - \alpha(\delta - 2\lambda) - \lambda\big)|S^{(i+1)}| < \lambda|S^{(i)}| \ . \tag{4}$$

Using the other claim precondition, $\delta > 3\lambda$, we see that

$$\delta - \alpha(\delta - 2\lambda) - \lambda = (1-\alpha)(\delta - 2\lambda) + \lambda > (1-\alpha)\lambda + \lambda = (2-\alpha)\lambda \ ,$$

which, applied to (**??**), gives $(2-\alpha)|S^{(i+1)}| < |S^{(i)}|$, which proves the claim. □

We have thus shown a geometric decline in the size of $S^{(i)}$;. Expressed in terms of $|S^{(1)}|$, $|S^{(i)}|$ drops exponentially as $1/(2-\alpha)^i$, and the algorithm will complete in $O\left(\frac{\lg n}{\lg(2-\alpha)}\right)$ rounds. □

The above code and decoding algorithm, together with explicit constructions for $\mathcal{C}$ and $\mathcal{G}$, give the following general theorem. We choose $\mathcal{C}$ to be a fixed-size linear code satisfying the Gilbert-Varshamov bound and $\mathcal{G}$ to be a family of Ramanujan graphs with $\lambda \approx 1/\sqrt{d}$ (in fact, any family of expander graphs with $\lambda = 1/o(d)$ will suffice.

Theorem 6.6 (Sipser and Spielman; Zémor)**.** *For all $0 < \delta, \epsilon < 1$ such that $1 - 2H(\sqrt{\delta}) < 1$, there exists an explicit family of codes with rate $1 - 2H(\sqrt{\delta})$ and relative distance $\delta - \epsilon$, with a decoding algorithm that corrects $\delta/4 - \epsilon$ errors in $O(\lg n)$ rounds, where n is the block-length of the code and $H(\cdot)$ is the binary entropy function.*

6.2.3 Expander codes with nearly optimal rate

The condition $1 - 2H(\sqrt{\delta}) < 1$ in Theorem **??** allows for only codes with relative distance $\delta < 0.0121$ and thus allows error-recovery from a small fraction $\approx 1\%$ of errors. Expanders and the expander mixing lemma can be used again (!!) to improve the relative distance to $1/2 - \epsilon$ and linear-time decoding upto $1/4 - \epsilon$ fraction of errors at the cost of increasing the alphabet size from binary to a large constant depending on ϵ. This construction is due to Guruswami and Indyk [**?**].

Theorem 6.7 (Guruswami Indyk [?]). *For every $0 < r, \epsilon < 1$, there is an explicit infinite family of linear codes with rate r and relative distance at least $(1 - r - \epsilon)$ such that the codes in the family can be decoded from a fraction $(1 - r - \epsilon)/2$ of errors in time linear in the block-length of the code.*

The details of this construction can be found in [?, ?].

References

[BZ] Alexander Barg and Gilles Zémor, "Error Exponents of Expander Codes", IEEE Transactions on Information Theory, 48(6):1725–1729 (2002).

[Gur1] Venkatesan Guruswami, Lecture notes for a course on Codes and Pseudorandom objects, University of Washington, Seattle. `http://www.cs.washington.edu/education/courses/590vg/03wi/notes.html`

[Gur2] Venkatesan Guruswami, "Error-correcting codes and Expander graphs", SIGACT News, 35(3): 25–41, September 2004.

[GI] Venkatesan Guruswami, Piotr Indyk, "Linear time encodable/decodable codes with near-optimal rate", To appear in IEEE Transactions on Information Theory. (Preliminary Version in STOC'02).

[SS] Michael Sipser, Daniel A. Spielman: "Expander codes". IEEE Transactions on Information Theory 42(6): 1710–1722 (1996)

[Zém] Gilles Zémor: "On Expander Codes", IEEE Transactions on Information Theory 47(2):835–837 (2001).