CSE590VG: Codes and Pseudorandom Objects January 27, 2003

Lecture 5

Lecturer: Venkatesan Guruswami *Scribe: Matt Cary*

1 Zémor's Decoding Algorithm

We will show a decoding algorithm due to Zémor [4] for the codes based on expander graphs that were described last lecture. These codes were based on a double-cover d-regular expander G, that is, a bipartite graph regular of degree d on both sides. Let L and R denote the left and right vertex sets of G, and E the edge set, and let $n = |L| = |R|$. We will use a $[d, r_0 d, \delta_0]$ code $\mathcal{C}_0$ to construct our code $\mathcal{C}$. Each bit x_e of a coded message is associated with an edge e of G. Given such a message x and a vertex v in L or R, define $x|_v$ to be the subsequence of x corresponding to all edges incident on v; define E_v the set of those edges. We summarize this in Figure 1. Our code $\mathcal{C}$ will consist of all codewords such that $x_v \in \mathcal{C}_0$ for all $v \in R, L$. Last time, we showed that $\mathcal{C}$ is a $[N = nd, (2r_0 - 1)N, \delta_0^2 N]$ code.

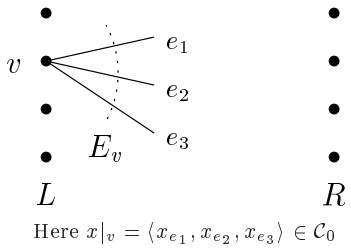


Figure 1: G and $\mathcal{C}$

The decoding algorithm alternates between the left and right vertex sets of G, correcting the codeword of $\mathcal{C}_0$ associated with each vertex of one side, then switching to the other side to do the same thing. Note that the edges associated with a vertex on one side of the graph, are not incident to any other vertex on that side; hence the order that the vertices are processed in does not matter. In fact, we can view (or even implement) the vertex processing of each round in parallel. See Figure 2 for the precise algorithm, and an example of correcting a the all-zeros codeword that has some erroneous 1-bits.

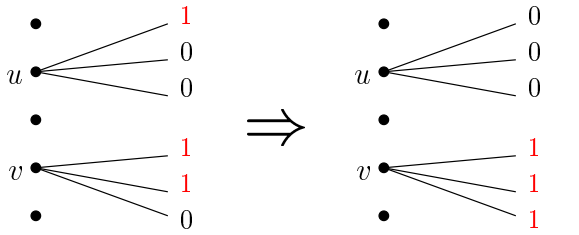
Theorem 1 *If G is a Ramanujan graph of sufficiently high degree, for any $\alpha < 1$, the decoding algorithm can correct $\frac{\alpha\delta_0^2}{4}(1-\varepsilon)N$ errors, in $O(\log n)$ rounds (where the big-O notation hides a dependence on α). This can be implemented in linear time on a single processor; on n processors each round can be implemented in constant time.*

Proof By linearity, and the fact that our decoding algorithm is insensitive to the values of the edges, we can assume the transmitted codeword is the all-zeros vector. Let the received codeword be x. We will be considering the edge-sets corresponding to the incorrect (value 1) bits of the codeword at various rounds of the decoding. Let $x = x^{(0)}$ be the initial value of the codeword, $x^{(1)}$ the value after the first round of decoding on L, $x^{(2)}$ the value after the first round of decoding on R, etc. Let $V^{(i)} = L$ or R as appropriate, depending on which side of G was used in the i^{th}round. Let $X^{(i)} = \{e \in E \mid x_e^{(i)} = 1\}$, and $S^{(i)} = \{v \in V^{(i)} \mid E_v \cap X^{(i+1)} \neq \emptyset\}$, that is, $S^{(i)}$ is the set of vertices that did not successfully decode their $\mathcal{C}_0$ codeword in the i^{th}round.

The Decoding Algorithm

$V^{(0)} = L, V^{(1)} = R$
$i = 0$
while $\exists x_v \notin \mathcal{C}_0$
  for $x_v \in L$ or R **do**
  for each $v \in V^{(i)}$ with $x_v \notin \mathcal{C}_0$
    Decode x_v to nearest $\mathcal{C}_0$ codeword
  $i \leftarrow i+1, V^{(i+1)} \leftarrow V^{(i-1)}$

A Round of Decoding



u has been corrected; the number of errors in v was too large for $\mathcal{C}_0$, so it was corrected to an incorrect value, from the point of view of $\mathcal{C}$. Those incorrect edges will be incident on different vertices in R, and hopefully will be corrected in the next round. In the notation of the proof of Theorem 1, $X^{(0)}$ is the set of edges that are labeled with 1; $S^{(0)} = \{v\}$ as u is incident on only correct edges, but v is incident to at least one incorrect edge.

Figure 2: Decoding $\mathcal{C}$

We will show that $|S^{(0)}|, |S^{(1)}|, |S^{(2)}|, \ldots$ form a decreasing sequence, in fact,

$$|S_{i+1}| \leq \frac{1}{2-\alpha}|S_i|. \tag{1}$$

As we assume $\alpha < 1$, this forms a geometric decreasing sequence, so that as $|S_i| < n$, at most $\log_{2-\alpha} n$ rounds are necessary to correct all errors. Furthermore, $\sum |S_i| = n \sum \frac{1}{(2-\alpha)^i} = O(n)$, so that as we can implement the i^{th}round in $O(|S_i|)$ time, we have that the total sequential running time is linear.

The task now is to justify (1). The key to this must lie in the expander property of G, in particular, the isoperimetric inequality: for all $A \subset L, B \subset R$,

$$\left| E(A,B) - d\frac{|A| \cdot |B|}{n} \right| \leq \lambda \sqrt{|A| \cdot |B|},$$

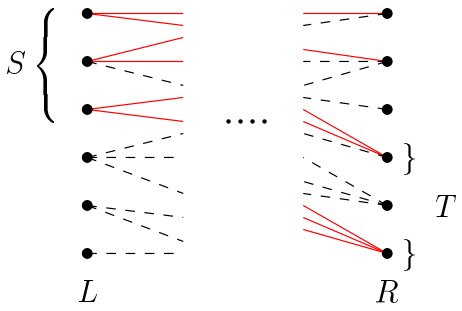
where λ is the second largest eigenvalue of G. Recall that this inequality says that in an expander graph, the number of edges between any two sets is close to what one would expect if the edges of the where chosen randomly, after the selection of the sets.

To ease notation, let $S = S^{(i)}$, $T = S^{(i+1)}$, and $Y = X^{(i+1)}$. Each vertex in S must have at least $\delta_0 d$ edges of Y leaving it, otherwise that vertex would have decoded its $\mathcal{C}_0$ word to zero. Each element v of T is a vertex in R that doesn't decode its codeword, so that the number of bad edges that enter v must be at least $\delta_0 d/2$; otherwise v would have correctly decoded. See Figure 3. The intuition with expander graphs is that now matter how S was chosen, the edges out of S look random, spreading out the bad edges into multiple nodes so that they will be corrected in the next round. More precisely, each vertex in T has lots of edges going to vertices in S, and the isoperimetric inequality tells us that this only happens if T is small.

Lemma 2 *Suppose $d\delta_0 \geq 3\lambda, S \subset L, |S| \leq \alpha n(\delta_0/2 - \lambda/d), T \subset R$, and $Y \subset E$ such that*

1. *every edge in Y leaves S, and*
2. *every $v \in T$ has $\geq \delta_0 d/2$ edges in Y.*

Then $|T| \leq \frac{1}{2-\alpha}|S|$.



The upper vertices of R are not in T, as they would be correctly decoded as a $\mathcal{C}_0$ codeword.

Figure 3: S and T

Proof

$$\begin{aligned}
|T|\frac{\delta_0 d}{2} &\le |E(S,T)| && \text{by assumption (ii) on } |T|,\\
&\le d\frac{|S|\cdot|T|}{n} + \lambda\sqrt{|S|\cdot|T|} && \text{by the isoperimetric inequality,}\\
&\le \alpha\left(\frac{\delta_0 d}{2} - \lambda\right)|T| + \lambda\frac{|S|+|T|}{2} && \text{by assumed bound on } |S|, \text{ and the arithmetic-geometric mean}
\end{aligned}$$

Thus,

$$\begin{aligned}
|T| &\le \frac{\lambda|S|}{(1-\alpha)\delta_0 d + \lambda(2\alpha-1)}\\
&\le \frac{|S|}{2-\alpha} && \text{if } \delta_0 d \ge 3\lambda.
\end{aligned}$$

■

Recall that G is Ramanujan, so that $\lambda \le 2\sqrt{d}$. Hence the assumption $\delta_0 d \ge 3\lambda$ of the lemma is reasonable, as $\lambda \sim \sqrt{d}$. In fact, as we have assumed we can take the degree of G to be arbitrarily large, we can assume $\lambda \ll \delta_0 d$.

By the bound on the number of erroneous bits in $x^{(0)}$ assumed for the theorem,

$$\begin{aligned}
|X^{(0)}| &\le \alpha\frac{\delta_0^2}{4}(1-\varepsilon)N\\
&\le \frac{\delta_0 d}{2}\alpha n\left(\frac{\delta_0}{2} - \frac{\lambda}{d}\right),
\end{aligned}$$

if $2\lambda/(\delta_0 d) \le \varepsilon$, which holds from the fact noted above that we can take $\lambda \ll \delta_0 d$. As each vertex in $S^{(0)}$ was incident (before being corrected) to at least $\delta_0 d/2$ edges of $X^{(0)}$,

$$|S^{(0)}| \le \frac{2|X^{(0)}|}{\delta_0 d} \le \alpha n\left(\frac{\delta_0}{2} - \frac{\lambda}{d}\right).$$

Thus the lemma can be applied inductively on $X^{(0)}, X^{(1)}$, etc., giving $|S^{(i+1)}| \le \frac{1}{2-\alpha}|S^{(i)}|$, so that the size of uncorrected nodes decreases geometrically with each round. This immediately gives the

$O(\log n)$ bound on the number of rounds. The linear sequential time claim follows from noting that a clever implementation can process round i in time $|S_i|$, so that the total time is $\sum |S_i| = O(n)$.

■

The original proof of Sipser and Spielman [2] only guaranteed a fraction $\sim d_0^2/48$ of errors could be corrected. The reason why Zémor's method gains is that his proof only asserts that the size of $S^{(i)}$ shrinks with each round; the number of erroneous edges may increase during a round. The proof of Sipser and Spielman showed that the number of errors decreased with each round; as this is a stronger condition, they could not prove that decoding worked with as large a fraction of errors.

Spielman [3] showed that a code (based on?) $\mathcal{C}$ could be encoded in linear time, so that together with Theorem 1, we have an asymptotically good family of linear-time encodable and decodable codes.

2 Improved Error-Correction: Distance Amplification

Recall that if a code $\mathcal{C}$ is able to correct a ρ fraction of errors, the Plotkin and Singleton bounds imply

$$0 < \rho < \frac{1}{4}, \qquad \text{if } \mathcal{C} \text{ is a binary code, and}$$
$$0 < \rho < \frac{1}{2}, \qquad \text{if } \mathcal{C} \text{ is over an arbitrary alphabet.}$$

The code described in the previous section has rate $r = 1 - 2H(\sqrt{4\rho})$; thus a bound on ρ, found by solving $r > 0$, has

$$\rho < (H^{-1}(1/2)^2/4 \sim 10^{-3},$$

which is not very close to the general bounds on ρ. Can we improve the error correction, perhaps by a code over a larger alphabet?

2.1 Distance Amplification

This question was addressed by Alon et. al. [1] with a technique known as *distance amplification*. Recall that the first expander-graph based code we developed used the expander graph as a parity check matrix. The distance amplification construction can be viewed as using instead the expander graph as a generator matrix.

Recall that to achieve $\rho = 1/2 - \varepsilon$, we need $\delta \geq 1 - 2\varepsilon$. Suppose we have an asymptotically good code $\mathcal{C}$ of rate r and relative distance $\geq \alpha$. We will use the double cover G of a d-regular expander graph to define a new code $\mathcal{C}^*$ over the alphabet 2^d, of rate k/d and relative distance $(1-\beta)$, for any β such that d satisfies $d \in \Omega\big(1/(\alpha\beta)\big)$. In particular, if $\mathcal{C}$ is a $\big[n, k, \alpha n\big]_2$ code, $\mathcal{C}^*$ will be a $\big(n, k/d, (1-\beta)n\big)_{q=2^d}$ code. Note that $\mathcal{C}^*$ will not be a linear code; it will only be "$GF(2)$-linear" or "additive", i.e. if c_1, c_2 are codewords then so is $c_1 + c_2$. Let G have n nodes on each side. Each codeword of $c^* \in \mathcal{C}^*$ will correspond to a codeword $c \in \mathcal{C}$ in the following way. Assign each character of c to a node on the left of G (L), and each character of c^* to a node on the right of G (R). Hence each $v \in L$ is associated with a bit $c_v \in \{0,1\}$, which it sends along each incident edge to vertices in R. Each vertex $u \in R$ collects the bits incoming on its edges into an ordered tuple (any fixed ordering will do), to form the character $c^*_u \in [q = 2^d]$. See Figure 4. To summarize, we have

Claim 3 *Given a d-regular Ramanujan expander and a $\big[n, k, \alpha n\big]_2$ code, for any $\beta > 4/(\alpha d)$ and $q = 2^d$, there exists a*

$$\left(n, \frac{k}{d}, (1-\beta)n\right)_q$$

$GF(2)$-linear code.

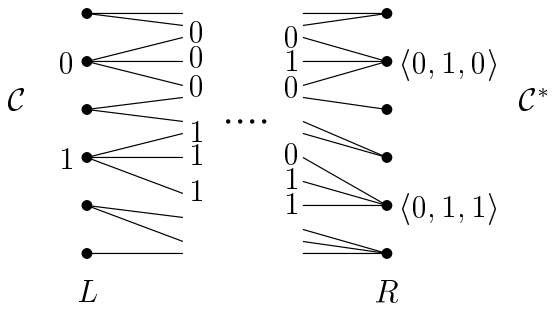


Figure 4: $\mathcal{C}$ and $\mathcal{C}^*$

Because of the one-to-one correspondence between $\mathcal{C}^*$ and $\mathcal{C}$, the two codes have the same number of codewords, namely 2^k. But as $\mathcal{C}^*$ is over a larger alphabet we must compute its rate in term of $q = 2^d$, that is, $2^k = \left(2^d\right)^{k/d}$, showing that the dimension of $\mathcal{C}^*$ is k/d as claimed. The claim of the distance of $\mathcal{C}^*$ is nearly as straightforward, and can be shown using the familiar technique of bounding the minimum weight of a codeword of $\mathcal{C}^*$. We have to convince ourselves that this will give the minimum distance in $\mathcal{C}^*$, as it is only $GF(2)$-linear, but this is straightforward and left to the reader.

Claim 4 *The minimum weight of a codeword in $\mathcal{C}^*$ is $(1-\beta)n$.*

Proof Take any nonzero codeword of $\mathcal{C}$, which has $\geq \alpha n$ ones by the minimum distance of $\mathcal{C}$. These ones correspond to a set $S \subset L$. Let $B \subset R$ be the nodes incident to a node in S; each node of T will correspond to a nonzero character of $\mathcal{C}^*$, hence it's enough to show that B is a $(1-\beta)$ fraction of R. A different way of saying this is that for any set T of size at least βn, $E(S,T) \neq \emptyset$. Suppose this is not true, that is, $E(S,T) = \emptyset$. Then, by the isoperimetric inequality, assuming G is Ramanujan,

$$\frac{d|S| \cdot |T|}{n} \leq 2\sqrt{d} \cdot \sqrt{|S| \cdot |T|}$$

implying

$$|T| \leq \frac{4}{d\alpha} n$$

but as $|T| > \beta n$,

$$d \leq \frac{4}{\alpha\beta}.$$

Hence, if $d > 4/(\alpha\beta)$, this cannot occur. ■

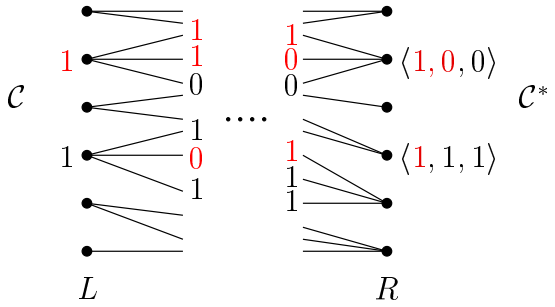
If we let $d = O(1/\varepsilon)$, we get linear time encodable codes over $GF(2^{O(1/\varepsilon)})$, of rate $\Omega(\varepsilon)$ and relative distance $1-\varepsilon$. Now, the question is, can we find and algorithm to decode to the desired $1/2 - \varepsilon$ fraction of error?

2.2 Decoding $\mathcal{C}^*$

Recall that $\mathcal{C}^*$ was constructed in two stages, by first encoding the message as a codeword in $\mathcal{C}$, then by using the expander graph G to create the final $\mathcal{C}^*$ codeword. The decoding process will be

similar: the received word is sent through the expander graph to give a word in $\{0,1\}^n$, and then the decoding algorithm for $\mathcal{C}$ is applied to reconstruct the message.

It remains to define how to send the received word the the expander graph. Recall that in coding, each bit of the message is associated with a node on the left side of the graph, this bit is replicated on each edge incident to the node, and collected into characters on the right side of the graph. Applying this process in reverse presents the problem that a left side node will not receive the same message on all of its incoming edges: errors in the received word will cause the inputs to the left side to be inconsistent, as shown in Figure 5. The easiest solution is to have each left-hand node take the majority of the bits on its incoming edges; it turns out this simple algorithm works quite well. The intuition is based, yet again, on the random-like quality of the expander. The neighbors of a left-hand node look random, so that the fraction of incorrect neighbors of a particular will usually approximate the fraction of incorrect nodes on the entire right-hand side. Hence, if slightly more than half of the right-hand nodes are correct, most of the errors will be corrected simply by passing them through the graph. The remaining errors are few enough that the error correction of $\mathcal{C}$ will be able to fix them.



The upper node on the left is incorrect because of the errors on the right-hand side, while the lower node was repaired.

Figure 5: Decoding $\mathcal{C}^*$

Claim 5 *Let d be given as in the construction of $\mathcal{C}^*$, and let γ be the error fraction that $\mathcal{C}$ can correctly decode from. If $d \geq \frac{4}{\gamma\epsilon^2}$, the decoding procedure outlined above can correct a $(1-\varepsilon)$ fraction of errors.*

Proof Let y be the received word with at least $(1-\varepsilon)n$ correct symbols, and let z be the result of sending y through G (but not yet decoding it with the $\mathcal{C}$ algorithm). Let c^* be the closest codeword of $\mathcal{C}^*$ to y. Let T be the right-hand nodes of G corresponding to *correct* characters of y, i.e. the u_i such that $y_i = c_i^*$. Let c be the $\mathcal{C}$ codeword corresponding to $\mathcal{C}^*$; our goal is to show z and c differ on at most γn bits.

Let $S = \{i \mid v_i \in L \text{ has } \leq d/2 \text{ neighbors in } T\}$, so that $z_i \neq c_i$ for all $v_i \in S$, as the majority rule will not decide on the correct bit for z_i. Hence, we need to show that $|S| < \gamma n$. We immediately have that

$$E(S,T) \leq \frac{d}{2}|S|$$

by the definition of S. By the isoperimetric inequality we have

$$\begin{aligned} E(S,T) &\geq \frac{d|S|\cdot|T|}{n} - \lambda\sqrt{|S|\cdot|T|} \\ &\geq d|S|\left(\frac{1}{2}+\varepsilon\right) - \lambda\sqrt{|S|n}, \end{aligned}$$

bounding $|T|$ by the number of received errors in the first term, and simply by n in the second term. Hence,

$$\varepsilon d|S| \leq \lambda\sqrt{|S|n},$$

implying the following,

$$|S| \leq \left(\frac{\lambda}{\varepsilon d}\right)^2 n.$$

Hence as G is Ramanujan, $\lambda \leq 2\sqrt{d}$, so that if $d \geq 4/(\gamma\varepsilon^2)$ as assumed, $|S| \leq \gamma n$ as required. ■

Note to get the distance bound, we needed $d \gtrsim 4/(\alpha\varepsilon)$. Now, we need $d \gtrsim 4/(\gamma\varepsilon^2)$, so that, like with previous asymptotically good constructions, we loose a square in the decoding.

If $\mathcal{C}$ is linear-time encodable and decodable, then one can show that $\mathcal{C}^*$ is linear-time encodable and decodable as well. This is a reasonable assertion, as we are assuming d is a constant, and the encoding and decoding algorithms are quite simple, but we will not formally prove this.

To summarize, we now have rate $\Omega(\varepsilon^2)$ codes with linear-time encoding and decoding algorithms to a $(1/2-\varepsilon)$ fraction of errors, over alphabets of size $2^{O(1/\varepsilon^2)}$. This approaches the Singleton bound, which asserts the rate is at most ε for such decoding capability.

References

[1] Noga Alon, Jehoshua Bruck, Joseph Naor, Moni Naor, and Ronny Roth. Construction of asymptotically good low-rate error-correcting codes through pseudo-random graphs. *IEEE Transactions on Information Theory*, 38:509–516, 1992.

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

[3] Daniel Spielman. Linear-time encodable and decodable error-correcting codes. *IEEE Transactions on Information Theory*, 42(6):1723–1732, 1996.

[4] Gillés Zémor. On expander codes. *IEEE Transactions on Information Theory*, 47(2):835–837, 2001.