6.440 Essential Coding Theory April 1, 2013

Lecture 14: Graph-Based Codes

Lecturer: Madhu Sudan *Scribe: George Xing*

1 Review

Up to this point, our class has focused on three major themes:

- *Limits on the existence of codes.* We've seen existence bounds, such as the Gilbert-Varshamov bound, which often have had corresponding codes which meet the bound tightly. (In our example, we used the probabilistic method to show that codes meeting the Gilbert-Varshamov bound exist.) We've also seen non-existence bounds.
- *Algebraic codes.* We've constructed explicit examples of codes, including the Reed-Solomon code, the Reed-Muller code, Algebraic Geometric codes, and the BCH code. These codes closely or exactly meet the bounds above.
- *Decoding algorithms.* For the codes we've described above, we've also been able to provide efficient decoding algorithms, which are able to correct many errors (say, half the distance of the code).

2 A Look Ahead

In the near future, we'll be exploring new topics:

- *Linear time algorithms.* Up to now, we've been primarily concerned with encoding and decoding algorithms that run in any polynomial time. However, we will now devote some attention to linear time algorithms. In particular, the graph-based codes we will be going over today will have linear encoding and decoding time.
- *Sublinear time algorithms.* We can't hope to decode an entire codeword in less than linear time. But suppose we had a very large codeword, and we wanted to decode only a small block of the message. We would like a coding scheme where it's possible to decode this small block by checking only some sublinear subset of the entire codeword. Codes for which this is possible are known as locally decodale, or locally testable, and have applications to computational complexity.
- *Codes in practice.* In practice, we may be more concerned with actually parameter values, as opposed to the asymptotics of these parameters. In

particular, we will be going over the tradeoff between gap and capacity: given a channel with flipping rate p, suppose we impose that our code have rate $1 - H(p) - \epsilon$. How high of a capacity can we achieve with such a code?

3 Gallager Codes

This idea comes from a 1963 paper due to Gallager. Suppose we had a linear binary error correcting code whose parity check matrix H was sparse, meaning that its matrix has few nonzero values.

What do we mean by few? Recall that for a linear binary code $[n, k, d]_2$, our parity matrix $H \in \mathbb{F}_2^{n \times (n-k)}$ has n rows and $m = n - k$ columns. Each of the n rows corresponds to a coordinate in our codeword, and each of the m columns corresponds to a constraint. (More specifically, since our codewords are the set $C = \{y | yH = 0\}$, each column imposes an additional orthogonality constraint on possible vectors $y \in \mathbb{F}_2^n$.) We would like to impose a constraint on H which makes it so it contains a number of ones which is linear in n, the codeword size. In particular, we will impose that the matrix is (c, d)**-bounded**:

Definition 1 *A matrix is* (c, d)**-bounded** *if each of its rows contains at most c ones and each of its columns contains at most d ones.*

For ease of analysis, we will further constrain H to be (c, d)**-regular**:

Definition 2 *A matrix is* (c, d)**-regular** *if each of its rows contains exactly c ones and each of its columns contains exactly d ones.*

(Note: We are holding c and d constant as n goes to infinity, so that the matrix has a linear number of ones.)

We now establish a correspondence between binary $n \times m$ matrices and bipartite graphs. For any binary matrix H of size $n \times m$, we associate it with a bipartite graph using the natural bipartite adjacency matrix interpretation:

- Our bipartite graph will be (V, E), where the vertices $V = L \cup R$ with $|L| = n, |R| = m$, and the edges $E \subseteq L \times R$. We label the vertices in L as $L_1, L_2, \ldots, L_n$ and the vertices in R as $R_1, R_2, \ldots, R_m$.
- For each $1 \leq i \leq n$ and $1 \leq j \leq m$, we add the edge (L_i, R_j) iff $H_{ij} = 1$.

For any binary vector y of size n, we can associate it with a subset of vertices $S \subseteq L$ such that $L_i \in S \iff y_i = 1$. Which S correspond to y's that are codewords? Recall that y is a codeword iff $yH = 0$. Notice that y times the j-th column of H is exactly the parity of the number of vertices in S which are neighbors of R_j. Thus, S corresponds to a codeword iff for each R_j, the number of neighbors of R_j which lie in S is even.

Let us now examine the rate of the code associated with this graph/matrix. By equating the sum of the degrees of vertices in L to the sum of degrees of vertices in R, we obtain

$$cn = dm.$$

Then, the rate of the code is

$$\frac{n-m}{n} = 1 - \frac{m}{n} = 1 - \frac{c}{d}.$$

Thus, for any desired rate, we can select c, d which achieves this rate. The following theorem gives a bound on the capacity of codes that achieve this rate:

Theorem 3 *Fix $0 < R < 1$. As $c \to \infty$ and $d = \frac{c}{1-R}$, the code corresponding to a (c, d)-regular matrix has rate R and distance approaching $H^{-1}(1-R)$. (This H^{-1} refers to the inverse entropy function not the inverse of the matrix.)*

We did not prove this theorem in class. However, we provided some motivation for the theorem. Consider a belief propagation algorithm. Roughly, we have individual agents, one per vertex on a graph. Each agent independently calculates his belief for what the value of some function is. Then, each agent propogates his information to his neighbors; from this, each agent can update on his belief. Iterating this process, we should see each agent converge to the same beliefs. This algorithm "seems" to be able to decode random errors well, so it makes sense that from the vertices in R, agents might be able to find the value of the nearest keyword.

Another more explicit intuition for why this theorem is true woul be the following: given the graph and an encoded word (with possible errors), we can see which constraints are violated. Each violated constraint tells us that there is an error somewhere in its neighborhood. If these neighborhoods are of constant size, this gives us a lot more information than if they were of linear size. Somehow, we are able to leverage this extra information.

4 Tanner Codes

The following comes from a 1984 paper due to Tanner. They are a generalization of the Gallager codes.

Consider the way in which we determine if a constraint in R is violated or not in the Gallager code. We check the total parity of the bits corresponding to its neighbors is 0. Equivalently, we might say that we check if the bitstring defined by its neighbors belongs to the parity Hamming code (the one with a parity bit, which has distance 2).

We can generalize this to instead say that the bit string (in $\mathbb{F}_2^d$) must be a codeword in another linear code on words of size d. As we will see, this construction can be used to create explicit codes which satisfy bounds like those of Theorem 3.

Tanner's paper also consider a certain graph parameter which was useful for analyzing distance in the corresponding code. He considered the **girth** of a graph, which is the length of the graph's smallest cycle. He showed that a larger girth implied a larger (lower bound on) the distance of a corresponding code. For explicitly constructed graphs, the best girth lower bounds were around $\frac{1}{2}\log n$, which corresponded to a distance of at least $\sqrt{n}$.

5 Expanders

These ideas are primarily due to a 1994 paper by Sipser and Spielman. This paper considered a new parameter on the graph, which was the "expansion" of a graph. (This will be fleshed out in more detail below.) They were able to give a code with a linear-time decoding algorithm which corrected a $\Omega(n)$ amount of errors.

To do this, they used expanders. We are still considering (c, d)-regular graphs, with the same notation we used before. To define an expander, we first define the notion of a neighborhood:

Definition 4 *Given a subset of vertices $S \subseteq L$, its* **neighborhood** $\Gamma(S)$ *is the set of vertices in R which are adjacent to at least one vertex in S; that is,*

$$\Gamma(S) = \{v \in R | \exists u \in S \ s.t. \ (u, v) \in E\}.$$

Now we are ready to define an expander.

Definition 5 *A graph is a (γ, δ)-***expander** *if for all $S \subseteq L$ such that $|S| < \delta n$, we have*

$$|\Gamma(S)| \geq \gamma |S|.$$

More informally, this condition says that any small enough subset of L has a neighborhood that is at least γ times larger than it; that is, it expands. Let's try to get some easy bounds on what sort of parameters are possible given c, d. Clearly, we must have $0 \leq \gamma \leq c$, since each vertex in L has exactly c neighbors. Now suppose, we fix γ. If we select m/γ vertices from L, then their neighborhood can only be of size m, so it can't expand by more than γ. Therefore, $\delta n \leq m/\gamma$, or using the relation $cn = dm$, $\delta \leq \frac{c}{d\gamma}$.

5.1 Random Expanders

Sipser and Spielman were able to show that by selecting a random (c, d)-regular graph, that with high probability, the associated graph is a (γ, δ)-expander, with $\gamma \to (c-1)$, $\delta \to\approx \frac{1}{d}$. These almost meet the bounds above. In fact, $c - 1$ is a roughly tight upper bound on γ, since if we start with a single vertex in L, then take its neighborhood, then take the vertices in L adjacent to some vertex in this neighborhood, each new vertex added this way can only contribute $c-1$ new neighbors.

To generate a random (c, d)-regular graph, one can simply start with a graph with cn vertices on the left, and dm vertices on the right, and create a random perfect matching on this graph (by permuting the right-side vertices). Then, the vertices in the left are contracted into n groups of c, and the vertices on the right are contracted into m groups of d. After this contraction, we wind up with a graph on $n + m$ vertices, which may not necessarily be simple. We may rejection sample until the resultant graph is simple, or we may simply delete parallel edges; with high probability, there are not very many collisions, and the results we show will still apply in this case.

5.2 Explicitly Described Expanders

Until 1999, the best explicit constructions of expanders achieved a value of $\gamma \to c/2$. In 2000, Capalbo, Reingold, Vadhan, and Wigderson achieved explicit constructions achieveing $\gamma \to c(1-\epsilon)$ (which have some corresponding $\delta > 0$), for any $\epsilon > 0$, for sufficiently large graphs. Given a particular ϵ, these constructions are polynomial in n; however, they are not also polynomial in $1/\epsilon$.

5.3 Expanders for Decoding

In this section, we argue that the code corresponding to an expander graph has good distance. To do this, we can show that any words with not enough ones cannot be codewords. In particular, if we show that for any $S \subseteq L$ such that $|S| < k$ for some upper bound k, there exists a violated constraint (that is, a vertex in R with an odd number of neighbors in S), then we have shown that the graph has distance at least k. Thus, we are interested in the following set:

Definition 6 *For $S \subseteq L$, let*

$$\Gamma^{odd}(S) = \{v \in R| \ \#(\{v \in S|(u,v) \in E\}) \text{ is odd}\}$$

be the set of neighbors of S which are adjacent to an odd number of vertices of S.

In particular, we would like to show that if $|S| < \delta n$, then $|\Gamma^{odd}(S)| \geq 1$. We will show this by lower bounding the size of the following set:

Definition 7 *For $S \subseteq L$, let*

$$\Gamma^{unique}(S) = \{v \in R| \ \#(\{v \in S|(u,v) \in E\}) = 1\}$$

be the set of neighbors of S which are adjacent to exactly one vertex in S.

Note that $\Gamma^{unique}(S) \subseteq \Gamma^{odd}(S)$ for all S. Thus, it suffices to show that $|\Gamma^{unique}(S)| \geq 1$ when $|S| < \delta n$ instead. Let us define a unique-expander as follows:

Definition 8 *A graph is a (γ, δ)-**unique-expander** if for all $S \subseteq L$ such that $|S| < \delta n$, we have*

$$|\Gamma^{unique}(S)| \geq \gamma |S|.$$

In our proof, we will use the following lemma:

Lemma 9 *If G is a (c,d)-regular (γ, δ)-expander, then G is a $(2\gamma - c, \delta)$-unique expander.*

Proof Fix S with $|S| < \delta n$. Partition $\Gamma(S)$ into $U = \Gamma^{unique}(S), D = \Gamma(S) \backslash U$. We have

$$|U| + |D| = |\Gamma(S)| \geq \gamma |S|.$$

Consider the edges incident to vertices in S. There are cS of them, since each vertex in S has degree c. Each such edge is also incident to either a vertex in U or in D. Each vertex in U has exactly 1 such incident edge, while each edge in D has at least 2. Thus,

$$cS \geq |U| + 2|D|.$$

Multiplying the first inequality by 2 and adding, we get

$$2|U| + 2|D| + cS \geq 2\gamma|S| + |U| + 2|D|,$$

or

$$|U| \geq (2\gamma - c)|S|,$$

as desired. ■

Notice that with our explicit $\gamma \to c/2$ construction, $2\gamma - c \leq 0$, the previous Lemma gives us no guarantees. But, for $\gamma > c/2$, this Lemma implies that the code associated with G has distance $\geq \delta n$, when G is a (c, d)-regular, (γ, δ)-expander.

5.4 Improving the Lemma for the $\gamma = c/2$ construction

Consider the Tanner graph interpretation, where whether a constraint in R is violated is determined through some linear code with distance Δ, rather than just parity. Since this code has distance Δ, if a vertex in R has some nonzero number of neighbors in S which is strictly less than Δ, then its constraint must be violated. Thus, we let

$$\Gamma^{\Delta-1}(S) = \{v \in \Gamma(S) |\ \#\{u \in S | (u, v) \in E\} \leq \delta - 1\}.$$

We also define a $(\gamma, \delta) - (\Delta - 1)$-expander similarly:

Definition 10 *G is a $(\gamma, \delta) - (\Delta - 1)$-**expander** iff for all $S \subseteq L$ such that $|S| < \delta n$, $|\Gamma^{\Delta-1}(S)| \geq \gamma|S|$.*

We show that if G is a (γ, δ)-expander (and (c, d)-regular), then it is also a $(\frac{\Delta\gamma - c}{\Delta - 1}, \delta) - (\Delta - 1)$-expander. The proof of this is similar to the proof of the Lemma in the previous section. Let $U = \Gamma^{\Delta-1}(S), D = \Gamma(S) \backslash U$. We have

$$|U| + |D| = |\Gamma^{\Delta-1}(S)| \geq \gamma|S|.$$

By counting edges incident to S, we have

$$c|S| \geq |U| + \Delta|D|.$$

We can multiply the first inequality by Δ and rearrange to get $|U| \geq \frac{\Delta\gamma - c}{\Delta - 1}|S|$, as desired.

5.5 Decoding Expander Codes

To decode, we can use the FLIP algorithm: given a vector $x_1, \ldots, x_n$, we want to find the codeword with closest Hamming distance to it. We associate each x_i with a vertex in L. Then, each constraint in R is either satisfied or not satisfied. While there exists a vertex $u \in L$ such that it has more unsatisfied neighbors than satisfied neighbors, we flip the bit associated with such a vertex (we may pick arbitrarily).

This algorithm is guaranteed to terminate, because every time we make such a flip, the number of unsatisfied constraints strictly decreases. However, we have yet to analyze its runtime or verify its correctness; this will be covered next lecture.