6.897 Algorithmic Introduction to Coding Theory November 4, 2002

Lecture 16

Lecturer: Madhu Sudan *Scribe: April Rasala*

1 Parity Check Matrix and Decoding

Recall that a binary linear code can be specified by an $n \times m$ parity check matrix H. Given H, the code is defined to be $C = \{y \mid y \cdot H = 0\}$.

The question we are interested in is: Can we use the parity check matrix for C to help us decode efficiently? The parity check matrix does give information about where an error occurred. For example, suppose we receive a vector $r \notin C$, then $r \cdot H \neq 0$. Suppose that the jth entry of $r \cdot H$ is non-zero. Then it follows that the received vector dotted with the jth column of H, $r \cdot H_j$, must be non-zero. Therefore an error must have occurred in some position i of r, such that $H_{ij} = 1$. While this might narrow down the set of possible indices where an error occurred, H_{ij} could be non-zero for around $n/2$ indices. In particular, if we choose the entries of H at random in order to produce a code that meets the Gilbert-Varshamov bound, then it is very likely that each column of H will have around $n/2$ non-zero entries.

2 Low Density Parity Check Matrices

Suppose that we know that H has low density. In particular, suppose that we have a upper bound on the number of non-zero entries in each column, then the parity check matrix *can* provide a lot of information about where an error might have occurred.

In fact, we require that H only has a constant number of ones in each column.

Definition 1 *Matrix H is d-sparse if every column has at most d ones in it.*

Since we wish to build a family of asymptotically good codes which can use information provided by the parity check matrix to aid in decoding, we define what it means for an infinite family of matrices to have low density.

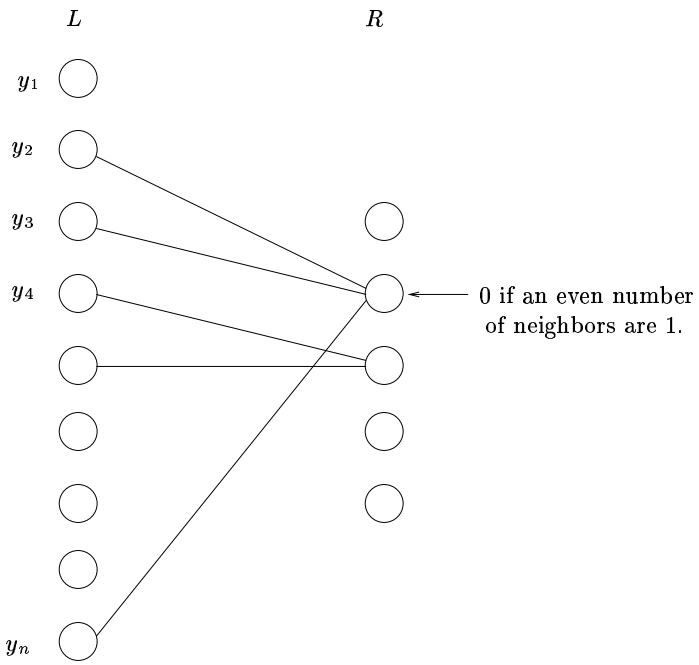
Definition 2 *A family of matrices $\{H_{n,m}\}_{n,m}$ is low density if there exists a constant d such that every matrix $H_{n,m} \in \{H_{n,m}\}_{n,m}$ is d-sparse.*

A family of codes generated by low density parity check matrices is referred to as a LDPC code. Thus far, our discussion has been motivated by trying to construct codes that are easily decodable. But how good are these codes? If the parity check matrix $H_{n,m}$, is of full rank, then the rate of the code is $(n - m)/n$. What about the distance? It turns out that there exist LDPC codes that meet the Gilbert-Varshamov bound. This was proven in 1963 by Gallager in the same paper in which he introduced LDPC codes. Gallager also showed that there exist LDPC codes that correct a constant fraction of errors in linear time.[2]

3 Graph-Theoretic Approach to LDPC codes

Subsequent work on LDPC codes approached the problem from a graph-theoretic framework. Suppose we view H as an adjacency matrix for an $n \times m$ bipartite graph $G = (L, R, E)$. Thus, an edge $(i, j) \in E$ if and only if $H_{i,j} = 1$.

How do we get codes from an adjacency matrix? Recall that the code $C = \{y \mid y \cdot H = 0\}$. Therefore, suppose we label the vertices on the left with the entries of y. We label the vertices on the right with the parity of its neighbors on the left (i.e. $v \in R$ gets label 0 if an even number of its neighbors on the left are set to 1 and v gets label 1 otherwise).



What sorts of graphs have a low density adjacency matrix?

Definition 3 *Graph $G = (L, R, E)$ is (c, d)-regular if every vertex $v \in L$ has degree at most c and every vertex $v \in R$ has degree at most d.*

Notice that insisting that each vertex on the right hand side of G has degree exactly d, corresponds to requiring the adjacency matrix H to have d entries that are 1 in each column. Likewise, requiring each vertex on the left hand side to have degree c means that each row of the adjacency matrix has c entries. This is actually stronger than we required in our definition of a low density family of parity check matrices.

At this point it doesn't seem that difficult to construct a low density parity check matrix or a (c, d)-regular graph. But when does a graph G produce a good code? To answer this question we'll actually consider when a graph doesn't produce a good code. First, since we want our code to have a good minimum distance we want to insure that there are no low weight code words in our code. A low weight code word w would correspond to labeling only a small set of vertices on the left with 1's in such a way that each vertex on the right is labeled 0. Thus, we want G to be such that for every small subset S of vertices on the left, there is a vertex on the right adjacent to an odd number of the vertices in S. If this condition is met for all subsets S such that $|S| \leq \delta n$, then we know that the code has minimum distance at least δn, since there can't exist lower weight non-zero codewords.

Lower bounding the number of neighbors of a set S with odd degree into S is tricky. However, people have considered graphs in which each small set S is guaranteed to have a large number of neighbors that are adjacent to exactly one node in S.

Definition 4 *For a set $S \subseteq L$, let $\Gamma(S) \subset R$ be the set of nodes adjacent to at least one node in S and let $\Gamma_1(S) \subseteq \Gamma(S)$ be the set of nodes adjacent to exactly one node in S. We refer to $\Gamma(S)$ as the* neighbors *of S and $\Gamma_1(S)$ as the* unique neighbors *of S.*

We are interested in showing that there exist graphs G such that every subset of small enough size has at least one unique neighbor. It turns out that graphs known as "expanders" achieve this property.

Definition 5 *A graph $G = (L, R, E)$ is a (γ, δ)-expander if every set $S \subset L$ with $|S| \leq \delta n$ has $|\Gamma(S)| \geq \gamma |S|$.*

4 Expander based codes

If G is (c,d)-regular and a (γ,δ)-expander, how large can the expansion factor γ be? Certainly $\gamma \leq c$, the degree of nodes on the left. For a random graph, typically $\gamma > c-1$. Furthermore, when $\gamma > c/2$, it is possible to show that there exists a $\delta > 0$ such that G is a (γ,δ) expander. This also implies that there exists a code with relative distance at least δ. In fact, if $\gamma > \frac{3c}{4}$, then it can be shown that there exists a decoding algorithm that can correct some ϵ fraction of errors in linear time. Although we won't be able to show that $\epsilon = \delta/2$, which is the best we could hope for, we will show that $\epsilon = \frac{\delta}{2c}$.

To prove a lower bound on the minimum distance of a code based on a (c,d)-regular (γ,δ)-expanding graph G, we first show that any small set of nodes on the left must have some unique neighbor on the right. Thus, if we label the vertices on the left according to a vector of weight at most δn, then some node on the right must be a unique neighbor of the non-zero vertices on the left. Therefore no low weight vector is such that every node on the right has even parity.

Lemma 1 *If $G = (L,R,E)$ is a (c,d)-regular graph and a (γ,δ)-expander, then for all $S \subset L$ such that $|S| \leq \delta n$, $|\Gamma_1(s)| > (2\gamma - c)\cdot|S|$.*

Proof Let $U = |\Gamma_1(S)|$ be the number of unique neighbors of S. Let D be the number of neighbors of S with at least two edges into S. Thus $U + D = |\Gamma(S)| \geq \gamma|S|$ since G is a (γ,δ)-expander. The number of edges incident to nodes in S is equal to $c\cdot|S|$. Each unique neighbor of S is incident to exactly one of these edges. Every other neighbor of a node in S is incident to at least two nodes in S. Thus $U + 2D \leq c|S|$ which implies that $2D \leq c\cdot|S| - U$. Therefore

$$\begin{aligned} U + D &\geq \gamma|S| \\ 2U &\geq 2\gamma|S| - 2D \\ 2U &\geq 2\gamma|S| - (c|S| - U) \\ U &\geq (2\gamma - c)|S| \end{aligned}$$

■

Thus, as long as $\gamma > c/2$, every small set of nodes on the left hand side has at least one unique neighbor. We use this to prove a lower bound on the minimum distance of the associated code.

Lemma 2 *If $G = (L,R,E)$ is a (c,d)-regular graph and a (γ,δ)-expander and $\gamma > \frac{c}{2}$, then the code corresponding to G has minimum distance at least δn.*

Proof Suppose that the code C corresponding G has minimum distance $\delta' n < \delta n$. Since C is a linear code, there exists a non-zero code word of weight $\delta' n$. Thus there exists a set $S \subset L$ of size $\delta' n$ such that $|\Gamma_1(S)| = 0$. However, by the previous lemma we know that $|\Gamma_1(S)| \geq (2\gamma - c)|S| > 0$ as long as $\gamma > c/2$ and thus we obtain a contradiction. ■

Until recently it wasn't known how to construct graphs with expansion better than $c/2$ explicitly. The bottleneck was roughly that the easiest way to lower bound the expansion of a graph is by considering the eigenvalues of the adjacency matrix. This technique is only capable of proving expansion of at most $c/2$. Recently, work by Capalbo, Reingold, Vadham and Wigderson showed how to construct expanders with γ close to $c(1-\epsilon)$ for arbitrarily small ϵ! [1]

5 Decoding Algorithms

Recall that our rational for considering low density parity check matrices is to use the result of $r \cdot H$ where r is a received vector and H is the parity check matrix, to determine what errors occurred. In

particular if the jth bit of $r \cdot H$ is non-zero, then an error must have occurred in the sent message at an index i such that $H_{ij} \neq 0$. Therefore a first attempt at an decoding algorithm would be to choose an index i such that $H_{ij} \neq 0$ and flip the bit r_i. This doesn't quite work but something almost as simple does. We say that a vertex on the right is *violated* if it has an odd number of non-zero neighbors.

Flip Algorithm[3]:

> While there exists a vertex r_i on the left with more violated neighbors than unviolated neighbors, flip r_i.

It is easy to prove that the Flip Algorithm does terminate. Suppose that there are initially k violated constraints. Each time we flip the setting of a vertex on the left we decrease the number of violated vertices on the right. Therefore, in k steps the algorithm must terminate. Furthermore, since the degree of each vertex is bounded by a constant, it is possible to run this algorithm with amortized constant cost for each iteration.

The more difficult aspect of the analysis is proving that the algorithm terminates with the correct assignment to the vertices on the left. There are three cases to consider.

Termination Possibilities:

1. Terminates with the sent codeword y.
2. Terminates with the wrong codeword y'.
3. Terminates with a non-codeword z.

Thus we need to show that as long as the number of errors, ϵn, is small then neither 2 nor 3 occur.

Lemma 3 *Let G be a (c,d)-regular graph and a (γ,δ)-expander with $\gamma > \frac{3c}{4}$. Suppose that $\epsilon n < \frac{\delta n}{2c}$ errors occur in transmission. Then the Flip Algorithm terminates with the correct code word assigned to the vertices on the left side of the graph.*

Proof Let y be the transmitted codeword, let r be the received vector and let z be the assignment to the left side of G when the Flip Algorithm terminates. The distance between y and r is at most ϵn. Therefore there are at most $c \cdot \epsilon n$ violated constraints at the start of the Flip Algorithm and thus the Flip Algorithm flips at most $c \cdot \epsilon n$ bits of r before it terminates. Therefore if z is the assignment to vertices on the left at the time of termination, then $\Delta(y,z) \leq \Delta(y,r) + \Delta(r,z) \leq \epsilon n + c\epsilon n < \frac{\delta n}{2c} + \frac{\delta n}{2} < \delta n$. Since z has distance less than δn from y, z cannot be a codeword other than y since this would imply that the code has minimum distance smaller than δn. Thus, we have ruled out termination condition 2.

All that remains is to rule out termination condition 3; that is, the possibility that z is not a codeword at all. To argue that z must equal y, let $z + y = x$ and note that x is a vector of weight less than δn. Furthermore, since y is a codeword and hence every node on the right has an even number of neighbors assigned 1 in the vector y, assigning z to the vertices on the left will induce the same assignments on the right as assigning the vertices on the left $z + y = x$.

Therefore, in order to argue that the Flip Algorithm would not have terminated with the vertices on the left assigned to $z \neq y$, we argue that many of the vertices on the right would be non-zero. Since z induces the same assignment to vertices on the right that x does, we argue this in terms of x.

Let S be the set of non-zero left side vertices under an assignment of x. Since x is a low weight vector, $|S| < \delta n$. Therefore, since G is (γ,δ)-expander, $|\Gamma_1(S)| > (2\gamma - c)|S| > \frac{c}{2}|S|$. Thus there must exist a vertex in S that has more violated constraints than unviolated constraints which contradicts the assumption that the Flip Algorithm terminated. ■

6 What if good expanders couldn't be constructed?

At the time of the Sipser-Spielman paper [3], expanders with the properties necessary to create good codes directly were not known to be constructible. It was known that random graphs typically had expansion properties that were sufficient for LDPC codes, but no one knew how to check that a random graph was a sufficiently good expander.

Tanner [4] and Sipser-Spielman[3] had the following idea for producing codes with constant relative distance. At the time it was known that random graphs could be tested to see if their expansion was at least c/Δ as long as $\Delta \geq 2$. Thus, suppose that $\gamma > c/\Delta$ and we are given a (c,d)-regular graph which is a (γ,δ)-expander. Although the expansion is not large enough to prove that sets of size at most δn have at least one unique neighbor, the expansion is good enough to prove that there is a neighbor of S with degree at most $\Delta - 1$ into S.

Lemma 4 *Let G be a (c,d)-regular graph and a (γ,δ)-expander with $\gamma > c/\Delta$. Then every set S such that $|S| \leq \delta n$ has at least $\frac{(\Delta\gamma - c)|S|}{\Delta - 1} > 0$ neighbors of degree less than Δ into S.*

Proof Let U be the set of neighbors of S with degree less than Δ into S and let D be all other neighbors of S. Then, $|U| + |D| \geq \gamma|S| > c\Delta|S|$. Also, $|U| + \Delta|D| \leq c|S|$ since each vertex in S has degree at most c. Therefore,

$$\begin{aligned} \Delta|U| + \Delta|D| &\geq \Delta\gamma|S| \\ \Delta|U| &\geq \Delta\gamma|S| - \Delta|D| \\ \Delta|U| &\geq \Delta\gamma|S| - (c|S| - |U|) \\ |U| &\geq \frac{(\Delta\gamma - c)}{\Delta - 1}|S| \\ &> 0 \end{aligned}$$

■

How does knowing that at least one node in $\Gamma(S)$ has degree at most $\Delta - 1$ into S help us create a good error-correcting code C? The idea is to use a small code with good minimum distance as well. Let D be an error-correcting code with minimum distance Δ and block length d (notice that the block length for D is equal to the degree of each vertex on the right). Now, instead of simply insisting that each vertex on the right has an even number of non-zero neighbors, we insist that the assignment to nodes on the left is such that the neighbors of each node on the right form a non-zero code word from D.

We can use this additional information to prove a lower bound on the minimum distance of C. By the previous lemma, if a vector has weight less than δn, then there exists a node v on the right that is adjacent to at most $\Delta - 1$ non-zero nodes. Therefore v is not adjacent to a non-zero codeword of D and therefore its constraint is not satisfied. Thus the vector must not be a code word of C. Therefore, C has minimum distance δn. Sipser and Spielman also give a linear time decoding algorithm for this construction [3].

References

[1] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson. Randomness conductors and constant-degree lossless expanders. In *Symposium on Theory of Computation (STOC 2002)*, 2002.

[2] R. G. Gallager. *Low Density Parity-Check Codes.* MIT Press, Cambridge, MA, 1963.

[3] M. Sipser and D. Spielman. Expander codes. *IEEE Transactions on Information Theory*, 49(6):1710–1722, 1996.

[4] R. M. Tanner. A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, 27(5):533–547, 1996.