Expander Graphs and their Applications*
Lecture 10: Error Correcting Codes

Notes taken by Elon Portugaly

March 2002

Summary: An error correcting code is set of words in $\{0,1\}^n$. Its distance is the minimal Hamming distance between two codewords. Therefore, if we transmit a codeword through a noisy channel that flips some of the bits, then we can correct the errors, as long as the number of bits flipped is bounded by half the distance. There is a trade-off between the size of the code and the number of errors it can correct.

There are lower bounds (Gilbert Varshamov) and upper bounds (The Balls Bound, MRRW) on the size and correction capabilities of codes.

A linear code is an error correcting code that is also linear subspace of $\{0,1\}^n$.

Expanders can be used to build error correcting codes, that have large size and distance. These codes are also efficiently decodable.

1 Definition of Error Correcting Codes

Definition 1. A Code is a set $C \subseteq \{0,1\}^n$.

Definition 2. The distance of C is $d \equiv \mathrm{d}ist(C) \equiv \min_{\substack{x \neq y \\ x,y \in C}} d_H(x,y)$, where $d_H(x,y)$ is the *hamming* distance between x and y (the number of coordinates x and y differ on).

Definition 3. The rate of C is $r \equiv \mathrm{r}ate(C) \equiv \frac{\log |C|}{n}$.

When defining a code, we desire both $|C|$ and d to be as large as possible.

1.1 Motivation

The setting we look at, is as follows:

We would like to send information through a noisy channel, that may flip some of the bits we send. We code our information using a set of words $C \subset \{0,1\}^n$ that we transmit through the channel, and assume that the number of bits the channel may flip in the transmitted word is bounded from above.

In this setting, it is clear that we would like the number of codewords available to be as large as possible, therefore we want $|C|$ to be large. We also would like to be able to reconstruct the codeword that was sent from the corrupted codeword received. Therefore, we would like that no two codewords could appear the same after they have been corrupted by the noise. If the number of bits the channel can flip is limited by k, and $d > 2k$, the last requirement is fulfilled. When d is larger, we can deal with noisier channels.

$|C|$ strings can be encoded using C. Therefore, we can transmit $\log |C|$ information bits, using n channel bits. This achieves channel utilization of $\frac{\log |C|}{n}$, which is the rate of the code.

> **Note:** A Code refers to the set of codewords and not to the process of encoding/decoding. However, in any practical aplication, we would like the code to be efficiently encodable and decodable.

* Lecture Notes for a course given by Nati Linial and Avi Wigderson at the Hebrew University, Israel.

2 Asymptotic bounds

2.1 A lower bound: Gilbert Varshamov

This bound shows that good codes can be built.

Theorem 4. *One can build a length n code with distance d and size $\geq \frac{2^n}{\text{volume of a radius } d \text{ hamming ball}}$.*

Proof. Following is an exponential time greedy algorithm that builds such a code:

- Initiate $S = \{0,1\}^n, C = \emptyset$.
- Repeat until S is empty:
  - Pick any point $x \in S$, and add it in the code.
  - Remove all the points in S that are within distance d from x.

Analysis: The volume of a hamming ball of radius d in $\{0,1\}^n$ is $\sum_{i=0}^{d} \binom{n}{i}$. Therefore, at most $\sum_{i=0}^{d} \binom{n}{i}$ points are removed from S in each iteration, and since the number of points in S at the beginning is 2^n, the number of iterations and thus the size of C at the end of the process must be at least $\frac{2^n}{\sum_{i=0}^{d} \binom{n}{i}} \equiv \frac{2^n}{\text{volume of a radius } d \text{ ball}}$. $\square$

Define $\delta \equiv \frac{d}{n}$. For $d \leq \frac{n}{2}$ we have $\sum_{i=0}^{d} \binom{n}{i} \approx \binom{n}{\delta n} \approx 2^{nH(\delta)}$, where H is the binary entropy function

$$H(x) = -x \log x - (1-x) \log(1-x).$$

Therefore

$$2^{rn} \geq \frac{2^n}{\binom{n}{\delta n}} \approx 2^{n(1-H(\delta))},$$

implying that for large n, the rate of the above code satisfies $r \geq 1 - H(\delta)$.

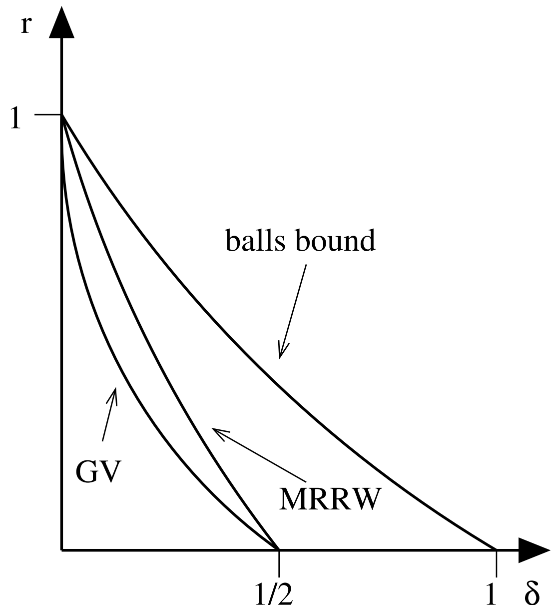

Figure 1: Illustrating Upper and Lower Bounds on rate vs. the relative distance

2.2 An Upper Bound - The Balls Bound

Theorem 5. *For any length n, distance d code C we have $|C| \leq \frac{2^n}{\text{volume of a ball of radius} \frac{d}{2}}$.*

This theorem implies the asymptotic bound: $r \leq 1 - H(\delta/2)$.

Proof. Given a distance d code C, a draw ball of radius $d/2$ around each of the points of C. If any two balls intersect, then the distance between them is smaller than d, which contradicts the definition of d. Therefore, the balls are disjoint, and their number is limited by the overall volume divided by the volume of each ball. $\square$

A much stronger upper bound was shown by McEliece, Rodemich, Rumsey and Welsh 77 (MRRW). The relations between the three bounds is illustrated in Figure 1.

3 Linear Codes

A linear code is a code that is a linear subspace of the n-dimensional space $\{0,1\}^n$. (In other words, it is closed under coordinate-wise addition mod 2.) Such codes have a polynomial size (in n) description. (For instance by specifying a basis.)

The Gilbert Varshamov algorithm can be modified to generate linear codes, and thus the Gilbert Varshamov bound applies to linear codes. Obviously, any general upper bound applies to linear codes as well.

Observation 6. *Since any linear code C must include the 0 codeword, then for linear codes,*

$$dist(C) = \min_{0 \neq x \in C} weight(x),$$

where $weight(x)$ is the number of non-zero coordinates of x.

> **Note:** Although, for a linear code, the encoding can be done in polynomial time, the decoding is in general NP-hard. I.e., given a linear code C (using some reasonable representation), and a vector x, the problem of finding the element of C that is closest to x is an NP-hard problem.

4 Using Expanders to generate Error Correcting Codes

4.1 Defining Codes using Bipartite Graphs

Consider a bipartite graph with n vertices on the left and m vertices on the right side. We call the vertices on left *variables*, and the vertices on the right *constraints*. Each variable may assume the value of 0 or 1, and we say that a constraint is satisfied if the sum of all the variables adjacent to it is zero mod 2.

Example 7. The graph in Figure 2 illustrates such a variables and constraints graph. For this graph, constraint $y4$ is satisfied iff $x3 + x6 + x9 + x14 = 0$ while constraint $y10$ is satisfied iff $x12 + x14 = 0$. Note that the same variable may appear in more than one constraint.

To define the code we refer to the variables as the coordinates of a vector $x \in \{0,1\}^n$. A vector x is in the code iff it satisfies all the constraints defined by the graph. We denote by $C(G)$, the code defined by G. The code then, is the set of all solutions of m linear equations on n variables. Therefore, $|C| \geq 2^{n-m}$ or $r \geq \frac{n-m}{n}$.

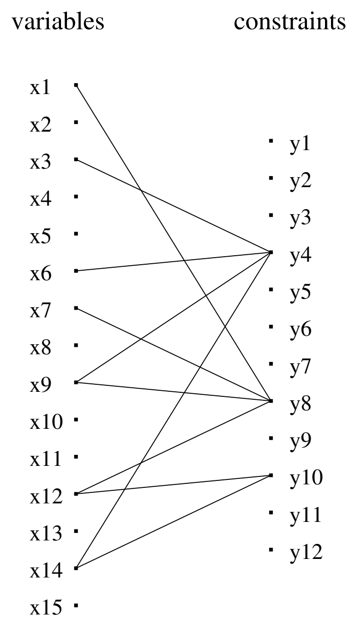

Figure 2: A Variables and Constraints Graph

4.2 Codes Using Left Side Expanders

Theorem 8. Sipser and Spielman (95): *Let $G = (V_L; V_R, E)$ be a bipartite graph of size $|V_L| = n$, $|V_R| = m$, that is k-regular on the left. Assume furthermore, that for any $S \subseteq V_L$ of size at most αn, $|\Gamma(S)| > \frac{k}{2}|S|$. Then $dist(C(G)) > \alpha n$.*

Proof.

Lemma 9. *Every set $S \subseteq V_L$ of size at most αn satisfies the* Unique Neighbor Property. *I.e, there exists $y \in V_R$ such that $|\Gamma(y) \cap S| = 1$*

> **Proof:** Consider a set $S \subseteq V_L$ of size $\leq \alpha n$. If the *Unique Neighbor Property* does not hold for S, then each $y \in \Gamma(S)$ has at least two neighbors in S. Therefore the number of edges leaving S is at least $2\Gamma(S) > 2\frac{k}{2}|S| = k|S|$, which contradicts the left regularity of G. □

Assume for the purpose of contradiction that $dist(C) \leq \alpha n$. Then there exists a nonzero codeword whose weight is at most αn (Observation 6). Let w be such a codeword, and X be its support (the set of coordinates v where $x_v = 1$), and let $y \in \Gamma(X)$ be a vertex guaranteed by the Unique Neighbor Property for X. The constraint defined by y is that the sum of all the variables that are neighbors of y is even, but in the assignment defined by w, only one of those variables is assigned the value 1. Therefore, the constraint cannot be satisfied, and w cannot be a codeword. A contradiction. □

Theorem 10. Efficient Decoding, Sipser and Speilman (95): *In the above conditions, if the expansion of sets of size at most αn is $> \frac{3}{4}k$, and if the distance of the input word ω' from a codeword ω is at most $\frac{\alpha}{2}n$, then the following decoding algorithm will return ω in a linear number of iterations:*

While there exists a variable such that most of its neighboring constraints are not satisfied, flip it.

Proof. Let A be the set of errors in ω', i.e. $A = \{v : \omega'_v \neq \omega_v\}$. If A is empty, we are done. Otherwise, assume that $|A| \leq \alpha n$. (We need the assumption to guaranty the expansion, and we will prove later that this assumption holds throughout the running of the algorithm.)

Partition $\Gamma(A)$ to satisfied neighbors S and unsatisfied neighbors U. Then:

$$|U| + |S| = |\Gamma(A)| > \frac{3}{4}k|A|. \tag{1}$$

Now, count the edges between A and $\Gamma(A) = U \cup S$. There are at least $|U|$ edges leaving U, and at least $2|S|$ edges leaving S (every vertex in S must have at least two neighbors in A). Therefore,

$$|U| + 2|S| \leq k|A|.$$

Combining this with (1) we get that

$$k|A| - |U| \geq 2|S| > 2(\frac{3}{4}k|A| - |U|),$$

and therefore,

$$|U| > \frac{1}{2}k|A|. \quad (2)$$

So more than $\frac{1}{2}k|A|$ neighbors of the $|A|$ members of A are unsatisfied. Therefore there is a variable in A that has $> \frac{1}{2}k$ unsatisfied neighbors. That means that as long as there are errors, there is a variable that most of its neighbors are unsatisfied. Since by definition, $|U|$ decreases with every step, we deduce that:

Corollary 11. *If the distance from ω does not exceed αn throughout the run of the algorithm, then U will reach the empty set, and the algorithm will halt with the codeword ω.*

To show that A is always $\leq \alpha n$, note that in the beginning, $|A_0| \leq \frac{\alpha}{2}n$, and therefore $|U_0| \leq |\Gamma(A_0)| \leq k\frac{\alpha}{2}n$. Therefore, since $|U|$ is decreasing, throughout the running of the algorithm

$$|U| \leq k\frac{\alpha}{2}n. \quad (3)$$

We know that at any step $|A_i|$ changes by ± 1. Therefore, if at any time $|A|$ exceeds αn, there must be a time i when $|A_i| = \alpha n$ (we can assume that αn is an integer). Then by (2), $|U| > k\frac{\alpha}{2}n$, which is a contradiction to (3). $\square$