6.440: Essential Coding Theory Spring 2013

Lecture 5 — February 20, 2013

Prof. Madhu Sudan *Scribe: Shravas Rao*

1 Administrative

Problem Set 1 is due next Wednesday.

2 Overview

Today we will do the following.

- Prove the Elias-Bassalygo-Johnson Bound introduced in the previous lecture. Recall that the proof consists of the two lemmas described roughly below
  - List decodable codes have poor rate.
  - Good error correcting codes are nicely decodable.
- Introduce some algebraic codes along with some helpful idea from algebra.

3 Elias-Bassalygo Theorem

The following is the Elias Bassalygo Theorem. Note that this version of the theorem is for the case $q = 2$. Although a similar statement exists for all q, the proof is significantly more involved.

Theorem 1. *Any code of relative distance δ has rate*

$$R \leq 1 - H(\tau)$$

where $\tau \leq \frac{1}{2}(1 - \sqrt{1 - 2\delta})$.

Recall that H is the entropy function defined a few lectures ago. The bound on τ comes in part from the solution for τ in the equation $2\tau(1 - \tau) = \delta$.

The proof of this theorem consists of the following two lemmas.

Lemma 2. *A $(\tau, poly(n))$-list decodable code has rate*

$$R \leq 1 - H(\tau).$$

Before continuing on to the next lemma we will define list decodable codes.

Definition 3. *A code $C \subseteq \Sigma^n$ is (τ, L)-list decodable if for all x in Σ^n*

$$|Ball(x, \tau n) \cap C| \leq L.$$

Here, we define $Ball(x, \tau n)$ to be the set of all y so that $\delta(x, y) \leq \tau n$.

Lemma 4. *(Johnson Lemma) If a code C has relative distance δ, then it is $(\tau, poly(n))$-list decodable for $\tau = \frac{1}{2}(1 - \sqrt{1 - 2\delta})$.*

3.1 Intuition behind Lemma 2

We will not give a rigorous proof of this lemma, but we will provide the intuition behind the bound. Consider the ball around x for any $x \in \{0, 1\}^n$. Without loss of generality, we can let x be 0^n, and adjust the other elements of $\{0, 1\}^n$ accordingly.

The ball sorrounding 0^n just consists of all words with at most τn 1's. If we choose two random words, y and z, from this ball, then it is very likely that both words have exactly τn 1's. The expected value of the distance between two words with exactly τn 1's is $2\tau(1 - \tau)$. If we let this value be δ, then it follows that τ is approximately equal to $\frac{1}{2}(1 - \sqrt{1 - 2\delta})$.

To see this, consider each coordinate seperately. The probability that two words differ on any given coordinate is $2\tau(1 - \tau)$. Either the coordinate of one word is 0 and the coordinate of the other is 1, or the coordinate of the first word is 1 and the coordinate of the other is 0. Note that the probability that a random coordinate of a word with τn 1's is τ.

3.2 Proof of the Johnson Lemma

Given a word $x \in \{0, 1\}^n$, consider all codewords $c_1, \ldots, c_m \in C \subseteq \{0, 1\}^n$ so that

- $\delta(x, c_i) \leq \tau n$ for all i.
- $\delta(c_i, c_j) \geq \delta n$ for all i and j.

Our goal is to find an upper bound m that is a polynomial in n. To do this, we will start by transforming words in $\{0, 1\}^n$ to vectors in $\{-1, 1\}^n$. For each coordinate, we map 0 to 1, and 1 to -1. The intuition behind this map is to give an isomorphism from an additive group consisting of 0 and 1 to a multiplicative group consisting of 1 and -1. Additionally, note that this map preserves distances up to a multiplicative factor.

Let the word x get mapped to the vector y, and codewords c_i get mapped to vectors v_i. Then the above relations on the distances can be rewritten as follows

- $\langle y, v_i \rangle \geq n(1 - 2\tau)$ for all i.
- $\langle c_i, c_j \rangle \leq (1 - 2\delta)n$ for all i and j.

Additionally, all vectors y and c_i have ℓ_2-norm n.

To find an upper bound on m, we will try to use the theorem stated last class that given a set of m vectors in n dimensions, $w_1, \ldots, w_m$, so that $\langle w_i, w_j \rangle \leq 0$ for all i and j, that $m \leq 2n$, and is upper bounded by a polynomial in n. Unfortunately, this might not always be able to be applied to the vectors v_i directly.

Instead, we can look at the vectors $v_i - \alpha y$, for some $\alpha \in (0, 1)$. The inner product between any two of these vectors is

$$\langle v_i - \alpha y, v_j - \alpha y \rangle = \langle v_i, v_j \rangle - \alpha \langle v_i, y \rangle - \alpha \langle v_j, y \rangle + \alpha^2 \langle y, y \rangle.$$

Using the bounds listed above, this value is upper bounded by $n[1 - 2\delta - 2\alpha(1 - 2\tau) + \alpha^2]$. It can be checked that if $2\tau(1 - \tau) \leq \delta$, then there exists an α so that $\langle v_i - \alpha y, v_j - \alpha y \rangle$ is less than or equal to 0 for all i and j, and therefore m is upper bounded by $2n$. This completes the proof of the Johnson Lemma.

3.3 Proof of polynomial bound on the number of vectors

In the last section, we use the following lemma.

Lemma 5. *Given unit vectors $v_1, \ldots, v_m \in \mathbb{R}^n$,*

1. *If for all i and j, $\langle v_i, v_j \rangle \leq -\epsilon$, then $m = O\left(\frac{1}{\epsilon}\right)$.*
2. *If for all i and j, $\langle v_i, v_j \rangle < 0$, then $m \leq n - 1$.*
3. *If for all i and j, $\langle v_i, v_j \rangle \leq 0$, then $m \leq 2n$.*

We will prove the third statement, which was used in the proof of the the Johnson Lemma. The proofs of the other statements use similar techniques. Assume that there exist $2n + 1$ unit vectors, $v_1, \ldots, v_{2n+1} \in \mathbb{R}^n$ so that $\langle v_i, v_j \rangle \leq 0$ for all i and j. There exists some unit vector x so that for all i, the inner product $\langle x, v_i \rangle$ is non-zero. In fact, this is the case for most unit vectors x. Without loss of generality, assume that $n + 1$ of these vectors, specifically the first $n + 1$ of them, satisfy $\langle x, v_i \rangle \geq 0$. If this is not the case, then we can just negate the coordinates of x.

Because we are working in n-dimensional space, these $n + 1$ vectors must be linearly dependent. Specifically, there exist $\alpha_1, \ldots, \alpha_{n+1}$ so that

$$\alpha_1 v_1 + \cdots + \alpha_{n+1} v_{n+1} = 0.$$

If we let the first k of these α_i be those that are positive, and if we denote those α_i that are not positive by $-\beta_i$, then we have that

$$\sum_{i \leq k} \alpha_i v_i = \sum_{j > k} \beta_j v_j.$$

Because $\sum_{i \leq k} \alpha_i v_i + \sum_{j > k} \beta_j v_j = 0$, it follows that both $\sum_{i \leq k} \alpha_i v_i$ and $\sum_{j > k} \beta_j v_j$ are 0. Therefore,

$$0 = \langle \sum_{i \leq k} \alpha_i v_i x \rangle = \sum_{i \leq k} \alpha_i \langle v_i, x \rangle.$$

But because $\langle x, v_i \rangle$ is strictly positive when i is less than or equal to k, and all the first k α_i are strictly positive, $k = 0$. This implies that $x = -\sum_{j>k} \beta_j v_j = 0$, which is a contradiction. Therefore, $2n$ is an upper bound on m.

As described in the previous class, such a bound can be achieved by considering the unit vectors in each dimension, and their negatives, and therefore the bound is tight. However, the question still remains if this bound is tight if we only consider words from $\{0,1\}^n$ and require that the distance between any two vectors be at least $n/2$.

We can answer this by mapping vectors from $\{0,1\}^n$ to $\{-1,1\}^n$ as before, and requiring that the inner product of any pair of vectors be less than or equal to 0. Such a construction can be achieved by using Hadamard matrices, which are $n \times n$ matrices, H, whose entries are -1 and 1 and are so that $HH^T = nI$ where I is the $n \times n$ identity matrix. In particular, the vectors used in the construction are the rows of a $n \times n$ Hadamard matrix, along with their negatives. It can be checked that this construction is valid.

In the case where n is a power of 2, there is a simple recursive construction of an $n \times n$ Hadamard matrix. In particular, we can let H_1 be the 1×1 Hadamard matrix [1], and let H_i be a $2^i \times 2^i$ matrix defined as

$$H_i = \begin{bmatrix} H_{i-1} & H_{i-1} \\ H_{i-1} & -H_{i-1} \end{bmatrix}.$$

3.4 Summary

This concludes our discussion for now on impossibility results for codes. Later in the course we will give an improvement to this bound called the Linear Programming Bound.

4 Algebraic Codes

The previous construction that used Hadamard matrices suggest that cleverly chosen codes could behave very nicely. Some of these codes include algebraic codes. In this section we will introduce some algebraic codes, along with a few ideas from algebra that will be needed.

4.1 Reed-Solomon Codes

All algebraic codes have the same general structure. In particular, the messages consists of some algebraic function, whose coordinates are some subset of points (or domain of the functions), and the encoding is to evaluate the message at all chosen points. These codes are useful because the functions chosen are so that two different functions are unlikely to agree on many points.

One particular example of an algebraic code that will appear later in the class is the Reed Solomon code. The functions in this case are polynomials from a certain field $\mathbb{F}$ of degree at most k, while the coordinates are some subset of $\mathbb{F}$. As in all algebraic codes, the encoding is to just evaluate the polynomial at all chosen points. If the number of points used is at least k, then we are guaranteed that no two functions will have the same encoding.

4.2 Fields

There are many useful properties of fields. The properties of fields allow for many theorems in linear algebra to work out. The polynomial ring of a field $\mathbb{F}$, denoted as $\mathbb{F}[x]$, is a unique factorization domain. Additionally, there are a few interactions that fields have with their respective polynomial rings. These include

1. A field extension of a field $\mathbb{F}$ can be created by taking the quotient ring $\mathbb{F}[x]/g(x)$ where $g(x)$ is a polynomial in $\mathbb{F}$.
2. A field extension of a field $\mathbb{F}$ can be greated by considering the field of fractions, $\mathbb{F}(x)$ that consist of elements f/g where f and g are polynomials, and g is non-zero. The two elements f/g and f'/g' are equal if $fg' = f'g$.

We can repeat this process to get a polynomial ring $\mathbb{F}(x)[y]$, which is again a unique factorization domain that consists of bivariate polynomials.

4.3 Reed-Muller Codes

We will introduce one more type of code, called Reed-Muller codes. These codes are defined by three parameters: $\mathbb{F}_q$, the finite field used, m, the number of variables, and r, the degree. The messages in this code are all polynomials of degree at most r and individual degree at most $q-1$, in m variables with coefficiants from $\mathbb{F}_q$. The coordinates are the points from $\mathbb{F}_q^m$, and the encoding is the usual encoding. Note that the Reed-Solomon codes are a special case of the Reed-Muller codes when $m = 1$.

The following lemmas give the distances of this code for various values of r.

Lemma 6. *If $r < q$ then the relative distance of the code is $1 - \frac{r}{q}$.*

The proof of this follows from the Schwartz-Zippel Lemma which states that at most r points of any given finite subset of points are rots of any multivariate polynomial with total degree at most r. The general idea behind the proof of this lemma is to do induction on the number of variables. In particular, we can express a polynomial $P(x_1, \ldots, x_m)$ as

$$P(x_1, \ldots, x_m) = x_1^0 P_0 + x_1^1 P_1 + \cdots + x_1^t P_t$$

where the P_i are polynomials in $x_2, \ldots, x_m$. Note that because the total degree of P is r, the degree of any P_i is at most $r - i$.

This lemma can be extended for larger values of r as follows.

Lemma 7. *If $r = a(q-1)+b$ where $1 \le b \le q$, then the relative distance of the code is $\left(1 - \frac{b}{q}\right) q^{-a}$.*

We will not prove this, but a good starting point is to consider the polynomial

$$(x_1^{q-1} - 1) \cdots (x_a^{q-1} - 1)(x_{a+1} - \alpha_1) \cdots (x_{a+1} - \alpha_b)$$