CSE522/Math581EB Coding Theory **Fall 2001**

Lecture 5: October 23

Lecturer: Chris Umans *Scribe: Jim Harmon*

Recall last time, we looked at Reed-Muller codes which are based on the evaluation of multivariate polynomials. With the exception of Hamming codes and Parity Check codes, every code we've looked at has been based on the evaluation of polynomials, and the distance properties have followed from the fact that low-degreee polynomials cannot have too many zeros. (And the distance of such codes is defined as $n -$ (number of zeroes)). Also, recall that a Reed-Muller code is denoted with the following notation $RM_{t,l,q}$ where t is the number of variables, l is the total degree of the polynomial, and q is the size of our field of evaluation. With this notation we can denote Reed-Solomon codes as a special case of Reed-Muller codes: $RS_{n,k,q} = RM_{1,k-1,q}$. With Reed-Muller codes we were able to get our alphabet size smaller than with Reed-Solomon codes.

5.1 Hadamard Codes

Let's look at the special case $RM_{t,1,q}$. These are called *Hadamard codes*. We define a message m as a sequence of coefficients

$$m = (m_0, m_1, \ldots, m_k)$$

for the polynomial $M(x)$ defined by

$$M(x) = \sum_{i=1}^{t} m_i x_i + m_0$$

and encoding of a message is equivalent ot evaluation of the polynomial at all points in the t-dimensional vector space over $\mathbf{F}_q$, $\mathbf{F}_q^t$. That is,

$$E(m) = (m(x_1, \ldots, x_t))_{(x_1,\ldots,x_t)\epsilon\mathbf{F}_q^t}$$

. Most of the time we will be considering the binary case $q = 2$. This gives us a code that we can parametrize with bracket notation as

$$[2^t, t+1, \frac{1}{2}(2^t)]_2$$

where 2^t is the size of the alphabet, $t+1$ is the number of coefficients, $\frac{1}{2}(2^t)$ is got from our computation in lecture four for the distance of a Reed-Muller code, and the brackets denote a linear code. Note: often, the constant term is not used in the coding. In this case, our notation for the code differs only by replacing $t+1$ with t.

At this point, we've exhausted the "tricks" that we can apply to codes based on polynomials. We've seen that Reed-Solomon codes give us a good rate and distance at the expense of the alphabet size, while Reed-Muller codes give us a good alphabet size at the expense of rate and distance. We wish to find some way to minimize rate and distance given an alphabet size. We do this through concatenation.

5.2 Concatenation

First, an example for motivation. Let's write a Reed-Solomon code as a binary code. We will assume $q = 2^p$ for some $p\epsilon\mathbf{Z}$, $p > 0$. Our code will have the following parameter representation $[n, k, n-k+1]_q$. Since

our code now contains elements from F_q, and $q = 2^p$, we can think of each bit in our code as encapsulating not 1, but p bits of information. To do this, we think of each element not as an element of $\mathbf{F}_q = \mathbf{F}_{2^p}$, but instead as an element of the p-dimensional vector space $\mathbf{F}_2^p$ over $\mathbf{F}_2$. This gives us a code with parameter representation $[np, kp, n-k+1]_2$.

Is this a linear code? We have to justify use of the square brackets, but, this is not hard to do. We just have to remember that there is an isomorphism $\mathbf{F}_q \leftrightarrow \mathbf{F}_2^p$. And so, if there exists a linear subspace of $\mathbf{F}_q^n$, then under the isomorphism, there is a corresponding linear subspace of $\mathbf{F}_2^{pn}$.

Now, this is less efficient than what we had before. But, if two codewords differ in one of the p-length blocks, we actually want them to differ in many positions. So, we will encode each p-length block into an n_2- length block so that each block differs from each other block in at least d_2 positions. Let me set out the whole scenario so that this all makes sense.

First, we have what we will call the *outer code*. It is an $[n_1, k_1, d_1]_{q^{k_2}}$ code. Explicitly, this means it takes our message string, $(m_1, \ldots, m_{k_1})$, to a code string, $(x_1, \ldots, x_{n_1})$, with each $x_i \epsilon \mathbf{F}_{q^{k_2}}$. Symbolically,

$$(m_1, \ldots, m_{k_1}) \rightarrow (x_1, \ldots, x_{n_1}) \text{ with } x_i \epsilon \mathbf{F}_{q^{k_2}}$$

Now, we apply the intermediate map $\pi : \mathbf{F}_{q^{k_2}} \rightarrow \mathbf{F}_q^{k_2}$ to each x_i and we get

$$\begin{aligned} \pi(x_1, \ldots, x_{k_1}) &= (\pi(x_1), \ldots, \pi(x_{k_1})) \\ &= ((\alpha_{1,1}, \ldots, \alpha_{1,k_2}), \ldots, (\alpha_{n_1,1}, \ldots, \alpha_{n_1,k_2})). \end{aligned}$$

Now, we apply the *inner code* to the k_2-length bits in the codeword on the far right. The inner code is a $[n_2, k_2, d_2]_q$ code which is applied to the n_1 strings of length k_2. If we denote this code by $E_2 : \mathbf{F}_q^{k_2} \rightarrow \mathbf{F}_q^{n_2}$, we get

$$\begin{aligned} E_2((\alpha_{1,1}, \ldots, \alpha_{1,k_2}), \ldots, (\alpha_{n_1,1}, \ldots, \alpha_{n_1,k_2})) &= (E_2(\alpha_{1,1}, \ldots, \alpha_{1,k_2}), \ldots, E_2(\alpha_{n_1,1}, \ldots, \alpha_{n_1,k_2})) \\ &= ((\beta_{1,1}, \ldots, \beta_{1,n_2}), \ldots, (\beta_{n_1,1}, \ldots, \beta n_1, n_2)) \end{aligned}$$

where each $\beta_{i,j} \varepsilon \mathbf{F}_q$. We can parametrize this new code with the following formula for code concatenation

$$[n_1, k_1, d_1]_{q^{k_2}} \circ [n_2, k_2, d_2]_q = [n_1 n_2, k_1 k_2, d_1, d_2]_q \tag{5.1}$$

and we are justified in denoting this as a linear code because both our inner and outer codes are linear.

5.3 Uses of Concatenation

So now that we have a concatenated code, how are we to put this to use? Recall our definition for the rate of a code $R = \frac{k}{n}$, and relative distance $\delta = \frac{d}{n}$. Earlier, we stated that at some point, we'd like to get rate and distance to be constant with respect to n in the asymptotic sense. That is, we wish to get codes with distance and rate on the order of $O(1)$ of n. We can do this with concatenation.

Let's try concatenation of a Reed-Solomon code with a Hadamard code. If we pick our codes judiciously, we can get a Reed-Solomon for our outer code that is $[n, \frac{1}{2}n, \frac{1}{2}n]_n$ and a Hadamard code for our inner code that is $[n, \log n, \frac{1}{2}n]_2$. This makes sense as long as $n = 2^m$ for some integer m. Now, if we concatenate these

codes, formula (5.1) tells us our new code is $[n^2, \frac{1}{2}n\log n, \frac{n^2}{4}]_2$. So, our relative distance becomes $\delta = \frac{1}{4}$ (which is a good constant!), and our rate becomes $R = \frac{\log n}{2n}$, (which is not a constant, but still pretty good).

However, we wish for better results on R. We wasted a lot of efficiency by using a Hadamard code as our inner code. Since our inner code has a message length of only $O(\log n)$ bits, we can afford to do things like exhaustive search, in polynomial time. Ideally, we would like to find the best linear code. But what is it? Well, frankly, we don't know, but we can get an idea with the Gilbert-Varshamov bound.

5.4 Gilbert-Varshamov Bound

As oppposed to the earlier bounds we've encountered, the Singleton bound for example, the Gilbert-Varshamov bound is an upper bound. It is found by constructing a linear code using the probabilistic method. The idea is to pick a random $k \times n$ matrix G as our generator matrix. Then, define the code space as $C = \{xG : x \epsilon \mathbf{F}_q^k\}$. The following theorem gives the bound on such.

Theorem 5.1 Gilbert-Varshamov Bound. *$\forall\ 0 < \delta < \frac{1}{2}$, $\forall \varepsilon > 0$, and for sufficiently large $n, \exists$ a $[n, 1 - H(\delta) - \varepsilon, \delta n]_q$ code.*

Proof: Recall that the volume of a ball of radius δn in the n-dimensional case is given by

$$Vol(\delta n, n) \leq 2^{(H(\delta)+o(1))n}.$$

Now, if we fix x, then

$$Pr[wt(xG) \leq \delta n] \leq \frac{Vol(\delta n, n)}{2^n} \leq \frac{2^{(H(\delta)+o(1))n}}{2^n} \leq 2^{(H(\delta)-\frac{\varepsilon}{2}-1)n}.$$

So, if we're just interested in the existence of such an x, we get

$$Pr[\exists x\colon wt(xG) \leq \delta n] \leq 2^{k+(H(\delta)-\frac{\varepsilon}{2}-1)n}.$$

Now, if we consider only those values of k such that $k \leq (1 - H(\delta) - \varepsilon)n$, then we get a bound on the existential probability

$$Pr[\exists x\colon wt(xG) \leq \delta n] \leq 2^{-\frac{\varepsilon}{2}n}.$$

Also, those values of k give us a rate

$$R = \frac{k}{n} \leq (1 - H(\delta) - \varepsilon)$$

which is the constant we required from the theorem. ■

As we have just shown, we have a code with asymptotically constant behavior in rate and relative distance as n goes to infinity.

Let us consider a modification to our previous example. We concatenate a Reed-Solomon code with a Random Linear code. Our Reed-Solomon code will be of the form

$$[n, \frac{1}{2}n, \frac{1}{2}n]_n$$

Our Random Linear code will be of the form

$$[\frac{1}{1 - H(\delta) - \varepsilon}\log n, \log n, \delta \log n]_2$$

By (5.1), this yields a code that is

$$[\frac{1}{1-H(\delta)-\varepsilon}\log n, \frac{1}{2}n\log n, \frac{\delta}{2}n\log n]_2$$

This new code has rate

$$\frac{1-H(\delta)-\varepsilon}{2} = O(1)$$

and a relative distance of

$$\frac{\delta}{2} = O(1)$$

as desired.

5.5 Wozencraft Ensemble

Recall, our original goal was to obtain a family of codes that are constructed in polynomial time. By thinking of the code generators as matrices, the messages as polynomials, and the alphabets as vector spaces, we have a lot of algebraic structure on hand that we've yet to exploit.

Let us now choose 2^k different linear subspaces from the vector space $\mathbf{F}_2^n$, and call them $S_1, \ldots, S_t$ such that they have the following properties

- $\forall i,\ |S_i| = 2^k$
- $\forall i \neq j,\ |S_i \cap S_j| = \{0\}$

Now, we shall pick our δ such that $Vol(\delta n, n) \leq 2^k$, so that *some* S_i avoids the ball of radius δn around 0 altogether – and hence is a code with distance at least δn. This provides us with the scenario for the following theorem.

Theorem 5.2 *$\forall \varepsilon > 0$, $\forall$ sufficiently large n, and for $k = \frac{n}{2}$, $\exists$ a family of $2^{\frac{n}{2}}$ subspaces of size $2^{\frac{n}{2}}$ each, such that at least one subspace is an $[n, \frac{n}{2}, (H^{-1}(\frac{1}{2}) - \varepsilon)n]_2$ code.*

Proof: Define $S_\alpha = \{(x, \alpha x) : x \in \mathbf{F}_{2^k}\}\ \forall \alpha \in \mathbf{F}_{2^k}$. Now check the following

$$\begin{array}{ll} (i) & \text{each } |S_\alpha| = 2^k \\ (ii) & \text{each } S_\alpha \text{ is a linear subspace} \\ (iii) & \forall i \neq j,\ |S_i \cap S_j| = \{0\} \\ (iv) & Vol(\delta n, n) \leq 2^k = 2^{\frac{n}{2}} \end{array}$$

(i) is obvious from the construction. (ii) is clear by the isomorphism we have constructed between the spaces. (iv) is done also by looking at the algebra we have at hand. The only statement that needs real justification is (iii). Suppose $(x_1, \alpha_1 x_1) = (x_2, \alpha_2 x_2)$. Then, this implies $x_1 = x_2$ and $\alpha_1 x_1 = \alpha_2 x_2 = \alpha_2 x_1$ and we can cancel the x_1's and get that $\alpha_1 = \alpha_2$ because we're operating in a domain. Now, the only exception is when $x_1 = x_2 = 0$, but this implies $\alpha_1 x_1 = \alpha_2 x_2 0$, and two S_i's are permitted to intersect at the zero vector. ■