CS395T Pseudorandomness and Combinatorial Constructions
Lecture 9, February 13, 2001
Lecturer: Prof. David Zuckerman, Scribe: Seth Pettie

1 Binary Codes

We have seen constructions of $[n, k, d]_2$ binary codes where the information rate k/n is constant and the distance between any two codewords is $d = \Omega(n)$. It would be nice if we could make $d = \frac{n}{2}$. Hadamard codes accomplish this though the information rate drops to $o(1)$.

1.1 Hadamard Codes

Let $x \in \{0,1\}^k$. (The *dimension* of the code is k.)

We encode x as $< x \cdot 0, x \cdot v_1, x \cdot v_2, \ldots, x \cdot v_{2^k-1} >$, where $v_i \in \{0,1\}^k$ and $\cdot$ is the dot product modulo 2.

By arguments from previous classes, this is an $[n = 2^k, k, d = n/2]_2$ code achieving the desired distance. Although the information rate $\frac{k}{n}$ is atrocious, it is essentially optimal.

Proposition 1 *Any $[n, k, d = \frac{1}{2}n]_2$ code has at most $2n$ codewords, hence $k \leq \log_2(2n)$.*

To give some intuition, we prove a simpler version assuming all pairs of codewords are at distance exactly $n/2$, rather than $\geq n/2$. Let $w_1, w_2, \ldots, w_m$ be the codewords for such a code, and let w_i' be derived from w_i by mapping

$$\begin{aligned} 0 &\mapsto 1 \\ 1 &\mapsto -1 \end{aligned}$$

Since $d(w_i, w_j) = n/2$ (the distance between codewords i and j), this implies that $w_i' \cdot w_j' = 0$ (the dot product taken over $\mathbb{R}$), that is, w_i' and w_j' are orthogonal. Because $\mathbb{R}^n$ has no more than n mutually orthogonal vectors, we conclude that $m \leq n$.

Note that there are at most n codewords in this simple case; for the general case there can be $2n$ codewords. We can see this by adding n more vectors to our original set, namely $-w_i'$ for each i.

2 Converting probability spaces to codebooks

Suppose there is a probability space of size n for k ϵ-biased $0-1$ variables. Let this space be represented by a $k \times n$ matrix G where the rows represent the k variables and the columns represent the n points in the prob. space.

By the definition of ϵ-biased (see Lecture 6), the summation (over $\mathbb{F}_2$) of any non-empty subset of the rows of G has between $n(1-\epsilon)/2$ and $n(1+\epsilon)/2$ ones.

We can also use G to generate an $[n, k, d \geq \frac{1-\epsilon}{2}n]_2$ binary code as follows. If we let $C = \{yG : y \in \{0,1\}^k\}$ be the set of codewords, it is clear that all 2^k codewords have length n. To prove the claim it suffices to show that all pairs of codewords in C are at distance at least $\frac{1-\epsilon}{2}n$ from each other.

First examine the distance from codeword $0^k G = 0^n$ to some other codeword w. This is simply the weight of w (number of 1s in w), which we argued is at least $n(1-\epsilon)/2$. By linearity this also lowerbounds the minimum distance.

Using the construction from Lecture 7, $n = O(k^2\epsilon^{-2})$. A different construction yields $n = O(k\epsilon^{-4})$. A probabilistic argument shows that there exists codes with $n = O(k\epsilon^{-2})$.

3 Decoding

3.1 Decoding with Erasures

Assuming the codeword $w \in \mathbb{F}_2^n$ is not altered, recovering the original $x \in \mathbb{F}_2^k$ amounts to solving for x in

$$xG = w$$

In the erasure model, code symbols may be replaced by *'s, a symbol not in the alphabet. Let w be derived from a codeword by placing *'s at positions $i_1, i_2, \ldots, i_\ell$. Let w' (resp. G') be derived from w (resp. G) by removing columns $i_1, i_2, \ldots, i_\ell$. Recovering the encoded word x, if it is possible, amounts to solving for x in

$$xG' = w'$$

3.2 Decoding Reed-Solomon With Errors

<u>The Berlekamp-Welch Decoder</u>

The Reed-Solomon coder over a field $\mathbb{F}$ encodes a degree $< k$ polynomial p by

$$\begin{aligned} p &\mapsto <p(a_1), p(a_2), \ldots, p(a_n)> \\ &= <b_1, \quad b_2, \quad \ldots, \quad b_n> \end{aligned}$$

A decoder takes all a_i, b_i and two naturals k and e. It returns a degree $< k$ polynomial p s.t. $p(a_i) \neq b_i$ for no more than e values of i.

The Berlekamp-Welch approach is to find an "error-locating" polynomial y such that

$p(a_i)y(a_i) = b_i y(a_i)$ for <u>all</u> i.

In other words we want $y(a_i) = 0$ whenever $p(a_i) \neq b_i$. Under the assumption that there are no more than e errors, we take the degree of y to be $\leq e$.

We find polynomials y and z such that
$y \not\equiv 0$
$deg(y) \leq e$
$deg(z) < e + k$
$z(a_i) = b_i \cdot y(a_i)$ for all i.

We output $p = z/y$. Finding z and y amounts to solving n linear equations in $2e + k + 1$ variables (the coefficients of z, y). Although we know there is not a unique solution to z, y (scaling both gives a new solution), we want to show z/y *is* unique.

Claim 1 *Suppose (z, y) and (z', y') are possible solutions returned by the Berlekamp-Welch decoder. Then $\frac{z}{y} \equiv \frac{z'}{y'}$.*

Proof: First observe that $deg(yz') < 2e + k$ and $deg(y'z) < 2e + k$.

Then for all $i \leq n$ and $n > 2e + k$,

$$y'(a_i) \cdot z(a_i) = y'(a_i) \cdot b_i \cdot y(a_i) = z'(a_i) \cdot y(a_i)$$

Hence $y'z \equiv yz'$, which gives the claim. □

Thus we can decode as long as the number of errors $e < (n-k)/2$. This is optimal, as the most we can hope to decode is $(d-1)/2 = (n-k)/2$.

4 Applications of Reed-Solomon Codes

4.1 Secret Sharing

We want to encode a secret in n pieces so that if one knows any k pieces the secret can be reconstructed, but if one only knows $< k$ pieces, no information about the secret is given away.

This can be done by constructing a random degree $k-1$ polynomial p over a field $\mathbb{F}$:

$$p(x) = p_0 + p_1 x + p_2 x^2 + \cdots + p_{k-1} x^{k-1}$$

where p_0 is the secret and all other p_i are chosen randomly.

The n pieces are then $p(a_1), p(a_2), \ldots, p(a_n)$ where the a_i are distinct and non-zero.

Knowing any k pieces is enough to determine p and hence p_0 as well. However any $k-1$ pieces gives no information about p_0.

4.2 Converting worst-case hardness to average-case hardness

Suppose that a function is hard to compute in the worst-case, though not on average. Error-correcting codes (in particular, the multidimensional analogue of Reed-Solomon) can be used to derive a new function that is hard to compute on the average. This will be explored later in the course.