6.897 Algorithmic Introduction to Coding Theory	September 12, 2001
Lecture 3	
Lecturer: Madhu Sudan	*Scribe: Arian Shahdadi*

Today we will talk about Hamming codes and the Hamming bound. In order to talk about Hamming's codes, we will define linear codes. The lecture also contained material on some basic and linear algebra, which is deferred to a separate handout.

1 Linear Codes

There are many ways to specifiy a code given our established formalisms. One convenient way is to give an encoding function, thereby specifying how to create arbitrary codewords. This generalized mapping can be used to decribe classes of codes in a succinct manner. For a special class of codes called *linear codes*, other succinct options are available. We will define these codes here.

Linear codes are obtained when the alphabet Σ is associated with the field $\mathbb{F}_q$ for some finite q. In such cases, one can think of the "ambient space" (Σ^n), which contains the codewords and received words, as a vector space.

Recall that $L \subseteq \mathbb{F}_q^n$ is a *linear subspace* of the vector space $\mathbb{F}_q^n$ if for every pair $\mathbf{x}, \mathbf{y} \in L$ and $\alpha \in \mathbb{F}_q$, it is the case that $\mathbf{x}+\mathbf{y} \in L$ and $\alpha \cdot \mathbf{x} \in L$. (Here $+$ is vector addition and $\cdot$ is scalar-vector multiplication. Specifically, if $\mathbf{x} = \langle x_1, \ldots, x_n \rangle$ and $\mathbf{y} = \langle y_1, \ldots, y_n \rangle$, then $\mathbf{x} + \mathbf{y} = \langle x_1 + y_1, \ldots, x_n + y_n \rangle$ and $\alpha \cdot \mathbf{x} = \langle \alpha \cdot x_1, \ldots, \alpha \cdot x_n \rangle$.)

When a code $C \subseteq \mathbb{F}_q^n$ is a linear subspace of $\mathbb{F}_q^n$, then the code is called a *linear code*. Notationally, we express the fact that an $(n, k, d)_q$ code is linear by using square brackets and denoting it an $[n, k, d]_q$ code.

1.1 Generator matrix and Parity check matrix

Linear algebra gives us several succinct ways of representing codes. Recall that a linear subspace $C \subseteq \mathbb{F}_q^n$ of dimension k can be specified by giving a basis for the subspace, i.e., by giving an independent set of vectors $\mathbf{x}_1, \ldots, \mathbf{x}_k \subseteq \mathbb{F}_q^n$ such that $C = \{\sum_{i=1}^k \alpha_i \cdot \mathbf{x}_i \mid \alpha_1, \ldots, \alpha_k \in \mathbb{F}_q\}$. Note in particular that the code has q^k vectors and so we are not abusing notation here by using k to mean two different things — the dimension does equal the message length.

Using matrix notation, we can thus represent the code C by a *generator matrix* $\mathbf{G} \in \mathbb{F}_q^{k \times n}$, where the rows of $\mathbf{G}$ are the basis vectors $\mathbf{x}_1, \ldots, \mathbf{x}_k$. (Throughout this course we will denote vectors by row vectors, unless otherwise specified.) In this notation, $C = \{\alpha \mathbf{G} \mid \alpha \in \mathbb{F}_q^k\}$.

Another way of describing a linear subspace is by enumerating the constraints that vectors in the subspace obey, or equivalently, by describing its "null space". For $\mathbf{x}, \mathbf{y} \in \mathbb{F}_q^n$, let $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^n x_i \cdot y_i$ where the multiplications and summation denote field operations. Recall that for every linear subspace C of dimension k, one can find a linear subspace, denoted $C^{\perp}$, of dimension $n-k$, such that for every $\mathbf{x} \in C$ and $\mathbf{y} \in C^{\perp}$, $\langle \mathbf{x}, \mathbf{y} \rangle = 0$. Let $\mathbf{H}^T \in \mathbb{F}_q^{(n-k)\times n}$ be the generator matrix for $C^{\perp}$. The matrix $\mathbf{H} \in \mathbb{F}_q^{n\times(n-k)}$ is called the *parity check matrix* of C. Note that one can express $C = \{\mathbf{y} \mid \mathbf{y}\mathbf{H} = \mathbf{0}\}$.

Note that under both representations, we can show trivially that the 0 word is a codeword for all linear codes. We will return to this fact later on.

Both the generator matrix and the parity check matrix representations give an $O(n^2 \log q)$ sized representation of the code C. These representations take $\text{poly}(n \log q)$ bits to describe and are thus fairly compact.

Furthermore these representations are equivalent computationally. With some rudimentary linear algebra, we can show that parity check matrices and generator matricies are essentially the same. That is, given G we can compute H and given H we can compute G.

Although we have shown that generator and parity check matrices are equivalent in both a mathematical and computational sense, do they work in the same manner? As we will see, parity check matrices have some properties that are very useful.

1.2 Applications of the parity check matrix

Error-detection: Suppose we are given a received vector $\mathbf{y}$ obtained from the transmission of a codeword from a code of minimum distance d. Further, suppose that we are guaranteed that that $d-1$ or fewer errors have occurred. As we noted earlier, it is possible to tell, given unlimited computational resources, if an error has occured or not. Can we do this computation efficiently? It turns out that the answer, for linear codes C given by their parity check matrix $\mathbf{H}$, is "YES"! We simply compute the vector $\mathbf{yH}$. If the result is the all zeroes vector, then $\mathbf{y}$ is a codeword and hence no errors occurred during transmission. If $\mathbf{yH} \neq \mathbf{0}$, then $\mathbf{y}$ is not a codeword, by the definition of the parity check matrix. Thus we conclude that error-detection is "easy" (in polynomial time) for linear codes.

Another nice property of parity check matricies is that they give a sense of the minimum distance of a code. One important question is that of how we can determine the minimum distance of a linear code. Unfortunately, there are no known efficient methods of computing this — and the problem is known to be NP hard [8, 2]. Nevertheless parity check matricies offer some aid in designing codes with reasonable minimum distance as we will see shortly. First, we digress in order to define a new concept, the *Hamming weight* of a codeword.

Definition 1 *The* Hamming weight *of a vector* $\mathbf{x} \in \mathbb{F}_q^n$*, denoted* $\text{wt}(\mathbf{x})$*, is the number of non-zero coordinates in* $\mathbf{x}$*.*

Now, in order to find the minimum distance of a linear code, we can find a non-zero codeword $\mathbf{x}$ of smallest weight such that $\mathbf{x} \cdot \mathbf{H} = \mathbf{0}$. The weight of this codeword, $\text{wt}(\mathbf{x})$, equals the minimum distance of the code. We can prove this assertion using elementary linear algebra.

Proposition 2 *The minimum distance of a linear code C with parity check* $\mathbf{H}$ *equals the smallest integer d such that there exist d rows of* $\mathbf{H}$ *that are linearly dependent.*

Proof First, note that the proposition is equivalent to the assertion that the minimum distance of C equals the weight of the non-zero vector $\mathbf{x}$ of smallest weight that satisfies $\mathbf{xH} = \mathbf{0}$. To see this, let $\mathbf{h}_i$ denote the ith row of $\mathbf{H}$. Now suppose $\mathbf{xH} = \mathbf{0}$ and let $x_{i_1}, \ldots, x_{i_d}$ be the non-zero elements of $\mathbf{x}$. Then $\sum_{j=1}^{d} x_{i_j} \mathbf{h}_{i_j} = \mathbf{0}$ and thus the rows $\mathbf{h}_{i_1}, \ldots, \mathbf{h}_{i_d}$ are linearly dependent. Similarly any collection of d linearly dependent rows lead to a vector $\mathbf{x}$ of weight d such that $\mathbf{xH} = \mathbf{0}$.

Now we turn to showing that $\min_{\{\mathbf{x} \neq \mathbf{0} | \mathbf{xH}=\mathbf{0}\}} \{\mathrm{wt}(\mathbf{x})\}$ is the minimum distance of the code. First note that if $\mathbf{xH} = \mathbf{0}$, then its weight is an upper bound on the minimum distance since its distance from $\mathbf{0}$ (which is also a codeword) is $\mathrm{wt}(\mathbf{x})$. On the other hand this quantity also lower bounds the minimum distance: If $\mathbf{y}$ and $\mathbf{z}$ are the nearest pair of codewords, then the vector $\mathbf{x} = \mathbf{y} - \mathbf{z}$ is a codeword whose Hamming weight equals the Hamming distance between $\mathbf{y}$ and $\mathbf{z}$. ■

1.3 Systematic Codes

Finally we introduce another side-effect of linearity, which is the notion of a *systematic code.*

Definition 3 *An* $(n, k, d)_q$ systematic code *is a code in which the encoding of a message consists of* k *message symbols followed by* $n - k$ *check symbols.*

In other words, in a systematic encoding, the message is recoverable without decoding if there are no errors. Systematic codes are very common in practical implementations of encoding and decoding schemes.

The term "systematic" is from the paper of Hamming [4]. Hamming observed that codes obtained from "linear" operations can be turned into systematic ones without loss of generality. (Hamming did not explicitly introduce linear codes, though linearity did play a dominant role in his codes.) To do so, note that if the generator matrix $\mathbf{G}$ is of the form $[\mathbf{I} \mid \mathbf{A}]$ where $\mathbf{I}$ is the $k \times k$ identity matrix, and $\mathbf{A}$ is a $k \times (n-k)$ matrix, then the code generated by $\mathbf{G}$ is systematic. So what can we do if $\mathbf{G}$ is not of this form? For simplicity, we describe the procedure when $q = 2$ and the code is binary. (The method also generalizes to other fields.) First we permute the columns and rows of $\mathbf{G}$ till the leftmost entry of the first row is a 1. (Permuting the rows does not alter the code, while permuting the columns only changes the naming of the coordinate indices.) Now we can arrange it so that all the leftmost entry of all remaining rows is 0. We do so by subtracting the first row from any given row if the row has a 1 in the leftmost entry. Again this process does not change the code — it only changes the basis we are using for the code. Repeating this process for the remaining indices, we arrive at a generator matrix with an identity matrix in the left part - as desired.

We conclude that every linear code is systematic. The converse need not be true.

2 Hamming Codes

The machinery of linear codes developed above allows us to arrive at the Hamming codes with minimum distance $d = 3$ naturally. In order to describe these codes, we construct their parity check matrix $\mathbf{H}$.

For simplicity, we will construct the matrix over the alphabet $\Sigma = \{0, 1\}$. This puts us in the field $\mathbb{F}_2$, where the addition and multiplication operations are just arithmetic modulo 2.

Notice our goal is to construct a matrix $\mathbf{H} \in \mathbb{F}_q^{n\times(n-k)}$ such that $\mathbf{xH} \neq \mathbf{0}$ for any vector $\mathbf{x}$ of weight 1 or 2. (Then the code will have distance ≥ 3. We will do so for as large an n as possible, given a fixed choice of $l = n - k$. Denote by $\mathbf{h}_i$ the ith row of this matrix. We will first determine what conditions $\mathbf{h}_i$ must satisfy.

First, consider a vector $\mathbf{x}$ of weight 1. Say this vector has a 1 in the ith position and is zero elsewhere. Then $\mathbf{xH} = \mathbf{h}_i$. Since we don't want $\mathbf{xH}$ to be zero, this imposes the constraint that $\mathbf{h}_i$ should not be zero. If $\mathbf{H}$ satisfies this property, then it gives the parity check matrix of a code with distance ≥ 2.

Now, consider a vector $\mathbf{x}$ of weight 2. Say, this vector has a 1 in the ith and jth coordinates and is zero elsewhere. Here, we get, $\mathbf{xH} = \mathbf{h}_i + \mathbf{h}_j$. If this result is to be non-zero, we need $\mathbf{h}_i \neq \mathbf{h}_j$, for every $i \neq j$. We conclude that $\mathbf{H}$ is the parity check matrix of a code of minimum distance ≥ 3 over $\mathbb{F}_2$ if and only if all its rows are distinct and non-zero.

Thus if $l = n - k$, then the largest n we can allow is the number of distinct non-zero vectors $\mathbf{h}_i \in \mathbb{F}_2^l$. This is a simple enumeration problem — there are exactly $2^l - 1$ such vectors and so we get $n = 2^l - 1$. Using our previously defined notation, we can now fully specify the parameters of the generalized Hamming codes:

Proposition 4 *For every positive integer l, there exists an $[2^l - 1, 2^l - l - 1, 3]_2$ code, called the* Hamming code.

Exercises:

- Describe the parameters and parity check matrix of the q-ary Hamming codes.
- Describe the generator matrix of the Hamming code. Put it in systematic form $[\mathbf{I} \mid \mathbf{A}]$ and describe the entries of the matrix $\mathbf{A}$.

To make some sense of the growth of parameters, notice that if $k = 2^l - l - 1$, then $l = \Theta(\log k)$. Thus the Hamming codes are obtained by appending $\Theta(\log k)$ parity check bits to the k-bit message. This gives a code of minimum distance 3, or equivalently, a 1-error-correcting code. The next section addresses the issue of whether one needs to expend so much redundancy to correct one error. But first we describe a simple error-correcting algorithm for the Hamming code.

2.1 Error-correction in the Hamming code

Given a received vector $\mathbf{y}$, where $\mathbf{y} = \mathbf{x} + \mathbf{e}$ for some codeword $\mathbf{x}$ and a vector $\mathbf{e}$ of weight one, how can we compute the location i such that $e_i = 1$?

The naive method for computing this would be by brute force. For $j = 1$ to n, we try flipping the jth bit of $\mathbf{y}$ and see if this gives us a codeword. We can check if such a vector $\mathbf{y}'$ is a codeword by computing $\mathbf{y}'\mathbf{H}$ and checking to see if it is zero. This gives an $O(n^3)$ algorithm.

Hamming wasn't satisfied with this approach and instead relied on the parity check matrix to simplify this computation. Since we know that $\mathbf{xH} = \mathbf{0}$, we have that $\mathbf{yH} = \mathbf{eH} = \mathbf{h}_i$, where i is the index of the error location. Furthermore, if we arrange $\mathbf{H}$ so that the ith row is simply the number i written in binary, then the vector $\mathbf{yH}$ simply gives the location of the error in binary. This speeds up our computation by a factor of n, giving an $O(n^2)$ algorithm to correct errors.

3 The Hamming bound and perfect codes

Is it necessary to add logarithmically many bits to correct 1 error? Hamming proved that this was the optimal result by introducing what is called the Hamming bound. We now derive this bound.

Let us recall some notation we have used earlier. Let $B(\mathbf{y}, r)$ denote the ball of radius r around vector $\mathbf{y}$ (where for this section the vectors will be from $\mathbb{F}_2^n$). Let $\mathrm{Vol}(r,n) = \sum_{i=0}^{r} \binom{n}{i}$ be the volume of such a ball. The Hamming bound gives the following:

Theorem 5 *If an $(n,k,d)_2$ code exists, then*

$$2^k \mathrm{Vol}\left(\left\lfloor \frac{d-1}{2} \right\rfloor, n\right) \le 2^n.$$

Proof Let $C = (n,k,d)_2$ code and let $r = \lfloor \frac{d-1}{2} \rfloor$. The Hamming bound is based on the simple observation that if $\mathbf{x}$ and $\mathbf{y}$ are codewords of C, then $B(\mathbf{x}, r)$ and $B(\mathbf{y}, r)$ are disjoint. Since $\bigcup_{\mathbf{x} \in C} B(\mathbf{x}, r) \subseteq \{0,1\}^n$ and the sets on the LHS are all disjoint, we get: $\sum_{\mathbf{x} \in C} |B(\mathbf{x}, r)| \le 2^n$. The bound follows immediately. ■

The Hamming bound generalizes naturally to q-ary alphabets. Let $\mathrm{Vol}_q(r,n) = \sum_{i=0}^{r} \binom{n}{i}(q-1)^i$. Then we have:

Theorem 6 *For every n, k, d and q, an $(n,k,d)_q$ code exists only if*

$$q^k \mathrm{Vol}_q\left(\left\lfloor \frac{d-1}{2} \right\rfloor, n\right) \le q^n.$$

For the case $d = 3$ and $q = 2$, the above bound says that $2^k(n+1) \le 2^n$, and this bound is met *exactly* by the Hamming codes (and this is true also for q-ary Hamming codes). The Hamming bound defines a class of codes known as *perfect* codes.

Definition 7 *An $(n,k,d)_q$ code is* perfect *if it meets the Hamming bound exactly: i.e.,*

$$q^k \mathrm{Vol}_q\left(\left\lfloor \frac{d-1}{2} \right\rfloor, n\right) = q^n.$$

An interesting fact here is that other than the Hamming codes and two specific codes proposed by Golay [3] no other perfect codes exist. This was supsected for long and finally proved by van Lint [7] and Tietavainen [6]. This proof is way beyond the scope of this class.

3.0.1 Converting odd minimum distance to even minimum distance

The Hamming bound is clearly "jerky" - it proves nothing interesting for $d = 2$, but jumps to proving that the Hamming codes for $d = 3$ are best possible. It does not improve for $d = 4$ but then jumps again when $d = 5$. Why does this discontinuity happen? Is it that the Hamming bound is too weak, or is it truly easier to construct codes of even minimum distance? Hamming showed, surprisingly enough, that the latter was the case. The argument is quite simple - the same that we used to construct distance 2 codes from distance 1 codes and we describe it next.

Proposition 8 *If an $(n, k, 2t+1)_2$ code exists, then so does an $(n+1, k, 2t+2)_2$ code.*

Proof Let C be an $(n, k, 2t+1)_2$ code. We construct C', an $(n+1, k, 2t+2)_2$ code from C as follows: Let $\mathbf{x} = \langle x_1, \ldots, x_n \rangle \in C$. Corresponding to $\mathbf{x}$, C' contains the codeword $\mathbf{x}' = \langle x_1, \ldots, x_n, \sum_{i=1}^n x_i \rangle$. (All summations in this proof are sums modulo 2.) It is clear that C' is an $(n+1, k, d)_2$ code for some d, and what we wish to show is that $d \geq 2t+2$.

It suffices to show that if $\mathbf{x}, \mathbf{y} \in C$, then $\Delta(\mathbf{x}', \mathbf{y}') \geq 2t+2$. Notice first that $\Delta(\mathbf{x}', \mathbf{y}') \geq \Delta(\mathbf{x}, \mathbf{y})$. So if the latter is at least $2t+2$ then we are done. Thus it suffices to consider the case where $\Delta(\mathbf{x}, \mathbf{y}) = 2t+1$. In this case, let us permute the coordinates of C so that $\mathbf{x}$ and $\mathbf{y}$ agree in the first $m = n - (2t+1)$ locations and disagree in the last $2t+1$ locations. Now consider the quantity $\sum_{i=1}^n (x_i + y_i)$. This quantity equals the parity of "(2$*$ # 1's in $\mathbf{x}$ in the first m coordinates) $+(2t+1)$". Clearly, this parity is odd (since the first expression as well as $2t$ are even and the last term, 1 is odd). We thus get that exactly one of $\sum_{i=1}^n x_i$ and $\sum_{i=1}^n y_i$ is one (modulo 2) and the other is zero. Thus $\mathbf{x}'$ and $\mathbf{y}'$ disagree in the last coordinate and so the distance between them is $2t+2$ in this case also. ■

Exercise: Does the proposition above extend for other values of q?

Applying the proposition above to the distance 3 codes, Hamming got a family of distance 4 codes. Going from $d = 4$ to $d = 5$ is non-trivial. In particular the Hamming bound implies that one needs approximately $2 \log k$ check bits to get such a code. Getting such a code proved to be non-trivial and was finally discovered, independently by Bose and Ray-Chaudhuri [1], and Hocquenghem [5], in 1959. These codes, which are quite famous under the label "BCH codes", give binary codes of block length n and odd minimum distance d using $\frac{d-1}{2} \log n$ check bits. The Hamming bound shows this is tight to within lesser order terms.

References

[1] R. C. Bose and D. K. Ray-Chaudhuri. On a class of error correcting binary group codes. *Information and Control*, 3:68–79, 1960.

[2] Ilya Dumer, Daniele Micciancio, and Madhu Sudan. Hardness of approximating the minimum distance of a linear code. *Proceedings of the 40th IEEE Symposium on Foundations of Computer Science*, pages 475–484, 1999.

[3] M. J. E. Golay. Notes on digital coding. *Proceedings of the IRE*, 37:657, June 1949.

[4] Richard W. Hamming. Error Detecting and Error Correcting Codes. *Bell System Technical Journal*, 29:147–160, April 1950.

[5] A. Hocquenghem. Codes correcteurs d'erreurs. *Chiffres (Paris)*, 2:147–156, 1959.

[6] Aimo Tietavainen. On the nonexistence of perfect codes over finite fields. *SIAM Journal of Applied Mathematics*, 24(1):88–96, January 1973.

[7] Jacobus H. van Lint. Nonexistence theorems for perfect error-correcting codes. In G. Birkhoff and M. Hall Jr., editors, *Proceedings of the Symposium on Computers in Algebra and Number Theory, New York, 1970*, pages 89–95. American Mathematical Society, Providence, RI, 1971.

[8] Alexander Vardy. The intractability of computing the minimum distance of a code. *IEEE Transactions on Information Theory*, 43:1757–1766, November 1997.