CSE 533: Error-Correcting Codes **(Autumn 2006)**

Lecture 1: Error Correcting Codes

Oct. 4, 2006

Lecturer: Venkatesan Guruswami *Scribe: Vibhor Rastogi*

1 Introduction

Consider the problem of storing information on an error prone media. In such cases, a mechanism to detect and correct errors caused due to noise would be very useful. Error correcting codes can be thought of as mathematical objects used to cope with noise. We can think of error correcting codes purely in a combinatorial fashion and focus on their existence and construction as combinatorial problems. This will be initial emphasis of the course, thogh we will keep track of important algorithmic challenges that arise. In order to effectively use a code, one needs efficient algorithms for encoding and error correction using that code. Such algorithmic aspects will be covered in the latter part of the course, once we have discussed the key combinatorial/existential aspects of the theory. A noticeable omission in the course content is related to the extraneous applications of codes to complexity theory, cryptography, explicit combinatorial constructions, etc.

In this lecture we shall look at some simple codes and make formal definitions which will help us understand codes in general.

2 Some Simple Codes

Suppose, we need to store 64 bit words in such a way that they can be correctly recovered even if a single bit per word gets flipped. One way is to store each information bit by duplicating it three times. We can thus store 21 bits of information in the word. This would permit only about a fraction $\frac{1}{3}$ of information to be stored per bit of the word. However, it would allow us to correct any single bit flip since the majority value of the three copies of the bit gives the correct value of the bit, even if one of the copies is flipped.
Hamming in 1950 introduced a code which could also correct 1-bit errors but used less number of redundant (or extra) bits. The code is defined in such a way that a chunk of 4 information bits x_1, x_2, x_3, x_4 gets mapped (or "encoded") to 7 bits as

$$x_1,\ x_2,\ x_3,\ x_4,\ x_2 \oplus x_3 \oplus x_4,\ x_1 \oplus x_2 \oplus x_4,\ x_1 \oplus x_3 \oplus x_4$$

Here $\oplus$ stands for the xor operation. This transformation can equivalently be represented as a mapping from $\mathbf{x}$ to $G\mathbf{x}$ (the operations are done modulo 2) where $\mathbf{x}$ is the column vector $[x_1\, x_2\, x_3\, x_4]^t$

and G is the matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

We prove that such a code can correct all single bit flips by proving that for two distinct 4-bit vectors $\mathbf{x}$ and $\mathbf{y}$ get mapped to code words $G\mathbf{x}$ and $G\mathbf{y}$ which differ in at least 3 bits. Thus for any 7-bit vector there is always a single unique code word which can be obtained by a single bit flip. Here by code word we mean a 7-bit vector corresponding to some $G\mathbf{x}$

Proposition 2.1. *If $\mathbf{x} \neq \mathbf{y}$ are two 4-bit vectors, then $G\mathbf{x}$ and $G\mathbf{y}$ differ in at least 3 locations.*

Proof: Let us define $\mathbf{w} = \mathbf{x} - \mathbf{y}$. Therefore, $|bfw \neq 0$. It is easy to see that for each non-zero $\mathbf{w}$, $G\mathbf{w}$ has at least 3 bits which are 1.

3 Notation

We will now define a few terms which can be used to characterize the properties of codes in general

Definition 3.1. *The Hamming distance between two strings x and y over alphabet Σ is defined as the number of positions at which the two strings differ. More formally, the Hamming distance $\Delta(x, y) = \|\{i|x_i \neq y_i\}\|$*

Definition 3.2. *The Hamming weight of a string x over alphabet Σ is defined as the number of non-zero symbols in the string. More formally, the Hamming weight of a string $wt(x) = |\{i|x_i \neq 0\}|$*

Remark 3.3. *It is trivial to see that $wt(x - y) = \Delta(x, y)$*

Definition 3.4. *An error correcting code C of block length n over a finite alphabet Σ is any subset of Σ^n*

Definition 3.5. *A code $C \subset \Sigma^n$ is said to have minimum distance d if*

$$d = \min_{c_1, c_2 \in C, c_1 \neq c_2} \Delta(c_1, c_2) \ .$$

In particular, for every pair of distinct codewords the Hamming distance between them is at least d.

Example 3.6. *The parity check code is an example of distance 2 code. It is defined over $\{0, 1\}^n$ and generated by inserting a redundant bit which is logically equivalent to the xor of the other $n - 1$ bits.*

Example 3.7. *The Hamming code discussed earlier is an example of distance 3 code.*

Remark 3.8. *There is a cute solution for the following puzzle which exploits the properties of Hamming codes*

Hat Puzzle : Fifteen players enter a room and a red or blue hat is placed on each person's head. The color of each hat is determined by a coin toss, with the outcome of one coin toss having no effect on the others. Each person can see the other players' hats but not his own.

No communication of any sort is allowed, except for an initial strategy session before the game begins. Once they have had a chance to look at the other hats, the players must simultaneously guess the color of their own hats or pass. The group shares a hypothetical $3 million prize if at least one player guesses correctly and no players guess incorrectly.

The same game can be played with any number of players. The general problem is to find a strategy for the group that maximizes its chances of winning the prize.

One obvious strategy for the players, for instance, would be for one player to always guess "red" while the other players pass. This would give the group a 50 percent chance of winning the prize. Can the group do better?

Definition 3.9. *Relative distance of a code is defined as the ratio of the minimum distance to the block length of the code.*

Definition 3.10. *Information rate of a code C over alphabet Σ with block length n is defined as* $\frac{log_{|\Sigma|}|C|}{n}$

Lemma 3.11. *Consider a code $C \subset \Sigma^n$. Then, the following statements are equivalent:*

1. *C has minimum distance $2t + 1$*
2. *C can be used to correct all t symbol errors*
3. *C can be used to detect all $2t$ symbol errors*
4. *C can be used to correct all $2t$ symbol erasures (In the erasure model, some symbols are erased and the rest are intact, and we* know *the locations of the erasures. The goal is to fill in the values of the erased positions, using the values of the unerased positions and the redundancy of the code.)*

Proof: Assume statement 1. We can imagine drawing disjoint balls of radius t centered at each codeword in C. Such balls are called Hamming balls. Thus, we can uniquely identify the initial codeword from an erroneous string s_1 in Σ^n as long as the number of errors is $\leq t$ by identifying the Hamming ball which contains s_1. Additionally, if one or more, but not more than $2t$ errors occur, then the codeword gets distorted to an erroneous string s_2 which must be different from every codeword. Thus such error patterns can be detected. We can also identify up to $2t$ missing symbols given their positions for any erroneous string s_3 by identifying the unique codeword that

is consistent with s_3.

Assume that statement 1 is false. Thus there exists two codewords c_1 and c_2 whose Hamming distance $d \leq 2t$. Consider the midpoint of c_1 and c_2; call it x. The Hamming distance of x from both c_1 and c_2 is $\leq t$. If t symbol errors are allowed then either c_1 or c_2 could give rise to x. Thus, on the string x it wouldn't be possible to correct the errors. Hence C cannot correct all t symbol errors If $2t$ symbol errors are allowed then c_1 could give rise to c_2 and thus we cannot be sure that a codeword corresponds to no errors. Hence C cannot detect all $2t$ symbol errors. If $2t$ symbols are erased then either c_1 or c_2 could give rise to the same string (take c_1 and erase all positions which are different from c_2) and thus we cannot correct $2t$ symbol erasures.

4 Linear Codes

Definition 4.1. *If Σ is a field and $C \subset \Sigma^n$ is a subspace of Σ^n then C is said to be a linear code.*

As C is a subspace, there exists a basis $c_1, c_2, .., c_k$ where k is the dimension of the subspace. Any codeword can be expressed as the linear combination of these basis vectors. A linear code of dimension k can be described in terms of a $n \times k$ matrix G as:

$$c = \{Gx | x \in \Sigma^k\}$$

A linear code over a field of q elements that has block length n, dimension k and minimum distance d will be denoted compactly as an $[n, k, d]_q$ code. Since Σ is a field q needs to be a prime power. The information rate for a $[n, k, d]_q$ code is simply $\frac{k}{n}$. The Hamming code we discussed earlier is a linear code and can be represented as $[7, 4, 3]_2$ code.

As we have seen a linear code $[n, k, d]_q$ forms a subspace and the generator matrix G is used to define the subspace uniquely. The dimension of G is $n \times k$. Another way of defining the subspace C is to define its null space, i.e. the subspace $C^\perp$ of all vectors which are orthogonal to every vector in C. Let H represent $(n-k) \times n$ generator matrix of $C^\perp$ which kills every vector in C. The matrix H is called the parity check matrix and can also be used to uniquely define the code C.

Proposition 4.2. *The minimum distance of code C with parity check matrix H is the smallest number d such that there exist d columns of H which are linearly dependent.*

Proof: The minimum distance of a linear code is equal to the smallest possible weight of a non-zero code. This is true as the all zeroes string is a codeword for any linear code. Let x be a non-zero code of smallest weight d. We also know that $Hx = 0$, therefore there exists d linearly dependent columns of H. Additionally for every set of dependent columns in H there exists a non-zero code in C. Thus, the minimum distance of C is exactly equal to the size of the smallest linearly dependent column set of H.