Notes taken by **Yuanjian Liu**

This document contains Notes to the Feb.2 Lecture of the course CMSC 33581 Topics of Big Data.

NOTHING ELSE ON THIS PAGE!

1. Review

In the review part, we ask three questions:

(1) Let x_1, x_2, x_3 be Boolean variables. What is a *Linear function* of x_1, x_2, x_3?
(2) Other than *AND* describe a non-linear function over x_1, x_2, x_3?
(3) What is the Hamming distance and why does it matter?

For the first and second question, the concept of *Linear* in Boolean variables is similar to the general linear concept, which includes ADD. It's just that the ADD opeartion is XOR in Boolearn variables. Note that AND is not a linear function, because it's simliar to MULTIPLY, so x_1x_2 is certainly not a linear operation. Another non-linear function is *majority*, which chooses the majority of x_1, x_2, x_3.

For the third question, Hamming distance is the number of bit positions in which the two bits are different. It matters because when we hope to use some type of coding to tolerate or detect faults, the conclusion is related to Hamming distance.

The easiest coding is 3X replication code. The specific one with block size of 2 ($B = 2$) and output size of 6 ($N = 6$) can tolerate 1 bit error. And 2X replication code can only detect one bit error. We will later see why this is the case. It's because the minimum Hamming distance of 3X replication code is $d = 3$ and it can tolerate $\frac{d-1}{2} = 1$ bit error; while the 2X replication code has $d = 2$ and it can detect $\frac{d}{2} = 1$ bit error.

2. Linear Channel Coding

In this lecture, we mainly discussed the problem of Channel Coding, introduced a "Generator Matrix" and talked about three different cases of the dimension of the Matrix and their consequences. In general, the linear coding can be represented by the following formula: $Out = G \cdot Input$ where G is the generator matrix of dimension $N \times B$, $Input$ is the 0-1 sequence of data to be transmitted, a vector of dimension $B \times 1$; and Out is the result of this linear coding, a vector of dimension $N \times 1$. Here B means the block size of input data, in other words, each time we encode B bits to N bits using the generator matrix G and send it to the channel.

We introduced another variable d denoting the **Minimum Hamming Distance** between codes. If we hope to tolerate k errors (correction), we need to have $d \geq 2k + 1$; while if we hope to detect k errors, we need $d \geq 2k$.

When $N > B$ the linear transform generates a coding that tolerates mistakes, two of the examples are **Hamming Code**(left) and **Reed-solomon Code**(right) which have the following forms:

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

Hamming Code has $B = 4, N = 7$ and $d = 3$ so we can tolerate/detect 1 error for each block; while Reed-solomon Code has $B = 4, N = 8, d = 4$, so we can tolerate 1 error and detect 2 errors.

Now comes the question: **What is the pattern of Generator Matrix?** The answer is **different perodicity**. Let's look at the first column of Reed-solomon Code, its period is 1; the second column, period 2; the thrid column, period 4; the third column, period 8. The G matrices look like **Fourier Transform**, which separates different signal waves from a synthetic wave.

3. Certain G matrices break up the information

Till now, we are talking about the specific linear codings that have the property $N > B$. What if $N = B$? Under this situation, the matrix G is a square matrix, and has a full rank, therefore it's invertible. G now becomes an encryption method. We now will show what's the information in *Output*. The conclusion is that certain types of G matrices break up the information in *Input*, and *Output* has a higher entropy.

Let's consider $B = 2$, and the two bits in the *Input* are random variables X_1, X_2 which follows 0-1 distribution with probablity p to be 1. So in total, there are four situations: 00, 10, 01, 11; and their probablities are $(1-p)^2, p(1-p), p(1-p), p^2$. Suppose our coding is a parity bit, the output now becomes 000, 101, 011, 110. The probablity distribution of the parity bit is 0:$1-2p(1-p)$, 1:$2p(1-p)$. If G matrix is big enough, the probablity distribution will lean towards 50-50.