COMPSCI 650 Applied Information Theory March 29, 2016

Lecture 16

Instructor: Arya Mazumdar *Scribe:* [redacted]

1 Review

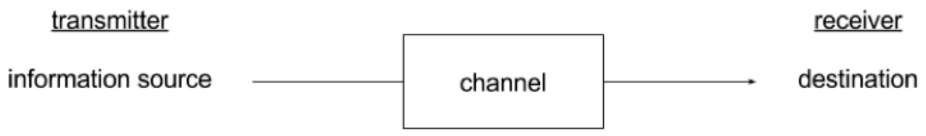

Figure 1: A general communication system

A **channel** is a block through which information is sent (from transmitter to receiver). Channels can act as random variables. Messages sent across channels are encoded into binary strings in order to enable error correction due to randomness present in the channel. **Binary symmetric channel (BSC)** are channels which will flip a bit from 0 to 1 or 1 to 0 between the transmitter and receiver with probability p. **binary erasure channel(BEC)** are channels which will erase a bit from the code with probability p.

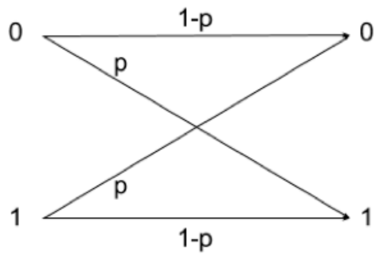

Figure 2: Binary symmetric channel (BSC)

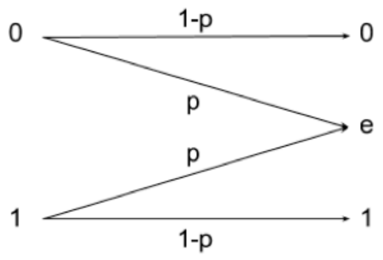

Figure 3: Binary erasure channel (BEC)

The **channel capacity** is defined as $\max_{p(x)} I(X;Y)$. There exist codes which can use a channel to its capacity. They can be generated randomly, but a random code is expensive to decode at the reciever, so how can we encode messages to use a channel's capacity and also decode the messages cheaply?

A simple method for decoding messages is to use the hamming distance between the recieved codeword and a table of codewords that maps to the actual message. Suppose code = {000, 111}, and we receive 001. Compute the Hamming distance between 000 and 001, and between 111 and 001, and pick the one with the minimum distance.

2 Linear codes

A linear code is a code which is comprised of all of the linear combinations of a set of lienar equations. The binary addition of any two valid codewords in a linear code will produce another valid codeword. However, when using hamming distance to decode a linear code, one must consider the following: $M = 2^k, \frac{k}{n} = \frac{1}{3}, k = O(n)$. Since M is exponential, you have to do exponential number of compressions/decoding, and we want to do better than that. Goal: A linear code that achieves that capacity of BSC

$(x_1, ..., x_n) \in \mathbb{R}^n$ is called a vector space. The field that we use is the field of 0 and 1, $\mathbb{F}_2 = \{0, 1\}$. Our codewords are vector spaces in $\mathbb{F}_2$. Defined operations are addition $(+)$ and multiplication$(\cdot)$.

XOR		
$+$	0	1
0	0	1
1	1	0

Table 1: Addition

AND		
$\cdot$	0	1
0	0	0
1	0	1

Table 2: Multiplication

The **dot product** of two vectors $\mathbf{x}$ and $\mathbf{y}$ is defined as $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_i x_i y_i$.
Example:

m_1	000	000000
m_2	001	001111
m_3	010	010110
m_4	011	011001
m_5	100	100101
m_6	101	101010
m_7	110	110011
m_8	111	111100

Table 3: Example of a linear code

The hamming distance between any two of the codes above is at least 3, so if there is an error, there is only one correct code that is closest in Hamming distance.

2.1 Single-parity check code

If even parity (i.e., number of ones), then append 0 at the end. Otherwise, if odd parity, then append 1 at the end.

The generator matrix G of the example 4 above is as follows:

$$G = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{1}$$

We would be able to recover single erasure because we can calculate the parity of the remaining bits. Always an even number of ones. C is a linear code in linear subspace of $\mathbb{F}_2^n$. A linear space has the following attributes:

1. Is an abelian group
2. The unit vector is the all-zero vector
3. Multiplcation by scalars is distributive
4. Multiplacation is associative

For any two codewords c_i, c_j:

$$\begin{aligned} c_i + c_j &= m_i G + m_j G \qquad (2)\\ &= (m_i + m_j)G \qquad (3)\\ &= m_k G = c_k \qquad (4) \end{aligned}$$

Therefore, linear codes are closed under addition.
Now, consider:

$$G = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{5}$$

then $C = \{0000, 0001, 0010, 0011\}$. But C is not a good code, because the minimum hamming distance is only 1 bit.
Now, consider code $A = \{00000, 11111\}$. The generator is $G = (11111)$.
The generator matrix of the **parity-check code** B, $n = 5$ is:

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \tag{6}$$

The last bit of G above is the parity bit.

m_1	0000	00000
m_2	0001	00011
m_3	0010	00101
m_4	0011	00110
m_5	0100	01001

Table 4: Example of a linear code

If there are m codewords, we have to compute $\binom{m}{2}$ distances. But it is possible to only compute m distances.

Codewords can be written as a solution of some form of linear equation. The completed code is all binary strings which are valid solutions to the linear equation. Consider a code of length 6: $\mathbf{x} = (x_1, x_2, ..., x_6)$. Suppose:

$$\begin{cases} x_1 + x_2 + x_3 + x_4 = 0 \\ x_2 + x_3 + x_5 = 0 \\ x_1 + x_3 + x_6 = 0 \end{cases}$$

Then we have:

$$H = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where H is the **parity-check matrix** of the code.

Let $C(n,k)$ be a linear code of length n and dimension k, such that $0 \leq k \leq n$. What is a **systematic code**? Anything linear code can do, systematic code can also do. Linear code is a mapping $C : \{0,1\}^k \to \{0,1\}^n$.

Tanner-graph representation:
A tanner graph is a bipartite graph used to construct a code. The edges between variable nodes (columns) and check nodes (rows) indicate a portion of the linear equation used to generate the code.