6.440 Essential Coding Theory Feb 13, 2005

Lecture 3

Lecturer: Madhu Sudan *Scribe: Shirley Shi*

1 Administrative

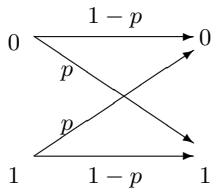
1. Problem set 1 due today at 11pm, by email.
2. Swastik (TA) will hold office hours on Wednesdays 5-7pm in G631.

2 Lecture Overview

1. Converse Coding Theorem for BSC
2. Error correcting codes
   - Parameters of interest
   - Greedy codes: Gilbert bound
   - Linear codes; Varshamov bound

3 Converse Coding Theorem for the BSC

Recall the Binary Symmetric Channel (BSC), a discrete memoryless channel which flips an input bit with probability p, and makes a correct transmission with probability $1-p$. The following picture illustrates the BSC:



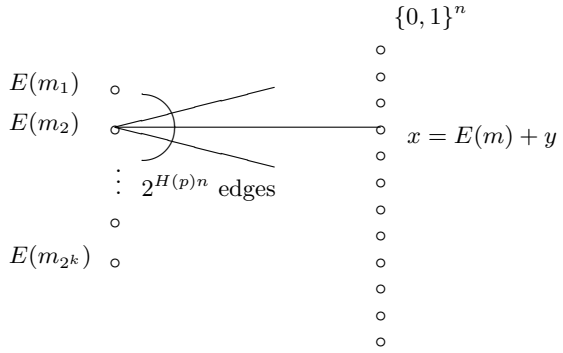
Shannon's Coding Theorem :

Capacity of the BSC is $C = C(p) = 1 - H(p)$; $\forall R < C$ and sufficiently large n, $\exists E : \{0,1\}^{Rn} \to \{0,1\}^n$ and $D : \{0,1\}^n \to \{0,1\}^{Rn}$, encoding and decoding functions mapping $k = Rn$ to n bits and back, such that the probability of decoding error is exponentially small, i.e., $\Pr[D(E(m) + error) \neq m] \to \exp(-n)$.

Converse Coding Theorem :

$\forall R > C$ and sufficiently large n, $\forall E : \{0,1\}^{Rn} \to \{0,1\}^n$, $D : \{0,1\}^n \to \{0,1\}^{Rn}$, the probability of correct decoding is exponentially small, i.e., Pr[Decoding correctly] $\to$ $\exp(-n)$.

Before proceeding to the proof, let us try to understand this statement intuitively. The theorem says that for any encoder and any decoder, the decoding result is wrong almost all the time. This would be expected if many errors can occur, for the probability of at least one of them happening would be high. To illustrate this idea, consider the following bipartite graph, where codewords are listed on the left, and received codewords are listed on the right. An edge connects a node on the left to one on the right if a likely (typical) error occurs.



We have shown during last lecture that the set of typical error sequences is exponential in size; hence there are exponentially many edges connected to each codeword on the left, where all error events are almost equally likely. Since each codeword is distributed over a much larger space of possibilities, any decoder would be hopeless!

Proof:

Fix R, n, E, and D, where

- $R > C + \epsilon$, ϵ is some positive constant;
- n is sufficiently large;
- $E : \{0,1\}^{Rn} \to \{0,1\}^n$ is the encoding function;
- $D : \{0,1\}^n \to \{0,1\}^k, k = Rn$, is some arbitrary decoding function which does not necessarily employ the minimum distance decoding strategy.

Also assume the message m has been uniformly generated on $\{0,1\}^k$, and y is an error sequence generated by the binary symmetric channel, i.e., $y \in BSC(p)\{0,1\}^n$. To find the probability of successful decoding, consider the following events that contribute to this probability:

Bad Events

E_1 : "Too few errors", defined as $\text{wt}(y) \leq (p - \frac{\epsilon}{2}) \cdot n$.

E_2 : $\exists x$ s.t. $E_2(x)$ is true, where $E_2(x)$ includes three independent cases:

(i) too many errors: # errors $\geq (p - \frac{\epsilon}{2})n$, i.e., $\text{wt}(E(D(x)) - x) \geq (p - \frac{\epsilon}{2})n$

(ii) the message sent is the decoding of x, i.e., $m = D(x)$

(iii) y takes $E(m)$ to x, i.e., $y = x - E(D(x))$

To prove the theorem we want to show that

1. $\Pr(E_1)$ is small. This can be achieved by using Chernoff bounds. In simple terms, since error sequences with "too few errors" are atypical, we can ignore their contributions, regardless if the decoder is able to decipher the received message correctly.

2. $\Pr(E_2)$ is small

   (i) this event is a function of x instead of being random, so it occurs with probability of either 1 or 0.

   (ii) $\Pr(m = D(x)) = 2^{-k} = 2^{-Rn}$

   (iii) Since E_1 occurs with low probability, it suffices to examine

$$\Pr[y = x - E(D(x))|\mathrm{wt}(x - E(D(x))) \geq (p - \frac{\epsilon}{2})n] \leq \frac{1}{\binom{n}{(p-\frac{\epsilon}{2})n}} \approx 2^{-H(p)n}$$

   Summarize:

$$\forall x\,, \quad \Pr(E_2) \leq 2^{-Rn} \cdot 2^{-H(p)n} = 2^{-(R+H(p))n}\,, \quad R + H(p) > C + \epsilon - C + 1 > 1.$$

   Applying the union bound then gives:

$$\Pr\left[\exists\, x \text{ s.t. } E_2(x) \text{ holds}\right] \leq \sum_x \Pr(E_2(x)) \leq 2^n 2^{-(R+H(p))n} \to \exp(-n)\,.$$

   Observe that both E_1 and E_2 occur with exponenetially small probabilities. Therefore the next case is dominating in the decoding process.

3. If neither E_1 nor E_2 occurs, we have a decoding failure.

   Assume $\neg E_1, \neg E_2$, and decoding is successful. Set $x = E(m) + y$, then

   - Since decoding is successful, $D(x) = m$.
   - $y = x - E(m) = x - E(D(x))$
   - $\neg E_1 \Rightarrow \mathrm{wt}(y) \geq (p - \frac{\epsilon}{2})n \Rightarrow \mathrm{wt}(x - E(D(x))) \geq (p - \frac{\epsilon}{2})n$

   These conditions correspond to E_2 exactly, leading to a contradiction.

4 Remarks on Shannon's Theory

1. Shannon's theory has much wider applicability than just the BSC, but a caveat is that there is no overall characterization of when the theory actually holds. Even when it does hold, capacity is not always computable. An example is the deletion channel, where bits are dropped with probability p. Shannon's theorem says there is a capacity to the channel, but we don't know what it is. Note a deletion channel differs from an erasure channel in the sense that the receiver does not know which bit has been corrupted (skipped).

2. (Shannon vs. Hamming) Shannon's theory is non-constructive. Although it discusses encoding and decoding functions, it doesn't tell how to design the encoder/decoder or the code itself, to achieve the derived bounds; nor does it give any criterion for ensuring a designed code works. By comparison, Hamming constructs the code C directly and proves whether the code works, and whether it is a good code. Nonetheless, although C is the image of the encoding function E, Hamming makes no explicit reference to the encoding and decoding functions.

   Another difference between Shannon's and Hamming's theories is the error models they are based on. Shannon assumes errors are random hence probabilistic, while Hamming considers the worse case error, subject to upper bounds on the total number of errors that can be tolerated.

   In this course we will use Hamming's theory, for it is more constructive, and can at least prove if a code C is good or not.

5 Error Correcting Codes

5.1 Notations

First, recall the distance between two strings $x, y \in \Sigma^n$ is the number of coordinates i where x_i differs from y_i. We denote this distance by $\Delta(x, y)$. A code C is a subset of Σ^n; the distance of a code, $\Delta(C)$, is the distance between the closest pair of points, and is given by $\Delta(C) = \min_{x \neq y, x, y \in C}\{\Delta(x, y)\}$. The ball of radius $r \in \mathbb{Z}^+$ centered at $x \in \Sigma^n$ is the set of all points within a distance r to x: $Ball(x, r) \triangleq \{y | \Delta(x, y) \leq r\}$. The volume of such a ball is equal to the number of points it contains: $Vol_q(n, r) = |Ball(x, r)|$; for binary codes, an appropriate approximation is $Vol_2(n, r) \approx 2^{H(\frac{r}{n})n}$.

Also recall from the previous lecture, that to correct t-bit errors, we want the distance of C to be at least $2t+1$. To achieve such error correction capabilities with block codes, we want to maximize distance of the code when given the set of parameters $(n, k, d)_q$, where

- $q =$ is the alphabet size: $q = |\Sigma|$;
- n is the block length: $C \subseteq \Sigma^n$;
- k is the message length: $|C| \leq |\Sigma|^k$;
- d is the distance of the code C: $d = \Delta(C)$.

5.2 Codes from Hamming's Paper

In his 1950 paper, Hamming constructed the following codes:

- $(n, n-1, 2)_q$: parity check code
- $(n, n - \log_2(n+1), 3)_2$: Hamming code
- $(n, n - \log_2(n+1) - 1, 4)_2$: Hamming code with additional parity check bit

Hamming proved that there exists a $(n, k, d)_2$ code with d odd. By adding one extra parity bit, a $(n+1, k, d+1)_2$ code can also be achieved. It can also be shown that there is a $(n, n, 1)_2$ code.

For an arbitrary block code, a Hamming bound exists to give a limit on the code parameters. In some cases this bound can be used to prove that certain codes do not exist.

In addition, observe that while more errors can be corrected when the distance of the code is larger, a tradeoff occurs on the rate of the code, since more redundancy is needed. The tradeoff between rate and distance is an important criterion for measuring the goodness of a code.

5.3 Linear Codes

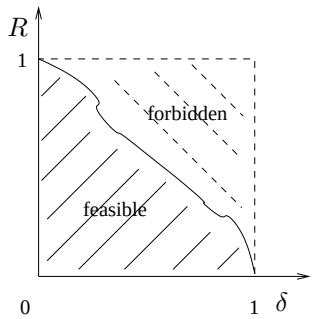
A code C is said to be linear if

- the alphabet is a finite field: $\Sigma = \mathbb{F}_q$ (guaranteed to exist when q is power of a prime number)
- $\forall x, y \in C \subseteq \mathbb{F}_q^n$, $\forall \alpha \in \mathbb{F}_q$, $\alpha x \in C, x + y \in C$

Hence any linear combinations of codewords is also a codeword, so a linear code is a closed subset of the vector space $\mathbb{F}_q^n$. In other words, C is linear if and only if $\exists\, G$ of size $k \times n$, such that $C = \{x \cdot G | x \in \mathbb{F}_q^k\}$. Since G can also be defined by its null space, we can equivalently conclude that a code is linear iff $\exists\, H$ of size $n \times (n-k)$ s.t. $C = \{y | yH = \mathbf{0}\}$.

Suppose we fix q to be some constant and let n be large. We want to study the relationship between the rate $R = \frac{k}{n}$ and distance $\delta \triangleq \frac{d}{n}$ of the code. Let us look at the case of binary codes, where $q = 2$. This offers a lot of insights, although some additional considerations may be needed when generalizing to larger values of q.

Before Shannon's theory, the only achievable rate-distance pairs are located on the two axes. Shannon showed that by using random codes, it is possible to have both values positive, although Shannon did not give a way of finding such codes.



5.4 Gilbert Greedy Algorithm

Given n and δ, the Gilbert Greedy algorithm constructs a code of distance $d = \delta n$. It does so by adding codewords to the cookbook one at a time, while removing all points within a distance of $d - 1$ from this the new codeword:

$C \leftarrow \emptyset$, $S \leftarrow \{0,1\}^n$

while $\exists\, x \in S$, do

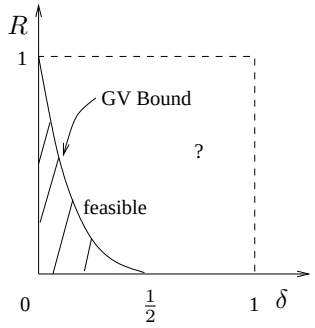
$$C \leftarrow C \cup \{x\}$$
$$S \leftarrow S - Ball(x, d-1)$$

output C

This algorithm runs in exponential time. To find out the performance of the resulting code, first observe that the following claims are true:

- $\Delta(C) \geq d$, for all points within a distance of $d-1$ from each codeword are discarded.
- $|C| \geq \dfrac{2^n}{Vol(n, d-1)} \approx 2^{(1-H(\delta))n}$

We conclude that $\exists\, (n,k,d)_2$ codes with rate $\frac{k}{n} \to 1 - H(\delta)$. The corresponding rate-distance curve is shown below. We call the upper bound of the feasible region obtained by the Gilbert greedy algorithm the GV bound, for Gilbert and Varshamov. It is unclear if any of the points above this bound is achievable.



For $d = 3$, let us consider the $(7,4,3)_2$ Hamming code. The GV bound gives $\dfrac{2^n}{n^2} = \dfrac{2^7}{49}$ as the code size, but clearly Hamming code achieves a much better rate. Can we find another algorithm that gives better performance?

5.5 Varshamov's Greedy Algorithm

Fact: for a matrix $H = \begin{bmatrix} h_1^T & h_2^T & \dots & h_n^T \end{bmatrix}^T$, H defines a code C with distance d iff $\forall$ subset $h_{i_1}, \dots, h_{i_{d-1}}$ is linearly independent. Varshamov proposed the following greedy algorithm based on this fact:

$H \leftarrow \emptyset$

while $\exists$ row $v \in \{0,1\}^{n-k}$ s.t. v is linearly independent of every subset of $d-2$ rows of H,

$H \leftarrow H \cup v$

output H

We will show next time that the code constructed with this algorithm achieves the Hamming bound for $d = 3$, namely

$$2^k \geq \frac{2^n}{Vol(n, d-2)}\,.$$