Part II – Coding and Cryptography (Rough)

Based on lectures by Dr R. Camina

Notes taken by Bhavik Mehta

Lent 2017

Contents

1 Noiseless Coding **4**

1.1 Mathematical Entropy . . . 6

1.2 Shannon-Fano coding . . . 9

1.3 Huffman coding . . . 10

2 Error Control Codes **15**

2.1 New codes from old . . . 19

2.2 Bound on codes . . . 19

2.3 Channel Capacity . . . 21

2.4 Conditional Entropy . . . 23

2.5 Linear Codes . . . 29

2.6 New codes from old (again) . . . 32

2.7 Reed-Muller Codes . . . 33

2.8 Cyclic Codes . . . 36

2.9 BCH (Bose-Chaudhuri-Hocquenghem) Codes . . . 38

2.10 Shift registers . . . 41

3 Cryptography **44**

3.1 Unicity distance . . . 45

3.2 Stream Cipher . . . 46

3.3 New key streams from old . . . 47

3.4 Public Key Cryptography . . . 48

3.5 Rabin-Williams Cryptosystem (1979) . . . 48

3.6 RSA . . . 49

3.7 Diffie-Hellman key exchange . . . 51

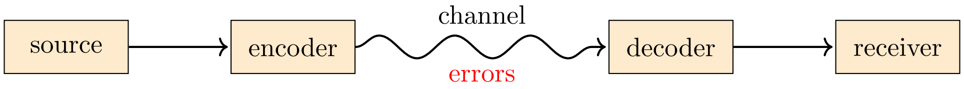
3.8 Authenticity and signatures . . . 51

3.9 Authenticity using RSA . . . 51

3.10 El-Gamal Signature Scheme . . . 52

Introduction to communication channels and coding

For example, given a message m=‘Call me!’ which we wish to send by email, first encode as binary strings using ASCII. So, $f(C) = 1000011, f(a) = 1100001$, and $f^*(m) = 1000011\ 1100001 \ldots 0100001$.



Basic problem: Given a source and a channel (described probabilistically) we aim to design an encoder and a decoder in order to transmit information both *economically* and *reliably* (coding) and maybe also to *preserve privacy* (cryptography).

Example.

- ‘Economically’: In Morse code, common letters have shorter codewords:

$$A = .\text{-} \quad E = . \quad Q = \text{--.-}$$

- ‘Reliably’: Every book has an ISBN of form $a_1 a_2 \ldots a_{10}$ where $a_i \in \{0, 1, \ldots, 9\}$ for $1 \leq i \leq 9$, $a_{10} \in \{0, 1, \ldots, 9, X\}$ such that

$$10a_1 + 9a_2 + \ldots + a_{10} \equiv 0 \pmod{11}$$

  so errors can be detected (but not corrected). Similarly a 13-digit ISBN has

$$x_1 + 3x_2 + x_3 + 3x_4 + \ldots + 3x_{12} + x_{13} \equiv 0 \pmod{10}$$

  for $0 \leq x_i \leq 10$, doesn’t necessarily spot transpositions.

- ‘Preserve privacy’ e.g. RSA.

A **communication channel** takes letters from an input alphabet $\Sigma_1 = \{a_1, \ldots, a_r\}$ and emits letters form an output alphabet $\Sigma_2 = \{b_1, \ldots, b_s\}$.

A channel is determined by the probabilities

$$\mathbb{P}(y_1 \ldots y_k \text{ received} \mid x_1 \ldots x_k \text{ sent})$$

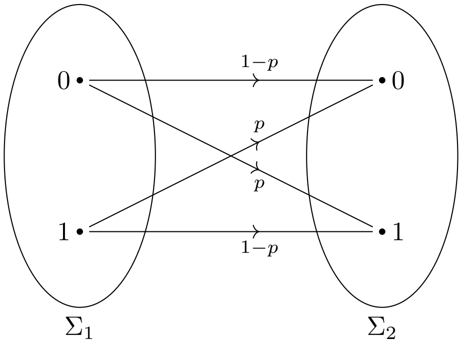
Definition (Discrete memoryless channel)**.** A **discrete memoryless channel** (DMC) is a channel for which

$$P_{ij} = \mathbb{P}(b_j \text{ received} \mid a_i \text{ sent})$$

is the same each time the channel is used and is independent of all past and future uses.

The channel matrix is the $r \times s$ matrix with entries p_{ij} (note the rows sum to 1).

Example. Binary Symmetric Channel (BSC) has $\Sigma_1 = \Sigma_2 = \{0, 1\}$, $0 \leq p \leq 1$:

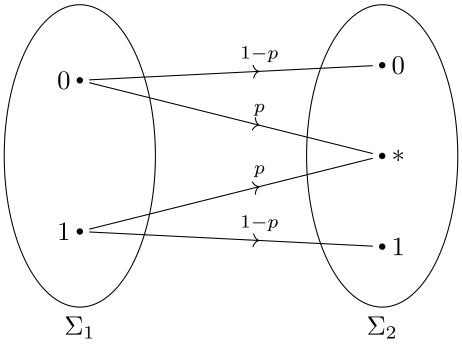


with channel matrix

$$\begin{pmatrix} 1-p & p \\ p & 1-p \end{pmatrix}$$

i.e. p is the probability a symbol is mistransmitted.

Another example is given by the Binary Erasure channel, $\Sigma_1\{0,1\}$, $\Sigma_2 = \{0,1,*\}$ and $0 \leq p \leq 1$.



with channel matrix

$$\begin{pmatrix} 1-p & p & 0 \\ 0 & p & 1-p \end{pmatrix}$$

i.e. p is the probability a symbol can't be read.

Informally, a channel's capacity is the highest rate at which information can be reliably transmitted over the channel. Rate refers to units of information per unit time, which we want to be high. Similarly, reliably means we want an arbitrarily small error probability.

1 Noiseless Coding

Notation. For Σ an alphabet, let $\Sigma^* = \bigcup_{n \geq 0} \Sigma^n$ be the set of all finite strings of elements of Σ.

If $x = x_1 \dots x_r$, $y = y_1 \dots y_s$ are strings from Σ, write xy for the concatenation $x_1 \dots x_r y_1 \dots y_s$. Further, $|x_1 \dots x_r y_1 \dots y_s| = r + s$, the length of the string.

Definition (Code)**.** Let Σ_1, Σ_2 be two alphabets. A **code** is a function $f : \Sigma_1 \to \Sigma_2^*$. The strings $f(x)$ for $x \in \Sigma_1$ are called **codewords**.

Example.

1) Greek fire code:

$$\begin{aligned} \Sigma_1 &= \{\alpha, \beta, \gamma, \dots, \omega\} \qquad \text{24 letters} \\ \Sigma_2 &= \{1, 2, 3, 4, 5\} \end{aligned}$$

so, $\alpha \mapsto 11, \beta \mapsto 12, \dots, \omega \mapsto 54$.

2) $\Sigma_1 = \{\text{all words in the dictionary}\}$, and $\Sigma_2 = \{A, B, \dots, Z, [\mathit{space}]\}$ and f='spell the word and a space'.

Send a message $x_1 \cdots x_n \in \Sigma_1^*$ as $f(x_1) \cdots f(x_n) \in \Sigma_2^*$ i.e. extend f to $f^* : \Sigma_1^* \to \Sigma_2^*$.

Definition (Decipherable)**.** A code f is **decipherable** if f^* is injective, i.e. every string from Σ_2^* arises from at most one message. Clearly we need f injective, but this is not enough.

Example. Take $\Sigma_1 = \{1, 2, 3, 4\}$, $\Sigma_2 = \{0, 1\}$ with

$$f(1) = 0, f(2) = 1, f(3) = 00, f(4) = 01.$$

f injective but $f^*(312) = 0001 = f^*(114)$ so f^* not decipherable.

Notation. If $|\Sigma_1| = m$, $|\Sigma_2| = a$, then we say f is an a-ary code of size m. (If $a = 2$ we say binary).

Aim. Construct decipherable codes with short word lengths.

Provided $f : \Sigma_1 \to \Sigma_2^*$ is injective, the following codes are always decipherable.

(i) A **block code** is a code with all codewords of the same length (e.g. Greek fire code).

(ii) In a **comma code**, we reserve one letter from Σ_2 that is only used to signal the end of the codeword (e.g. Example 2 above).

(iii) A **prefix-free code** is a code where no codeword is a prefix of another (if $x, y \in \Sigma_2^*$, x is a prefix of y if $y = xz$ for some $z \in \Sigma_2^*$.)

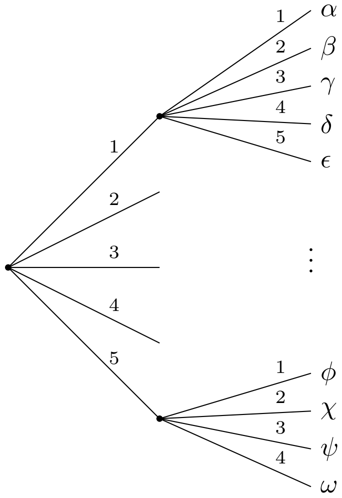
Remark. (i) and (ii) are special cases of (iii).

Prefix-free codes are also known as **instantaneous codes** (i.e. a word can be recognised as soon as it is complete) or **self-punctuating codes**.

Theorem 1.1 (Kraft's inequality)**.** Let $|\Sigma_1| = m, |\Sigma_2| = a$. A prefix-free code $f : \Sigma_1 \to \Sigma_2^*$ with word lengths $s_1, \dots, s_m$ exists iff

$$\sum_{i=1}^{m} a^{-s_i} \leq 1.$$

Proof. ($\Rightarrow$) Consider an infinite tree where each node has a descendant, labelled by the elements of Σ_2. Each codeword corresponds to a node, the path from the root to this node spelling out the codeword. For example,



Assuming f is prefix-free, no codeword is the ancestor of any other. Now view the tree as a network with water being pumped in at a constant rate and dividing the flow equally at each node.

The total amount of water we can extract at the codewords is $\sum_{i=1}^{m} a^{-s_i}$, which is therefore ≤ 1.

($\Leftarrow$) Conversely, suppose we can construct a prefix-free code with word lengths $s_1, \dots, s_m$, wlog $s_1 \leq s_2 \leq \dots \leq s_m$. We pick codewords of lengths $s_1, s_2, \dots$ sequentially ensuring previous codewords are not prefixes. Suppose there is no valid choice for the rth codeword. Then reconstructing the tree as above gives $\sum_{i=1}^{r-1} a^{-s_i} = 1$, contradicting our assumption. So we can construct a prefix-free code. (There is a more algebraic proof in Welsh.) □

Theorem 1.2 (McMillan)**.** Every decipherable code satisfies Kraft's inequality.

Proof. (Karush) Let $f : \Sigma_1 \to \Sigma_2^*$ be a decipherable code with word lengths $s_1, \dots, s_m$, let $s = \max_{1 \leq i \leq m} s_i$. Let $r \in \mathbb{N}$,

$$\left(\sum_{i=1}^{m} a^{-s_i} \right)^r = \sum_{l=1}^{rs} b_l a^{-l}$$

where b_l is the # of ways of choosing r codewords of total length l. f decipherable $\implies$ $b_l \leq |\Sigma_2|^l = a^l$.

Thus

$$\left(\sum_{i=1}^{m} a^{-s_i}\right)^r \le \sum_{l=1}^{rs} a^l a^{-l} = rs$$
$$\implies \sum_{i=1}^{m} a^{-s_i} \le (rs)^{\frac{1}{r}} \to 1 \text{ as } r \to \infty.$$

(As $\frac{\log r + \log s}{r} \to 0$ as $r \to \infty$).

$$\therefore \sum_{i=1}^{m} a^{-s_i} \le 1.$$

□

Corollary. A decipherable code with prescribed word lengths exists iff there exists a prefix-free code with the same word lengths.

So we can restrict our attention to prefix-free codes.

1.1 Mathematical Entropy

Definition (Entropy)**.** The entropy of X:

$$H(X) = H(p_1, \dots, p_n) = -\sum_{i=1}^{n} p_i \log p_i$$

where, in this course, $\log = \log_2$.

Remark.

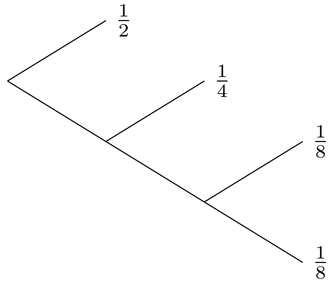
(i) If $p_i = 0$, we take $p_i \log p_i = 0$.

(ii) $H(X) \ge 0$.

Example.

1. Suppose $p_1 = p_2 = p_3 = p_4 = \frac{1}{4}$. We identify $\{x_1, x_2, x_3, x_4\}$ with {HT, HT, TH, TT}. Then $H(X) = 2$.

2. Take $(p_1, p_2, p_3, p_4) = (\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8})$.

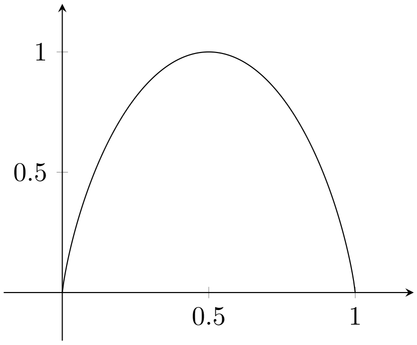


$$H(X) = \frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{8} \times 3 = \frac{7}{4}.$$

So example 1 is more random than example 2.

Entropy is a measure of 'randomness' or 'uncertainty'. Consider a random variable X taking values $x_1, \dots, x_n$ with probability $p_1, \dots, p_n$ ($\sum p_i = 1, 0 \le p_i \le 1$). The entropy $H(X)$ is roughly speaking the expected number of tosses of a fair coin needed to simulate X (or the expected number of yes/no questions we need to ask in order to establish the value of X).

Example. We toss a biased coin, $\mathbb{P}(\text{heads}) = p, \mathbb{P}(\text{tails}) = 1-p$. Write $H(p) = H(p, 1-p) = -p \log p - (1-p)\log(1-p)$. If $p = 0$ or 1, the outcome is certain and so $H(p) = 0$. Entropy is maximal where $p = \frac{1}{2}$, i.e. a fair coin.



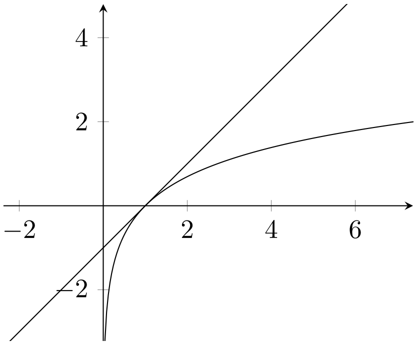
Note the entropy can also be viewed as the expected value of the information of X, where information is given by $I(X = x) = -\log_2 \mathbb{P}(X = x)$. For example, if a coin always lands heads we gain no information from tossing the coin. The entropy is the average amount of information conveyed by a random variable X.

Lemma 1.3 (Gibbs' Inequality)**.** Let $p_1, \dots, p_n$ and $q_1, \dots, q_n$ be probability distributions. Then

$$-\sum p_i \log p_i \le -\sum p_i \log q_i$$

with equality iff $p_i = q_i$.

Proof. Since $\log x = \frac{\ln x}{\ln 2}$ it suffices to prove the inequality with log replaced with ln. Note $\ln x \le x - 1$, equality iff $x = 1$.



Let $I = \{ 1 \leq i \leq n \mid p_i \neq 0 \}$

$$\begin{aligned} \ln \frac{q_i}{p_i} &\leq \frac{q_i}{p_i} - 1 \quad \forall i \in I \\ \sum_{i \in I} p_i \ln \frac{q_i}{p_i} &\leq \sum q_i - \underbrace{\sum p_i}_{=1} \leq 0 \\ \implies -\sum_{i \in I} p_i \ln p_i &\leq -\sum_{i \in I} p_i \ln q_i \\ \implies -\sum_{i=1}^{n} p_i \ln p_i &\leq -\sum_{i=1}^{n} p_i \ln q_i \end{aligned}$$

If equality holds then $\frac{q_i}{p_i} = 1 \ \forall i \in I$. So, $\sum_{i \in I} q_i = 1$ and hence $p_i = q_i$ for $1 \leq i \leq n$. □

Corollary. $H(p_1, \dots, p_n) \leq \log n$ with equality iff $p_1 = p_2 = \cdots = p_n = \frac{1}{n}$.

Proof. Take $q_1 = q_2 = \ldots = q_n = \frac{1}{n}$ in Gibbs' Inequality. □

Suppose we have two alphabets Σ_1, Σ_2 with $|\Sigma_1| = m$ and $|\Sigma_2| = a$, for $m \geq 2$ and $a \geq 2$. We model the source as a sequence of random variables $X_1, X_2, \ldots$ taking values in Σ_1.

Definition (Memoryless source)**.** A **Bernoulli** or **memoryless** source is a sequence of independently, identically distributed random variables.

That is, for each $\mu \in \Sigma_1$, $\mathbb{P}(X_i = \mu)$ is independent of i and independent of all past and future symbols emitted. Thus

$$\mathbb{P}(X_1 = x_1, X_2 = x_2, \ldots, X_k = x_k) = \prod_{i=1}^{k} \mathbb{P}(X_i = x_i).$$

Let $\Sigma_1 = \{\mu_1, \ldots, \mu_n\}$, $p_i = \mathbb{P}(X = \mu_i)$ (assume $p_i > 0$).

Definition (Expected word length)**.** The **expected word length** of a code $f : \Sigma_1 \to \Sigma_2^*$ with word lengths $s_1, \ldots, s_m$ is $E(S) = \sum_{i=1}^{m} p_i s_i$.

Definition (Optimal code)**.** A code $f : \Sigma_1 \to \Sigma_2^*$ is **optimal** if it has the shortest possible expected word length among decipherable codes.

Theorem 1.4 (Shannon's Noiseless Coding Theorem)**.** The minimum expected word length of a decipherable code $f : \Sigma_1 \to \Sigma_2^*$ satisfies

$$\frac{H(X)}{\log a} \leq E(S) < \frac{H(X)}{\log a} + 1$$

Proof. The lower bound is given by combining Gibbs' Inequality and Kraft's inequality. Let $q_i = \frac{a^{-s_i}}{c}$ where $c = \sum a^{-s_i} \le 1$ by Kraft's inequality. Note $\sum q_i = 1$.

$$\begin{aligned} H(X) = -\sum p_i \log p_i \le -\sum_i p_i \log q_i \\ &= \sum p_i (s_i \log a + \log c) \\ &= \left(\sum p_i s_i\right) \log a + \underbrace{\log c}_{\le 0} \le E(S) \log a \\ \implies \frac{H(X)}{\log a} \le E(S) \end{aligned}$$

We get equality $\iff$ $p_i = a^{-s_i}$ for some integers s_i. For the upper bound put

$$s_i = \lceil -\log_a p_i \rceil$$

where $\lceil x \rceil$ means least integer $\ge x$.

We have

$$\begin{gathered} -\log_a p_i \le s_i < -\log_a p_i + 1 \\ \implies a^{-s_i} \le p_i \implies \sum a^{-s_i} \le \sum p_i \le 1. \end{gathered}$$

So by Theorem 1.1, $\exists$ a prefix-free code with word lengths $s_1, \dots, s_m$. Also,

$$\begin{aligned} E(S) &= \sum p_i s_i \\ &< p_i(-\log_a p_i + 1) \\ &= \frac{H(X)}{\log a} + 1 \end{aligned}$$

□

Remark. The lower bound holds for all decipherable codes.

1.2 Shannon-Fano coding

Follows from above proof. Set $s_i = \lceil -\log_a p_i \rceil$ and construct a prefix-free code with word lengths $s_1, \dots, s_m$ by taking the s_i in increasing order ensuring that previous codewords are not prefixes. Kraft's inequality ensures there is enough room.

Example. Suppose $\mu_1, \dots, \mu_5$ are emitted with probabilities $0.4, 0.2, 0.2, 0.1, 0.1$.

A possible Shannon-Fano code (with $a = 2$, $\Sigma_2 = \{0, 1\}$) has

p_i	$\lceil -\log_2 p_i \rceil$	
0.4	2	00
0.2	3	010
0.2	3	100
0.1	4	1100
0.1	4	1110

This has expected word length

$$\begin{aligned} &= 2 \times 0.4 + 3 \times 0.2 + 3 \times 0.2 + 4 \times 0.1 + 4 \times 0.1 \\ &= 2.8. \end{aligned}$$

compare $H(X) \approx 2.12$.

1.3 Huffman coding

For simplicity, take $a = 2$. Take $\Sigma_1 = \{\mu_1, \dots, \mu_m\}$ with $p_i = \mathbb{P}(X = \mu_i)$. Without loss of generality, $p_1 \geq p_2 \geq \cdots \geq p_m$. Huffman coding is defined inductively.

If $m = 2$, assign codewords 0 and 1. If $\mu > 2$, find a Huffman coding in the case of messages $\mu_1, \mu_2, \dots, \nu$, with probabilities $p_1, p_2, \dots, p_{m-1} + p_m$.

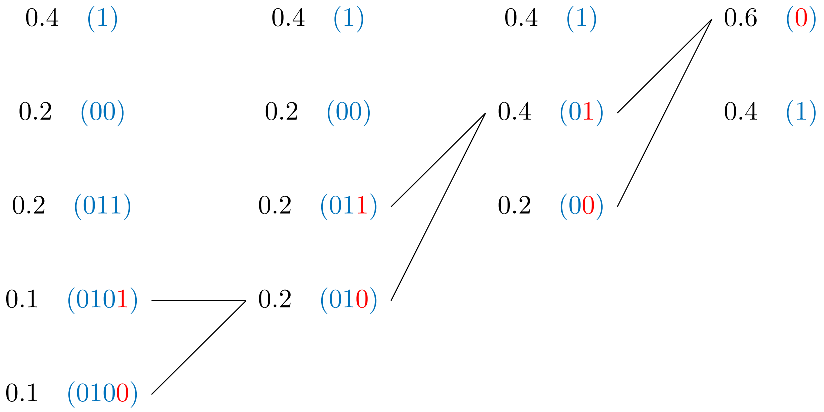
Append 0 (resp, 1) to the codeword for ν to give a codeword for μ_{m-1} (resp, μ_m).

Remark.

i) This construction gives a prefix-free code.

ii) We exercise some choice when some of the p_i are equal. So Huffman codes are not unique.

Example. Use the same example probabilities as earlier.



So $\{1, 00, 011, 0101, 0100\}$ is the prefix-free code constructed. The expected word length is:

$$\begin{aligned} &= 1 \times 0.4 + 2 \times 0.2 + 2 \times 0.2 + 4 \times 0.1 + 4 \times 0.1 \\ &= 0.4 + 0.4 + 0.6 + 0.4 + 0.4 \\ &= 2.2. \end{aligned}$$

This is better than Shannon-Fano, which gave 2.8.

Theorem 1.5. Huffman coding is optimal.

Lemma 1.6. Suppose we have $\mu_1, \dots, \mu_m \in \Sigma_1$ emitted with probabilities $p_1, \dots, p_m$. Let f be an optimal prefix-free code, with word lengths $s_1, \dots, s_m$. Then

i) If $p_i > p_j$, then $s_i \leq s_j$.

ii) $\exists$ two codewords of maximal length which are equal up to the last digit.

Proof.

i) If not, then swap the ith and jth codewords. This decreases the expected word length, contradicting f optimal.

ii) If not, then either only one codeword of maximal length, or any two codewords of maximal length differ before the last digit. In either case, delete the last digit of each codeword of maximal length. This maintains the prefix-free condition, contradicting f optimal. □

Proof of Theorem 1.5. We only take the case $a = 2$. We show by induction on m that any Huffman code of size m is optimal.
For $m = 2$, codewords $0, 1$ are optimal.
For $m > 2$, say source X_m emits $\mu_1, \dots, \mu_m$ with probabilities $p_1 \geq p_2 \geq \dots \geq p_m$ while source X_{m-1} emits $\mu_1, \dots, \mu_{m-2}, \nu$ with probabilities $p_1, \dots, p_{m-2}, p_{m-1} + p_m$.

We construct a Huffman coding f_{m-1} for X_{m-1} and extend to a Huffman coding for X_m. Then the expected codeword length satisfies:

$$E(s_m) = E(s_{m-1}) + p_{m-1} + p_m \tag{†}$$

Let f'_m be an optimal code for X_m, WLOG f'_m prefix-free. Lemma 1.6 tells us that by shuffling codewords, we may assume that the last two codewords are of maximal length and differ only in the last digit, say y_0 and y_1 for some string y.

We define a code f'_{m-1} for X_{m-1} with

$$\begin{aligned} f'_{m-1}(\mu_i) &= f'_m(\mu_i) \quad \forall 1 \leq i \leq m-2 \\ f'_{m-1}(\nu) &= y. \end{aligned}$$

Then f'_{m-1} is a prefix-free code, and the expected word length satisfies

$$E(s'_m) = E(s'_{m-1}) + p_{m-1} + p_m \tag{‡}$$

Induction hypothesis tells us f_{m-1} is optimal, so $E(s_{m-1}) \leq E(s'_{m-1})$. Hence $E(s_m) \leq E(s'_m)$ by (†), (‡). So f_m optimal. □

Remark. Not all optimal codes are Huffman. For instance, take $m = 4$, and probabilities $0.3, 0.3, 0.2, 0.2$. An optimal code is given by $00, 01, 10, 11$, but this is not Huffman.

But, the previous result says that if we have a prefix-free optimal code with word lengths $s_1, \dots, s_m$ and associated probabilites $p_1, \dots, p_m$, $\exists$ a Huffman code with these word lengths.

Definition (Joint entropy)**.** The **joint entropy** of X and Y is

$$H(X, Y) = -\sum_{x \in \Sigma_1} \sum_{y \in \Sigma_2} \mathbb{P}(X = x, Y = y) \log \mathbb{P}(X = x, Y = y).$$

Lemma 1.7. $H(X, Y) \leq H(X) + H(Y)$, with equality $\iff$ X, Y independent.

Proof. Take $\Sigma_1 = \{x_1, \dots, x_m\}$, $\Sigma_2 = \{y_1, \dots, y_n\}$. Let $p_{ij} = \mathbb{P}(X = x_i, Y = y_j)$, as well as $p_i = \mathbb{P}(X = x_i)$, $q_i = \mathbb{P}(Y = y_i)$. Apply Gibbs' Inequality to the distributions p_{ij} and $p_i q_j$:

$$\begin{aligned} -\sum p_{ij} \log(p_{ij}) &\leq -\sum p_{ij} \log(p_i q_j) \\ &= -\sum_i \left(\sum_j p_{ij} \log p_i \right) - \sum_j \left(\sum_i p_{ij} \log q_j \right) \\ &= -\sum p_i \log p_i - \sum q_j \log q_j \end{aligned}$$

That is, $H(X, Y) \leq H(X) + H(Y)$. Equality $\iff$ $p_{ij} = p_i q_j \forall i, j \iff X, Y$ independent. □

Suppose a source Ω produces a stream $X_1, X_2, \dots$ of random variables with values in Σ. The probability mass function (p.m.f.) of $X^{(n)} = (X_1, \dots, X_n)$ is given by

$$p_n(x_1, \dots, x_n) = \mathbb{P}(X_1, \dots, X_n = x_1, \dots, x_n) \quad \forall x_1, \dots, x_n \in \Sigma^n$$

Now,

$$\begin{aligned} p_n &: \Sigma^n \to \mathbb{R} \\ X^{(n)} &: \Omega \to \Sigma^n \end{aligned}$$

can form

$$p_n(X^{(n)}) : \Sigma \xrightarrow{X^{(n)}} \Sigma^n \xrightarrow{p_n} \mathbb{R}$$

a random variable sending $\omega \mapsto p_n(X^{(n)} = X^{(n)}(\omega))$.

Example. Take $\Sigma = \{A, B, C\}$, with

$$X^{(2)} = \begin{cases} AB & p = 0.3 \\ AC & p = 0.1 \\ BC & p = 0.1 \\ BA & p = 0.2 \\ CA & p = 0.25 \\ CB & p = 0.05 \end{cases}$$

So, $p_2(AB) = 0.3$, etc, and $p_2(X^{(2)})$ takes values

$$p_2(X^{(2)}) = \begin{cases} 0.3 & p = 0.3 \\ 0.1 & p = 0.2 \\ 0.2 & p = 0.2 \\ 0.25 & p = 0.25 \\ 0.05 & p = 0.05 \end{cases}$$

Definition (Convergence in probability)**.** A sequence of random variables $X_1, X_2, \dots$ **converges in probability** to $c \in \mathbb{R}$, written $X_n \xrightarrow{p} c$ as $n \to \infty$, if

$$\forall \epsilon > 0 \quad \mathbb{P}(|X_n - c| \le \epsilon) \to 1 \quad \text{as } n \to \infty.$$

So, X_n and c can take very different values for large n, but only on a set with small probability.

Theorem (Weak law of large numbers)**.** $X_1, X_2, \dots$ an independent, identically distributed sequence of random variables with finite expected value μ, then

$$\frac{1}{n} \sum_{i=1}^{n} X_i \xrightarrow{p} \mu \quad \text{as } n \to \infty.$$

Example (Application)**.** Take $X_1, X_2, \dots$ a Bernoulli source. Then $p(X_1), p(X_2), \dots$ are i.i.d. random variables

$$\begin{aligned} p(X_1, \dots, X_n) &= p(X_1) \dots p(X_n) \\ -\frac{1}{n} \log p(X_1, \dots, X_n) &= -\frac{1}{n} \sum_{i=1}^{n} \log p(X_i) \xrightarrow{p} E(-\log p(X_1)) = H(X_1) \quad \text{as } n \to \infty. \end{aligned}$$

Definition (Asymptotic Equipartition Property)**.**

A source $X_1, X_2, \dots$ satisfies the **Asymptotic Equipartition Property** (AEP) if for some $H \geq 0$ we have

$$-\frac{1}{n}\log p(X_1, \dots, X_n) \xrightarrow{p} H \quad \text{as } n \to \infty.$$

Example. Consider a coin, $p(H) = p$. If coin tossed N times, expect approximately pN heads and $(1-p)N$ tails.

$$\begin{aligned}
&\mathbb{P}(\text{particular sequence of } pN \text{ heads and } (1-p)N \text{ tails}) \\
=&p^{pN}(1-p)^{(1-p)N} \\
=&2^{N(p\log p)+(1-p)\log(1-p)} = 2^{-NH(A)}
\end{aligned}$$

where A is the result of an independent coin toss. So, with high probability we will get a typical sequence, and its probability will be close to $2^{-NH(A)}$.

Lemma 1.8. A source $X_1, \dots, X_n$ satisfies AEP iff it satisfies the following.

$$\forall \epsilon > 0,\ \exists n_0(\epsilon) \text{ such that } \forall n \geq n_0(\epsilon)\, \exists \text{ a 'typical set' } T_n \subset \Sigma^n \text{ with}$$

$$\begin{gathered}
\mathbb{P}((X_1, \dots, X_n) \in T_n) > 1 - \epsilon \\
2^{-n(H+\epsilon)} \leq p(x_1, \dots, x_n) \leq 2^{-n(H-\epsilon)} \quad \forall (x_1, \dots, x_n) \in T^n
\end{gathered} \tag{$*$}$$

Sketch proof. AEP $\Rightarrow$ $(*)$. Take

$$\begin{aligned}
T_n &= \left\{ (x_1, \dots, x_n) \in \Sigma^n \;\middle|\; \left| -\frac{1}{n}\log p(x_1, \dots, x_n) - H \right| < \epsilon \right\} \\
&= \left\{ (x_1, \dots, x_n) \in \Sigma^n \;\middle|\; 2^{-n(H+\epsilon)} \leq p(x_1, \dots, x_n) \leq 2^{-n(H-\epsilon)} \right\}
\end{aligned}$$

$(*)$ $\Rightarrow$ AEP

$$\mathbb{P}\left(\left| -\frac{1}{n} p(X_1, \dots, X_n) - H \right| < \epsilon \right) \geq \mathbb{P}(T_n) \to 1 \quad \text{as } n \to \infty. \qquad \square$$

Definition (Reliably encodable)**.** A source $X_1, X_2, \dots$ is **reliably encodable** at rate r if $\exists A_n \subset \Sigma^n$ for each n such that

(i) $\frac{\log|A_n|}{n} \to r \quad \text{as } n \to \infty$

(ii) $\mathbb{P}((X_1, \dots, X_n) \in A_n) \to 1 \quad \text{as } n \to \infty$.

So, in principle you can encode at rate almost r with negligible error for long enough strings.

So, if $|\Sigma| = a$, you can reliably encode at rate $\log a$. However you can often do better. For example, consider telegraph English with 26 letters and a space. $27 \approx 2^{4.756}$, so can encode at rate of 4.76 bits/letter. But much lower rates suffice, as there is a lot of redundancy in the English language. Hence the following definition.

Definition (Information rate)**.** The **information rate** H of a source is the infimum of all values at which it is reliably encodable.

Roughly, nH is the number of bits required to encode $(X_1, \dots, X_n)$.

Theorem 1.9 (Shannon's first coding theorem)**.** If a source satisfies AEP with some constant H, then the source has information rate H.

Proof. Let $\epsilon > 0$ and let $T_n \subset \Sigma^n$ be typical sets. Then for sufficiently large $n \geq n_0(\epsilon)$,

$$p(x_1, \dots, x_n) \geq 2^{-n(H+\epsilon))} \quad \forall (x_1, \dots, x_n) \in T^n.$$

So,

$$\begin{aligned} \mathbb{P}((X_1, \dots, X_n) \in T_n) &\geq 2^{-n(H+\epsilon)} \left| T_n \right| \\ \implies \left| T_n \right| &\leq 2^{n(H+\epsilon)} \end{aligned}$$

therefore the source is reliably encodable at rate $H + \epsilon$.

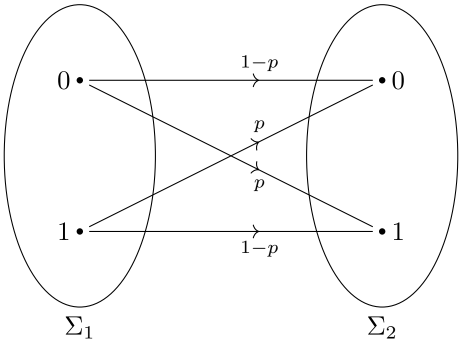
Conversely, if $H = 0$, done. Otherwise, pick $0 < \epsilon < \frac{H}{2}$. Suppose the source is reliably encodable at rate $H - 2\epsilon$, say with sets A_n.

$$\begin{aligned} p(x_1, \dots, x_n) &\leq 2^{-n(H-\epsilon)} \quad \forall (x_1, \dots, x_n) \in T_n \\ \implies \mathbb{P}((x_1, \dots, x_n) \in A_n \cap T_n) &\leq 2^{-n(H-\epsilon)} \left| A_n \right| \\ \implies \frac{\log \mathbb{P}(A_n \cap T_n)}{n} &\leq -(H - \epsilon) + \frac{\log \left| A_n \right|}{n} \\ &\to -(H - \epsilon) + (H - 2\epsilon) = -\epsilon \quad \text{as } n \to \infty. \end{aligned}$$ □

2 Error Control Codes

Definition (Binary code)**.** A $[n, m]$ **binary code** is a subset $C \subset \{0,1\}^n$ of size $|C| = m$. We say C has length n. The elements of C are called **codewords**.

We use a $[n, m]$-code to send one of m possible messages through a BSC making use of the channel n times.



Definition (Information rate)**.** The **information rate** of C is

$$\rho(C) = \frac{\log m}{n}$$

where we continue to use $\log = \log_2$.

Note since $C \subset \{0,1\}^n$, $\rho(C) \le 1$, with equality iff $C = \{0,1\}^n$. A code with size $m = 1$ has information rate 0.

We aim to design codes with both a large information rate and a small error rate, which are contradicting aims. The error rate depends on the decoding rule. We consider 3 possible rules.

(i) The **ideal observer** decoding rule decodes $x \in \{0,1\}^n$ as the codeword c maximising $\mathbb{P}(c \text{ sent} \mid x \text{ received})$.

(ii) The **maximum likelihood** decoding rule decodes $x \in \{0,1\}^n$ as the codeword c maximising $\mathbb{P}(x \text{ received} \mid c \text{ sent})$.

(iii) The **minimum distance** decoding rule decodes $x \in \{0,1\}^n$ as the codeword c minimising $\#\{\, 1 \le i \le n \mid x_i \ne c_i \,\}$.

Remark. Some convention should be agreed in the case of a 'tie', e.g. choose at random, or ask for message to be sent again.

Lemma 2.1. If all messages are equally likely, then the ideal observer and maximum likelihood rule agree.

Proof. By Bayes' rule,

$$\begin{aligned}\mathbb{P}(c \text{ sent} \mid x \text{ received}) &= \frac{\mathbb{P}(c \text{ sent}, x \text{ received})}{\mathbb{P}(x \text{ received})} \\ &= \frac{\mathbb{P}(c \text{ sent})}{\mathbb{P}(x \text{ received})}\mathbb{P}(x \text{ received} \mid c \text{ sent}).\end{aligned}$$

We suppose $\mathbb{P}(c \text{ sent})$ is independent of c. So for fixed x, maximising $\mathbb{P}(c \text{ sent} \mid x \text{ received})$ is the same as maximising $\mathbb{P}(x \text{ received} \mid c \text{ sent})$. □

Definition (Hamming distance)**.** Let $x, y \in \{0,1\}^n$. Then **Hamming distance** between x and y is

$$d(x,y) = \#\{\, 1 \le i \le n \mid x_i \ne y_i \,\}\,.$$

Lemma 2.2. If $p < \frac{1}{2}$, then maximum likelihood and minimum distance agree.

Proof. Suppose $d(x,c) = r$,

$$\mathbb{P}(x \text{ received} \mid c \text{ sent}) = p^r(1-p)^{n-r} = (1-p)^n \left(\frac{p}{1-p}\right)^r.$$

Since $p < \frac{1}{2}$, $\frac{p}{1-p} < 1$. So choosing c to maximise $\mathbb{P}(x \text{ received} \mid c \text{ sent})$ is the same as choosing c to minimise $d(x,c)$. □

Note we assumed $p < \frac{1}{2}$, which is not unreasonable.

Example. Suppose codewords 000 and 111 are sent with probabilities $\alpha = \frac{9}{10}$ and $1-\alpha = \frac{1}{10}$ respectively. We use a BSC with $p = \frac{1}{4}$. If we receive 110, how should it be decoded? Clearly minimum distance and therefore maximum likelihood (by Lemma 2.2) say decode as 111. For ideal observer:

$$\begin{aligned}
\mathbb{P}(000 \text{ sent} \mid 110 \text{ received}) &= \frac{\mathbb{P}(000 \text{ sent}, 110 \text{ received})}{\mathbb{P}(110 \text{ received})} \\
&= \frac{\alpha p^2(1-p)}{\alpha p^2(1-p) + (1-\alpha)p(1-p)^2} \\
&= \frac{\alpha p}{\alpha p + (1-\alpha)(1-p)} \\
&= \frac{9/40}{9/40 + 3/40} = \frac{3}{4} \\
\mathbb{P}(111 \text{ sent} \mid 110 \text{ received}) &= \frac{(1-\alpha)p^2(1-p)}{(1-\alpha)p^2(1-p) + \alpha p(1-p)^2} \\
&= \frac{(1-\alpha)(1-p)}{(1-\alpha)(1-p) + \alpha p} \\
&= \frac{3/40}{9/40 + 3/40} = \frac{1}{4}.
\end{aligned}$$

So the ideal observer rule says decode as 000.

Remark. The ideal observer rule is also known as the minimum-error rule. But it does rely on knowing the probabilities of the codewords sent.

From now on, we use minimum distance decoding.

Definition. For a binary code C,

(i) C is **d-error detecting** if changing at most d letters of a codeword cannot give another codeword.

(ii) C is **e-error correcting** if knowing that the string received has at most e errors is sufficient to determine which codeword was sent.

Example.

1. The repetition code of length n has:

$$C = \{\underbrace{0\cdots0}_{n}, \underbrace{1\cdots1}_{n}\}$$

This is an $[n, 2]$-code. It is $n-1$ error detecting, and $\lfloor \frac{n-1}{2} \rfloor$ error correcting. But it has information rate $= \frac{1}{n}$.

2. The simple parity check code of length n (also known as paper tape code). We view $\{0, 1\} = \mathbb{F}_2 = \mathbb{Z}/2\mathbb{Z}$ (i.e. do arithmetic modulo 2).

$$C = \left\{ (x_1, \dots, x_n) \in \{0, 1\}^n \;\middle|\; \sum_{i=1}^{n} x_i = 0 \right\}.$$

This is a $[n, 2^{n-1}]$-code. It is 1-error detecting and 0-error correcting. It has information rate $\frac{n-1}{n}$.

3. Hamming's original code (1950). Let $C \subseteq \mathbb{F}_2^7$ be defined by

$$\begin{aligned} c_1 + c_3 + c_5 + c_7 &= 0 \\ c_2 + c_3 + c_6 + c_7 &= 0 \\ c_4 + c_5 + c_6 + c_7 &= 0 \end{aligned}$$

where all arithmetic is modulo 2. Since we may choose c_3, c_5, c_6, c_7 freely, the c_1, c_2, c_4 are uniquely determined. So we get $|C| = 2^4$ i.e. C is a $[7, 16]$-code. It has information rate $\frac{\log m}{n} = \frac{4}{7}$. Note $c_{3,5,6,7}$ are free, $c_{1,2,4}$ are check digits.

Suppose we receive $x \in \mathbb{F}_2^7$. We form the syndrome $z_x = (z_1, z_2, z_4)$ where

$$\begin{aligned} z_1 &= x_1 + x_3 + x_5 + x_7 \\ z_2 &= x_2 + x_3 + x_6 + x_7 \\ z_4 &= x_4 + x_5 + x_6 + x_7. \end{aligned}$$

If $x \in C$, then $z = (0, 0, 0)$. If $d(x, c) = 1$ for some $c \in C$, then the place where x and c differ is given by $z_1 + 2z_2 + 4z_4$ (not modulo 2). Since if $x = c + e_i$ where $e_i = 0\dots010\dots0$ (1 in the ith place), then the syndrome of x is the syndrome of e_i. For example the syndrome of e_3 is $(1, 1, 0)$, the binary expansion of 3. True in fact for each $1 \le i \le 7$.

Remark. Suppose we change our code $C \subset \{0, 1\}^n$ by using the same permutation to reorder each codeword. This gives a code with the same mathematical properties (e.g. information rate, error detection rate). We say such codes are equivalent.

Lemma 2.3. The Hamming distance is a metric on $\mathbb{F}_2^n$.

Proof. Clearly $d(x, y) \geq 0$ with equality iff $x = y$. Also $d(x, y) = d(y, x)$. Check triangle inequality, let $x, y, z \in \mathbb{F}_2^n$:

$$\begin{aligned} \{\, 1 \leq i \leq n \mid x_i \neq z_i \,\} &\subseteq \{\, 1 \leq i \leq n \mid x_i \neq y_i \,\} \cup \{\, 1 \leq i \leq n \mid y_i \neq z_i \,\} \\ \implies d(x, z) &\leq d(x, y) + d(y, z) \end{aligned}$$

□

Remark. We could also write $d(x, y) = \sum_{i=1}^{n} d_1(x_i, y_i)$ where d_1 is the discrete metric on $\{0, 1\}$.

Definition (Minimum distance)**.** The **minimum distance** of a code C is the smallest Hamming distance between distinct codewords.

Notation. An $[n, m]$-code with minimum distance d is sometimes called an $\boldsymbol{[n, m, d]}$**-code**.

Remark.

- $m \leq 2^n$, with equality if $C = \mathbb{F}_2^n$, this is called the trivial code.
- $d \leq n$, with equality in the case of the repetition code.

Lemma 2.4. Let C be a code with minimum distance d.

(i) C is $(d-1)$-error detecting, but cannot detect all sets of errors.

(ii) C is $\lfloor \frac{d-1}{2} \rfloor$-error correcting, but cannot correct all sets of $\lfloor \frac{d-1}{2} \rfloor + 1$ errors.

Proof.

(i) If $x \in \mathbb{F}_2^n$ and $c \in C$ with $1 \leq d(x, c) \leq d - 1$ then $x \notin C$. So errors are detected. Suppose $c_1, c_2 \in C$ with $d(c_1, c_2) = d$. Then c_1 can be corrupted to c_2 with just d errors, this set of errors will not be detected.

(ii) Let $e = \lfloor \frac{d-1}{2} \rfloor$, so $e \leq \lfloor \frac{d-1}{2} \rfloor < e + 1$, i.e. $2e < d \leq 2(e + 1)$. Let $x \in \mathbb{F}_2^n$. If $\exists c_1 \in C$ with $d(x, c_1) \leq e$, we want to show $d(x, c_2) > e \quad \forall c_2 \in C, c_2 \neq c_1$. By the triangle inequality,

$$\begin{aligned} d(x, c_2) &\geq d(c_1, c_2) - d(x, c_1) \\ &\geq d - e \\ &> e \end{aligned}$$

so C is e-error correcting.

Let $c_1, c_2 \in C$ with $d(c_1, c_2) = d$. Let $x \in \mathbb{F}_2^n$ differ from c_1 in precisely $e + 1$ places, where c_1 and c_2 differ. Then $d(x, c_1) = e + 1$ and $d(x, c_2) = d - (e + 1) \leq e + 1$. So C cannot correct all $e + 1$ errors.

□

Example.

1) The repetition code is a $[n, 2, n]$-code, it is $n - 1$ error detecting and $\lfloor \frac{n-1}{2} \rfloor$ error correcting.

2) The simple parity check code is a $[n, 2^{n-1}, 2]$-code, it is 1-error detecting and 0-error correcting.

3) Hamming's original $[7, 16]$-code is 1-error correcting $\implies d \geq 3$. Since 0000000 and 1110000 are both valid codewords, $d = 3$. That is, it is a $[7, 16, 3]$-code.

2.1 New codes from old

Let C be an $[n, m, d]$-code.

i) The parity extension of C is

$$\overline{C} = \left\{ (c_1, c_2, \dots, c_n, \sum c_i) \,\middle|\, (c_1, \dots, c_n) \in C \right\}.$$

It is a $[n+1, m, d']$ code, for $d' = d$ or $d+1$, depending on d being odd or even.

ii) Fix $1 \leq i \leq n$. Deleting the ith letter from each codeword gives a punctured code. If $d \geq 2$, the new code is $[n-1, m, d'']$ for $d'' = d-1$ or d.

iii) Fix $1 \leq i \leq n$ and $a \in \{0, 1\}$. The shortened code is

$$\{ (c_1, \dots, c_{i-1}, c_{i+1}, \dots, c_n) \mid (c_1, \dots, c_{i-1}, a, c_{i+1}, \dots, c_n) \in C \}.$$

It is a $[n-1, m', d']$-code, where $d' \geq d$ and some choice of a gives $m' \geq \frac{m}{2}$.

2.2 Bound on codes

Definition (Hamming ball)**.** Let $x \in \mathbb{F}_2^n$ and $r \geq 0$. The (closed) **Hamming ball** with centre x and radius r is

$$B(x, r) = \{ y \in \mathbb{F}_2^n \mid d(x, y) \leq r \}.$$

Note the 'volume'

$$V(n, r) = |B(x, r)| = \sum_{i=0}^{r} \binom{n}{i}$$

which is independent of x.

Lemma 2.5 (Hamming's bound)**.** If $C \subset \mathbb{F}_2^n$ is e-error correcting, then

$$|C| \leq \frac{2^n}{V(n, e)}.$$

Proof. Since C is e-error correcting, the Hamming balls $B(c, e)$ are disjoint for $c \in C$. So

$$\begin{aligned} \sum_{c \in C} |B(c, e)| &\leq |\mathbb{F}_2^n| \\ \implies |C| \, V(n, e) &\leq 2^n \\ \implies |C| &\leq \frac{2^n}{V(n, e)}. \end{aligned}$$

□

Definition (Perfect code)**.** A $[n, m]$-code which can correct e errors is called **perfect** if

$$m = \frac{2^n}{V(n, e)}.$$

Remark. If $\frac{2^n}{V(n,e)} \notin \mathbb{Z}$, then no perfect e-error correcting code of length n can exist.

Example. Hamming's original $[7, 16, 3]$-code, can correct 1 error.

$$\frac{2^n}{V(n,e)} = \frac{2^7}{V(7,1)} = \frac{2^7}{1+7} = 2^4 = m$$

so it is a perfect code.

Remark. A perfect e-error correcting code will always incorrectly decode $e+1$ errors.

Definition (Maximum size of code)**.**

$$A(n,d) = \max\{\, m \mid \exists \text{ a } [n,m,d]\text{-code}\,\}$$

i.e. size of largest code with parameters n and d .

Example. $A(n,1) = 2^n$, the trivial code $= \mathbb{F}_2^n$. $A(n,n) = 2$, for the repetition code.

Proposition 2.6 (Gilbert-Shannon-Varshanov bound)**.**

$$\frac{2^n}{V(n,d-1)} \underset{\substack{\uparrow \\ \text{GSV}}}{\leq} A(n,d) \underset{\substack{\uparrow \\ \text{Hamming}}}{\leq} \frac{2^n}{V(n,\lfloor\frac{n-1}{2}\rfloor)}$$

Proof. Let C be a code of length n and minimum distance d of largest possible size. Then $\nexists x \in \mathbb{F}_2^n$ such that $d(x,c) \geq d \quad \forall c \in C$, otherwise we would replace C with $C \cup \{x\}$

$$\begin{aligned} &\implies \mathbb{F}_2^n \subseteq \bigcup_{c\in C} B(c, d-1) \\ &\implies 2^n \leq |C|\, V(n,d-1) \\ &\implies |C| \geq \frac{2^n}{V(n,d-1)}. \end{aligned}$$

□

Example. Take $n = 10$, $d = 3$.

$$\begin{aligned} V(n,1) &= 1 + 10 = 11 \\ V(n,2) &= 1 + 10 + \binom{10}{2} = 56 \end{aligned}$$

Proposition 2.6 gives $\frac{2^{10}}{56} \leq A(10,3) \leq \frac{2^{10}}{11}$, so $19 \leq A(10,3) \leq 93$. The exact value $A(10,3) = 72$ was only found in 1999.

There exist asymptotic versions of the Gilbert-Shannon-Varshanov bound and Hamming's bound. Let

$$\alpha(\delta) = \limsup \frac{1}{n} \log A(n, \delta n), \quad 0 \leq \delta \leq 1.$$

Notation. $H(\delta) = -\delta \log \delta - (1-\delta)\log(1-\delta)$.

The asymptotic GSV says

$$\alpha(\delta) \geq 1 - H(\delta) \quad \text{for } 0 < \delta < \frac{1}{2}$$

while the asymptotic Hamming bound says

$$\alpha(\delta) \le 1 - H\left(\frac{\delta}{2}\right).$$

We prove the asymptotic GSV bound

Proposition 2.7. Let $0 < \delta < \frac{1}{2}$. Then

(i) $\log V(n, \lfloor n\delta \rfloor) \le nH(\delta)$

(ii) $\frac{\log A(n, \lfloor n\delta \rfloor)}{n} \ge 1 - H(\delta)$

Proof (i) ⇒ (ii). The Gilbert-Shannon-Varshanov bound gives

$$\begin{aligned} A(n, \lfloor n\delta \rfloor) &\ge \frac{2^n}{V(n, \lfloor n\delta \rfloor - 1)} \\ &\ge \frac{2^n}{V(n, \lfloor n\delta \rfloor)} \\ \implies \log A(n, \lfloor n\delta \rfloor) &\ge n - \log V(n, \lfloor n\delta \rfloor) \\ &\ge n - nH(\delta) \qquad \text{by (i)} \\ \implies \frac{\log A(n, \lfloor n\delta \rfloor)}{n} &\ge 1 - H(\delta) \end{aligned}$$ □

Proof of (i).

$$\begin{aligned} 1 = (\delta + (1-\delta))^n &= \sum_{i=0}^{n} \binom{n}{i} \delta^i (1-\delta)^{n-i} \\ &\ge \sum_{i=0}^{\lfloor n\delta \rfloor} \binom{n}{i} \delta^i (1-\delta)^{n-i} \\ &= (1-\delta)^n \sum_{i=0}^{\lfloor n\delta \rfloor} \binom{n}{i} \left(\frac{\delta}{1-\delta}\right)^i \\ &\ge (1-\delta)^n \sum_{i=0}^{\lfloor n\delta \rfloor} \binom{n}{i} \left(\frac{\delta}{1-\delta}\right)^{n\delta} \\ 1 &\ge \delta^{n\delta}(1-\delta)^{n(1-\delta)} V(n, \lfloor n\delta \rfloor) \\ \implies 0 &\ge n(\delta \log \delta + (1-\delta)\log(1-\delta)) + \log V(n, \lfloor n\delta \rfloor) \\ \implies \log V(n, \lfloor n\delta \rfloor) &\le nH(\delta). \end{aligned}$$ □

2.3 Channel Capacity

Let $|\Sigma| = q$. A code of length n is a subset of Σ^n (usually we take $q = 2$).

A code is used to send messages through a discrete memoryless channel with q input letters. For each code a decoding rule is chosen.

Definition (Maximum error probability)**.** The **maximum error probability** is

$$\hat{e}(C) = \max_{c \in C} \mathbb{P}(\text{error} \mid c \text{ sent}).$$

Definition (Reliable transmission)**.** A channel can transmit **reliably at rate** R if there exists a sequence of codes $C_1, C_2, \dots$ where C_n is a code of length n and size $\lfloor 2^{nR} \rfloor$ such that

$$\hat{e}(C_n) \to 0 \text{ as } n \to \infty.$$

Definition. The (operational) **channel capacity** is the supremum over all reliable transmission rates.

Lemma 2.8. Let $\epsilon > 0$. A BSC with error probability p is used to send n digits. Then

$$\mathbb{P}(\text{BSC makes} \ge n(p+\epsilon) \text{ errors}) \to 0 \text{ as } n \to \infty.$$

Proof. Let

$$\mu_i = \begin{cases} 1 & \text{if digit mistransmitted} \\ 0 & \text{otherwise} \end{cases}$$

$\mu_1, \mu_2, \mu_3, \dots$ are i.i.d. random variables.

$$\left.\begin{aligned} \mathbb{P}(\mu_i = 1) &= p \\ \mathbb{P}(\mu_i = 0) &= 1 - p \end{aligned}\right\} \implies \mathbb{E}(\mu_i) = p$$

$$\begin{aligned} \mathbb{P}(\text{BSC makes} \ge n(p+\epsilon) \text{ errors}) &= \mathbb{P}\left(\sum_{i=1}^n \mu_i \ge n(p+\epsilon) \right) \\ &\le \mathbb{P}\left(\left| \frac{1}{n} \sum \mu_i - p \right| \ge \epsilon \right) \end{aligned}$$

This $\to 0$ as $n \to \infty$ by WLLN. □

Remark. $\sum_{i=1}^n \mu_i$ is a binomial random variable with parameters n and p.

Proposition 2.9. The capacity of a BSC with error probability $p < \frac{1}{4}$ is $\ne 0$.

Proof. Choose δ with $2p < \delta < \frac{1}{2}$. We prove reliably encoding at rate $R = 1 - H(\delta) > 0$. Let C_n be the largest code of length n and minimum distance $\lfloor n \rfloor \delta$.

$$|C_n| = A(n, \lfloor n\delta \rfloor) \ge 2^{n(1-H(\delta))} \qquad \text{by Proposition 2.7} \qquad = 2^{nR}.$$

Replacing C_n by a subcode gives $|C_n| = \lfloor 2^{nR} \rfloor$ and still minimum distance $\ge \lfloor n\delta \rfloor$. Using minimum distance decoding,

$$\begin{aligned} \hat{e}(C_n) &\le \mathbb{P}(\text{BSC makes} \ge \lfloor \frac{\lfloor n\delta \rfloor - 1}{2} \rfloor \text{errors}) && (1) \\ &\le \mathbb{P}(\text{BSC makes} \ge \lfloor \frac{n\delta - 1}{2} \rfloor \text{errors}). && (2) \\ & && (3) \end{aligned}$$

Pick $\epsilon > 0$ such that $p + \epsilon < \frac{\delta}{2}$. Then $\frac{n\delta - 1}{2} = n\left(\frac{\delta}{2} - \frac{1}{2n}\right) > n(p+\epsilon)$ for n sufficiently large. Therefore, $\hat{e}(C_n) \le \mathbb{P}(\text{BSC makes} \ge n(p+\epsilon)\text{errors}) \to 0$ as $n \to \infty$ by Lemma 2.8. □

2.4 Conditional Entropy

Definition (Conditional Entropy)**.** Let X and Y be random variables taking values in Σ_1 and Σ_2. We define

$$\begin{aligned} H(X \mid Y = y) &= -\sum_{x \in \Sigma_1} \mathbb{P}(X = x \mid Y = y) \log \mathbb{P}(X = x \mid Y = y) \\ H(X \mid Y) &= \sum_{y \in \Sigma_2} \mathbb{P}(Y = y) H(X \mid Y = y). \end{aligned}$$

Lemma 2.10. $H(X, Y) = H(X \mid Y) + H(Y)$

Proof.

$$\begin{aligned} H(X \mid Y) &= -\sum_{y \in \Sigma_2} \sum_{x \in \Sigma_1} \mathbb{P}(X = x \mid Y = y) \mathbb{P}(Y = y) \log \mathbb{P}(X = x \mid Y = y) \\ &= -\sum_{y \in \Sigma_2} \sum_{x \in \Sigma_1} \mathbb{P}(X = x, Y = y) \log \left(\frac{\mathbb{P}(X = x, Y = y)}{\mathbb{P}(Y = y)} \right) \\ &= -\sum_{y \in \Sigma_2} \sum_{x \in \Sigma_1} \mathbb{P}(X = x, Y = y) \log \mathbb{P}(X = x, Y = y) \\ &\qquad + \sum_{y \in \Sigma_2} \left(\sum_{x \in \Sigma_1} \mathbb{P}(X = x, Y = y) \right) \log \mathbb{P}(Y = y) \\ &= H(X, Y) - H(Y). \end{aligned}$$

□

Example. A fair six-sided dice is thrown. X is the value on the dice,

$$Y = \begin{cases} 0 & \text{if } X \text{ even} \\ 1 & \text{if } X \text{ odd} \end{cases}$$

$$\begin{aligned} H(X, Y) &= H(X) = \log 6 \qquad H(Y) = \log 2 = 1 \\ H(X \mid Y) &= H(X, Y) - H(Y) = \log 3 \\ H(Y, X) &= 0. \end{aligned}$$

Corollary. $H(X \mid Y) \leq H(X)$ with equality iff X and Y are independent.

Proof. Since $H(X \mid Y) = H(X, Y) - H(Y)$, this is equivalent to showing $H(X, Y) \leq H(X) + H(Y)$ with equality iff X and Y are independent. We showed this in Lemma 1.7. □

Notation. In the definition of conditional entropy we can replace random variables X and Y with random vectors $\mathbf{X} = (X_1, \ldots, X_r)$ and $\mathbf{Y} = (Y_1, \ldots, Y_s)$. This defines

$$H(X_1, \ldots, X_r \mid Y_1, \ldots, Y_s) \coloneqq H(\mathbf{X}, \mathbf{Y}).$$

Lemma 2.11. $H(X \mid Y) \leq H(X \mid Y, Z) + H(Z)$.

Proof. We expand $H(X, Y, Z)$ using Lemma 2.10 in two different ways.

$$\begin{aligned} H(X, Y, Z) &= H(Z \mid X, Y) + H(X \mid Y) + H(Y) \\ H(X, Y, Z) &= H(X \mid Y, Z) + H(Z \mid Y) + H(Y) \end{aligned}$$

Since $H(Z \mid X, Y) \geq 0$,

$$H(X \mid Y) \leq H(X \mid Y, Z) + H(Z \mid Y) \leq H(X \mid Y, Z) + H(Z).$$

□

Lemma 2.12 (Fano's inequality)**.** Let X, Y be random variables taking values in Σ_1 with $|\Sigma_1| = m$. Let $p = P(X \neq Y)$. Then $H(X \mid Y) \leq H(p) + p \log(m-1)$.

Proof. Let

$$Z = \begin{cases} 1 & \text{if } X \neq Y \\ 0 & \text{if } X = Y \end{cases}$$

so $P(Z = 1) = p$ and $P(Z = 0) = 1 - p$. By Lemma 2.11,

$$H(X \mid Y) \leq H(X \mid Y, Z) + H(Z) = H(X \mid Y, Z) + H(p).$$

Now,

$$\begin{aligned} H(X \mid Y = y, Z = 0) &= 0, && \text{as } Z = 0 \text{ implies } X = Y \\ H(X \mid Y = y, Z = 1) &\leq \log(m-1) && \text{since } m-1 \text{ choices for } X \text{ remain.} \end{aligned}$$

So,

$$\begin{aligned} H(X \mid Y, Z) &= \sum_{y,z} \mathbb{P}(Y = y, Z = z) H(X \mid Y = y, Z = z) \\ &\leq \sum_{y} \mathbb{P}(Y = y, Z = 1) \log(m-1) \\ &= \mathbb{P}(Z = 1) \log(m-1). \end{aligned}$$ □

Definition (Mutual information)**.** Let X, Y be random variables. The **mutual information** is $I(X;Y) = H(X) - H(X \mid Y)$ i.e. the amount of information about X conveyed by Y.

By Lemma 1.7 and Lemma 2.10, $I(X;Y) = H(X) + H(Y) - H(X,Y) \geq 0$ with equality iff X and Y are independent. Note the symmetry $I(X;Y) = I(Y;X)$.

Consider a DMC. Let X take values in Σ_1, where $|\Sigma_1| = m$ with probabilities $p_1, \dots, p_m$. Let Y be the random variables output when the channel is given input X.

Definition (Information channel capacity)**.** The **information channel capacity** is

$$\max_X I(X;Y).$$

Remark.

(i) The maximum is over all choices of $p_1, \dots, p_m$.

(ii) The maximum is attained, since we have a continuous function on the compact set $\{ (p_1, \dots, p_m) \mid p_i \geq 0, \sum p_i = 1 \} \subset \mathbb{R}^n$.

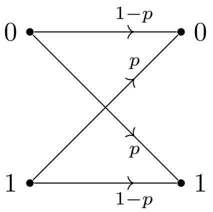
(iii) The information capacity only depends on the channel matrix.

Theorem 2.13 (Shannon's Second Coding Theorem)**.** Operational capacity = information capacity.

We will show $\leq$ in general, $\geq$ for a BSC. We now compute the capacity of certain channels using Theorem 2.13.

Capacity of a Binary Symmetric Channel

Take error probability p.

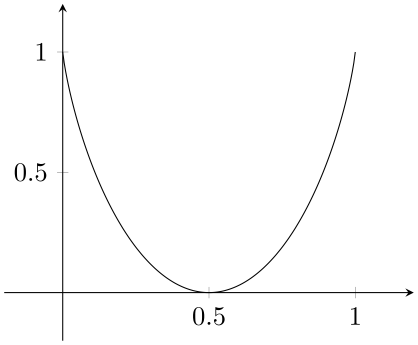


$$\begin{aligned}
\text{Input } X\text{:} \quad & \mathbb{P}(X=0) = 1-\alpha \\
& \mathbb{P}(X=1) = \alpha \\
\text{Output } Y\text{:} \quad & \mathbb{P}(Y=0) = (1-\alpha)(1-p) + \alpha p \\
& \mathbb{P}(Y=1) = \alpha(1-p) + (1-\alpha)p
\end{aligned}$$

$$\begin{aligned}
\text{Capacity is } \ C &= \max_\alpha I(X;Y) \\
&= \max_\alpha (H(Y) - H(Y \mid X)) \\
&= \max_\alpha (H(\alpha(1-p) + (1-\alpha)p) - H(p))
\end{aligned}$$

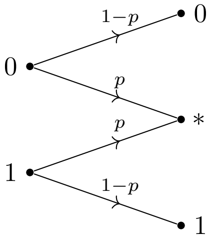
since $H(Y \mid X) = \mathbb{P}(X=0)H(p) + \mathbb{P}(X=1)H(p) = H(p)$

$$\begin{aligned}
&= 1 - H(p) \\
&= 1 + p\log p + (1-p)\log(1-p)
\end{aligned}$$



Capacity of Binary Erasure Channel

Again, take error probability p.



$$\begin{aligned} \text{Input } X\text{:} \quad & \mathbb{P}(X=0) = 1-\alpha \\ & \mathbb{P}(X=1) = \alpha \\ \text{Output } Y\text{:} \quad & \mathbb{P}(Y=0) = (1-\alpha)(1-p) \\ & \mathbb{P}(Y=1) = \alpha(1-p) \\ & \mathbb{P}(Y=*) = p \end{aligned}$$

We have $H(X \mid Y=0) = 0$, $H(X \mid Y=1) = 0$.

$$H(X \mid Y=*) = -\sum_x \mathbb{P}(X=x \mid Y=*) \log \mathbb{P}(X=x \mid Y=*)$$
$$\mathbb{P}(X=0 \mid Y=*) = \frac{\mathbb{P}(X=0, Y=*)}{\mathbb{P}(Y=*)} = \frac{(1-\alpha)p}{p} = 1-\alpha.$$

Similarly, $\mathbb{P}(X=1 \mid Y=*) = \alpha$. So $H(X \mid Y=*) = H(\alpha)$, so $H(X \mid Y) = pH(\alpha)$.

$$\begin{aligned} \text{Capacity is } \; C &= \max_\alpha I(X;Y) \\ &= \max_\alpha (H(X) - H(X \mid Y)) \\ &= \max_\alpha (H(\alpha) - pH(\alpha)) \\ &= (1-p) \max_\alpha H(\alpha) \\ &= 1-p, \quad \text{attained when } \alpha = \tfrac{1}{2}. \end{aligned}$$

We model using a channel n times as the nth extension, i.e. replace input and output alphabets Σ_1 and Σ_2 by Σ_1^n and Σ_2^n with channel probabilities

$$\mathbb{P}(y_1, \dots, y_n \text{ received} \mid x_1, \dots, x_n \text{ sent}) = \prod_{i=1}^n \mathbb{P}(y_i \text{ received} \mid x_i \text{ sent})$$

Lemma 2.14. The nth extension of a DMC with information capacity C has information capacity nC.

Proof. We take random variable input $(X_1, \dots, X_n) = \mathbf{X}$ producing random variable output $(Y_1, \dots, Y_n) = \mathbf{Y}$. Now, $H(\mathbf{Y} \mid \mathbf{X}) = \sum_{\mathbf{x}} \mathbb{P}(\mathbf{X} = \mathbf{x}) H(\mathbf{Y} \mid \mathbf{X} = \mathbf{x})$. Since the channel is memoryless, $H(\mathbf{Y} \mid \mathbf{X} = \mathbf{x}) = \sum_i H(Y_i \mid \mathbf{X} = \mathbf{x}) = \sum_i H(Y_i \mid X_i = x_i)$. So,

$$\begin{aligned} H(\mathbf{Y} \mid \mathbf{X}) &= \sum_{\mathbf{x}} \mathbb{P}(\mathbf{X} = \mathbf{x}) \sum_i H(Y_i \mid X_i = x_i) \\ &= \sum_i \sum_n H(Y_i \mid X_i = u) \mathbb{P}(X_i = u) \\ &= \sum_i H(Y_i \mid X_i). \end{aligned}$$

Now $H(\mathbf{Y}) \leq H(Y_1) + \cdots + H(Y_n)$ by Lemma 1.7, thus

$$\begin{aligned} I(X;Y) &= H(\mathbf{Y}) - H(\mathbf{Y} \mid \mathbf{X}) \\ &\leq \sum_i (H(Y_i) - H(Y_i \mid X_i)) \\ &= \sum_i I(X_i; Y_i) \\ &\leq nC \quad \text{by definition of information capacity.} \end{aligned}$$

For equality, need $Y_1, \dots, Y_n$ to be independent. This can be achieved by taking $X_1, \dots, X_n$ independent and choosing the distribution such that $I(X_i; Y_i) = c$. □

Proposition 2.15. For a DMC, operational capacity $\leq$ information capacity.

Proof. Let C be the information capacity. Suppose we can transmit reliably at rate $R > C$, i.e. there is a sequence of codes $(C_n)_{n \geq 1}$ with C_n of length n and size $\lfloor 2^{nR} \rfloor$ such that $\hat{e}(C_n) \to 0$ as $n \to \infty$.

$$\begin{aligned} \text{Recall } \hat{e}(C_n) &= \max_{c \in C_n} \mathbb{P}(\text{error} \mid c \text{ sent}). \\ \text{Let } e(C_n) &= \frac{1}{|C_n|} \sum_{c \in C_n} \mathbb{P}(\text{error} \mid c \text{ sent}). \end{aligned}$$

Clearly $e(C_n) \leq \hat{e}(C_n)$, so $e(C_n) \to 0$ as $n \to \infty$. Take random variable input X, equidistributed over C_n. Let Y be the random variable output when X is transmitted and decoded. So $e(C_n) = P(X \neq Y) = p$, say.

$$\begin{aligned} H(X) &= \log |C_n| \geq nR - 1 \text{ for large } n. \\ H(X \mid Y) &\leq H(p) + p \log(|C_n| - 1) \text{ by Fano's inequality} \\ &\leq 1 + pnR \\ I(X;Y) &= H(X) - H(X \mid Y) \\ nC &\geq nR - 1 - (1 + pnR) \\ \implies pnR &\geq n(R - C) - 2 \\ \implies p &\geq \frac{n(R-C) - 2}{nR} \nrightarrow 0 \text{ as } n \to \infty. \end{aligned}$$

Thus our sequence of codes cannot exist. □

Proposition 2.16. Consider a BSC, error probability p. Let $R < 1 - H(p)$. Then $\exists$ a sequence of codes $(C_n)_{n \geq 1}$ of length n and size $\lfloor 2^{nR} \rfloor$ such that $e(C_n) \to 0$ as $n \to \infty$.

Proof. Idea: Construct codes by picking codewords at random. Without loss of generality $p < \frac{1}{2}$, so $\exists \epsilon > 0$ such that $R < 1 - H(p + \epsilon)$. We use minimum distance decoding (in case of tie, make arbitrary choice). Let $m = \lfloor 2^{nR} \rfloor$. We pick a $[n, m]$-code C at random (i.e. each word with probability). Say $C = \{c_1, \dots, c_m\}$.

Choose $1 \leq i \leq m$ at random (i.e. each with probability $\frac{1}{m}$). We send c_i through the channel and get output Y. Then $\mathbb{P}(Y \text{ is not decoded as } c_i)$ is the average value of $e(C)$ as C runs over all $[n, m]$-codes. We can pick C_n a $[n, m]$-code with $e(C_n)$ at most this average. So it will suffice to show $\mathbb{P}(Y \text{ is not decoded as } c_i) \to 0$ as $n \to \infty$.

Let $r = \lfloor n(p+\epsilon) \rfloor$.

$$\mathbb{P}(Y \text{ is not decoded as } c_i) \le \mathbb{P}(c_i \notin B(y,r)) + \mathbb{P}(B(Y,r) \cap C \supsetneq \{c_i\}).$$

We now split into two cases: (i) $d(c_i, Y) > r$ or (ii) $d(c_i, Y) \le r$.

(i)

$$\begin{aligned} \mathbb{P}(d(c_i,Y) > r) &= \mathbb{P}(\text{BSC makes} > r \text{ errors}) \\ &= \mathbb{P}(\text{BSC makes} > n(p+\epsilon) \text{ errors}) \\ &\to 0 \text{ as } n \to \infty \text{ by Lemma 2.8.} \end{aligned}$$

(ii) If $j \ne i$,

$$\begin{aligned} \mathbb{P}(c_j \in B(Y,r) \mid c_i \in B(Y,r)) &= \frac{V(n,r)-1}{2^n-1} \le \frac{V(n,r)}{2^n} \\ \text{So } \mathbb{P}(B(Y,r) \cap C \supsetneq \{c_i\}) &\le \frac{(m-1)V(n,r)}{2^n} \\ &\le 2^{nR} 2^{nH(p+\epsilon)} 2^{-n} \text{ by Proposition 2.7(i)} \\ &= 2^{n(R-(1-H(p+\epsilon)))} \\ &\to 0 \text{ as } n \to \infty \text{ since } R < 1 - H(p+\epsilon). \end{aligned}$$

□

Proposition 2.17. Consider a BSC with error probability p. Let $R < 1 - H(p)$. Then $\exists$ a sequence of $[n,m]$-codes $(C_n)_{n \ge 1}$ with C_n of length n, size $\lfloor 2^{nR} \rfloor$ and $\hat{e}(C_n) \to 0$ as $n \to \infty$.

Proof. Pick R' such that $R < R' < 1 - H(p)$. By Proposition 2.16 we construct a sequence of codes $(C'_n)_{n \ge 1}$ with C'_n of length n, size $\lfloor 2^{nR'} \rfloor$ and $e(C'_n) \to 0$ as $n \to \infty$. Throwing out the worst half of the codewords in C'_n gives a code C_n with $\hat{e}(C_n) \le 2e(C'_n)$.

So $\hat{e}(C_n) \to 0$ as $n \to \infty$. Note C_n has length n and size $\lfloor 2^{nR'}/2 \rfloor$.

$$\begin{aligned} \lfloor 2^{nR'} \rfloor / 2 &= \lfloor 2^{nR'-1} \rfloor \\ &= 2^{n(R' - \frac{1}{n})} \\ &\ge 2^{nR} \text{ for large } n. \end{aligned}$$

We can replace C_n by a code of smaller size $\lfloor 2^{nR} \rfloor$ and still get $\hat{e}(C_n) \to 0$ as $n \to \infty$. □

Conclusion A BSC with error probability p has operational capacity $1 - H(p)$.

Remark.

(i) How does it work? Say capacity is 0.8, we have a message (a string of 0's and 1's). Let $R = 0.75 < 0.8$. Then $\exists$ a set of $2^{0.75n}$ codewords of length n that have error probability below some prescribed threshold. Hence, to encode message,

a) break message into blocks of size $3\lceil \frac{n}{4} \rceil = m$ sufficiently large

b) encode these m-blocks into C_n by using codewords of length $\frac{4}{3}m$ for each m-block

c) transmit new message through channel.

(ii) The theorem shows good codes exist. But the proof does not construct them for us.

2.5 Linear Codes

In practise we consider codes with extra structure to allow efficient decoding.

Definition (Linear code)**.** A code $C \subseteq \mathbb{F}_2^n$ is **linear** if

(i) $0 \in C$

(ii) $x, y \in C \implies x + y \in C$.

Recall $\mathbb{F}_2 = \{0, 1\}$ is the field of two elements. So C is linear if it is a $\mathbb{F}_2$-vector space.

Definition (Rank)**.** The **rank** of C is its dimension as a $\mathbb{F}_2$ vector space.

Notation. A linear code C of length n and rank k is called a (n, k)-code.

Say C has a basis $v_1, \dots, v_k$. Then $C = \{ \sum \lambda_i v_i \mid \lambda_i \in \mathbb{F}_2 \}$ so $|C| = 2^k$, so a (n, k)-code is a $[n, 2^k]$-code. Thus, the information rate is $\frac{k}{n}$.

Now for $x, y \in \mathbb{F}_2^n$, define $x \cdot y = \sum_{i=1}^n x_i y_i \in \mathbb{F}_2$. Note $\cdot$ is symmetric and bilinear, but $x \cdot x = 0 \not\Rightarrow x = 0$.

Lemma 2.18. Let $P \subset \mathbb{F}_2^n$ be a subset. Then $C = \{ x \in \mathbb{F}_2^n \mid p \cdot x = 0 \quad \forall p \in P \}$ is a linear code.

Proof.

(i) $0 \in C$ since $p \cdot 0 = 0 \quad \forall p \in P$.

(ii) If $x, y \in C$, then $p \cdot (x + y) = p \cdot x + p \cdot y = 0 \implies x + y \in C$. □

Definition (Parity check code)**.** P is called a set of **parity checks** and C is a **parity check code**.

Definition (Dual code)**.** Let $C \subset \mathbb{F}_2^n$ be a linear code. The **dual code** is

$$C^\perp = \{ x \in \mathbb{F}_2^m \mid x \cdot y = 0 \quad \forall y \in C \}.$$

This is a code by Lemma 2.18.

Lemma 2.19. $\dim C + \dim C^\perp = n$.

Warning: we can have $C \cap C^\perp \neq \{0\}$.

Proof. $V = \mathbb{F}_2^n$, $V^* = \{\text{linear maps: } V \to \mathbb{F}_2\}$. Consider

$$\begin{aligned} \phi : V &\longrightarrow V^* \\ x &\longmapsto \theta_x \quad \text{where } \theta_x : y \longmapsto x \cdot y. \end{aligned}$$

ϕ is a linear map.

Suppose $x \in \ker \phi$, then $x \cdot y = 0\ \forall y \in V$. Taking $y = e_i = (0, \dots, 0, 1, 0, \dots, 0)$ (with 1 in the ith place) gives $x_i = 0$. So $\ker \phi = \{0\}$. Since $\dim V = \dim V^*$, it follows that ϕ is an isomorphism. Thus

$$\begin{aligned} \theta(C^\perp) &= \{ \theta \in V^* \mid \theta(x) = 0\ \forall x \in C \} \\ &= \text{'annihilator of } C\text{'} = C^\circ \end{aligned}$$

so $\dim C + \dim \phi(C^\perp) = \dim V$, and $\dim C + \dim C^\perp = n$. □

Corollary. $(C^\perp)^\perp = C$ for any linear code C. In particular, any linear code is a parity check code.

Proof. Let $x \in C$. Then $x \cdot y = 0 \ \forall y \in C^\perp$, so $x \in (C^\perp)^\perp$, i.e. $C \subseteq (C^\perp)^\perp$. By Lemma 2.19 (twice), $\dim(C^\perp)^\perp = \dim C$, so $C = (C^\perp)^\perp$. □

Definition. Let C be a (n, k)-code.

(i) A **generator matrix** for C is a $k \times n$ matrix whose rows are a basis for C.

(ii) A **parity check matrix** for C is a generator matrix for $C^\perp$. It is a $(n-k) \times n$ matrix.

Lemma 2.20. Every (n, k) linear code is equivalent to a linear code with generator matrix $\left(\ I_k \ \middle| \ B \ \right)$.

Proof. We can perform row operations:

- swap 2 rows
- add one row to another

(multiplying by a scalar is not useful in $\mathbb{F}_2$). By Gaussian elimination, we get G, the generator matrix in row echelon form, e.g.

$$G = \begin{pmatrix} 1 & * & * & * & \dots \\ 0 & 1 & * & * & \dots \\ 0 & 0 & 0 & 1 & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

i.e. $\exists l(1) < l(2) < \cdots < l(n)$ such that

$$G_{ij} = \begin{cases} 0 & \text{if } j < l(i) \\ 1 & \text{if } j = l(i) \end{cases}$$

Permuting the columns of G gives an equivalent code, i.e. $l(i) = i$ for $1 \le i \le k$, i.e.

$$G = \left(\begin{array}{cccc|cc} 1 & * & \dots & * & * & * \\ 0 & 1 & \dots & * & * & * \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & * & * \end{array} \right)$$

More row operations put G in the form $\left(\ I_k \ \middle| \ B \ \right)$ with B a $k \times (n-k)$ matrix. □

Remark. A message $y \in \mathbb{F}_2^k$ (a row vector) is sent as yG. If $G = \left(\ I_k \ \middle| \ B \ \right)$ then

$$yG = (\ y \mid yB\ )$$

where y forms the message and yB are the check digits.

Lemma 2.21. A (n, k) linear code with generator matrix $G = (\ I_k \mid B\ )$ has parity check matrix $H = (\ -B^T \mid I_{n-k}\ )$.

Proof.

$$GH^T = (\ I \mid B)\begin{pmatrix} -B \\ \hline I_{n-k} \end{pmatrix} = -B + B = 0$$

So the rows of H generate a subcode $C^\perp$. But

$$\begin{aligned} \dim(C^\perp) &= n-k \quad \text{by Lemma 2.19} \\ &= \operatorname{rank} H \quad \text{since } H \text{ has } I_{n-k} \text{ as a submatrix.} \end{aligned}$$

Therefore the rows of H are a basis for $C^\perp$, as required. □

Definition. The **Hamming weight** of $x \in C$ is $w(x) \coloneqq d(x, 0)$.

Lemma 2.22. The minimum distance of C, a linear code, is the minimum weight of a non-zero codeword.

Proof. Let $x, y \in C$. Then $x + y \in C$ and $d(x, y) = d(x - y, 0) = d(x + y, 0) = w(x + y)$. Note x, y distinct $\iff$ $x + y \neq 0$. So

$$d(C) = \min_{\substack{x,y \in C \\ \text{distinct}}} d(x, y) = \min_{\substack{x \in C \\ x \neq 0}} w(x)$$ □

Definition. The **weight** $w(C)$ of a linear code C is the minimum weight of a non-zero codeword.

By Lemma 2.22 this is the same as minimum distance.

Syndrome decoding

Let C be a $(n, \hat{k})$-linear code with parity check matrix H. Then $C = \{\, x \in \mathbb{F}_2^n : Hx = 0 \,\}$ where x is a column vector.

Suppose we receive $y = c + e$ where $c \in C$ is a codeword and $e \in \mathbb{F}_2^n$ is an error. We compute the **syndrome** Hy. Suppose we know C is K-error correcting. Then we tabulate the syndromes He for all $e \in \mathbb{F}_2^n$ with $w(e) \leq K$. If we receive y we search for Hy in our list. If successful we get $Hy = He$ for some $e \in \mathbb{F}_2^n$ with $w(e) \leq K$. We decode y as $c = y - e$. Then $c \in C$ as $He = Hy - He = 0$, and $d(y, c) = w(e) \leq k$.

Recall Hamming's original code:

$$\begin{aligned} c_1 + c_3 + c_5 + c_7 &= 0 \\ c_2 + c_3 + c_6 + c_7 &= 0 \\ c_4 + c_5 + c_6 + c_7 &= 0 \end{aligned}$$

So

$$C^\perp = \langle (1\ \ 0\ \ 1\ \ 0\ \ 1\ \ 0\ \ 1), (0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1), (0\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 1) \rangle$$

So

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} \quad \text{and} \quad Hy = z = \begin{pmatrix} z_1 & z_2 & z_4 \end{pmatrix}$$

In general, we have

Definition (Hamming codes)**.** Let $d \geq 1$, $n = 2^d - 1$. Let H be the $d \times n$ matrix whose columns are the non-zero elements of $\mathbb{F}_2^d$. The Hamming $(n, n-d)$ linear code is the linear code with parity check matrix H (original is $d = 3$).

Lemma 2.23. Let C be a linear code with parity check matrix H. Then $w(C) = d$ iff

(i) any $(d-1)$-columns of H are linearly independent.

(ii) some d columns of H are linearly dependent.

Proof. Suppose C has length n. Then $C = \{ x \in \mathbb{F}_2^n \mid Hx = 0 \}$. If H has columns $v_1, \dots, v_n$ then

$$(x_1, \dots, x_n) \in C \iff \sum_{i=1}^n x_i v_i = 0$$

i.e. codewords are dependence relations between columns. □

Lemma 2.24. The Hamming $(n, n-d)$ linear code has minimum distance $d(C) = 3$ and is a perfect 1-error correcting code.

Proof. Any two columns of H are linearly independent (where H is the parity check matrix of C), but $\exists$ 3 that area linearly dependent. Hence $d(C) = 3$ by Lemma 2.23. By Lemma 2.4, C is $\lfloor \frac{3-1}{2} \rfloor = 1$ error correcting.

To be perfect,

$$|C| = \frac{2^n}{V(n,e)}.$$

Here, $n = 2^d - 1$, $e = 1$ so

$$\frac{2^n}{V(n,e)} = \frac{2^n}{1 + 2^d - 1} = 2^{n-d} = |C|.$$ □

2.6 New codes from old (again)

The following construction is specific to linear codes.

Definition (Bar product)**.** Let C_1, C_2 linear codes of length n with $C_2 \subseteq C_1$, i.e. C_2 is a subcode of C_1. The **bar product** is

$$C_1 \mid C_2 = \{ (x \mid x + y) \mid x \in C_1, y \in C_2 \}$$

a linear code of length $2n$.

Lemma 2.25. Take C_1, C_2 as above.

(i) $\operatorname{rank}(C_1 \mid C_2) = \operatorname{rank}(C_1) + \operatorname{rank}(C_2)$

(ii) $w(C_1 \mid C_2) = \min\{2w(C_1), w(C_2)\}$

Proof.

(i) Let $x_1, \dots, x_k$ a basis for C_1. Let $y_1, \dots, y_l$ be a basis for C_2. Then

$$\{ (x_i \mid x_i) \mid 1 \leq i \leq k \} \cup \{ (0 \mid y_j) \mid 1 \leq j \leq l \}$$

is a basis for $C_1 \mid C_2$. Hence $\operatorname{rank}(C_1 \mid C_2) = \operatorname{rank}(C_1) + \operatorname{rank}(C_2)$.

(ii) Let $x \in C_1$, $y \in C_2$ not both zero. If $y \neq 0$,

$$\begin{aligned} w(x \mid x+y) &= w(x) + w(x+y) \\ &\geq w(y) \\ &\geq w(C_2). \end{aligned}$$

If $y = 0$ ($x \neq 0$), $w(x \mid x) = 2w(x) \geq 2w(C_1)$. So $w(C_1 \mid C_2) \geq \min\{2w(C_1), w(C_2)\}$.
But $\exists 0 \neq x \in C_1$ such that $w(x) = w(C_1)$ so $w(x \mid x) = 2w(x) = 2w(C_1)$.
Also $\exists 0 \neq y \in C_2$ such that $w(y) = w(C_2)$ so $w(0 \mid y) = w(y) = w(C_2)$.
Therefore $w(C_1 \mid C_2) = \min\{2w(C_1), w(C_2)\}$. □

2.7 Reed-Muller Codes

Let $X = \mathbb{F}_2^d = \{p_1, \ldots, p_n\}$ where $n = 2^d$ (we have chosen an ordering). For $A \subseteq X$, we get a vector $\mathbb{1}_A \in \mathbb{F}_2^n$ by the rule

$$(\mathbb{1}_A)_i = 1 \iff p_i \in A,$$

i.e. $\mathbb{1}_A$ is the indicator function of A. For $x, y \in \mathbb{F}_2^n$ we have

$$\begin{aligned} x + y &= (x_1 + y_1, \ldots, x_n + y_n) \\ x \wedge y &= (x_1 y_1, \ldots, x_n y_n) \end{aligned}$$

Then $(\mathbb{F}_2^n, +, \wedge)$ is a ring. Note for $A, B \subseteq X$, we have

$$\begin{aligned} \mathbb{1}_A + \mathbb{1}_B &= \mathbb{1}_{A \triangle B} \quad \text{(symmetric difference)} \\ \mathbb{1}_A \wedge \mathbb{1}_B &= \mathbb{1}_{A \cap B} \\ w(\mathbb{1}_A) &= |A|. \end{aligned}$$

Define $v_0 = \mathbb{1}_X = (1, \ldots, 1)$ (multiplicative identity). For $1 \leq i \leq d$, let

$$v_i = \mathbb{1}_{H_i} \text{ where } H_i = \{\, p \in X \mid p_i = 0 \,\}$$

Definition (Reed-Muller code)**.** Let $0 \leq r \leq d$. The **Reed-Muller code** $RM(d, r)$ of order r and length 2^d is the vector subspace of $\mathbb{F}_2^n$ spanned by v_0 and wedge products of at most r of the v_i.

By convention, the empty wedge product is v_0.

Example. Take $d = 3$.

X	000	001	010	011	100	101	110	111
v_0	1	1	1	1	1	1	1	1
v_1	1	1	1	1	0	0	0	0
v_2	1	1	0	0	1	1	0	0
v_3	1	0	1	0	1	0	1	0
$v_1 \wedge v_2$	1	1	0	0	0	0	0	0
$v_2 \wedge v_3$	1	0	0	0	1	0	0	0
$v_1 \wedge v_3$	1	0	1	0	0	0	0	0
$v_1 \wedge v_2 \wedge v_3$	1	0	0	0	0	0	0	0

- $RM(3,0)$ is spanned by v_0. It is the repetition code of length 8.
- $RM(3,1)$ is spanned by v_0, v_1, v_2, v_3. Deleting the 1st component gives Hamming's $(7,4)$-code, i.e. the highlighted rectangle in the diagram is the generator matrix for the Hamming $(7,4)$-code. Note 1110000 corresponds to

$$\begin{pmatrix}0\\0\\1\end{pmatrix} + \begin{pmatrix}0\\1\\0\end{pmatrix} + \begin{pmatrix}0\\1\\1\end{pmatrix}$$

  Notice also v_0, v_1, v_2, v_3 all have even weight. So $RM(3,1)$ is (equivalent to) the parity check extension of the Hamming $(7,4)$-code.
- $RM(3,2)$ is spanned by $v_0, v_1, v_2, v_3, v_1 \wedge v_2, v_1 \wedge v_3, v_2 \wedge v_3$. These are linearly independent (see next theorem), so $RM(3,2)$ is a $(8,7)$-code. Each codeword has even weight so $RM(3,2)$ is the simple parity check code of length 8.
- $RM(3,3) = \mathbb{F}_2^8$, the trivial code.

Theorem 2.26.

(i) The vectors $v_{i_1} \wedge \cdots \wedge v_{i_s}$ for $1 \leq i_1 < \cdots < i_s \leq d$ and $0 \leq s \leq d$ are a basis for $\mathbb{F}_2^n$.

(ii) $\mathrm{RM}(d,r)$ has rank $\sum_{s=0}^{r} \binom{d}{s}$

(iii) $\mathrm{RM}(d,r) = \mathrm{RM}(d-1,r) / \mathrm{RM}(d-1,r-1)$

(iv) $\mathrm{RM}(d,r)$ has weight 2^{d-r}.

Proof.

(i) We have a set of $\sum_{s=0}^{d} \binom{d}{s} = (1+1)^d = 2^d = n$ vectors. So it suffices to show they span $\mathbb{F}_2^n$, equivalently $RM(d,d) = \mathbb{F}_2^n$. Let $p \in X$, and let

$$y_i = \begin{cases} v_i & \text{if } p_i = 0 \\ v_0 + v_i & \text{if } p_i = 1 \end{cases}$$

giving $\mathbb{1}_{\{p\}} = y_1 \wedge y_2 \wedge \cdots \wedge y_d$.

Expanding using the distributive law gives that $\mathbb{1}_{\{p\}} \in RM(d,d)$. But the $\mathbb{1}_{\{p\}}$ for $p \in X$ span $\mathbb{F}_2^n$, so $RM(d,d) = \mathbb{F}_2^n$.

(ii) By definition, $RM(d,r)$ is spanned by the vectors $v_{i_1} \wedge \cdots \wedge v_{i_s}$ with $1 \leq i_1 < \cdots < i_s \leq d$ and $0 \leq s \leq r$. By (i), these vectors are linearly independent so form a basis of $RM(d,r)$. There $\sum_{s=0}^{r} \binom{d}{s}$ such vectors, giving the required rank.

(iii) **missing proof**

(iv) $RM(d,0)$ is the repetition code of length 2^d, which has weight 2^d. $RM(d,d) = \mathbb{F}_2^n$ by (i), so has weight $1 = 2^{d-d}$. By induction, $RM(d-1,r)$ has weight 2^{d-1-r} and $RM(d-1,r-1)$ has weight 2^{d-r}. Using Lemma 2.25 and (iii), $RM(d,r)$ has weight $\min(2 \times 2^{d-1-r}, 2^{d-r}) = 2^{d-r}$.

□

Remark.

(1) A different ordering gives an equivalent code.

(2) We could alternatively define the Reed-Muller code recursively using the bar product, starting with $RM(d,d) = \mathbb{F}_2^{2^d}$ and $RM(d,0) = \{1\cdots 1, 0\cdots 0\}$.

Algebraic aside

A **ring** R is a set with two operations, $+$ and $\times$. $(R,+)$ is an additive group and $\times$ is distributive over addition i.e. $a(b+c) = ab + ac$. Think of $\mathbb{Z}$ or $\mathbb{Z}_{10}$.

An **ideal** $I \trianglelefteq R$ is an additive subgroup, closed under external mutliplication, i.e. if $a \in I$ and $r \in R$ then $ra \in I$. Think of $2\mathbb{Z} \trianglelefteq Z$.

Lemma 2.27. Let I be an ideal in ring R and let $q : R \to R/I$ be the quotient map. Then there is a bijection between the set of ideals $J \subseteq R$ containing I and the set of ideals in R/I. The bijection is given by $J \mapsto q(J) = J/I$ and the inverse $K \mapsto q^{-1}(K) = \{\, r \in R \mid q(r) \in K \,\}$.

An ideal is **principal** if it is generated by one element, e.g. $6\mathbb{Z} = \langle 6 \rangle$. Note if $a, b \in R$ then

$$\langle b \rangle \subseteq \langle a \rangle \iff b \in \langle a \rangle \iff a \mid b.$$

If all ideals are principal the ring is called a **Principal Ideal Domain** or **PID**. For example, $\mathbb{Z}$ is a PID.

A **field** is a commutative ring (i.e. multiplication is commutative) with a multiplicative identity such that every nonzero element has a multiplicative inverse, e.g. $\mathbb{Q}$, $\mathbb{R}$, $\mathbb{F}_2$ and $\mathbb{Z}_5 = \mathbb{F}_5$, but not $\mathbb{Z}$ or $\mathbb{Z}_{10}$. Note if $\mathbb{F}$ is a field then the polynomial ring $\mathbb{F}[X]$ (the set of polynomials in X with coefficients in $\mathbb{F}$) is a PID.

Example.

(i) $R = \mathbb{Z}$ and $I = 6\mathbb{Z}$. Then

$$6\mathbb{Z} \subseteq J \subseteq \mathbb{Z} \iff J = \mathbb{Z}, 2\mathbb{Z}, 3\mathbb{Z}, 6\mathbb{Z}.$$

(ii) $R = \mathbb{F}_2[X]$ and $I = \langle X^3 + 1 \rangle$. Then

$$\langle X^3 + 1 \rangle \subseteq J \subseteq F_2[X] \iff J = \langle 1 \rangle, \langle X + 1 \rangle, \langle X^2 + X + 1 \rangle, \langle X^3 + 1 \rangle.$$

Theorem.

(i) Let K be a finite field. Then $K \supseteq \mathbb{F}_p$ for some prime p and $|K| = p^r$ for some $r \geq 1$.

(ii) Let $q = p^r$, a prime power. Then, up to isomorphism, there exists a unique field $\mathbb{F}_q$ with q elements.

Warning. $\mathbb{F}_p \cong \mathbb{Z}_p$ but if $q = p^r$ and $r \geq 2$ then $\mathbb{F}_q \not\cong \mathbb{Z}_q$.

Proposition. (i) Let q be a prime power. Then there exists $\alpha \in \mathbb{F}_q$ (not unique) such that $\mathbb{F}_q^* = \mathbb{F}_q \setminus \{0\} = \{1, \alpha, \ldots, \alpha^{q-2}\}$ i.e. $\mathbb{F}_q^* \cong C_{q-1}$ under multiplication. α is called a **primitive element**.

(ii) $\mathbb{F}_{p^r}$ is a subfield of $\mathbb{F}_{p^s}$ if and only if $r \mid s$.

(iii) Let $f(x) \in \mathbb{F}_q[X]$. Then there exists $r \geq 1$ such that $f(X)$ factors completely into linear factors in $\mathbb{F}_{q^r}[X]$.

Lemma 2.28. Let $\mathbb{F}$ be a field and

$$f(X) = \sum_{i=0}^{\infty} a_i X^i \in \mathbb{F}[X].$$

Define

$$f'(X) = \sum_{i=1}^{\infty} i a_i X^{i-1} \in \mathbb{F}[X].$$

Let $a \in \mathbb{F}$. If $(X-a)^2$ divides $f(X)$ then $f(a) = f'(a) = 0$.

Proof. We have $f(X) = (X-a)^2 g(X)$ and $f'(X) = 2(X-a)g(X) + (X-a)^2 g'(X)$. Then $f(a) = f'(a) = 0$. □

In particular we consider $X^N - 1 \in \mathbb{F}_2[X]$ for N odd. Then there exists a field $K \supseteq \mathbb{F}_2$ such that $X^N - 1$ factorises into linear factors in K. Furthermore, $X^N - 1$ has N distinct roots by Lemma 2.28. The set R of roots in K form a group under multiplication and so is cyclic as it is a subgroup of K^*. An element $\alpha \in R$ is a primitive N-th root of unity if $\alpha^N = 1$ and $\alpha^i \neq 1$ for $1 \leq i < N$, the other roots are $\alpha^2, \alpha^3, \dots, \alpha^N$. (Note if N even we can have repeated roots, e.g. $X^2 - 1 = (X-1)^2 \in \mathbb{F}_2[X]$.)

2.8 Cyclic Codes

Definition. $C \subset \mathbb{F}_2^n$ is a **cyclic code** if C is linear and

$$(a_0, a_1, \dots, a_{n-1}) \in C \implies (a_{n-1}, a_0, \dots, a_{n-2}) \in C.$$

We identify $\mathbb{F}_2^n$ with $\mathbb{F}_2[X]/\langle X^n - 1\rangle$ via

$$\begin{aligned} \mathbb{F}_2^n &\overset{\pi}{\longmapsto} \mathbb{F}_2[X]/\langle X^n - 1\rangle \\ (a_0, \dots, a_{n-1}) &\longrightarrow a_0 + a_1 X + \dots + a_{n-1}X^{n-1} + \langle X^n - 1\rangle \end{aligned}$$

Lemma 2.29. A code $C \subseteq \mathbb{F}_2^n$ is cyclic iff $\mathcal{C} = \pi(C)$ satisfies

(i) $0 \in \mathcal{C}$

(ii) $f, g \in \mathcal{C} \implies f + g \in \mathcal{C}$

(iii) $f \in \mathcal{C}, g \in \mathbb{F}_2[X] \implies gf \in \mathcal{C}$.

Proof. (i) and (ii) follow by linearity. For (iii), observe that Xf 'cycles' f, i.e.

$$\begin{aligned} f &= a_0 + a_1 X + \dots + a_{n-1}X^{n-1} + \langle X^n - 1\rangle \\ Xf &= a_{n-1} + a_0 X + \dots + a_{n-2}X^{n-1} + \langle X^n - 1\rangle \end{aligned}$$

So C cyclic $\iff (f \in \mathcal{C} \implies Xf \in \mathcal{C})$, and the general statement follows by linearity. □

Remark. So C cyclic of length n if and only iff $\mathcal{C}$ is an ideal in the PID $\mathbb{F}_2[X]/\langle X^n - 1\rangle$.

From now on C identifies with $\mathcal{C}$.

Definition. A **generator polynomial** $g(X)$ for a cyclic code C is a polynomial g such that $g \mid X^n - 1$ and

$$\mathcal{C} = \langle g\rangle = \{\, fg \mid f \in \mathbb{F}_2[X] \,\}\,.$$

Theorem 2.30. Every cyclic code has a generator polynomial.

Proof. $\mathcal{C}$ is an ideal in $\mathbb{F}_2[X]/\langle X^n - 1\rangle$. By Lemma 2.27, $\mathcal{C} = J/\langle X^n - 1\rangle$, for an ideal J with $\langle X^n - 1\rangle \subseteq J \subseteq \mathbb{F}_2[X]$. $F_2[X]$ is a PID, so $J = \langle g\rangle$ for some g. $X^n - 1 \in \langle g\rangle \implies g \mid X^n - 1$. □

Note generator polynomials are unique if we insist on g monic. But this is true over $\mathbb{F}_2$ for any g.

Corollary. There is a bijection

$$\left\{\begin{array}{c}\text{cyclic codes}\\ \text{of length } n\end{array}\right\} \longleftrightarrow \left\{\begin{array}{c}\text{factors of } X^n - 1\\ \text{in } \mathbb{F}_2[X]\end{array}\right\}$$

Further, if cyclic codes C_1, C_2 have generator polynomials g_1, g_2 respectively then

$$C_1 \supset C_2 \implies g_1 \mid g_2.$$

Also if n odd, $f(X) = X^n - 1$ has no repeated roots. So

$$X^n - 1 = f_1(X) \cdots f_r(X)$$

where $f_1, \dots, f_r$ are distinct and irreducible in $\mathbb{F}_2[X]$. So, there are 2^r cyclic codes of length n.

Lemma 2.31. Say C is cyclic with generator g:

$$g(X) = a_0 + a_1 X + \cdots + a_k X^k$$

with $a_k = 1$. Then $\{g, Xg, \dots, X^{n-k-1}g\}$ is a basis for C.

Proof. **LI**: Say $fg = 0 \in \mathbb{F}_2[X]/\langle X^n - 1\rangle$, for some $f \in \mathbb{F}_2[X]$ with $\deg f \le n - k - 1$. So

$$\begin{aligned}\deg fg \le n - 1 &\implies fg = 0\\ &\implies f = 0\end{aligned}$$

Spanning: Say $p \in \mathbb{F}_2[X]$ representing an element of C. WLOG say $\deg p \le n - 1$. Then $p = fg$ for some $f \in \mathbb{F}_2[X]$ with $\deg f = \deg p - \deg g \le n - k - 1$. Hence

$$p \in \operatorname{span}\{g, Xg, \dots, X^{n-k-1}g\}$$ □

.

Corollary. C has rank $n-k$, and has generator matrix

$$G = \begin{pmatrix} a_0 & a_1 & a_2 & \dots & a_k & 0 & 0 & \dots & 0 \\ 0 & a_0 & a_1 & \dots & a_{k-1} & a_k & 0 & \dots & 0 \\ 0 & 0 & a_0 & \dots & a_{k-2} & a_{k-1} & a_k & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & a_0 & a_1 & a_2 & \dots & a_k \end{pmatrix}$$

with dimensions $(n-k) \times n$.

Recall $g(X) = a_0 + a_1 X + \cdots a_k X^k$ and $gh = X^n - 1$ for some h, say

$$h = b_0 + b_1 X + \cdots + b_{n-k} X^{n-k}$$

for $b_{n-k} \neq 0$.

Let

$$H = \begin{pmatrix} b_{n-k} & b_{n-k-1} & \dots & b_0 & 0 & \dots & 0 \\ 0 & b_{n-k} & \dots & b_1 & b_0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & b_{n-k} & b_{n-k} & \dots & b_0 \end{pmatrix}$$

a $k \times n$ matrix.

Then we claim H is a parity check matrix for C. Note the ith row of $G \cdot j$th row of $H =$ coefficient of $X^{n-k+(j-i)}$ in $X^n - 1$. As $b_{n-k} \neq 0$, $\operatorname{rank}(H) = k = \operatorname{rank}(C^\perp)$.

Lemma 2.32. The parity check polynomial is the generator polynomial for the 'reverse' of $C^\perp$, i.e. reverse all codewords.

2.9 BCH (Bose-Chaudhuri-Hocquenghem) Codes

Definition. Let K be a field, $K \supset \mathbb{F}_2$, $n \in \mathbb{N}$ and $A \subset \{\, x \in K \mid x^n = 1 \,\}$. The cyclic code of length n defined by A is

$$C = \{\, f(X) \pmod{X^n - 1} \mid f(x) = 0 \quad \forall x \in A \,\}$$

Observe if $f, g \in C$ then $f + g \in C$ and $Xf \in C$. So C is a cyclic code.

Definition. Say $K \supset \mathbb{F}_2$, n odd, $\alpha \in K$ a primitive nth root of unity (i.e. the roots of $X^n - 1$ are $\{1, \alpha, \dots, \alpha^{n-1}\}$). Let $A = \{\alpha, \dots, \alpha^{\delta - 1}\}$. Then A defines the **BCH code** of design distance δ.

Remark. (i) The minimal polynomial for α over $\mathbb{F}_2$ is the polynomial of least degree satisfied by α.

(ii) The generator polynomial $g(X)$ for a BCH code C is

$$\operatorname{lcm}\{m_1(X), \dots, m_{\delta-1}(X)\}$$

for m_i the minimal polynomial of α^i over $\mathbb{F}_2$.

Theorem 2.33. The minimum distance of a BCH code with design distance δ is at least δ.

Lemma.

$$\det \begin{pmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & \dots & x_n \\ x_1^2 & x_2^2 & \dots & x_n^2 \\ \vdots & \vdots & \ddots & \vdots \\ x_1^{n-1} & x_2^{n-1} & \dots & x_n^{n-1} \end{pmatrix} = \prod_{1 \le i < j \le n} (x_j - x_i)$$

Proof. Work in $\mathbb{Z}[x_1, \dots, x_n]$. Note if $x_i = x_j$ then the determinant is 0. So, $(x_i - x_j) \mid \text{LHS}$ for each $i \neq j \implies \text{RHS} \mid \text{LHS}$.

But both polynomials have degree $\binom{n}{2}$, and the coefficient of $x_2 x_3^2 \cdots x_n^{n-1}$ is 1 on both sides. So LHS = RHS. □

Proof of Theorem 2.33. Consider the $(\delta - 1) \times n$ matrix H:

$$H = \begin{pmatrix} 1 & \alpha & \alpha^2 & \dots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \dots & \alpha^{2(n-1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{\delta-1} & \alpha^{2(\delta-1)} & \dots & \alpha^{(\delta-1)(n-1)} \end{pmatrix}$$

Using the previous lemma, pulling out factors as needed, we get that any $\delta - 1$ columns of H are linearly independent. But a codeword in C is a dependence relation on the columns of H. Hence $w(C) \geq \delta$. □

Remark. H is not the parity check matrix in the usual sense (entries are in K, not in $\mathbb{F}_2$).

Example.

(1) $n = 7$, $X^7 - 1 = (1+X)(1+X+X^3)(1+x^2+X^3)$ is the factorisation into irreducibles in $\mathbb{F}_2[X]$. Suppose $g(X) = 1 + X + X^3$, then $h(X) = 1 + X + X^2 + X^4$.

The parity check matrix is

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

whose columns are exactly $\mathbb{F}_2^3 \setminus \{0\}$. So the code generated by g is Hamming's (7, 4) code.

(2) Say $K \supset \mathbb{F}_2$, the splitting field of $X^7 - 1$. Let $\alpha \in K$ a root of $g = X^3 + X + 1$, then α is a primitive root of unity.

$$\begin{aligned} g(\alpha) = 0 &\implies \alpha^6 = (\alpha + 1)^2 \\ &\implies \alpha^6 = \alpha^2 + 1 \\ &\implies g(\alpha^2) = 0. \end{aligned}$$

So the roots of g are $\alpha, \alpha^2, \alpha^4$. The BCH code of length 7 and design distance 3 (i.e. determining set is $\{\alpha, \alpha^2\}$) has generator polynomial g. By (1) this is Hamming's original code. By Theorem 2.33, the weight is ≥ 3.

Decoding BCH codes

Note by Theorem 2.33, if C is a length n code of design distance δ, then C is at least $t = \lfloor \frac{\delta-1}{2} \rfloor$ error correcting.

Say $c \in C$ is sent and we receive $r = c + e$, where e is the 'error pattern'. By the identification of $\mathbb{F}_2^n$ with $\mathbb{F}_2[X]/\langle X^n - 1\rangle$, say we have polynomials $r(X)$, $c(X)$, $e(X)$. Let

$$\mathcal{E} = \{\, 0 \le i \le n-1 \mid e_i \neq 0 \,\}$$

Definition. The **error locator polynomial** is

$$\sigma(X) = \prod i \in \mathcal{E}(1 - \alpha^i X).$$

Assuming $\deg(\sigma) = |\mathcal{E}| \le t$, our aim is to recover $\sigma(X)$ from $r(X)$.

Theorem 2.34. $\sigma(X)$ has constant term 1, and satisfies

$$\sigma(X) \sum_{j=0}^{2t} r(\alpha^j) X^j \equiv w(X) \pmod{X^{2t+1}}$$

where $w(X)$ is a polynomial of degree $\le t$. Moreover, $\sigma(X)$ is the unique polynomial of least degree satisfying the above.

Application Taking coefficients of X^i for $t+1 \le i \le 2t$ allows us to solve for the coefficients of $\sigma(X)$. Then

$$\xi = \{\, 0 \le i \le n-1 \mid \sigma(a^{-i}) = 0 \,\}.$$

This determines e and we decode as $r - e$.

Proof of Theorem 2.34. Let

$$w(X) = -X\sigma'(X) = \sum_{i \in \xi} \alpha^i X \prod_{j \neq i} (1 - \alpha^j X).$$

So $w(X)$ is a polynomial of degree $=\deg(\sigma)$. We work in $k[[X]]$ the ring of formal power series $\sum_{i=0}^{\infty} \beta_i X_i$, $\beta_i \in K$. Note

$$\frac{1}{1 - \alpha^i X} = \sum_{j=0}^{\infty} (\alpha^i X)^j \in k[[X]].$$

So

$$\begin{aligned}
\frac{w(X)}{\sigma(X)} &= \sum_{i\in\xi} \frac{\alpha^i X}{1-\alpha^i X} \\
&= \sum_{i\in\xi}\sum_{j=1}^{\infty} (\alpha^i X)^j \\
&= \sum_{j=1} \left(\sum_{i\in\xi} (\alpha^j)^i \right) X^j \\
&= \sum_{j=1}^{\infty} e(\alpha^j) X^j \\
\implies w(X) &= \left(\sum_{j=1}^{\infty} e(\alpha^j) X^j \right) \sigma(X).
\end{aligned}$$

By defnition of C we have $c(\alpha^j) = 0$ for $1 \le i \le \delta - 1$ so for $1 \le j \le 2t$. So $r(\alpha^j) = e(\alpha^j)$ for $1 \le j \le 2t$. Thus

$$\sigma(X) \sum_{j=1}^{2t} r(\alpha^j) X^j \equiv w(X) \pmod{X^{2t+1}}.$$

Now to show uniqueness. Note $\sigma(X)$ has distinct nonzero roots, so $\sigma(X)$ and $w(X) = -X\sigma'(X)$ are coprime. Suppose $\tilde{\sigma}(X)$ and $\tilde{w}(X)$ are another solution. Assume $\deg(\tilde{\sigma}) \le \deg(\sigma)$. Then

$$\sigma(X)\tilde{w}(X) \equiv \tilde{\sigma}(X) w(X) \pmod{X^{2t+1}}.$$

But $\sigma, \tilde{\sigma}, w, \tilde{w}$ all have degree $\le t$. So

$$\sigma(X)\tilde{w}(X) = \tilde{\sigma}(X) w(X).$$

Since $\sigma(X)$ and $w(X)$ are coprime $\implies \sigma(X) \mid \tilde{\sigma})(X)$. But $\deg(\tilde{\sigma}) \le \deg(\sigma)$, so $\tilde{\sigma}$ is a scalar multiple of σ. Both have constant term 1 so $\sigma = \tilde{\sigma}$. □

2.10 Shift registers

Definition (General feedback shift register)**.** A **general feedback shift register** is a fucntion $f : \mathbb{F}_2^d \to \mathbb{F}_2^d$ of the form

$$f(x_0, \dots, x_{d-1}) = (x_1, \dots, x_{d-1}, c(x_0, \dots, x_{d-1}))$$

where $c : \mathbb{F}_2^d \to \mathbb{F}_2$.

Definition (Linear feedback shift register)**.** A **linear feedback shift register** is a function $f : \mathbb{F}_2^d \to \mathbb{F}_2^d$ as above with c linear i.e.

$$c(x_0, \dots, x_{d-1}) = a_0 x_0 + \cdots + a_{d-1} x_{d-1} \quad \text{for some } a_0, \dots, a_{d-1} \in \mathbb{F}^2.$$

The stream associated to the initial fill $y_0, \dots, y_{d-1}$ is the sequence $y_0, y_1, \dots$ where

$$y_n = a_{d-1}y_{n-1} + a_{d-2}y_{n-2} + \dots + a_0 y_{n-d} \quad \forall n \geq d$$

or more generally $y_n = c(y_{n-d}, \dots, y_{n-1})$.

The stream produced by a LFSR is a recurrence relation (= difference relation). The feedback (auxiliary) polynomial is

$$P(X) = X^d + a_{d-1}X^{d-1} + \dots + a_0.$$

Definition. A sequence of elements in $\mathbb{F}_2$ has generating function

$$G(X) = \sum_{j=0}^{\infty} x_j X^j \in \mathbb{F}_2[[X]].$$

Theorem 2.35. The stream comes from a LFSR with feedback polynomial $P(X)$ iff $G(X) = \frac{B(X)}{A(X)}$ where $A(X)$ is the reverse of $P(X)$ and $B(X)$ a polynomial of degree $< \deg(A)$.

Proof. Suppose $P(X) = a_d X^d + a_{d-1}X^{d-1} + \dots + a_0$ with $a_d = 1$. Then $A(X) = a_0 X^d + a_1 X^{d-1} + \dots + a_d$. So,

$$A(X)G(X) = \left(\sum_{i=0}^{d} a_{d-i} X^i \right) \left(\sum_{j=0}^{\infty} x_j X^j \right).$$

So $A(X)G(X)$ is a polynomial of degree $< d$ iff the coefficient of X^r in $A(X)G(X)$ is 0 $\forall r \geq d$

$$\begin{aligned} &\iff \sum_{i=0}^{d} a_{d-1} x_{r-i} = 0 \quad \forall r \geq d \\ &\iff (x_n)_{n \geq 0} \text{ comes from a LFSR with feedback polynomial } P(X). \end{aligned}$$ □

Remark. The problems

(i) Recover the LFSR from its sequence output.

(ii) Decoding BCH codes.

both involve recognising a power series as a quotient of polynomials.

Berlekamp-Massey algorithm

Let $x_0, x_1, \dots$ be the output of a binary LFSR. We can find the unknown d and $a_0, \dots, a_{d-1}$ such that

$$x_n + \sum_{i=1}^{d} a_{d-i} x_{n-i} = 0.$$

In this case

$$\begin{pmatrix} x_d & x_{d-1} & \dots & x_1 & x_0 \\ x_{d+1} & x_d & \dots & x_2 & x_1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{2d} & x_{2d-1} & \dots & x_{d+1} & x_d \end{pmatrix} \begin{pmatrix} 1 \\ a_{d-1} \\ \vdots \\ a_0 \end{pmatrix} = \mathbf{0} \qquad (*)$$

We look successively at the matrices

$$A_0 = \begin{pmatrix} x_0 \end{pmatrix}, \quad A_1 = \begin{pmatrix} x_1 & x_0 \\ x_2 & x_1 \end{pmatrix}, \quad A_2 = \begin{pmatrix} x_2 & x_1 & x_0 \\ x_3 & x_2 & x_1 \\ x_4 & x_3 & x_2 \end{pmatrix}$$

Start at A_r, if we know $d \geq r$. For each i, compute $\det(A_i)$. If $\det(A_i) \neq 0$, then $d \neq i$. If $\det(A_i) = 0$, we solve $(*)$ assuming that $d = i$.

We then check our candidate for $a_0, \ldots, a_{d-1}$ over as many terms of the sequence as we wish.

If this fails we know $d > i$, so start again with A_{i+1}.

Remark. It is easier to use Gaussian elimination rather than expanding rows/columns.

3 Cryptography

Idea: Modify the message to make them unintelligible to all but the intended recipient.

Terminology: unencrypted message is called the **plain text**, and the encrypted message is referred to as the **cipher text**. Before transmission, the parties share some secret information known as a **key**. Let

$$\begin{aligned} \mathcal{M} &= \{\text{all possible unencrypted messages}\} \\ \mathcal{C} &= \{\text{all possible encrypted messages}\} \\ \mathcal{K} &= \{\text{all possible keys}\} \end{aligned}$$

Definition (Cryptosystem)**.** A **cryptosystem** consists of sets $\mathcal{M}, \mathcal{C}, \mathcal{K}$ and functions

$$\begin{aligned} e &: \mathcal{M} \times \mathcal{K} \to \mathcal{C} \\ d &: \mathcal{C} \times \mathcal{K} \to \mathcal{M} \end{aligned}$$

such that $d(e(m, k), k) = m$, encoding and decoding functions.

Example. Take $\mathcal{M} = \mathcal{C} = \{A, B, \dots, Z\}^* = \Sigma^*$.

(i) Simple substitution:

$$\mathcal{K} = \{\text{perms of } \Sigma\}$$

Each letter is encoded by replacing by its image under the permutation.

(ii) Vigenère cipher:

Identify $\Sigma = \mathbb{Z}/26\mathbb{Z}$. Also let $K = \Sigma^d$. Then write out the key k below m and add up letters. (For $d = 1$ this is a Caesar cipher, but for $d > 1$, a letter in m gets encrypted differently depending on its place in m. So not as weak to frequency analysis.)

What does is mean to 'break' a cryptosystem? We assume the enemy *may* know:

- d and e
- probability distribution of $\mathcal{M}$ and $\mathcal{K}$

but does not know the key.

We consider three possible levels of attack:

Level 1 Cipher text only
The enemy only knows some piece of cipher text.

Level 2 Known plaintext
The enemy has some m and $e(m, k)$ (but not k).

Level 3 Chosen plaintext
The enemy has the ability to generate $e(m, k)$ for any m (but still doesn't know k).

Remark.

(1) Substitution and Vigenère fail at level 1 if messages are in English (highly predictable) and are sufficiently long. But even for quite random plaintext, they both fail at level 2.

(2) For modern applications, level 3 is desirable. Note exhaustive searches are possible for all systems, so the only hope is to make such a method take a prohibitively long time.

3.1 Unicity distance

Take $(\mathcal{M}, \mathcal{K}, \mathcal{C})$ a cryptosystem. Say m, k are random variables taking values in $\mathcal{M}, \mathcal{K}$. Also let $c = e(m, k)$ (also random). Then define:

Definition (Equivocation)**.**

(1) The **key equivocation** is $H(k|c)$.

(2) The **message equivocation** is $H(m|c)$.

Lemma 3.1.

$$H(m|c) \leq H(k|c)$$

Proof. Note $m = d(c, k)$ so get $H(m|c, k) = 0$. Then

$$\begin{aligned} H(k|c) &= H(k, c) - H(c) \\ &= H(m, k, c) - H(m|k, c) - H(c) \\ &= H(m, k, c) - H(c) \\ &= H(k|m, c) + H(m, c) - H(c) \\ &= H(k|m, c) + H(m|c) \\ &\geq H(m|c) \end{aligned}$$

□

Remark. Say $(\mathcal{M}, \mathcal{K}, \mathcal{C})$ has **perfect secrecy** if $H(m|c) = H(m)$.

Suppose we send a sequence of messages $m^{(n)} = (m_1, \ldots, m_n)$ using the same key.

Definition (Unicity distance)**.** The **unicity distance** is

$$\mathcal{U} = \min \{ n \mid H(k|c^{(n)}) = 0 \} ;$$

i.e. the smallest n such that k becomes known.

Note

$$\begin{aligned} H(k|c^{(n)}) &= H(k, c^{(n)}) - H(c^{(n)}) \\ &= H(k, m^{(n)}, c^{(n)}) - H(c^{(n)}) \\ &\leq H(k) + H(m^{(n)}) - H(c^{(n)}) \end{aligned}$$

We assume

(1) All keys are uniformly likely, so $H(k) = \log |\mathcal{K}|$

(2) $H(m^{(n)}) = nH(m)$

(3) All assumed ciphertext is equally likely. So get

$$H(c^{(n)}) = n \log |\Sigma|$$

with $\mathcal{C} = \Sigma^*$.

So get

$$H(k|c^{(n)}) = \log|\mathcal{K}| + nH(m) - n\log|\Sigma|$$
$$\mathcal{U} = \frac{\log|\mathcal{K}|}{\log|\Sigma| - H(m)}$$

Definition. The **unicity distance** $\mathcal{U}$ of a system is the least length of ciphertext required to uniquely determine the key.

Remark. To make $\mathcal{U}$ large:

- make key space large
- send messages with little redundancy

Also to be secure, should not use a single key for more messages than the unicity distance.

3.2 Stream Cipher

Consider streams (sequences) in $\mathbb{F}_2$.

$$\begin{aligned} \text{Plain text is} & p_0, p_1, p_2, \dots \\ \text{Key stream is} & k_0, k_1, k_2, \dots \\ \text{Cipher text is} & c_0, c_1, c_2, \dots, z = p + k. \end{aligned}$$

A **one time pad** is where k is generated randomly. $k_i = 0, 1$ with chance $\frac{1}{2}$ and k_i's are iid.

Then $z = p + k$ gives a stream of iid random variables with $z_i = 0, 1$ with chance $\frac{1}{2}$ also. So without the key stream, deciphering is impossible.

A one time pad has perfect secrecy.

Problems:

(i) How to generate k? Quite difficult.

(ii) How do we share the key stream? Same as initial problem.

In most applications a one time pad is not viable. Instead we share $k_0, \dots, k_{d-1}$ and also use a shared FSR to generate k.

Lemma 3.2. Let $x_0, x_1, \dots$ be a sequence produced by a LFSR. Then $\exists M, N \leq 2^d - 1$ s.t. $x_{r+N} = x_r \ \forall r \geq M$.

Proof. Pigeonhole principle. If any of the first $2^d - 1$ d-blocks is 0, the sequence is 0 always. □

Remark. (i) The same result holds for a general FSR but $N, M \leq 2^d$.

(ii) The Berlekamp-Massey method gives that stream cipher are unsafe at level 2 (known plaintext).

(iii) Stream ciphers still get used, as they are cheap and easy, and can encrypt/decrypt on the fly.

3.3 New key streams from old

Recall a stream produced by a LFSR of form

$$x_n = \sum_{i=1}^{d} a_{d-i} x_{n-i}$$

has feedback polynomial $P(X) = X^d + a_{d-1}X^{d-1} + \cdots + a_0$. The solution of this recursion relation are linear combinations of the powers of the roots of $P(X)$.

Lemma 3.3. Let x, y be LFSR streams from $P(X), Q(X)$ respectively. Say $P(X)$ has roots $\alpha_1, \ldots, \alpha_M$ and $Q(X)$ has roots $\beta_1, \ldots, \beta_N$ in a field $K \supset \mathbb{F}_2$.

(i) Then $x + y$ is the output from a LFSR with feedback poly PQ.

(ii) $(x_n y_n)_n$ is the output of a LFSR with feedback poly

$$\prod_{i=1}^{M} \prod_{j=1}^{N} (X - \alpha_i \beta_j).$$

Proof (sketch). Assume (for simplicity) roots of P, Q are distinct i.e. $\alpha_i \neq \beta_j \forall i, j$. Then $x_n = \sum_{i=1}^{M} \lambda_i \alpha_i^n$, $y_n = \sum_{j=1}^{N} \mu_j \beta_j^n$ for some $\lambda_i, \mu_j \in K$.

$$\text{(i)} x_n + y_n = \sum_{i=1}^{M} \lambda_i \alpha_i^n + \sum_{j=1}^{N} \mu_j \beta_j^n$$

So $x_n + y_n$ solves the difference equation with feedback polynomial $P(X)Q(X)$.

$$\text{(ii)} x_n y_n = \sum_{i=1}^{M} \sum_{j=1}^{N} \lambda_i \mu_j (\alpha_i \beta_j)^n$$

solves the difference equation of $\prod_i \prod_j (X - \alpha_i \beta_j)$. (Check!) □

Remark. $x + y$ is from the LFSR of length $M + N$. xy is from LFSR of length MN.

So, we conclude (i) adding streams is no better than calculating the stream from a single register, and (ii) xy has 0's 75% of the time, this predictability is not desirable.

Example. Say x, y, z are streams from LFSRs. Let

$$k_n = \begin{cases} x_n & z_n = 0 \\ y_n & z_n = 1 \end{cases}$$

This might look complicated compared to an LFSR, but

$$\begin{aligned} k &= yz + x(1 + z) \\ &= x + (x + y)z \end{aligned}$$

so k is just a LFSR by Lemma 3.3.

Remark. Using an FSR stream is an example of a symmetric cryptosystem; i.e. one where the decryption algorithm equals or is easily deduced from the encryption algorithm.

3.4 Public Key Cryptography

We now use two keys: one for encryption and one for decryption. Then we aim to create a system such that even while knowing the encryption algorithm, decryption algorithm and public key, it should still be hard to find the private key and decrypt messages. Achieving this aim gives security at level 3. Note this system avoids the issue of key exchange.

The idea is to use 'one way' operations like

(1) Factoring/multiplying

(2) Discrete logs: Say p is a large prime, g a primitive element mod p, i.e. generates $\mathbb{F}_p^*$. Given x, find a such that $x \equiv g^a \pmod p$.

Definition. An algorithm runs in **polynomial time** if the number of operations is $\leq Cn^d$ for n the size of input and C,d are some constants.

Example. If N is input into an algorithm for factoring N, and N is written in binary, then the size of the input is $|log_2 N$, i.e. the number of digits used in binary.

Some polynomial time algorithms:

- Arithmetic on integers, i.e. $+, \times, -, \div$ with remainder.
- Computing gcd (Euclid's algorithm)
- Modular exponentiation (via repeated squaring method).
- Testing primality - (Agrawal, Kayal, Saxena, '02). Their algorithm was the first to be:
  - General
  - Polynomial time
  - Deterministic
  - Not conditional (e.g. not based on the Riemann hypothesis).

No polynomial time algorithms are known for (1) or (2) from earlier, however there are some elementary methods:

(1) Dividing by primes $\leq \sqrt{N}$. Works in $\mathcal{O}(\sqrt{N}) = \mathcal{O}(2^{\frac{B}{2}})$ time ($B = \log_2 N$).

(2) Shanks' Baby-step giant-step method: Say $m = \lceil\sqrt{p}\rceil$, $a = qm + r$, $0 \leq q, r < m$.

$$g^a \equiv x \pmod p \iff (g^m)^q \equiv xg^{-r} \pmod p$$

So make lists for $(g^m)^q$ and $xg^{-r} \pmod p$, for $q, r = 0, \ldots, m-1$. Then look for a match. Takes $\mathcal{O}(\sqrt{p}\log p)$ time.

3.5 Rabin-Williams Cryptosystem (1979)

Private key: p, q large distinct primes with $p, q \equiv 3 \pmod 4$. Public key: $N = pq$, $\mathcal{M}, \mathcal{C} = \{0, \ldots, N-1\}$. Encrypt $m \in \mathcal{M}$ as $c \equiv m^2 \pmod N$. The cipher text is c. (Note we should avoid using $m < \sqrt{N}$ and $(m, N) \neq 1$).

Lemma 3.4. Suppose $p = 4k - 1$ a prime, and d an integer. If $x^2 \equiv d \pmod{p}$ is soluble, then $x \equiv d^k \pmod{p}$ is a solution.

Proof.

$$(d^k)^2 \equiv d \cdot (x^2)^{2k-1} \equiv dx^{p-1} \equiv d \pmod{p}. \qquad \square$$

So if we know p, q and receive c, we can find x_1, x_2 such that $x_1^2 \equiv c \pmod{p}$, $x_2^2 \equiv c \pmod{q}$. Then CRT gives x such that $x^2 = c \pmod{N}$.

Lemma 3.5. Suppose $p = 4k - 1$ prime, $d \equiv 0 \pmod{p}$ an integer.

(i) If $x^2 \equiv d \pmod{p}$ is soluble, then there are exactly 2 solutions.

(ii) Hence if $N = pq$, p, q distinct odd primes and $\gcd(d, N) = 1$. If $x^2 \equiv d \pmod{N}$ soluble, there are exactly four solutions.

Proof.

(i) If x is a solution, $-x \pmod{p}$ is also a solution.

(ii) Say $x_0^2 \equiv d \pmod{N}$. Then for any choice of signs, x such that

$$\begin{cases} x \equiv \pm x_0 \pmod{p} \\ x \equiv \pm x_0 \pmod{q} \end{cases}$$

is also a solution, so there exactly four solutions. □

To decrypt RW, we compute all four solutions. Our message should involve sufficient redundancy such that only one solution makes sense.

Theorem 3.6. Breaking RW code is essentially equivalent to factoring N.

Proof. It is clear that factoring N allows decryption. Conversely, suppose we have a decryption algorithm, i.e. an algorithm for finding square roots mod N. Pick x mod N randomly. Use this algorithm to find y such that $y^2 \equiv x^2 \pmod{N}$. With probability $\frac{1}{2}$, $x \not\equiv \pm y \pmod{N}$ by Lemma 3.5.

Then in this case $\gcd(x-y, N)$ is a non-trivial factor of N. If $y \equiv \pm x$ repeat with another x. After r trials, probability of failure is $\frac{1}{2^r}$, which can be made arbitrarily small. □

3.6 RSA

Suppose $N = pq$, for p, q large distinct primes. Recall

$$\phi(N) = \#\{\, 1 \le x \le N \mid \gcd(x, N) = 1 \,\} = (p-1)(q-1).$$

By Euler-Fermat, if $(x, N) = 1$ then $x^{\phi(N)} \equiv 1 \pmod{N}$. We pick an integer e such that $\gcd(e, \phi(N)) = 1$. Then find d such that $de \equiv 1 \pmod{\phi(N)}$. Then the public key is (N, e) and the private key is (N, d).

Given a message m, it is encrypted as $c \equiv m^e \pmod{N}$, then decrypted as $m' \equiv c^d \pmod{N}$. By Euler-Fermat, $m \equiv m' \pmod{N}$. Note the probability that $(m, N) \equiv 1$ is so small we neglect this possibility.

Notation. $\mathrm{ord}_p(x)$ = order of x in $\mathbb{F}_p^\times = (\mathbb{Z}/p\mathbb{Z})^\times$.

Theorem 3.7. Let $N = p, q$ as above. Suppose $\phi(N)|2^a b$, for b odd. Also let $1 \leq x \leq N$, $\gcd(x, N) = 1$.

(i) If $\mathrm{ord}_p(x^b) \neq \mathrm{ord}_q(x^b)$ then $\exists 0 \leq t < a$ such that $\gcd(x^{2^t b} - 1, N)$ is a non-trivial factor of N.

(ii) Let $X = \#\{\, x \in (\mathbb{Z}/N\mathbb{Z})^\times \mid \mathrm{ord}_p(x^b) \neq \mathrm{ord}_q(x^b) \,\}$. Then $X \geq \frac{1}{2}\phi(N)$.

Proof.

(i) Say $y \equiv x^b \pmod N$. Euler-Fermat $\implies$ $y^{2^a} \equiv 1 \pmod N$, so $\mathrm{ord}_p(y), \mathrm{ord}_q(y)$ are powers of 2. Note also $\mathrm{ord}_p(y) \neq \mathrm{ord}_q(y)$ by hypothesis. Swapping p and q if needed, we get that for some $0 \leq t < a$

$$\left.\begin{aligned} y^{2^t} &\equiv 1 \pmod p \\ y^{2^t} &\not\equiv 1 \pmod q \end{aligned}\right. \implies \gcd(y^{2^t} - 1, N) = p.$$

(ii) Recall $(\mathbb{Z}/N\mathbb{Z})^\times = \{\, x + N\mathbb{Z} \mid x, N \text{ coprime} \,\}$. Also note CRT gives a bijection

$$\begin{aligned} (\mathbb{Z}/N\mathbb{Z})^\times &\longleftrightarrow (\mathbb{Z}/p\mathbb{Z})^\times \times (\mathbb{Z}/q\mathbb{Z})^\times \\ x &\longleftrightarrow (x \bmod p, x \bmod q) \end{aligned}$$

We will partition $(\mathbb{Z}/p\mathbb{Z})^\times$ into sets of elements where $\mathrm{ord}_p(x^b)$ is the same.

Claim: each set in this partition has size $\leq \frac{p-1}{2}$.

Assuming the claim, we note then that if $y \in (\mathbb{Z}/q\mathbb{Z})^\times$, by the claim,

$$\#\{\, x \in (\mathbb{Z}/p\mathbb{Z})^\times \mid \mathrm{ord}_p(x^b) \neq \mathrm{ord}_q(y^b) \,\} \geq \frac{1}{2}(p-1)$$

which implies $X \geq \frac{1}{2}(p-1)(q-1)$, by reverse CRT.

Proof of claim: We find a partition set of size $\frac{p-1}{2}$. Let g be a primitive root mod p. Then $(g^b)^{2^a} \equiv 1 \pmod p$, so $\mathrm{ord}_p(g^b)$ is a power of 2.

Say $x \equiv g^\delta$. Then $x^b = g^{\delta b}$. So $\mathrm{ord}_p(x^b) = \mathrm{ord}_p(g^b)$ in the case δ odd, or $\mathrm{ord}_p(x^b) \leq \frac{1}{2}\mathrm{ord}_p(g^b)$ for δ even. Hence $\{\, g^{\delta b} \pmod p \mid \delta \text{ odd} \,\}$ is the required subset. □

Corollary. Finding (N, d) from (N, e) is essentially as difficult as factoring N.

Proof. If we know N's factors, can easily get keys. Conversely, if we know d and e then $de \equiv 1 \pmod{\phi(N)}$ if and only if $\phi(N) \mid de - 1 = 2^a b$ for some a, b. Then we can use Theorem 3.7 to factor N. The probability of failure after r attempts is $< \frac{1}{2^r}$. □

Remark. We have shown that finding (N, d) from (N, e) is as hard as factoring N. But it is not known whether *decrypting messages* sent via RSA is as hard as factoring.

RSA avoids the issue of key sharing, but it is slow. Symmetric systems are often faster. So we are still interested in sharing keys. Shamir proposed the following analogy of the 'padlock example':
A chooses $a \in (\mathbb{Z}/p\mathbb{Z})^*$, computes a' such that $a'a \equiv 1 \pmod{p-1}$
B chooses $b \in (\mathbb{Z}/p\mathbb{Z})^*$, and computes b' such that $b'b \equiv 1 \pmod{p-1}$.

$$m \xrightarrow[A]{} m^a \xrightarrow[B]{} m^{ab} \xrightarrow[A]{} m^{aba'} \xrightarrow[B]{} m^{aba'b'} \equiv m \pmod p$$

3.7 Diffie-Hellman key exchange

A and B wish to agree on a secret key for communication. Let p be a large prime, g a primitive root mod p. Then:

A chooses α and sends $g^\alpha \pmod p$ to B

B chooses β and sends $g^\beta \pmod p$ to A.

Both parties then compute $K = g^{\alpha\beta}$, and use this as the secret key. An attacker has the problem of computing $g^{\alpha\beta}$ from g, g^α and g^β. It is conjectured that this is as difficult as the discrete log problem.

3.8 Authenticity and signatures

Say A, B send messages.

3.9 Authenticity using RSA

Suppose A has private key (N, d) and public key (N, e). A now uses the private key (N, d) for encryption. Anyone can decrypt using the public key (N, e), but cannot forge messages sent by A. If B sends a random message and then receives back a message from A which upon decryption ends in μ, then B knows they are in communication with A.

Some examples to show integrity is important. **Homomorphism attack**: A bank creates a message of the form (M_1, M_2) where M_1 is the client name and M_2 is the amount to be credited to their account. Messages are encoded using RSA as $(z_1, z_2) = (M_1^e \pmod N), M_2^e \bmod N)$.

I enter into a transaction which credits £100 to my account. I intercept the resulting (z_1, z_2) and then send (z_1, z_2^3). I become a millionaire without having to break RSA.

Copying: Even if I didn't know RSA was being used, I could still repeatedly transmit (z_1, z_2) - can stop this by time-stamping.

Remark. We consider the 'signature of a message', not the signature of a sender.

We suppose all users have a private key and a public key. We have a map $s : \mathcal{M} \times \mathcal{K} \to \mathcal{S}$, where $\mathcal{M}$ is the set of all messages, $\mathcal{K}$ is the set of all keys and $\mathcal{S}$ is the set of all possible signatures. A signs a message m with $s(m, K_a)$ where K_A where K_A is A's private key. Then B checks the signature using A's public key. s should be a 'trapdoor function' (one-way) i.e. no-one can sign a message from A without A's private key.

For example using RSA, A has private key (N, d) and public key (N, e). A signs message m with $s = m^d \pmod N$. Anyone can verify that (m, s) is a valid signed message using A's public key.

Remark (Existential forgery)**.** Anyone can sign a message of the form $(s^e \pmod N), s)$, but we hope such a message will not be meaningful.

In practice, rather than sign a message m we sign $h(m)$ where $h : \mathcal{M} \to \{1, \dots, N-1\}$ is a collision-resistant hash function, i.e. a publicly known function chosen so that it is easy to verify that some input data maps to a given hash value, but if the input data is unknown it is deliberately difficult to reconstruct it, or find another input that maps to the same hash value.

Thus if the message is changed, then it is very likely that the hash value will also change. So we can use the hash value to test the integrity of a message, e.g. combat a homomorphism attack.

3.10 El-Gamal Signature Scheme

Take p a large prime, g a primitive root mod p. A chooses a random integer $1 < u < p$.

Now, the public key is $p, g, y = g^u \pmod p$, and the private key is u. Let $h : \mathcal{M} \to \{1, 2, \ldots, p-1\}$ be a collision-resistant hash-function.

To send a message, A chooses a random exponent k, with $(k, p-1) = 1$ and computes $1 \leq r \leq p-1$ and $1 \leq s \leq p-2$ satisfying

(i) $r \equiv g^k \pmod p$

(ii) $h(m) \equiv ur + ks \pmod{p-1}$

(note: $(k, p-1) = 1$, so k has an inverse mod $p-1$, so can solve for s). A signs the message m with (r, s).

B accepts the signature if

$$g^{h(m)} \equiv y^r r^s \pmod p$$

Now,

$$\begin{aligned} g^{h(m)} &\equiv g^{ur+ks} \pmod p \\ &\equiv (g^u)^r (g^k)^s \\ &\equiv y^r r^s \pmod p \end{aligned}$$

It is believed that the only way to forge signatures is to find u from $y \equiv g^u \pmod p$, i.e. by solving the discrete log problem.

Choice of k. It is essential that a different choice of k is used to sign each message, otherwise messages m_1, m_2 are signed (r, s_1) and (r, s_2) with

$$\begin{aligned} h(m_1) &\equiv ur + ks_1 \pmod{p-1} \\ h(m_2) &\equiv ur + ks_2 \pmod{p-1} \\ \implies h(m_1) - h(m_2) &\equiv k(s_1 - s_2) \pmod{p-1} \end{aligned}$$

Let $d = (s_1 - s_2, p-1)$, and put

$$h' \equiv \frac{h(m_1) - h(m_2)}{d}, \quad s' \equiv \frac{s_1 - s_2}{d}, \quad p \equiv \frac{p-1}{d}$$

Then $h' \equiv ks' \pmod{p'}$.

As $(s', p') = 1$, we can solve for $k \pmod{p'}$. So $k \equiv k_0 \pmod{p'}$ for some k_0. Then $k \equiv k_0 + \lambda p' \pmod{p-1}$ for some $0 \leq \lambda \leq d-1$. For those d possibilities determine correct value of k using

$$g^k \equiv r \pmod p.$$

Similarly we solve $h(m) \equiv ur + ks \pmod{p-1}$ for u, which is A's private key.