X Lecture 3,

$$P_r(X_{n+1}=x_{n+1} \mid X_n=x_n, X_{n-1}=x_{n-1}, \cdots, X_1=x_1)$$

$$= P_r(X_{n+1}=x_{n+1} \mid X_n=x_n) .$$

In this case:

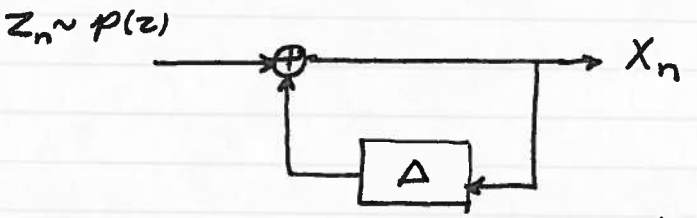
$$p(x_1, x_2, \cdots, x_n) = p(x_1)p(x_2|x_1)p(x_3|x_2) \cdots p(x_n|x_{n-1}) .$$

Defn: A Markov chain is time-invariant if

$$P_r(X_{n+1}=a \mid X_n=b) = P_r(X_2=a \mid X_1=b) \quad \forall a,b \in \mathcal{X}.$$

Examples of Markov Source

1)



$$X_n = X_{n-1} + Z_n$$

$$p(z) = \begin{cases} p & z=1 \\ 1-p & z=0 \end{cases}$$

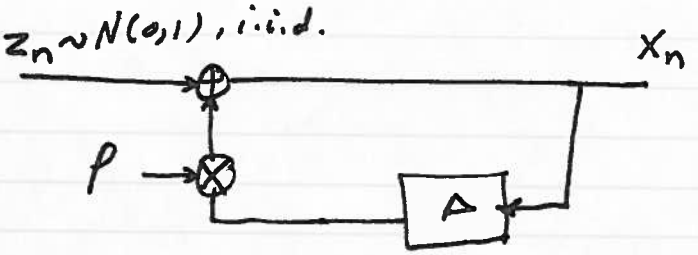
$$P(X_n=0|X_{n-1}=0) = 1-p$$

$$P(X_n=1|X_{n-1}=0) = p$$

$$P(X_n=0|X_{n-1}=1) = p$$

$$P(X_n=1|X_{n-1}=1) = 1-p$$

2)



$$X_n = \rho\, x_{n-1} + Z_n$$

The entropy rate of a stationary Markov Chain is given by:

$$H(\mathcal{X}) = H'(\mathcal{X}) = \lim_{n\to\infty} H(X_n|X_{n-1},\cdots X_1) = \lim_{n\to\infty} H(X_n|X_{n-1}) = H(X$$

A Markov Chain can be visualized using a finite state diagram were X_n is the state a time n.
A Markov chain, visualized as a finite state machine can be completely specified by its initial state and the transisition probabilities & At any give time n, $P_r(X_n)$ can be written as

$$P_r(X_n) = \sum_{x_{n-1}} p(x_{n-1}) P_{x_n x_n}$$

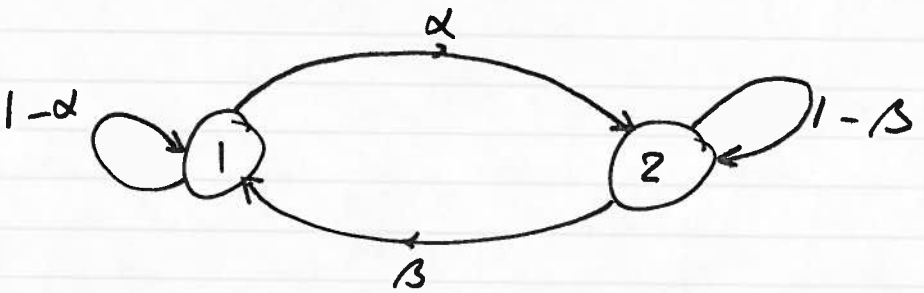
If the probability distribution at times $n-1$ and n are the same, then we call that distribution stationary.
If we denote the stationary probability distribution by $\mu = [p(X_n = x_1), \cdots, p(X_n = x_j)]$ then it can be calculated by solving

$$\mu P = \mu \quad \text{where } P \text{ is}$$

the state_transition probabilities [matrix].

Example: Two state Markov Process

$$P = \begin{bmatrix} 1-\alpha & \alpha \\ \beta & 1-\beta \end{bmatrix}$$



This for example, can model a fading channel with a good (no or low fading) and a bad state (high fading). It can also model a traffic source where in busy (on) state traffic bursts are generated where in idle (off) state no traffic is generated.

To find $\mu = [\mu_1, \mu_2]$ where $\mu_i = P_r[X_n = i]$, $i = 1, 2$ we solve:

$$\mu_1 = (1-\alpha)\mu_1 + \mu_2 \beta$$

$$\mu_2 = \mu_1 \alpha + \mu_2 (1-\beta)$$

with the condition $\mu_1 + \mu_2 = 1$.

Doing this, we get:

$$\mu_1 = \frac{\beta}{\alpha+\beta}$$

$$\mu_2 = \frac{\alpha}{\alpha+\beta}$$

Theorem:

The entropy of state X_n is:

$$H(X_n) = H\left(\frac{\alpha}{\alpha+\beta}\right)$$

Theorem: The entropy rate of a stationary Markov chain with stationary distribution μ and transition matrix P is:

$$H(\mathcal{X}) = -\sum_i \sum_j \mu_i P_{ij} \log P_{ij}$$

Proof:

$$H(\mathcal{X}) = H(X_2|X_1) = \sum_i \mu_i \sum_j (-P_{ij} \log P_{ij})$$

Example: the entropy rate of the two state Markov chain discussed earlier is:

$$H(\mathcal{X}) = H(X_2|X_1) = \frac{\beta}{\alpha+\beta} H(\alpha) + \frac{\alpha}{\alpha+\beta} H(\beta).$$

Lossless Source Coding

Definition: A source code for the random variable X is a mapping from $\mathcal{X}$, the range of X to D^*, the set of finite length strings of symbols from a D-ary alphabet.

Definition: The expected length of a source code $C(x)$ is given by

$$L(C) = \sum_{x \in \mathcal{X}} p(x) l(x)$$

where $p(x)$ is the probability of the event x and $l(x)$ is the length of $C(x)$, the code assigned to x.

Example:

$$\mathcal{X} = \{1, 2, 3, 4\}$$

$$\underline{p} = \left\{\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{8}\right\}$$

let's define:

$C(1) = 0$
$C(2) = 10$
$C(3) = 110$
$C(4) = 111$

then $L(C) = 1 \times \frac{1}{2} + 2 \times \frac{1}{4} + 3 \times \frac{1}{8} + 3 \times$

or $L(C) = 1.75$ bits

$H_2(X) = 1.75$ bits

so: $L(C) = H_2(X)$

This is an example of D-adic (in this case) 2-adic distribution.

Definition: A probability distribution is D-adic w.r.t. D if each probability is equal to D^{-n} for some n.

Some desirable properties of Codes:

1) A Code should be non-singular, i.e.,

Definition: A Code is non-singular if every element of $\mathcal{X}$ maps into a different string in D^*

$$x_i \neq x_j \Rightarrow C(x_i) \neq C(x_j)$$

A Code that is non-singular allows us to encode different symbols of $\mathcal{X}$ unambiguously, for strings of element of $\mathcal{X}$ we may need to separate the codewords for different x's by comma. This involves an overhead.

Example of Morse Code: letter space and word space in addition to Mark and Space are needed.

2) To avoid comma (delimiter), we may wish that any finite sequence of codewords be non-singular.

Definition: The extension C^* of a code C is the mapping from finite length strings of $\mathcal{X}$ to finite length strings of $\mathcal{D}$, defined as

$$C(x_1, x_2, \cdots x_n) = C(x_1)C(x_2)\cdots C(x_n).$$

where $C(x_1)\cdots C(x_n)$ is the concatenation of $C(x_1), C(x_2), \cdots C(x_n)$.

Example: For the previous example

$1 \leftrightarrow 0$
$2 \rightarrow 10$
$3 \rightarrow 110$
$4 \rightarrow 111$

and

$$C(1\,3\,2\,1) = 0\,1\,1\,0\,1\,0\,0$$

Definition: A code is called uniquely decodable if its extension is non-singular.

Example:

$1 \rightarrow 0$
$2 \rightarrow 010$
$3 \rightarrow 01$
$4 \rightarrow 10$

} This is not a uniquely decodable code since

$2 \rightarrow 010$
$3,1 \rightarrow 01,0$ $\Rightarrow$ need for comma
$1,4 \rightarrow 0,10$

Example

1 → 0
2 → 00
3 → 11
4 → 110

This is uniquely decodable. However, you may need to wait until the end of sequence to decode, e.g., 110110

11 → is it 11 or preamble to 110 ?

110 → is it 110 or preamble to 11,00 ?

1101 → is it 11,01 or 110 and preamble to 11 (or 11

11011 → is it 110,11 or 11,011 or 11,01 and 1x etc.

finally 11011 0 → 110,110 (decodes to 4,4).

Examples:

3) It is desirable that the code be decodable after receiving each codeword. Codes with this property are called instantaneous or prefix codes.

Definition: A code is a prefix code or an instantaneous code if no codeword is prefix to any other codeword.

Example:

$1 \rightarrow 0$
$2 \rightarrow 10$
$3 \rightarrow 110$
$4 \rightarrow 111$

This code is a prefix code since:

1) when you come across a single 0, you know it represents a 1
2) when you see a 1 you wait for the next bit, if it is 0 you know to decode 2
3) if it is 11 then you wait for the next bit and depending on it you decode 3 or 4.

Kraft inequality for prefix codes:

Theorem: For any prefix code over an alphabet of size D, the codeword lengths $l_1, \ldots l_m$ must satisfy

$$\sum_i D^{-l_i} \leq 1$$

Conversely, given a set of lengths satisfying the above (Kraft) inequality, there exists a prefix code with those lengths.

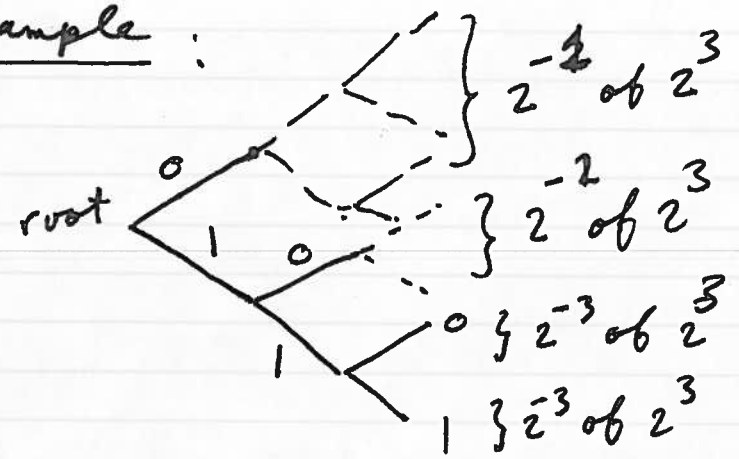
Proof: Consider a D-ary tree whose branches are labeled by the symbols of the codewords. A codeword of length l_i is a leaf at depth l_i. The total depth of the tree is $l_{max} = \max_i l_i$. A codeword of length l_i precludes a fraction $\frac{1}{D^{l_i}}$ of the total number $D^{l_{max}}$ of the potential leafs (nodes) from being a codeword (otherwise it would a prefix to its descendants). So, a codeword of length l_i (a node at depth l_i) deprives $D^{l_{max}-l_i}$ of the nodes at the depth of l_{max} from the right of being a codeword. The total number of excluded nodes (of depth l_{max}) is $\sum_i D^{l_{max}-l_i}$. This sum can not exceed the total number of nodes $D^{l_{max}}$. So

$$\sum_i D^{l_{max}-l_i} \leq D^{l_{max}}$$

or

$$\boxed{\sum_i D^{-l_i} \leq 1}$$

Example:



Conversely, given $l_1, l_2, \ldots l_m$ satisfying the Kraft inequality, we can construct a tree like the one above and then label the first node of depth l_1 as codeword 1 and remove all the descendants from the tree. Then label the 1st remaining node of depth l_2 as codeword 2, etc. Continuing this, we construct a prefix code with lengths $l_1, l_2, \ldots, l_m$.

Optimal Codes

We would like to minimize:

$$L = \sum_i p_i l_i$$

Subject to:

$$\sum_i D^{-l_i} \leq 1.$$

we use the method of Lagrange multipliers:

$$J = \sum_i p_i l_i + \lambda\left(\sum_i D^{-l_i}\right)$$

$$\frac{\partial J}{\partial l_i} = p_i - \lambda D^{-l_i} \log_e D = 0$$

$$\Rightarrow D^{-l_i} = \frac{p_i}{\lambda \log_e D}$$

$$\sum_i D^{-l_i} = \sum_i \frac{p_i}{\lambda \log_e D} = \frac{1}{\lambda \log_e D} = 1 \Rightarrow \lambda = \frac{1}{\log_e D}$$

or $\begin{cases} D^{-l_i} = p_i \\ l_i^* = -\log_D p_i \end{cases}$

Of course l_i^* found as $-\log_D p_i$ can be non-integer. This non-integer optimal length results in

$$L^* = \sum_i p_i l_i^* = -\sum_i p_i \log_D p_i = H_D(X).$$

In practice, we may set

$$l_i = \lceil \log_D \left(\frac{1}{p_i}\right) \rceil \qquad \{\text{this is called a Shannon code}\}$$

So

$$-\log_D p_i \leq l_i < -\log_D p_i + 1$$

an averageng gives

$$\boxed{H_D(X) \leq L < H_D(X) + 1}$$

Note that the lengths $l_i = \lceil \log_D(\frac{1}{P_i}) \rceil$ satisfy the Kraft's inequality:

$$\sum_i D^{-\lceil \log_D \frac{1}{P_i} \rceil} \leq \sum_i D^{-\log_D \frac{1}{P_i}} = \sum P_i = 1$$

So, it is possible to have a prefix code with these lengths.

We can reduce the 1 bit overhead in the above inequality be encoding n symbols of the source at a time. Then the expected codeword length per input symbol is

$$L_n = \frac{1}{n} \sum p(x_1, \cdots x_n) l(x_1, \cdots, x_n) = \frac{1}{n} E\{l(X_1, \cdots X_n)\}$$

But,

$$H(X_1, \cdots X_n) \leq E\{l(X_1, \cdots X_n)\} \leq H(X_1, \cdots X_n) + 1$$

Since $X_1, \cdots X_n$ are i.i.d.,

$$H(X_1, \cdots X_n) = \sum_i H(X_i) = H(X)$$

So:

$$H(X) \leq L_n \leq H(X) + \frac{1}{n}$$

For a sequence of source symbols that are not i.i.d. (i.e., a source with memory):

$$\frac{H(X_1, \ldots X_n)}{n} \leq L_n^* < \frac{H(X_1, \ldots X_n)}{n} + \frac{1}{n}$$

So:

$$L_n^* \to H(\mathcal{X}) \quad \text{the entropy rate.}$$

Kraft inequality for uniquely decodable codes:

Theorem (McMillan): For any uniquely decodable code, with code lengths $l_1, \ldots l_n$ must satisfy:

$$\sum_i D^{-l_i} \leq 1$$

Conversely, given a set of lengths, l_i, satisfying the Kraft's inequality, one can construct a uniquely decodable code with those codeword lengths

Proof: Take C^k as the kth extension of C, i.e.,

$$x_1, \ldots x_k \to C(x_1)\, C(x_2) \cdots C(x_k)$$

Then, $l(x_1, \ldots x_k) = \sum_{i=1}^{k} l(x_i)$

We want to prove that

$$\sum_{x \in \mathcal{X}} D^{-\ell(x)} \leq 1$$

Take the k-th power of the above sum:

$$\left(\sum_{x \in \mathcal{X}} D^{-\ell(x)} \right)^k = \sum_{x_1 \in \mathcal{X}} \sum_{x_2 \in \mathcal{X}} \cdots \sum_{x_k \in \mathcal{X}} D^{-\ell(x_1)} D^{-\ell(x_2)} \cdots D^{-\ell(x_k)}$$

$$= \sum_{\underline{x} \in \mathcal{X}^k} D^{-[\ell(x_1) + \cdots + \ell(x_k)]}$$

$$= \sum_{\underline{x}^k \in \mathcal{X}^k} D^{-\ell(x^k)} \qquad x^k = (x_1, \cdots x_k)$$

Now, we gather the terms by word lengths.

$$\sum_{x^k \in \mathcal{X}^k} D^{-\ell(x^k)} = \sum_{m=1}^{k \ell_{max}} a(m) D^{-m}$$

where $a(m)$ is the number of sequences $x^k = (x_1, \cdots, x_k)$ mapped into a codeword of length m

$$\left(\sum_{x \in \mathcal{X}} D^{-\ell(x)} \right)^k = \sum_{m=1}^{k \ell_{max}} a(m) D^{-m} \leq \sum_{m=1}^{k \ell_{max}} D^m D^{-m} = k \ell_{max}$$

or

$$\sum_{x \in \mathcal{X}} D^{-\ell(x)} \leq (k \ell_{max})^{1/k}$$

since this is true $\forall k$ it is true as $k \to \infty$

$\lim_{k \to \infty} (k l_{max})^{1/k} = 1$. So,

$$\sum_{x \in \mathcal{X}} D^{-l(x)} \leq 1$$

Conversely, if l_i's satisfy the Kraft's inequality then, we can design a prefix code (from the theorem ~~proof~~ of Kraft inequality for prefix codes) and since a prefix code is a uniquely decodable code the theorem is proven.

Interpretation: The only interesting members of the class of uniquely decodable codes are the prefix codes, and any uniquely decodable code can be transformed into a prefix code. So, it is not a bad idea to just concentrate on the prefix codes.

Optimum prefix codes:

To motivate: give the example of a source generating 1 with probability 0.001 and zero with probability of 0.999 and show that the shannon code is not suitable. Conclude the equality of the ~~two~~ length of the two least likely messages. ~~dbph~~ Then give the following lemma

~~Theorem~~:

Lemma

~~Theorem~~: For any distribution, there is an optimal prefix (instantaneous) code that satisfies the following properties:

1) If $p_j > p_k$ then $l_j \leq l_k$

2) The two longest codewords have the same length.

3) The two longest codewords differ only in the last bit and correspond to the least likely symbols.

{ in general: if there are two or more codewords of the same length, then two of them differ only in the last digit }

Proof:

Assume that we have a code C that assigns $l_j > l_k$ to symbols j and k when $p_j > p_k$. We form another code C' whose elements are the same as that of C except that the codewords for j and k are swapped:

$$c_i' = \begin{cases} c_i & i \neq j,\ i \neq k \\ c_j & i = k \\ c_k & i = j \end{cases}$$

Then the expected length of C and C' will be:

$$L(C) = \sum_i l_i p_i = \sum_{\substack{i \neq j \\ i \neq k}} l_i p_i + l_j p_j + l_k p_k$$

$$L(C') = \sum_i l_i' p_i = \sum_{\substack{i \neq j \\ i \neq k}} l_i p_i + l_j p_k + l_k p_j$$

$$L(C') - L(C) = l_j(p_k - p_j) + l_k(p_k - p_j) = (l_j - l_k)(p_k - p_j) <$$

So

$$L(C') < L(C) \Rightarrow L(C) \text{ is not optimal.}$$

2) If the two longest codewords do not have the same length, the last bit of the longer codeword can be deleted. This reduces the expected length while not violating the prefix condition.

According to part 1) these longest codewords should belong to least likely symbols.

3) If the two longest codewords are different in more than the last bit, we can delete the last bit and still have different (but shorter) codewords. So, they must be different only in the last bit.

This Lemma provides us with a way to construct optimal prefix codes.

Assume a source with probabilities $p_1, \ldots p_m$ and a code for it, C_m, with lengths $l_1, \ldots l_m$

The expected length of this code is

$$L(C_m) = \sum_{i=1}^{m} p_i l_i$$

Now, we define another source with $m-1$ symbols. The first $m-2$ symbols are the same as those of the original source and $m-1$st is one with probability $p_{m-1} + p_m$. So:

$$p'_i = \begin{cases} p_i & i = 1, 2, \cdots, m-2 \\ p_{m-1} + p_m & i = m-1 \end{cases}$$

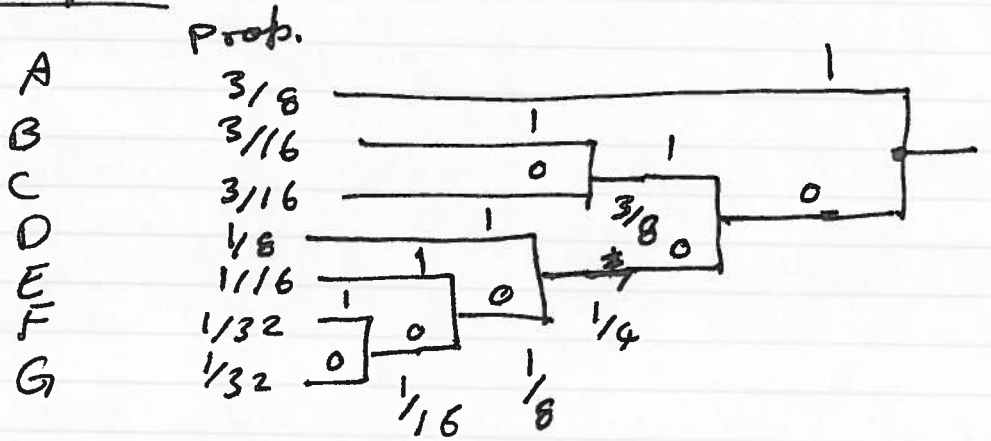
Assume that we have a code for the second source, called C_{m-1} with codelength $l'_1, \cdots l'_{m-1}$. We form a code for the original source from this code by taking the first $m-2$ codewords and use them for the first $m-2$ ~~Codes~~ Symbols of the original source and expand the $(m-1)$st one into two codewords of length $l'_{m-1} + 1$ and use it for $(m-1)$st and m-th symbols. Then

$$L(C_m) = \sum_{i=1}^{m} l_i p_i = \sum_{i=1}^{m-2} l'_i p_i + (l'_{m-1} + 1) p_{m-1} + (l'_{m-1} + 1) p_m$$

$$= \sum_{i=1}^{m-1} p'_i l'_i + p_{m-1} + p_m = L(C_{m-1}) + p_{m-1} + p_m$$

Note that the difference between $L(C_m)$ and $L(C_{m-1})$ is $P_{m-1} + P_m$, i.e., a quantity that is not dependent on the C_{m-1}. Therefore, in order to minimize $L(C_m)$, we can minimize $L(C_{m-1})$ for a source generated by merging the two least likely source symbols. This procedure is continued recursively to form a prefix code which is optimal and is called a Huffman Code.

Example:



So:

A → 1
B → 0 1 1
C → 0 1 0
D → 0 0 1
E → 0 0 0 1
F → 0 0 0 0 1
G → 0 0 0 0 0

$L = E(l) = 2.44$ bits/symbol

$H(X) = 2.37$