Module 1

Introduction to Digital Communications and Information Theory

Lesson 4

Coding for discrete sources

After reading this lesson, you will learn about

- ***Need for coding source letters***
- ***Variable length coding***
- ***Prefix – condition code***
- ***Kraft Inequality Theorem***
- ***Huffman Coding***
- ***Source Coding Theorem***

All source models in information theory may be viewed as random process or random sequence models. Let us consider the example of a discrete memory less source (DMS), which is a simple random sequence model.

A DMS is a source whose output is a sequence of letters such that each letter is independently selected from a fixed alphabet consisting of letters; say a_1, a_2, …….a_k. The letters in the source output sequence are assumed to be random and statistically independent of each other. A fixed probability assignment for the occurrence of each letter is also assumed. Let us, consider a small example to appreciate the importance of probability assignment of the source letters.

Let us consider a source with four letters a_1, a_2, a_3 and a_4 with $P(a_1)=0.5$, $P(a_2)=0.25$, $P(a_3)= 0.13$, $P(a_4)=0.12$. Let us decide to go for binary coding of these four source letters. While this can be done in multiple ways, two encoded representations are shown below:

Code Representation#1: a_1: 00, a_2:01, a_3:10, a_4:11

Code Representation#2: a_1: 0, a_2:10, a_3:001, a_4:110

It is easy to see that in method #1 the probability assignment of a source letter has not been considered and all letters have been represented by two bits each. However in the second method only a_1 has been encoded in one bit, a_2 in two bits and the remaining two in three bits. It is easy to see that the average number of bits to be used per source letter for the two methods are not the same. ($\bar{a}$ for method #1=2 bits per letter and $\bar{a}$ for method #2 < 2 bits per letter). So, if we consider the issue of encoding a long sequence of letters we have to transmit less number of bits following the second method. This is an important aspect of source coding operation in general. At this point, let us note the following:

a) We observe that assignment of small number of bits to more probable letters and assignment of larger number of bits to less probable letters (or symbols) may lead to efficient source encoding scheme.

b) However, one has to take additional care while transmitting the encoded letters. A careful inspection of the binary representation of the symbols in method #2 reveals that it may lead to confusion (at the decoder end) in deciding the end of binary representation of a letter and beginning of the subsequent letter.

So a source-encoding scheme should ensure that

1) The average number of coded bits (or letters in general) required per source letter is as small as possible and

2) The source letters can be fully retrieved from a received encoded sequence.

In the following we discuss a popular variable-length source-coding scheme satisfying the above two requirements.

Variable length Coding

Let us assume that a DMS U has a K- letter alphabet $\{a_1, a_2, \ldots\ldots\ldots a_K\}$ with probabilities $P(a_1)$, $P(a_2)$,……. $P(a_K)$. Each source letter is to be encoded into a codeword made of elements (or letters) drawn from a code alphabet containing D symbols. Often for ease of implementation a binary code alphabet (D = 2) is chosen. As we observed earlier in an example, different codeword may not have same number of code symbols. If n_k denotes the number of code symbols corresponding to the source letter a_k , the average number of code letters per source letter ($\bar{n}$) is:

$$\bar{n} = \sum_{k=1}^{K} P(a_k) n_k \qquad 1.4.1$$

Intuitively, if we encode a very long sequence of letters from a DMS, the number of code letters per source letter will be close to $\bar{n}$.

Now, a code is said to be *uniquely decodable* if for each source sequence of finite length, the sequence of code letters corresponding to that source sequence is different from the sequence of code letters corresponding to any other possible source sequence.

We will briefly discuss about a subclass of uniquely decodable codes, known as *prefix condition code*. Let the code word in a code be represented as

$\overline{x_k} = (x_{k,1}, x_{k,2}, ..., x_{k,n_k})$, where $x_{k,1}, x_{k,2}, ..., x_{k,n_k}$ denote the individual code letters (when D=2, these are 1 or 0). Any sequence made up of an initial part of $\overline{x_k}$ that is $x_{k,1}, x_{k,2}, ..., x_{k,i}$ for $i \le n_k$ is called a prefix of $\overline{x_k}$.

A prefix condition code is a code in which no code word is the prefix of any other codeword.

Example: consider the following table and find out which code out of the four shown is / are prefix condition code. Also determine $\bar{n}$ for each code.

Source letters:- a_1, a_2, a_3 and a_4
P(a_k) :- $P(a_1)=0.5$, $P(a_2)=0.25$, $P(a_3)= 0.125$, $P(a_4)=0.125$
Code Representation#1: a_1: 00, a_2:01, a_3:10, a_4:11
Code Representation#2: a_1: 0, a_2:1, a_3:00, a_4:11
Code Representation#3: a_1: 0, a_2:10, a_3:110, a_4:111
Code Representation#4: a_1: 0, a_2:01, a_3:011, a_4:0111

A prefix condition code can be decoded easily and uniquely. Start at the beginning of a sequence and decode one word at a time. Finding the end of a code word is not a problem as the present code word is not a prefix to any other codeword.

Example: Consider a coded sequence 0111100 as per Code Representation #3 of the previous example. See that the corresponding source letter sequence is a_1, a_4, a_2, a_1.

Now, we state one important theorem known as Kraft Inequality theorem without proof.

Kraft Inequality Theorem

If the integers n_1, n_2, …….n_K satisfy the inequality

$$\sum_{k=1}^{K} D^{-n_k} \leq 1 \qquad 1.4.2$$

then a prefix condition code of alphabet size D exists with these integers as codeword lengths.

This also implies that the lengths of code words of any prefix condition code satisfy the above inequality.

It is interesting to note that the above theorem does not ensure that any code whose codeword lengths satisfy the inequality is a prefix condition code. However it ensures that a prefix condition code exists with code word length n_1, n_2, …….n_K.

Binary Huffman Coding (an optimum variable-length source coding scheme)

In Binary Huffman Coding each source letter is converted into a binary code word. It is a prefix condition code ensuring minimum average length per source letter in bits.

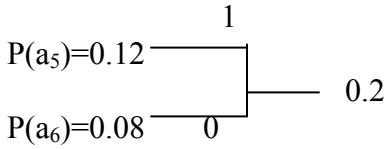
Let the source letters a_1, a_2, …….a_K have probabilities $P(a_1)$, $P(a_2)$,……. $P(a_K)$ and let us assume that $P(a_1) \geq P(a_2) \geq P(a_3) \geq \ldots \geq P(a_K)$.

We now consider a simple example to illustrate the steps for Huffman coding.

Steps to calculate Huffman Coding

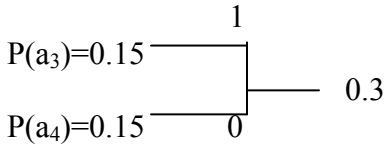
Example Let us consider a discrete memoryless source with six letters having $P(a_1)=0.3$, $P(a_2)=0.2$, $P(a_3)=0.15$, $P(a_4)=0.15$, $P(a_5)=0.12$ and $P(a_6)=0.08$.

1. Arrange the letters in descending order of their probability (here they are arranged).
2. Consider the last two probabilities. Tie up the last two probabilities. Assign, say, 0 to the last digit of representation for the least probable letter (a_6) and 1 to the last digit of representation for the second least probable letter (a_5). That is, assign '1' to the upper arm of the tree and '0' to the lower arm.



3. Now, add the two probabilities and imagine a new letter, say b_1, substituting for a_6 and a_5. So $P(b_1) = 0.2$. Check whether a_4 and b_1 are the least likely letters. If not, reorder the letters as per Step#1 and add the probabilities of two least likely letters. For our example, it leads to:
   $P(a_1)=0.3$, $P(a_2)=0.2$, $P(b_1)=0.2$, $P(a_3)=0.15$ and $P(a_4)=0.15$

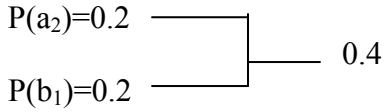
4. Now go to Step#2 and start with the reduced ensemble consisting of a_1 , a_2 , a_3 , a_4 and b_1. Our example results in:



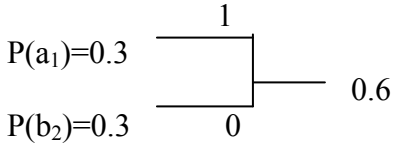
   Here we imagine another letter b_1, with $P(b_2)=0.3$.

5. Continue till the first digits of the most reduced ensemble of two letters are assigned a '1' and a '0'.

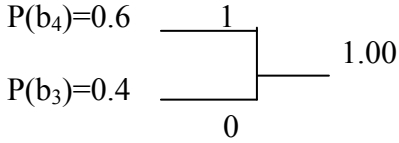
   Again go back to the step (2): $P(a_1)=0.3$, $P(b_2)=0.3$, $P(a_2)=0.2$ and $P(b_1)=0.2$. Now we consider the last two probabilities:



   So, $P(b_3)=0.4$. Following Step#2 again, we get, $P(b_3)=0.4$, $P(a_1)=0.3$ and $P(b_2)=0.3$.
   Next two probabilities lead to:



   with $P(b_4) = 0.6$. Finally we get only two probabilities



6. Now, read the code tree inward, starting from the root, and construct the codewords. The first digit of a codeword appears first while reading the code tree inward.

Hence, the final representation is: a_1=11, a_2=01, a_3=101, a_4=100, a_5=001, a_6=000.

A few observations on the preceding example

1. The event with maximum probability has least number of bits.
2. Prefix condition is satisfied. No representation of one letter is prefix for other. Prefix condition says that representation of any letter should not be a part of any other letter.
3. Average length/letter (in bits) after coding is
   $= \sum_i P(a_i)n_i = 2.5$ bits/letter.
4. Note that the entropy of the source is: H(X)=2.465 bits/symbol. Average length per source letter after Huffman coding is a little bit more but close to the source entropy. In fact, the following celebrated theorem due to C. E. Shannon sets the limiting value of average length of codewords from a DMS.

Source Coding Theorem (for coding source letters with variable length codes)

Given a finite source ensemble U with entropy H (U) and given a code alphabet of D symbols, it is possible to assign code words to the source letters in such a away that the prefix condition is satisfied and the average length of the code words $\bar{n}$, satisfies the inequality.

$$\bar{n} < \frac{H(U)}{\log_2 D} + 1,$$

Further, for any uniquely decodable set of code words,

$$\bar{n} \geq \frac{H(U)}{\log_2 D}$$

As all prefix condition codes are uniquely decodable, for such a code,

$$\frac{H(U)}{\log D} \leq \bar{n} \leq \frac{H(U)}{\log D} + 1$$

For binary representation (i.e. D = 2, as in our example),

$$H(U) \leq \bar{n} < H(U) + 1$$

The equality holds if,

$$P(a_k) = D^{-n_k} \quad \text{for } 1 \leq k \leq K, \qquad 1.4.3$$

where n_k is the length of the k-th source letter.

Problems

Q1.4.1) What is a prefix-condition code?

Q1.4.2) Is optimum source coding possible when the length of coded letters is the same (i.e. fix length coding)?

Q1.4.3) What is the significance of source coding theorem?