EXPERIMENT NO. 8

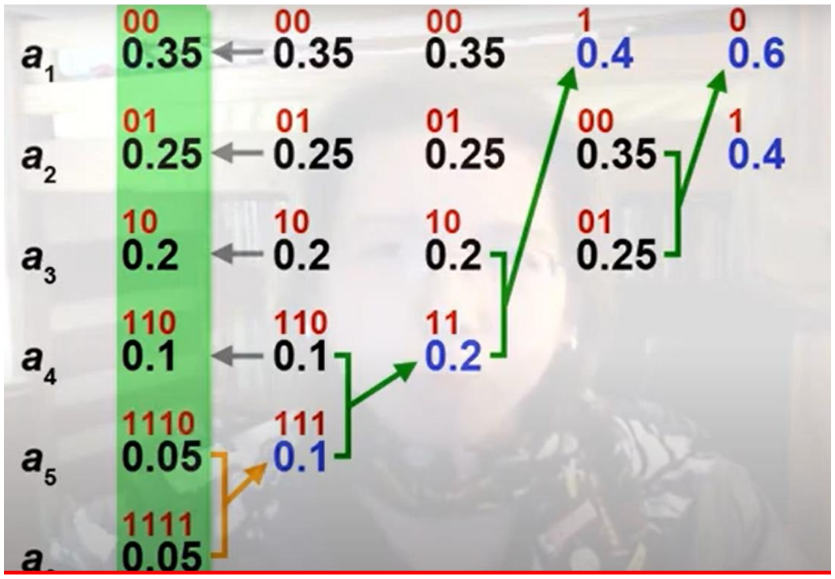
Aim: To understand Source Coding Theorem. Study of Huffman coding.

Objective: Study of Huffman coding Encoder and Decoder.

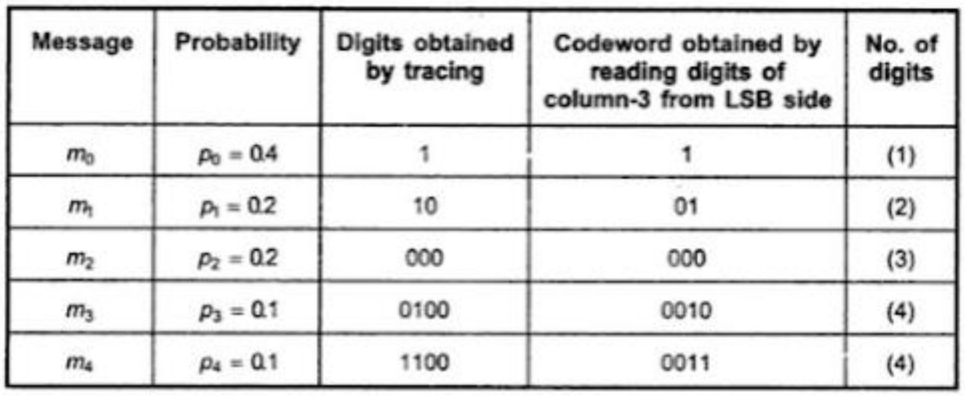
Software Used: MATLAB 7.6 software

Huffman coding is a lossless data compression algorithm. In this algorithm, a variable-length code is assigned to input different characters. The code length is related to how frequently characters are used. Most frequent characters have the smallest codes and longer codes for least frequent characters.

Huffman coding is an entropy coding algorithm used for lossless data compression, developed by David Huffman in 1952. The idea is to assign variable-length codes to input characters, the most frequent character gets the smallest code and the least frequent character gets the largest code. The Huffman algorithm is a so-called "greedy" approach to solving this problem in the sense that at each step, the algorithm chooses the best available option. It turns out that this is sufficient for finding the best encoding.



a_1 00 0.35 — 00 0.35 — 00 0.35 — 1 0.4 — 0 0.6
a_2 01 0.25 — 01 0.25 — 01 0.25 — 00 0.35 — 1 0.4
a_3 10 0.2 — 10 0.2 — 10 0.2 — 01 0.25
a_4 110 0.1 — 110 0.1 — 11 0.2
a_5 1110 0.05 — 111 0.1
a_6 1111 0.05



Message	Probability	Digits obtained by tracing	Codeword obtained by reading digits of column-3 from LSB side	No. of digits
m_0	$p_0 = 0.4$	1	1	(1)
m_1	$p_1 = 0.2$	10	01	(2)
m_2	$p_2 = 0.2$	000	000	(3)
m_3	$p_3 = 0.1$	0100	0010	(4)
m_4	$p_4 = 0.1$	1100	0011	(4)

Consider that the source generates five messages $m_0, m_1, \ldots m_4$. The probabilities of these messages are as shown in 2nd column of Table 1.8.3.

1.The messages are arranged according to their decreasing probabilities. For example m_3 and m_4 have lowest probabilities and hence they are put at the bottom ~~in~~ column of stage-I.

2.The two messages of lowest probabilities are assigned binary '0' and '1'.

3. The two lowest probabilities in stage-I are added. Observe that the sum of two probabilities is 0.1 + 0.1 = 0.2.

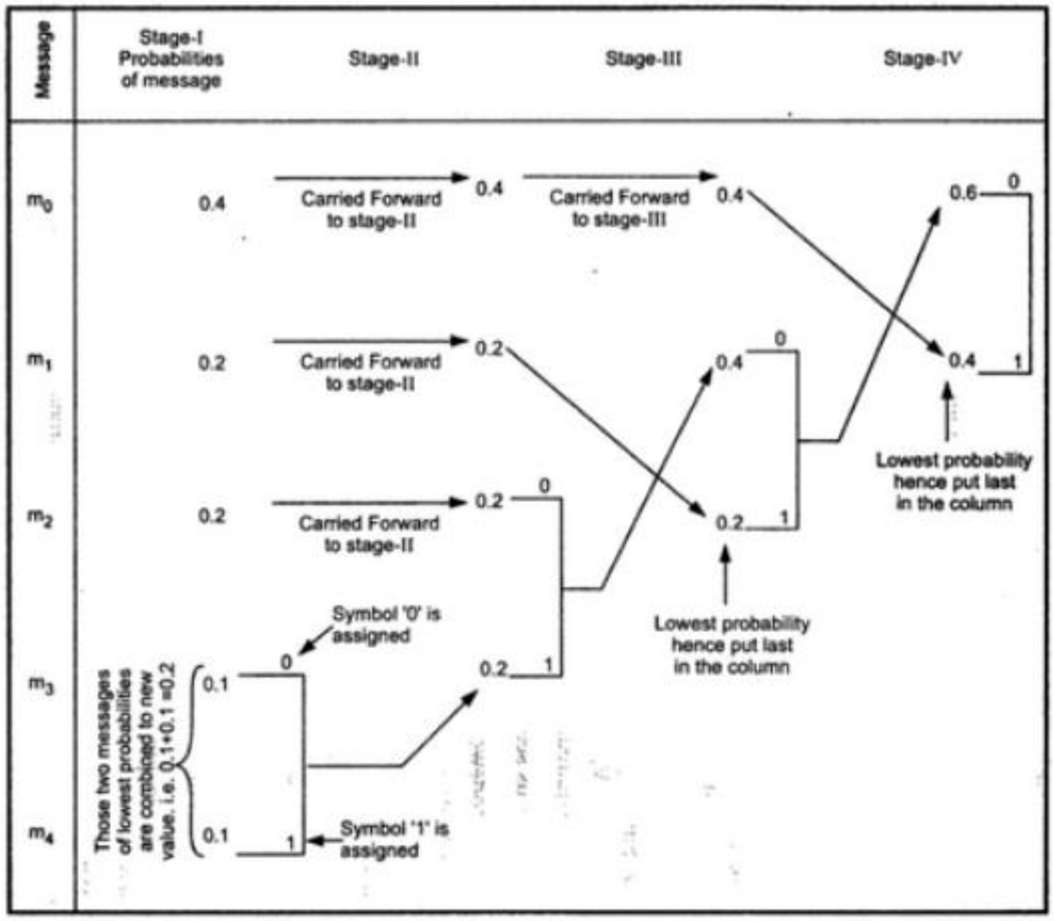
4. The sum of probabilities in stage-I is placed in stage-II such that the probabilities are in descending order. Observe that 0.2 is placed last in stage-II.

5. Now the last two probabilities are assigned '0' to '1' and they are added. Thus the sum of last two probabilities in stage-II is 0.2 + 0.2 = 0.4.

6. The sum of last two probabilities (i.e. 0.4) is placed in stage-III such that the probabilities are in descending order. Again '0' and '1' is assigned to the last two probabilities.

7. Similarly the values in stage-IV are obtained. Since there are only two values in stage-IV, these two values are assigned digits 0 and 1 and no further repetition is required.

Now let us see how the codewords for messages are obtained.


Message
Stage-I Probabilities of message
Stage-II
Stage-III
Stage-IV
m0
m1
m2
m3
m4
0.4
0.2
0.2
0.1
0.1
Carried Forward to stage-II
Carried Forward to stage-III
0.4
0.2
0.2
0.2
0.4
0.4
0.2
0.6
0.4
0
1
Symbol '0' is assigned
Symbol '1' is assigned
Those two messages of lowest probabilities are combined to new value. i.e. 0.1+0.1 =0.2
Lowest probability hence put last in the column

We know that equation 1.4.6 that average information per message (entropy) is given as,

$$H = \sum_{k=1}^{M} p_k \log_2\left(\frac{1}{p_k}\right)$$

For five messages above equation can be expanded as,

$$H = p_0 \log_2\left(\frac{1}{p_0}\right) + p_1 \log_2\left(\frac{1}{p_1}\right) + p_2 \log_2\left(\frac{1}{p_2}\right) + p_3 \log_2\left(\frac{1}{p_3}\right) + p_4 \log_2\left(\frac{1}{p_4}\right)$$

Here we started from $k = 0$. Putting values of probabilities in above equation from Table 1.8.5 we get,

$$\begin{aligned} H &= 0.4 \log_2\left(\frac{1}{0.4}\right) + 0.2 \log_2\left(\frac{1}{0.2}\right) + 0.2 \log_2\left(\frac{1}{0.2}\right) + 0.1 \log_2\left(\frac{1}{0.1}\right) + 0.1 \log_2\left(\frac{1}{0.1}\right) \\ &= 0.52877 + 0.46439 + 0.46439 + 0.33219 + 0.33219 \\ &= 2.12193 \text{ bits of information / message} \qquad \dots (1.8.8) \end{aligned}$$

Now let us calculate the average number of binary digits (binits) per message. Since each message is coded with different number of binits, we should use their probabilities to calculate average number of binary digits (binits) per message. It is calculated as follows :

$$\begin{aligned} \begin{matrix}\text{Average number of binary}\\ \text{digits per message}\end{matrix} &= \sum \begin{pmatrix}\text{Probability of}\\ \text{message}\end{pmatrix} \times \begin{pmatrix}\text{No. of digits}\\ \text{in codeword}\end{pmatrix} \\ &= (0.4 \times 1) + (0.2 \times 2) + (0.2 \times 3) + (0.1 \times 4) + (0.1 \times 4) \\ &= 2.2 \text{ binary digits / message} \qquad \dots (1.8.9) \end{aligned}$$

Thus it is clear from equation 1.8.8 and equation 1.8.9 that Huffman coding assigns binary digits to each message such that Average number of binary digits per message are nearly equal to average bits of information per message (i.e. H). This means because of Huffman coding one binary digit carries almost one bit of information, which is the maximum information that can be conveyed by one digit.

Advantages of Huffman Coding

- Huffman coding is very efficient. It can compress data very quickly and effectively.
- Huffman coding is relatively simple to implement.
- Huffman coding can be used with any type of data.
- Huffman coding is very effective at compressing large files.

Flowchart:

Algorithm:

Program:

Input,

Output,

Result:

Questions:

Solve the following two numerical:

1. Apply Huffman Coding for the symbols [A E H N G S] generated by a DMS with probabilities [0.19,0.15,0.2,0.16,0.4,0.08] Also calculate coding efficiency.
2. Encode the following symbols using Huffman source coding technique and calculate coding efficiency [1/4,1/8,1/16, 1/16, 1/16,1/4,1/16,1/8]

Conclusion:

Timely Submission(10)	Journal Presentation(10)	Performance (10)	Understanding(10)	Oral(10)	Total (50)
Sub Teacher Sign					