Topic Huffman Coding Unit No. 01

- It was First developed by David Huffman in 1951
- It is data Compression technique in which size of data / message is reduced.
- Variable length Coding algorithm
- Reducing size of file / Message without losing any of the details

Huffman Coding

Suppose the string below is to be sent over a network.

B	C	A	A	D	D	D	C	C	A	C	A	C	A	C

Initial String

As we know that, Each character occupies 8 bits,
According to ASCII,

A = 65 = 01000001
B = 66 = 01000010
:
& so on,

As in given string total 15 strings so total 15 X 8 = 120 bits required to store / send data

Unit No. 01

Topic

We can send the above data by using fixed length coding

Let us consider the following string

ABBCDBCCDAABBEEEBEAB

In the above string total 20 characters

Each character required 8 bits to store (ASCII)

∴ Total Bits required = 20 × 8 = 160 bits

Now another method, fixed length coding

2 bits can represent 4 characters

ex. 00, 01, 10, 11 } 2 bits

& 3 bits can represent 8 characters

000, 001, ..., 111 } 4 bits

& so, on.

as in the given string only 5 different character i.e. A, B, C, D, E so we required 3 bits (8 combination)

Table:1

000	A
001	B
010	C
011	D
100	E
101 }	unused
110 }	
111 }	

∴ The string. ABBCDBCC DAABBEEEBE
AB

000 001 001 010 011 ---- & so on

So here for each character 3 bits required

Total no. of bits = 20X3 = 60 bits

Now for Decoding the message which is sent on the network, need required Table 1

so for sending the Table required 5 character (5X8 = 40 bits) & 3 bits for each character (3X5 = 15 bits)

Topic

Unit No.

∴ Total Number of bits Required for this method

Table 1 info

$= 20 \times 3 + 5 \times 8 + 5 \times 3$

(3 bits for each character) (5 different char A, B, C, D, E 8 bits each) + (3 bits for each char. Total 5 char)

Now Consider Variable length Coding

Huffman Coding

Here length of bits is not fixed for each character it is different

Now Consider the string

A B B C D B C C D A A B B E E E B E A B

Step 1 Write the frequency of each (count) character.

∴

char	Frequency / Count of Each char
A	4
B	7
C	3
D	2
E	4

Step 2

Now Design Huffman Tree.

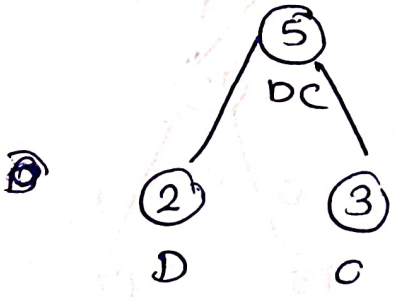
So here arrange all character in increasing order according to their frequency count

∴

~~char~~	D	C	E	A	B
	2	3	4	4	7

Step 3 Select Two least Frequency Count character. (Left side of Tree .. Lowest count value)

D	C	E	A	B
2	3	4	4	7

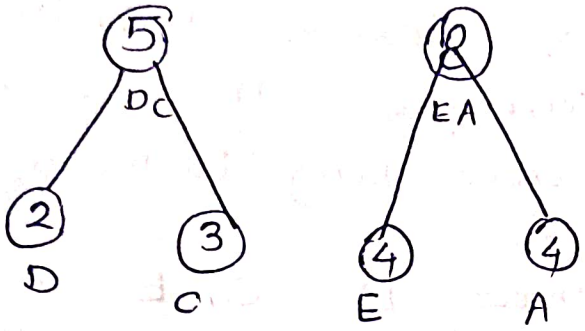


Now, arrange the count again

E	A	DC	B
4	4	5	7

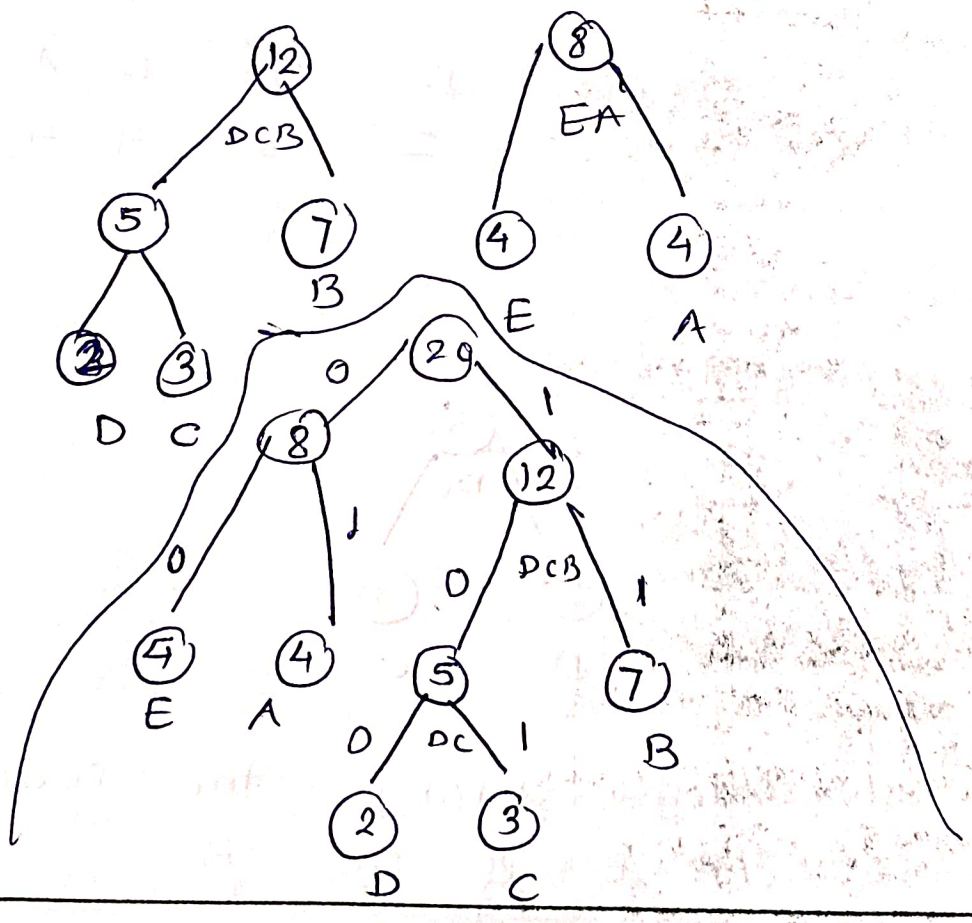
Topic

Unit No.



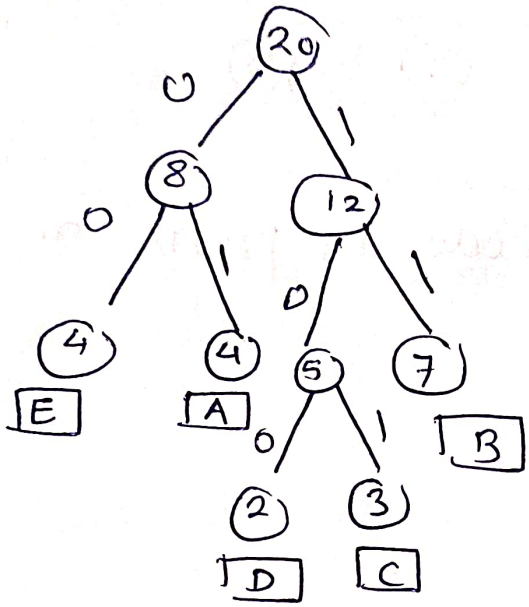
Now, Again Arrange the character Acc. to new frequency count

5 7 8
DC B EA



Topic

Unit No.



Now find the huffman coding for each node

Char	Huffman code	Frequency count	Total Bit for Each character = Freq count * Total Bit
A	= 01	4	4X2 = 8
B	= 11	7	7X2 = 14
C	= 101	3	3X3 = 9
D	= 100	2	2X3 = 6
E	= 00	4	4X2 = 8
		Total	= 45 bits

Also Table need to send & for Table required Total Bits.

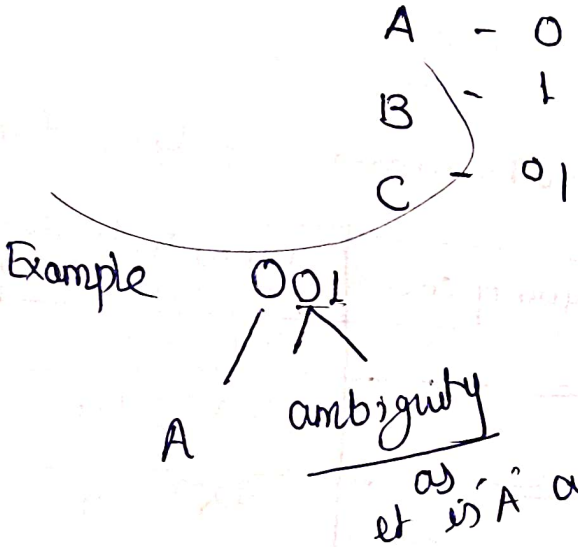
= 5(char) X 8 bit(Each char)
+ Total Bits 12 bits (All char)
= 40+12 = 52

Total Number of Bits = 52 + 45
= 97 bits

Time Complexity O(n log n)

Prefix Code: NO code is prefix of another Code.

for Example



001 ⇒ AAB

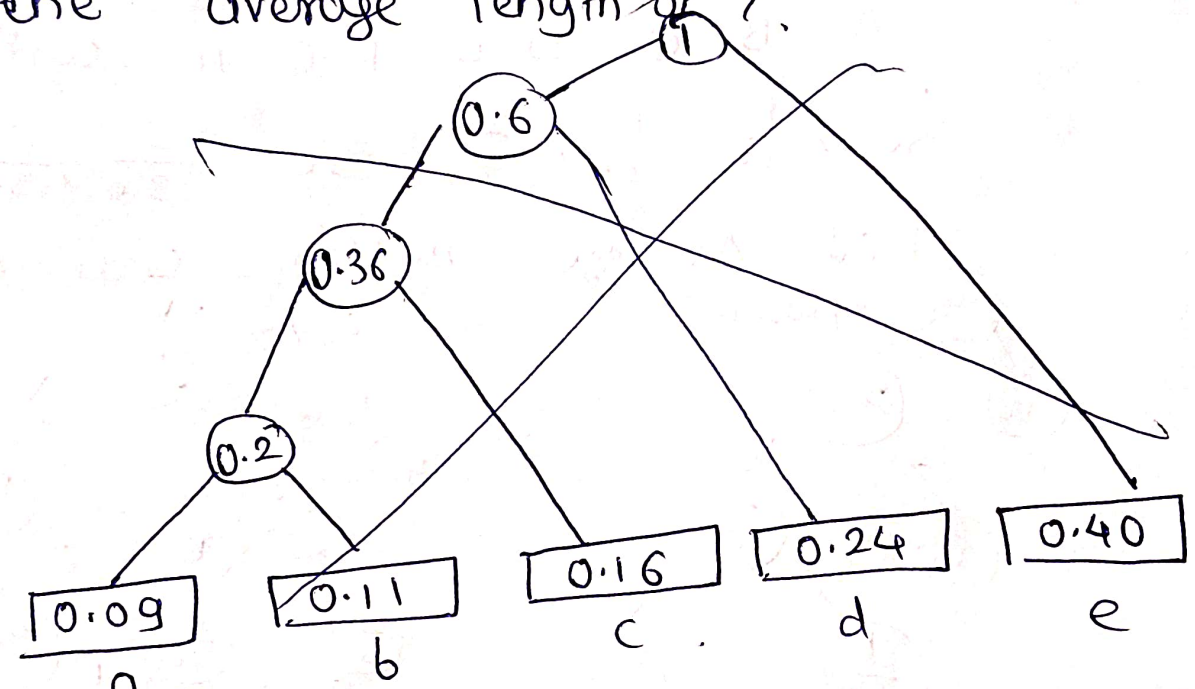
OR 001 ⇒ AC

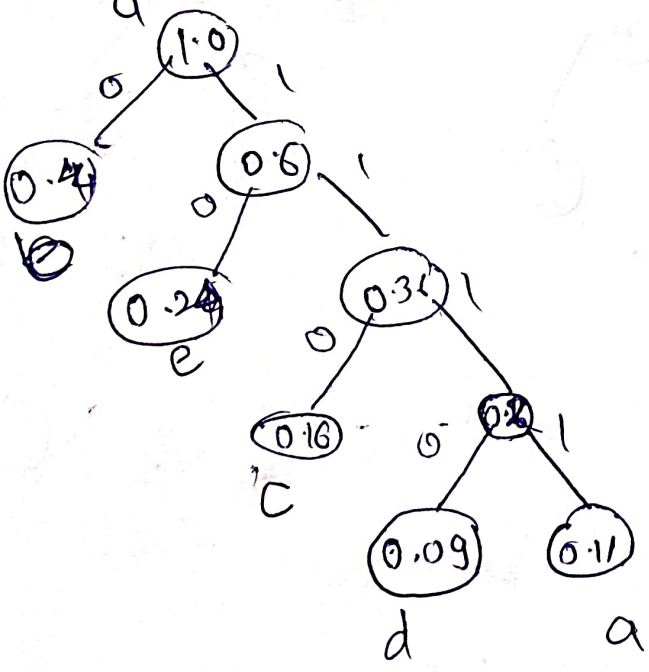
- Encoding follows the prefix rule
- most generated char will get the small code & least generated char will get the large code.
- Time Complexity O(n log n)

Topic

Unit No.

A Text is made up of the characters a, b, c, d, e each occuring with the probability 0.11, 0.40, 0.16, 0.09 & 0.24 resp. The optimal Huffman coding tech will have the average length of ?.





char	Probabilities	Huffman code
a	0.11	1111 = 4
b	0.40	0 = 1
c	0.16	110 = 3
d	0.09	1110 = 4
e	0.24	10 = 2

Total Bits = char * problem $= \sum_{i=1}^{n} (Pp * bi)$

Total Bits

Total Bits = $a \times 4 + b \times 1 + c \times 3 + d \times 4 + e \times 2$

$= 0.11 \times 4 + 0.40 \times 1 + 0.16 \times 3 + 0.09 \times 4 + 0.24 \times 2$

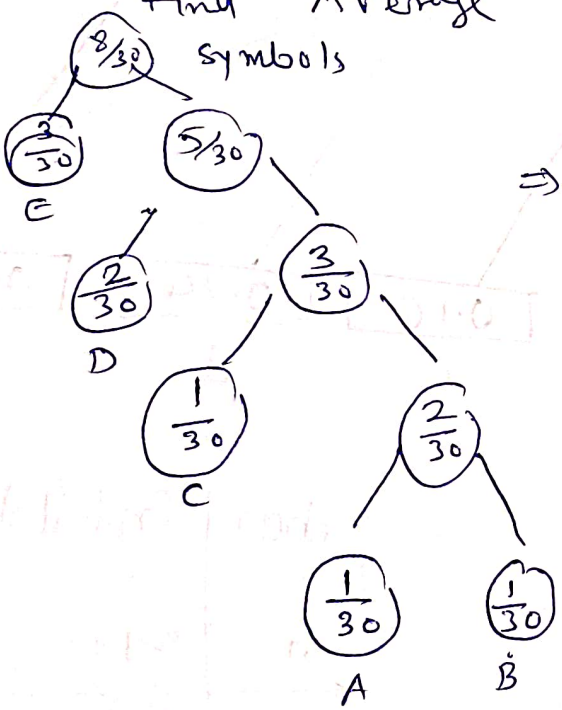
$= 0.44 + 0.40 + 0.48 + 0.36 + 0.48$

$= 2.16$ bits / symbol

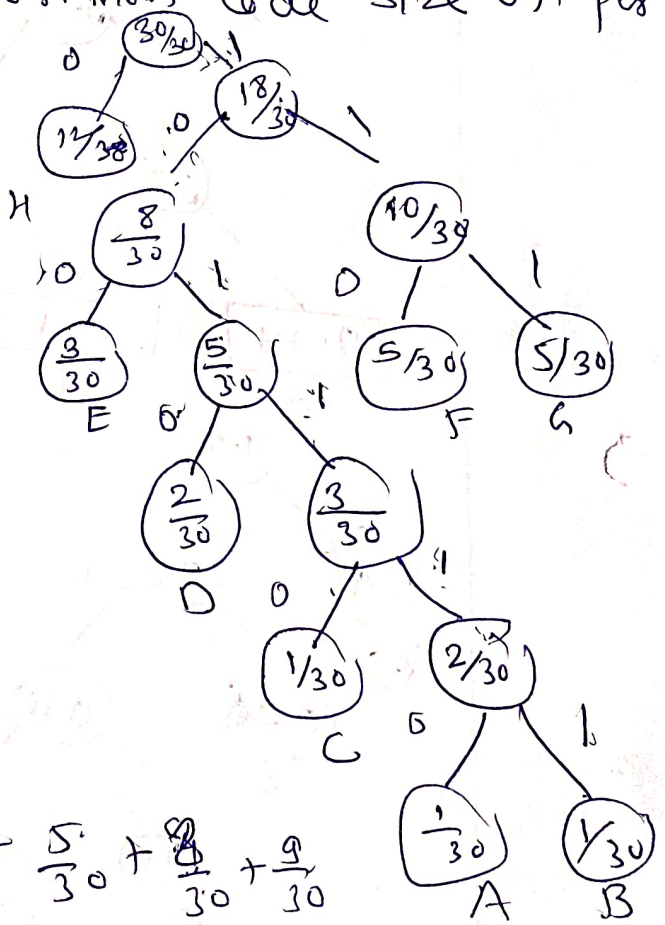
Ex ③

A B C D E F G H with probab $\frac{1}{30}, \frac{1}{30}, \frac{1}{30}, \frac{2}{30}, \frac{3}{30}, \frac{5}{30}, \frac{5}{30}, \frac{12}{30}$ resp.

Find Average Huffman code size bit per symbols



⇒



A

B

C H = 0

D

E

F

G

H 0

$\frac{6}{30} + \frac{6}{30} + \frac{5}{30} + \frac{8}{30} + \frac{9}{30} + \frac{15}{30} + \frac{15}{30} + \frac{12}{30}$

$= \frac{76}{30}$ bits / symbol

(1)