Huffman Coding:-

- Data Compression Algorithm.
- Reduce the size of the data.
- Real life example compressed the file to reduce the size of the file.
- Every message you sent and each message type go in the form of ASCII codes and ascii convert to binary of individual character.
- Example:-

A — 65 — 01000001 = 8 bits

↙ message ↓ Ascii ↓ Binary ↓ size

- On the transfer of simple message no compressed every character takes 8 bits. so your message consists of 20 characters the total space

$20 \times 8 = 160$ bits.

- This is known as fixed length coding in which each character take 8 bits of a memory.
- After Applying compression technique we can reduce the size of the message depends on the message may be it comes to 100, 110, 90, etc.
- If we move sometimes back, we learn binary tables we know $2^2 = 4$ (combinations to store different character).

$2^3 = 8$ (combination " " " ")

$2^8 = 256$(" " " " ")

My message is ABBCDBCCDAABBEEFBEAB = 20.

- On my message 5 different characters (A, B, C, D, E), no need to go on 256 combinations.
- $2^{3 \to bits} = 8$ combinations (use 3 bits).

- Rather than taking 8-bits we use 3-bits combination to represent character.

0	0	0	– A
0	0	1	– B
0	1	0	– C
0	1	1	– D
0	0	0	– E
0	0	1	[unsed.
0	1	0	
0	1	1	

- We use 000 instead of A and same for B, C, D and E.
- Total bits are 20 x 3 = 60 bits.
- In this message the user don't know what is the meaning of 000 so for this we sent the table with the message.
- So, the table also take memory in this we have 8 rows and in 5 rows we have some character. The character take 8 bits.

  8x5 = 40 bits.

  Remaining the binary of these 5 character take 3 bits.

  5x3=15 bits.

  Total no. of bits to send this message is

  60+55 = 115 bits.

- This is also known as fixed length coding.
- Huffman coding is known as variable length coding.
- In Huffman we declare the variables like 1, 2, 3 that in not fixed.

On Huffman coding design a tree

char	Frequency / count	
A	4	01
B	7	11
C	3	101
D	2	100
E	4	00

- Draw with the increase of count. (Priority)

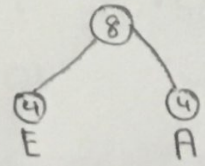
  2 D 3 C 4 E 4 A 7 B

- Take first two character in the priority queue and merge the count.

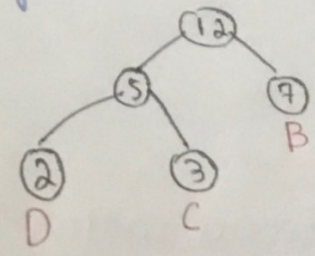
- Now draw the again with increase of count.

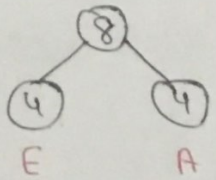
  4 E 4 A 5 DC 7 B

- Now again least two in priority queue.

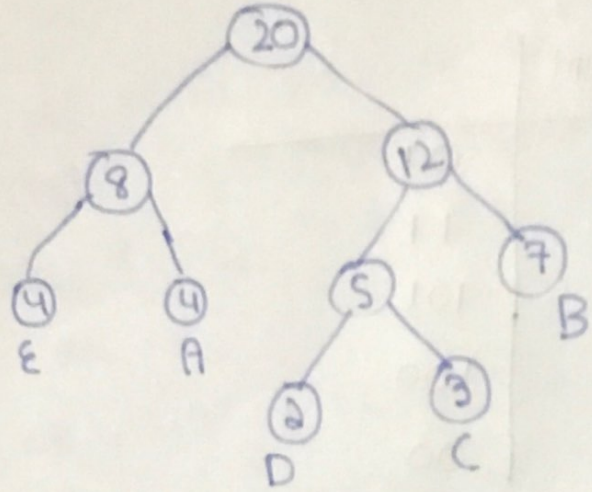


- Again do the same step.



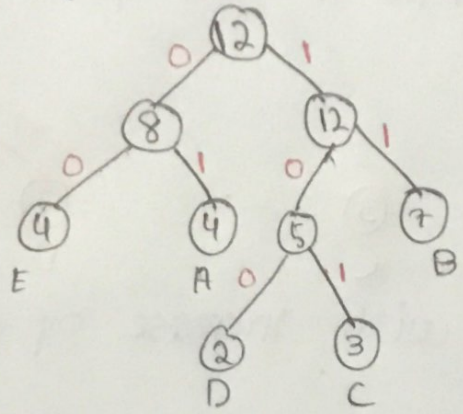


- Now we have only 8 and 12 minimum frequency always on the left side and greater on right. So, the tree become.



→ Final Tree.
→ character always on leaf

- Now in step 2 we assign variables to each edge left = 0 , right = 1.



- Now calculate the Huffman coding of each character from root node to that node.

A = 01

B = 11

C = 101

D = 100

E = 00

→ To calculate total no. of bits in Huffman coding is multipy the frequesy/count with the no. of binary.

= 4 x 2 + 7 x 2 + 3 x 3 + 2 x 3 + 4 x 2

= 45 bits.

- same problem the reciever can't understand what is 01, 11.
- so, for this we send table or may be tree.
- The table consists of five characters and each character take 8-bits. and the binary of character the sum of all.

$5 \times 8 = 40 + 12 = 52.$

Total = $52 + 45 = \boxed{97 \text{ bits.}}$

- Reciever Decode the tree using traversal technique.

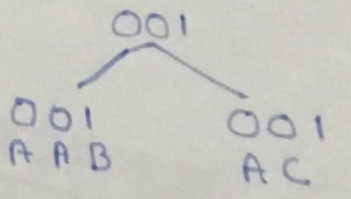
Time Complexity:

$\Theta(n \log n)$

Note:-

NO code is prefix of another code.

Example:- A – 0
B – 1
C – 01



- Two possiblilites reciever can't understand which one is correct.
- Huffman coding follow the prefix Rule.

Huffman Coding:-

- David Huffman in 1951
- encoding follow the prefix Rule
- Most generated char. will get the small code & least generated char. will get the large code.
- Time complexity $\Theta(n \log n)$