HUFFMAN CODING TECHNIQUE:

In computer science, all data is encoded as strings of ones and zeros. The goal of information theory is to convey information in the smallest possible number of bits while ensuring the clarity of each encoding, and to achieve this goal we will use the Huffman encoding method.

ALGORITHM:

Input: A string with different characters.

Output: The codes for each individual characters.

Code idea:

Begin

define a node with character, frequency, left and right child of the node for Huffman tree.

create a list ‘freq’ to store frequency of each character, initially, all are 0

for each character c in the string do

increase the frequency for character ch in freq list.

done

for all type of character ch do

if the frequency of ch is non zero then

add ch and its frequency as a node of priority queue Q.

done

while Q is not empty do

remove item from Q and assign it to left child of node

remove item from Q and assign to the right child of node

traverse the node to find the assigned code

done

End

EXPLAINING PROCESS WITH A EXAMPLE:

An example of how Huffman coding techniques can help reduce the size of a message:

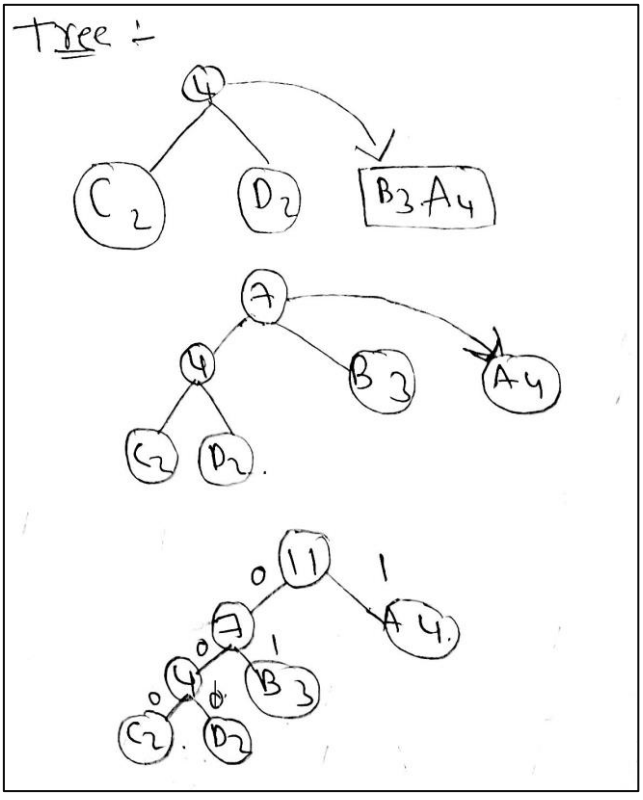
The specified text is "BBCACABADDA",

When sending a message using 8-bit ASCII code, the size of the message is 11 * 8 bits = 88 bit

CHARACTERS	FREQUENCY
A	4
B	3
C	2
D	2

TREE REPRESENTATION OF MESSAGE:

First, the basic rule for building a tree for Huffman: You must first connect the two least frequent nodes, then the previous tree to the next least frequent node. Therefore, the tree must contain all nodes of the message.



The least-repeated nodes in our message are C and D, and their frequency (number of times presented in the message) is 2, so they are both linked first, and the size of this message in that linked tree should be It becomes the root node in the first tree of frequency C, and D is equal to 2.2, so the size is 4 (2 + 2). In the second tree, we connect B to the previous tree because B is the node with the next lowest frequency. And the size of the root node is now 7 because the previous size was 4 and B was added. B has a frequency of 3, so the root node has a value of 7 (4 + 3). And in the third tree we connect A to the previous tree. This is because A is the node containing the next lowest frequency. And the root node size is now 11 because the previous node was 7 and A was added, and the frequency is 4, so the root node value is 11 (7 + 4).

And after building the tree the right nodes edges are valued as “1” and left node edges are valued as “0” and for getting the code of each node we will traverse from root node to node containing the character the and values of that edges will the code for that corresponding character.

Example: To get the character code C, in the root node, the code for C would be 000 and the code for A would be 1. Because only one edge goes through and this edge has a value of " 1".

CHARACTERS	FREQUENCY	CODE	MESSAGE SIZE AFTER USING ALGORITHM
A	4	1	4*1 = 4
B	3	01	3*2 = 6
C	2	000	2*3 = 6
D	2	001	2*3 = 6
4*8 = 32		9	22

There are 4 characters so if you want to check if this particular character should send ASCII code 32(8 * 4)

The receiver should know the huffman generated code also because this help the user to decode the message

So now our message will be translated into the binary code generated by this technique and if we send the table to the receiver then it will be easy to decode the binary code for the receiver.

Basically the user can know only the ascii codes only because it is well known for everyone but, the size of message is 88 bits. So, we should send our table generated after Huffman technique so that the codes generated by our self can be understood to the user and the size of the message now is 22 bits and as we mentioned above we should the table too, size of table is 41 bits. So, now totally the message size is decreased to 63 bits. In this way the text in compressed using this Huffman coding.

BY,

N.MONEESH

AM.EN.U4AIE20150