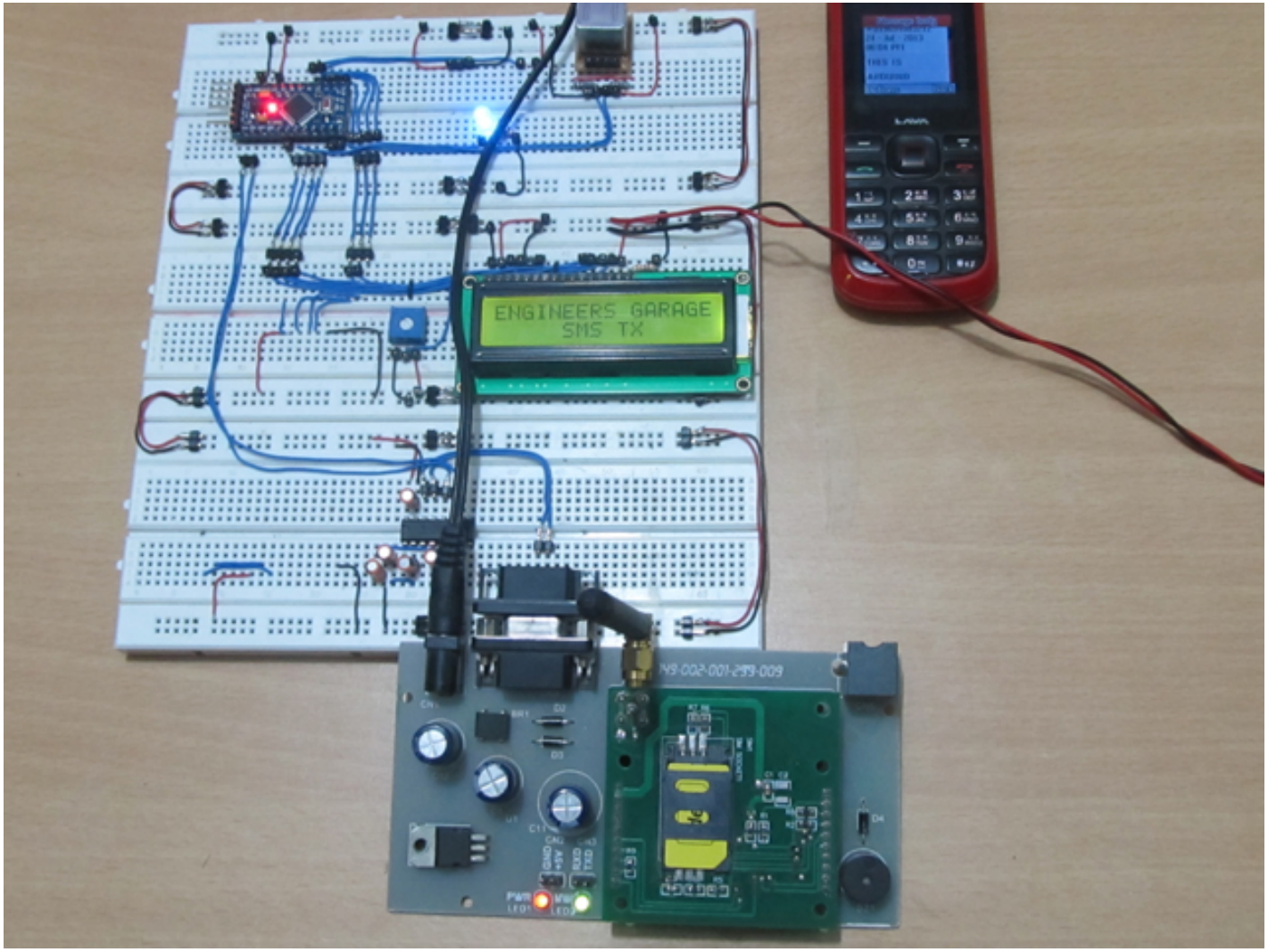
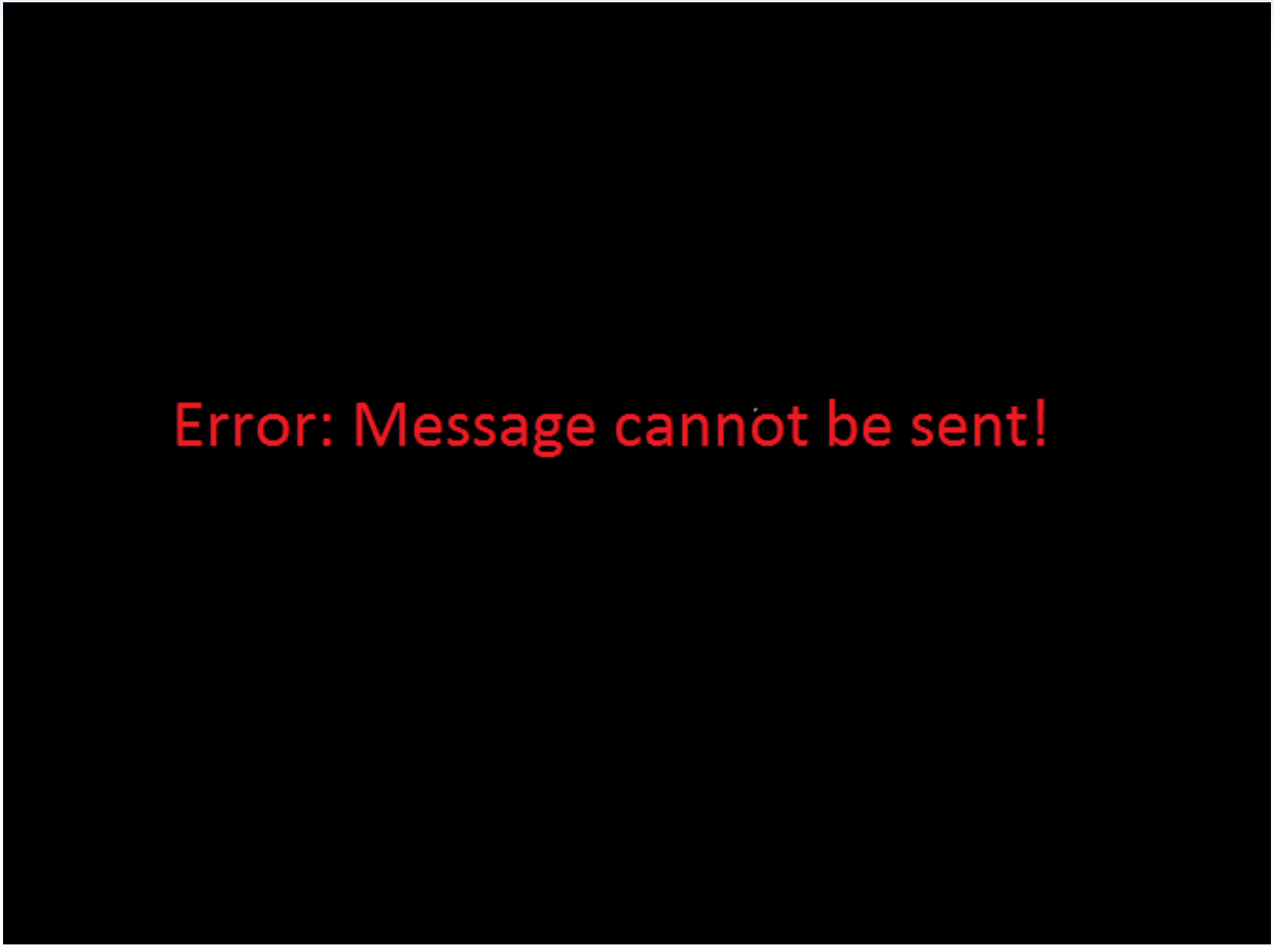

HUFFMAN
CODING
COMPRESSION ALGORITHM

Huffman Coding

Mohimenul Karim
And
Ankur Lahiry

May 23, 2015


ENGINEERS GARAGE
SMS TX


Error: Message cannot be sent!

How To Solve This Problem???

How To Solve This Problem???

Can we minimize the number of bits for representing the character?

How To Solve This Problem???

Can we minimize the number of bits for representing the character?

Yes

How To Solve This Problem???

Can we minimize the number of bits for representing the character?

Yes

How.....???

How To Solve This Problem???

Can we minimize the number of bits for representing the character?

Yes

How.....???

We can solve this problem using **Huffman Coding**

Principle of Huffman Coding

Principle of Huffman Coding

- **Huffman Coding** works in **Greedy Approach**

Principle of Huffman Coding

- **Huffman Coding** works in **Greedy Approach**
- It focuses on **frequency** of each character appearing in the text to construct an optimal binary representation of the characters.

Greedy Approach

- Greedy Algorithm always makes the choice that looks best at the moment

Greedy Approach

- Greedy Algorithm always makes the choice that looks best at the moment
- It makes a locally optimal choice for achieving optimal solution

Greedy Approach

- Greedy Algorithm always makes the choice that looks best at the moment
- It makes a locally optimal choice for achieving optimal solution
- Example: 'Activity Selection Problem' , 'Coin Changing Problem' , 'Fractional Knapsack Problem' etc

Retriving Lowest Frequency Of Characters

Retriving Lowest Frequency Of Characters

To implement this technique Huffman Algorithm uses Priority Queue

Retriving Lowest Frequency Of Characters

To implement this technique Huffman Algorithm uses Priority Queue

Concepts Of Priority Queue

- Is an abstract data type like array where additionally each element has a "priority" associated with it.
- An element with high priority is served before an element with low priority
- If two elements have the same priority, they are served according to their order in the queue.

Back To Huffman Coding

- Is a procedure to generate a binary code tree
- this was invented by David Huffman in 1952

Huffman Coding Algorithm

- Initializzation: put all symbols on a list sorted according to their frequency counts
- Repeat until the list has only one symbol left:
  a) From the list pick two symbols with the lowest frequency counts
  b) From a Huffman sub-tree that has these two symbols as child nodes and create a parent node
  c) Assign the sum of the children's frequency counts to the parent and insert it into the list such that the order is maintained
  d) Delete the children from the list
- Assign A codeword for each leaf based on the path from the root

Continued....

```
n = |C|
Q = C
for i =1 to n-1:
        allocate a new node z
        z.left = x= EXTRACT-MIN(Q)
        z.right = y = EXTRACT-MIN(Q)
        z.freq = x.freq + y.freq
        Insert(Q,z)
return EXTRACT-MIN(Q)
```

Number of Bits

To encode a file , the total number of bits , B(T) = $\Sigma c.freq * d_t(c)$
Where T is the encoded tree
c belongs to the character set

Back To The MISSISSIPPI RIVER

Spliting the word

Back To The MISSISSIPPI RIVER

Spliting the word

M	I	S	P	R	V	E	””
1	5	4	2	2	1	1	1

Back To The MISSISSIPPI RIVER

Spliting the word

M	I	S	P	R	V	E	””
1	5	4	2	2	1	1	1

Decending Order of Numbers

I	S	P	R	M	V	E	””
5	4	2	2	1	1	1	1

Before Huffman Coding Tree:

Before Huffman Coding Tree:

5∗8+4∗8+2∗8+2∗8+1∗8+1∗8+1∗8+1∗8=136

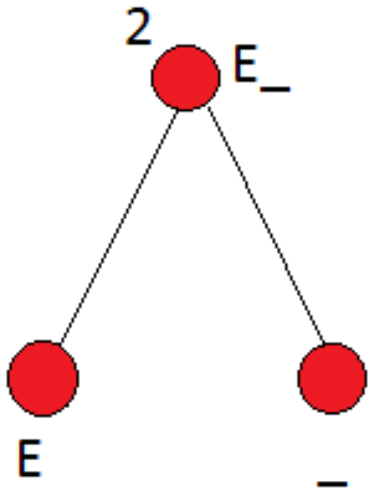
Before Huffman Coding Tree:

5∗8+4∗8+2∗8+2∗8+1∗8+1∗8+1∗8+1∗8=136

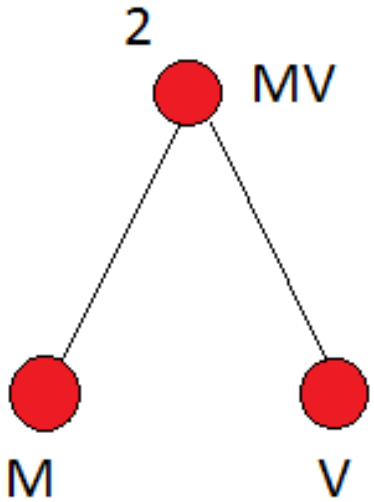
Huge!!!!!

Graphical Overview Of Huffman Coding

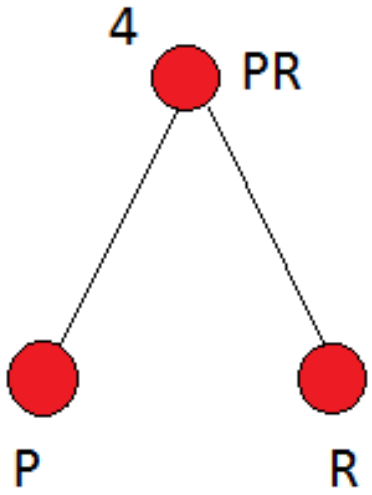
Graphical Overview Of Huffman Coding



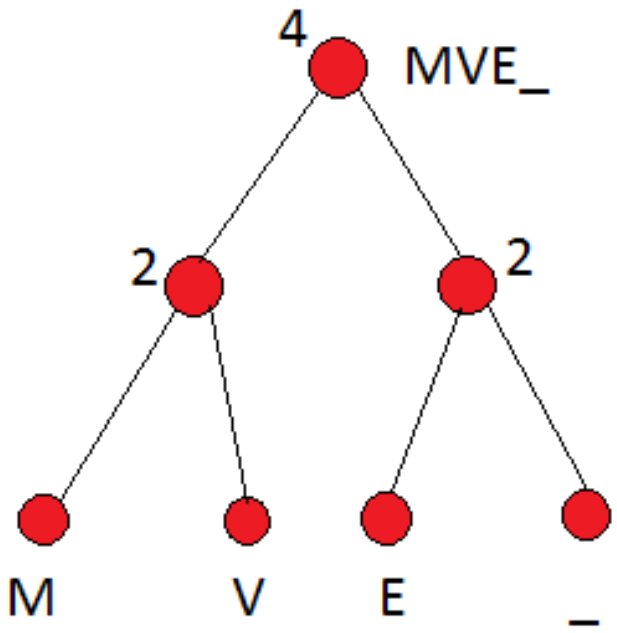
Graphical Overview Of Huffman Coding



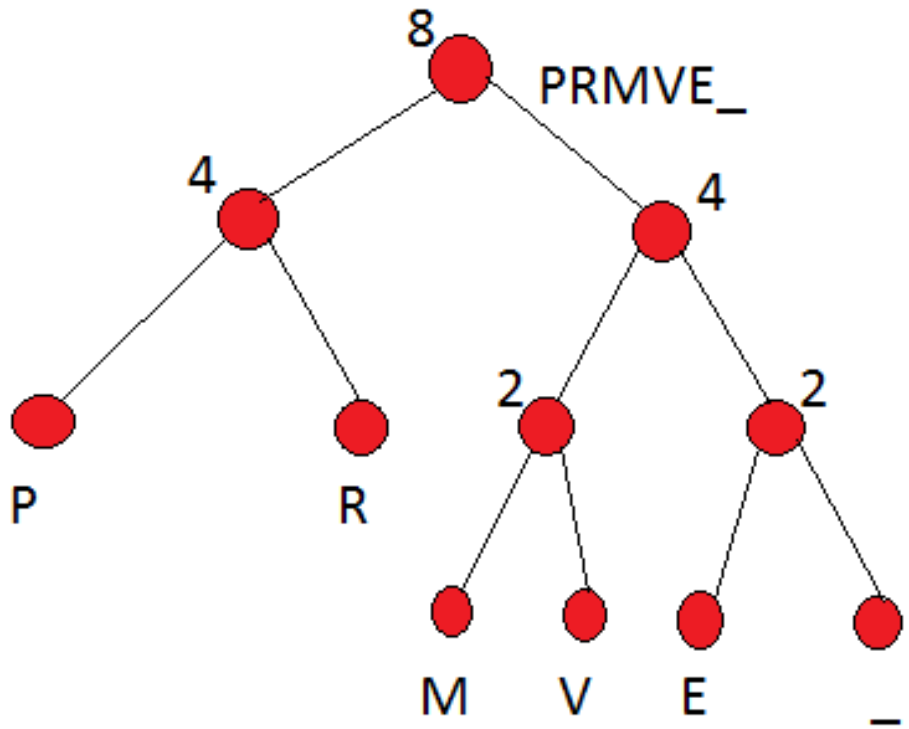
Graphical Overview Of Huffman Coding



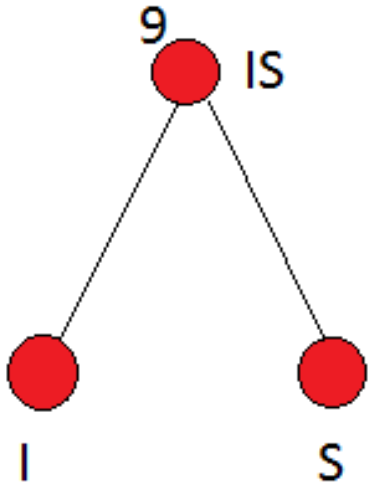
Graphical Overview Of Huffman Coding



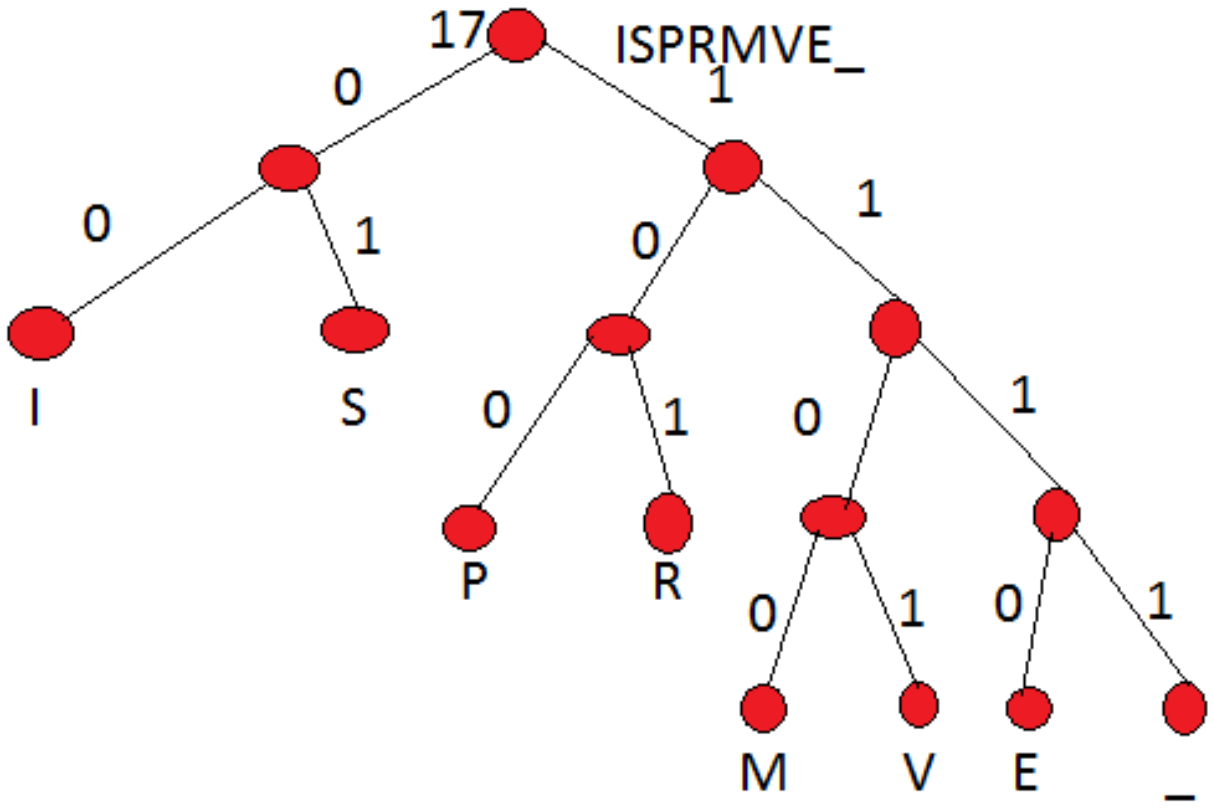
Graphical Overview Of Huffman Coding



Graphical Overview Of Huffman Coding



Graphical Overview Of Huffman Coding



Finally ....

Finally ....

I	S	P	R	M	V	E	””
00	01	100	101	1100	1101	1110	1111

After Huffman Coding Tree

After Huffman Coding Tree

Number of bits=
$5*2+4*2+2*3+2*3+4*1+4*1+4*1+4*1=46$

After Huffman Coding Tree

Number of bits=
$5*2+4*2+2*3+2*3+4*1+4*1+4*1+4*1=46$
Improvement: 66.2%

Complexity

- time complexity is nlogn
- can be improved by using Van Embde Boss

Usefullness

Usefullness

- Easy To decode

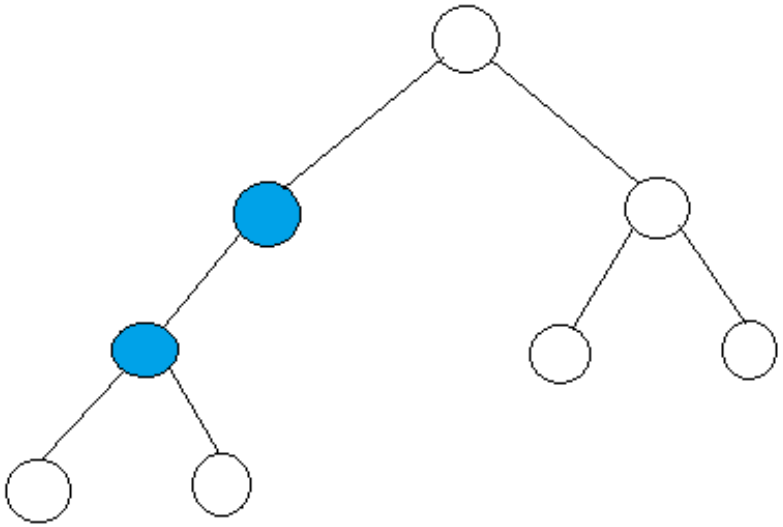
Usefullness

- Easy To decode
- Prefix free code

Usefullness

- Easy To decode
- Prefix free code
- The tree generated by Huffman Code is a Full Binary Tree

Full Binary Tree




Message sent!

Resources

Introduction To Algorithms by Thomas H. Cormen , Charles E. Leiserson , Ronald L.Rivest, Clifford Stein.