For complete proof we have to go through two steps.

(1.) let, $C \equiv$ alphabet

char $c \in C$ where frequency $= c.freq$

$c \equiv$ each char

Let. $x \in C$ & $y \in C$

x, y having the lowest frequency.

Then, we say, there exist an optimal prefix code for C in which x & y have the same length ~~an~~ code differ by only in the last bit. ——— (1)

proof of (1): Let's take a tree T that representing an arbitrary optimal prefix code and modify it to make a tree representing another optimal tree.

In new tree, x & y should have the max depth & they are sibling.

If we can construct such new tree proof of statement (1) will complete.

Let, a, b are sibling and their depth is max. in T.

let a.freq ≤ b.freq

& x.freq ≤ y.freq.

then, we get. x.freq ≤ a.freq.

and, y.freq ≤ b.freq.

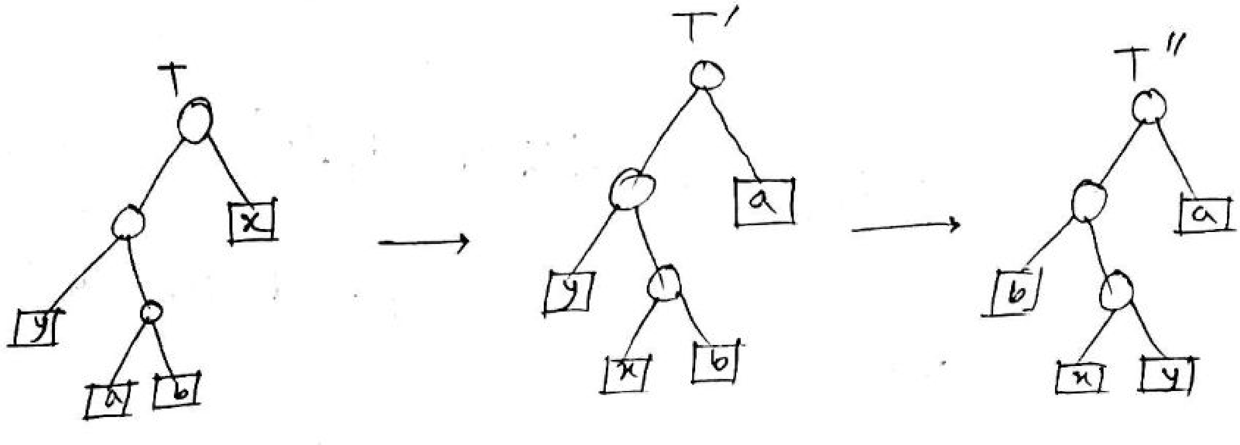
as, x, y have smallest frequency.

if we have, $x.freq = y.freq = a.freq = b.freq$

then statement (2) would be true trivially.

We will assume that, $x.freq \neq b.freq$

so $(x \neq b)$



In tree T'' x & y have the max depth and they are sibling.

Scanned by CamScanner

Let's Calculate the cost difference

$$B(T) - B(T') = \sum_{c \in C} c.freq \cdot d_T(c) - \sum_{c \in C} c.freq \cdot d_{T'}(c)$$

$$= x.freq \cdot d_T(x) + a.freq \cdot d_T(a) - x.freq \cdot d_{T'}(x) - a.freq \cdot d_{T'}(a)$$

$$= (a.freq - x.freq)(d_T(a) - d_T(x))$$

$$\geq 0. \qquad [\because d_{T'}(x) = d_T(a)]$$

$$[\because (a.freq - x.freq) \geq 0 \;\&\; (d_T(a) - d_T(x)) \geq 0]$$

as, $a.freq - x.freq \geq 0$

so, $x.freq$ is minimum frequency leaf.

$d_T(a) - d_T(x) \geq 0$

so, ~~x~~ a has the max depth in T.

Again if we exchange y & b the cost won't be increased. so, $B(T') - B(T'') \geq 0$

$$\Rightarrow B(T'') \leq B(T)$$

as T is optimal $B(T) \leq B(T'')$

Therefore, $B(T) = B(T'')$

Thus T'' is an optimal tree.

Step 2: Let, C = Alphabet.

each char $c \in C$. has frequency c.freq

Let, x, y be two char ~~whi~~ with min freq

Let, C' is another alphabet whic can be found by removing x & y from C and adding new char z.

so, $C' = C - \{x, y\} \cup \{z\}$

$z.freq = x.freq + y.freq$.

T' ~~b~~ is a tree representing optimal code for C'.

T is obtained from T' by replacing the leaf node for z with ~~the~~ an internal node, x and y as children, represents ~~the~~ an optimal prefix code for C. —— (11)

proof of S(ii) :

For each $c \in C - \{x, y\}$ we have $d_T(c) = d_{T'}(c)$;

$c.freq \cdot d_T(c) = c.freq \cdot d_{T'}(c)$.

$d_T(x) = d_T(y) = d_{T'}(z) + 1$

we have,

$$x.freq \cdot d_T(x) + y.freq \cdot d_T(y) =$$

$$(x.freq + y.freq)(d_{T'}(z) + 1)$$

$$= z.freq \cdot d_{T'}(z) + (x.freq + y.freq)$$

therefore, $B(T) = B(T') + (x.freq + y.freq)$

Assuming that T in not optimal. Then there exist an optimal tree T'' such that $B(T'') < B(T)$

T'' has x and y as sibling. Let T''' be T'' with the common parent of x and y replaced by z leaf. and $z.freq = x.freq + y.freq$

Then, $$B(T''') = B(T'') - (x.freq + y.freq)$$
$$< B(T) - (x.freq + y.freq)$$
$$= B(T')$$

it contradict with the previous assumption that T' is optimal for C'.

Thus T must represent an optimal prefix code for C.

Through this two steps we can conclude that Huffman produce an optimal prefix code that is huffman encode is optimal.

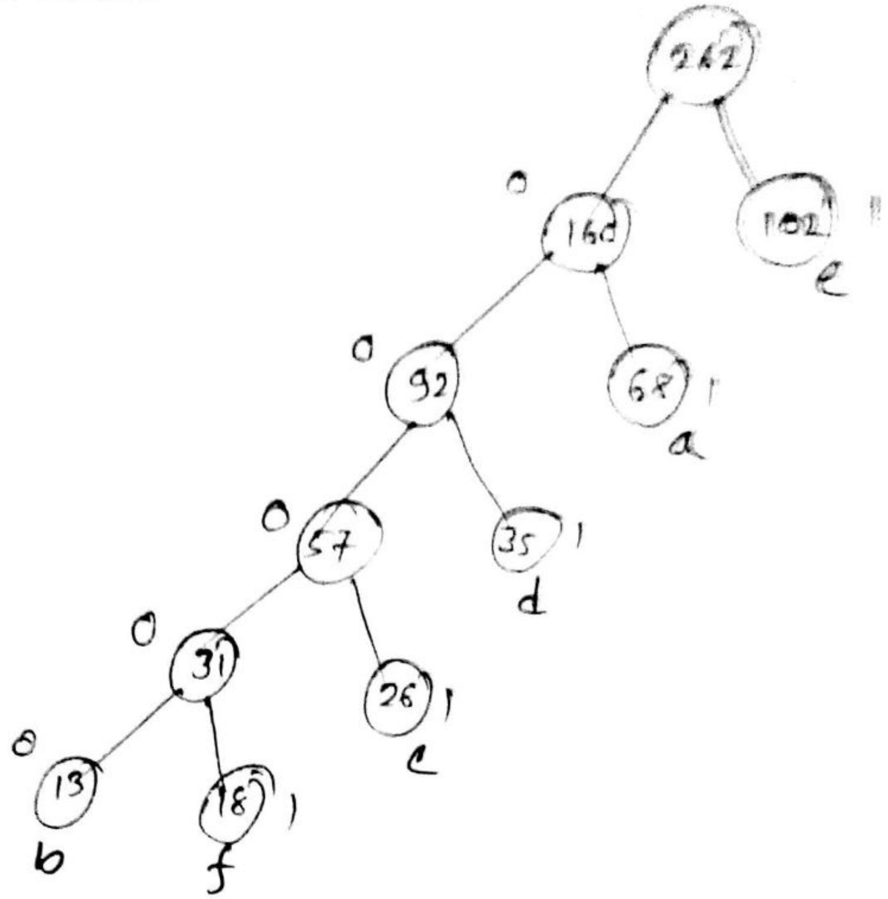
```
Struct Node {
        int freq;
        char c;
        Node left, right;
}
```

```
Huffman (n) {
    MeanHeap meap;

    for (i=0 to n-1) {
        meap. push (Node (freqi, ci))
    }
```

(P.T.O)

```
While( meap.size() >1) {

      New Node z :

      z.left =x=meap.top() ; meap.pop() ;

      z.right =y=meap.top() ;

      meap.pop() ;

       z.freq = x.freq + y.freq

       meap.push (z) ;

   }

   return  meap.top() ;

}
```

Trees:



Codes:

a → 01

b → 00000

c → 0001

d → 001

e → 1

f → 00001