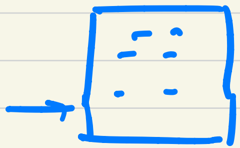
Lecture 08

Date: 25 Jan 2024

Huffman Code

Alice $\longrightarrow$ Bob



Alphabet : Set of letters.

a	b ...	c	d
000	001	010	011

Fixed length encoding

communication cost 33×3 bit

a	b	c	d
10	2	20	1.

a	b	c	d
00	01	10	11

66.

Variable length encoding

$a \to 10$ $\quad b \to 110$ $\quad c \to 0,$ $\quad d \to 111$ ←

$2 \times 10 + 2 \times 3 + 1 \times 20 + 3 \times 1 = 49$

$\underbrace{110}_{b}\,\underbrace{0}_{c}\,\underbrace{10}_{a}\,\underbrace{111}_{d}$

Example:

$a \to 0$ $\quad b \to 01$ $\quad c \to 001$

01001 not uniquely decipherable.

$bc \to bab.$

Unique Decipherability ← **want**

Example:-

$a = 0$

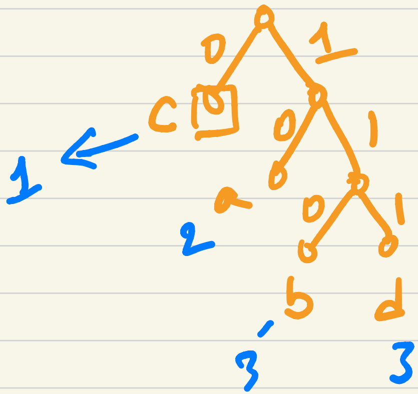
$b = 01$

$c = 011$

$01 \, \boxed{001011}$

Prefix code:- No codeword is a prefix of any other codeword.

$a \to \underbrace{10}$ $b \to 1\underbrace{10}$ $c \to 0$, $d \to 111$ $\leftarrow$



Prefix code $\longleftrightarrow$ Binary trees.

letter appear in the leaves.

Consider a prefix code given by a binary tree T.

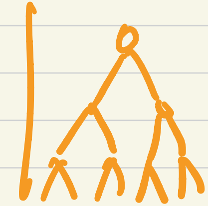
$A = \{a_1, \ldots, a_n\}$

$f[a_1], \ldots, f[a_n]$

cost of communication

$$\text{cost}(T) = \sum f[a_i] \cdot \text{depth}_T(a_i)$$

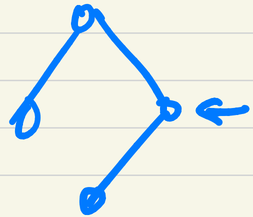
<u>Goal:</u> Find a binary tree with min cost.

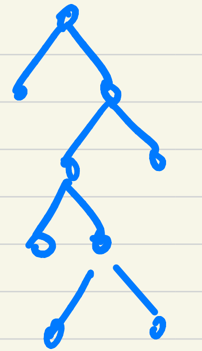


<u>Full binary Tree:-</u> A node can have

0 child

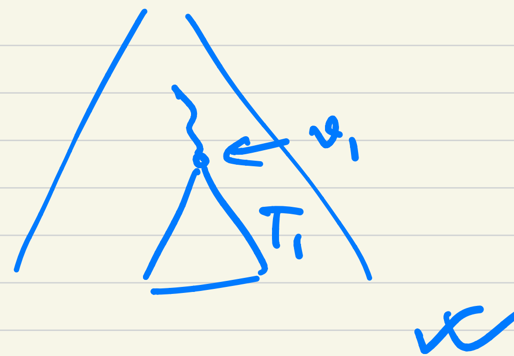
or 2 children





Observation: The optimal tree can be considered to be a full binary tree.

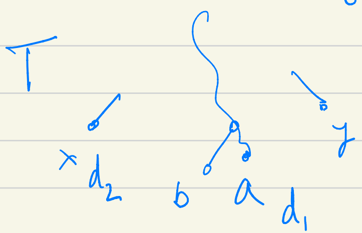
contract edge $\longrightarrow \{v_1, v_2\}$



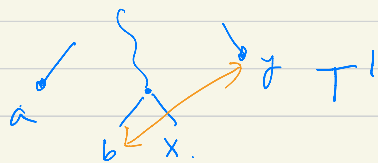
Lemma: Let C be an alphabet where each $c \in C$ has frequency $f[c]$. Let x & y be two characters having the lowest frequencies. Then there is an optimal prefix code for C in which the codewords for x & y have same length & they differ only in the last bit.

Proof:- Let T be an optimal tree. Let a be the letter with max depth. Let b be its sibling.

$f[x] \le f[y]$
$f[a] \le f[b]$ | w.l.g



$\longrightarrow$



$$\underbrace{cost(T)}_{\text{optimal}} - cost(T') = f[x]\cdot d_2 + f[a]\, d_1 - f[x]\cdot d_1 - f[a]\cdot d_2$$

$$= d_2\,(f[x] - f[a]) + d_1\,(f[a] - f[x])$$

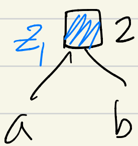
$$= -d_2\,(f[a] - f[x]) + d_1\,(f[a] - f[x])$$

$$= \underbrace{(d_1 - d_2)}_{\ge 0}\ \underbrace{(f[a] - f[x])}_{\ge 0}$$

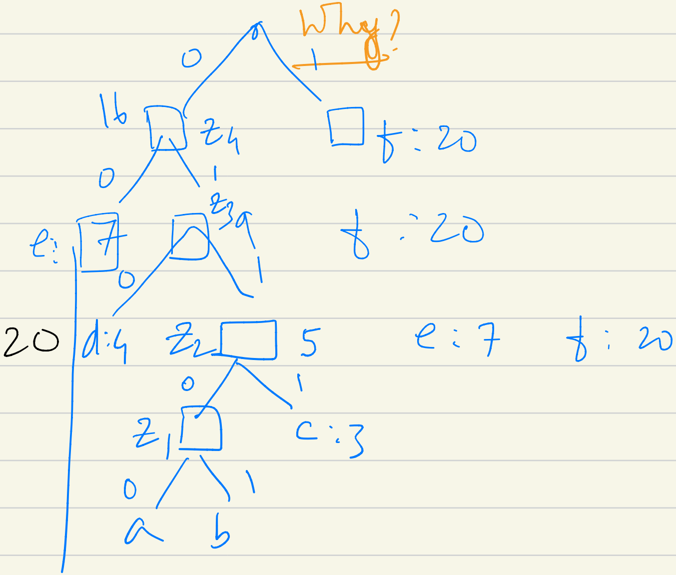
$cost(T) = cost(T')$
as T is optimal.

Next exchange $b \longleftrightarrow y$, note $\underbrace{f[b] \geq f[y]}_{\text{Why?}}$

a	b	c	d	e	f
1	1	3	4	7	20



z_1 2   c:3   d:4   e:7   f:20



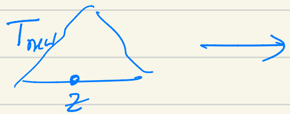
a: 01100

Lemma:- C - alphabet, $x, y \in C$ have lowest frequencies.

$C'_{new} = (C \setminus \{x, y\}) \cup \{z\}$, where z is a new symbol.

The frequency of all $c \in C'_{new}$ is $f[c]$ except for z, $f[z] = f[x] + f[y]$

Let T_{new} be an optimal tree for C'. T obtained from T_{new} by replacing z with an internal node having x & y as the children.



Thus T is optimal for C.

a : 4 b : 5 c : 6 d : 10 e : 12

a = 010 e ⇐ 11

```
n ← |C|
Q ← C          // Q is a min priority queue.
for i ← 1 to n-1
    do allocate a new node z
       left[z] ← x ← Extractmin(Q)
       right[z] ← y ← Extractmin Q.
       f[z] ← f[x] + f[y]
       Insert (Q, z)

return ExtractMin (Q)
```