Lecture 08

Date: 25 Jan 2024

Huffman Code

Alphabet : Set of letters.

a	b	c	d
000	001	010	011

Communication cost 33×3 bit

Fixed length encoding

a	b	c	d
10	2	20	1.

a	b	cd
00	01	1011

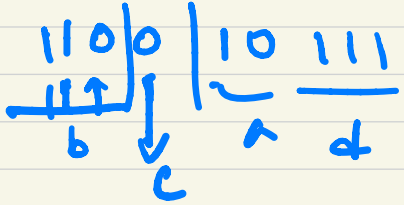
66.

Variable length encoding

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

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

Example:-



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

01001 not uniquely decipherable.

bc a → bad.

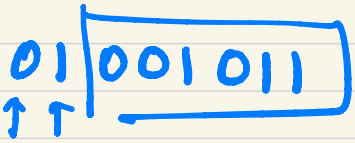
Unique decipherability ← want

Example:-

$a = 0$

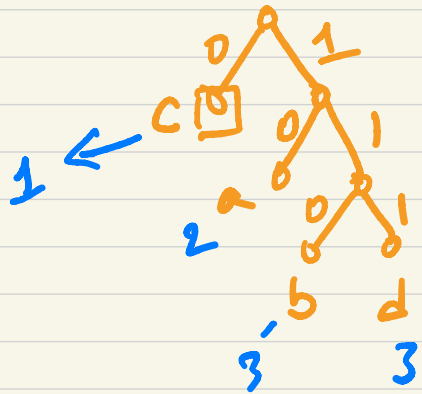
$b = 01$

$c = 011$



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

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



Prefix code ⟷ Binary trees.
letter appear in the leaves.

Consider a prefix code given by a binary tree T.

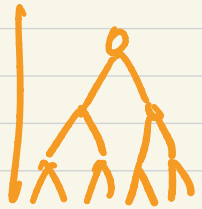
$A = \{a_1, \ldots, a_n\}$

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

Cost of communication

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

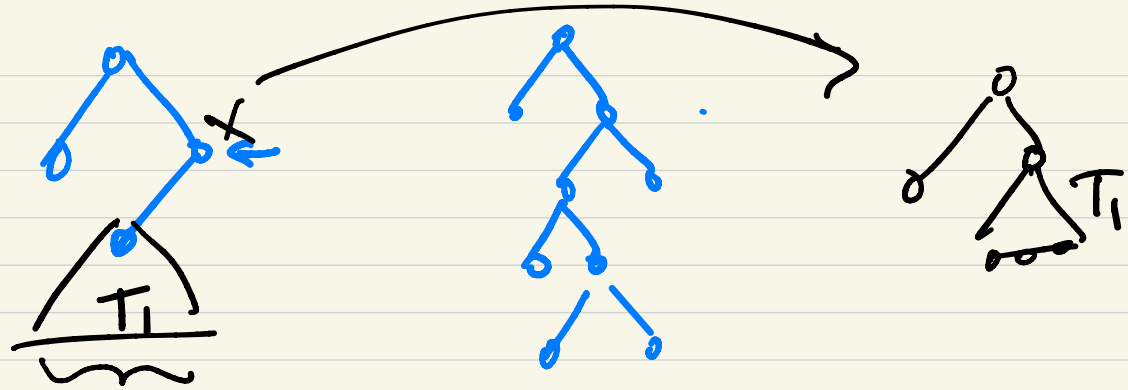
Goal: Find a binary tree with min cost.



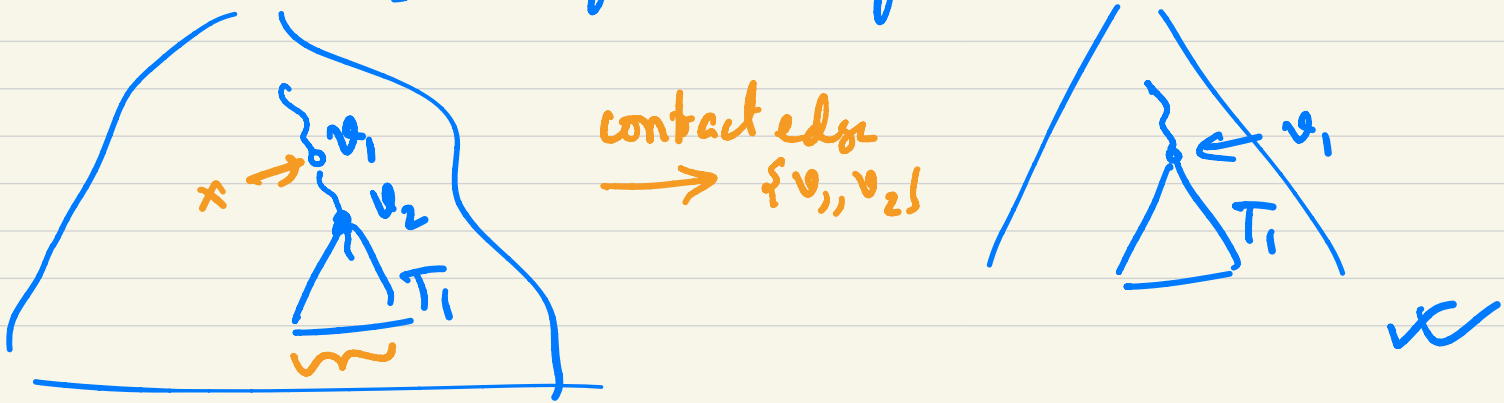
Full binary Tree:- A node can have

0 child

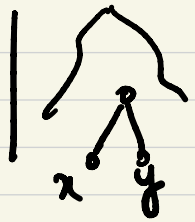
or 2 children



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

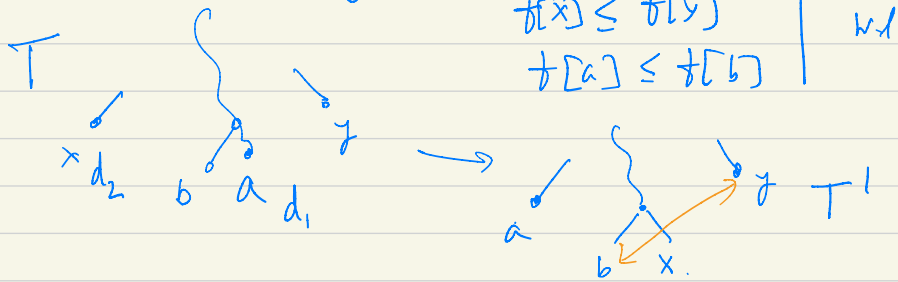


Lemma: Let C be an alphabet where each $c \in C$ has frequency $f[c]$. Let x & y be two characters having the lowest frequencies. Then there is an optimal prefix code for C in which the codewords for x & y have same length & they differ only in the last bit.



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



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

$\text{cost}(T)$: optimal

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

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

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

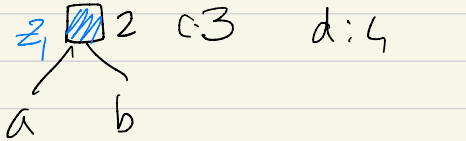
$(d_1 - d_2) \ge 0$, $(f[a] - f[x]) \ge 0$

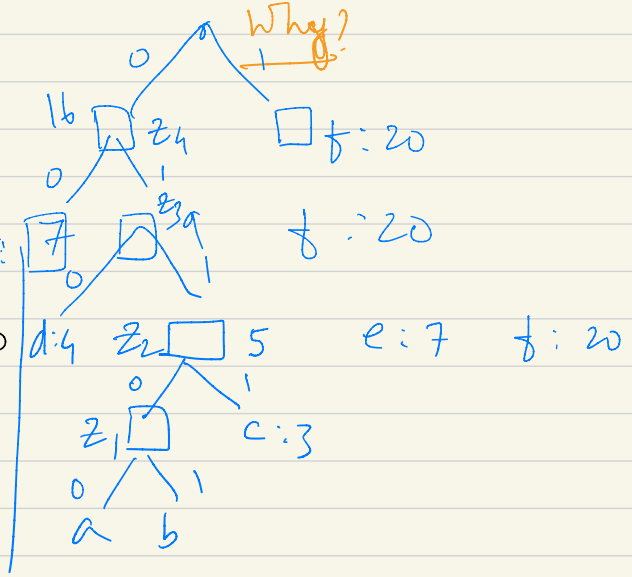
$\text{cost}(T) = \text{cost}(T')$ as T is optimal.

Next exchange $b \leftrightarrow y$, note $f[b] \geq f[y]$

Why?

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





a: 0 1 1 0 0

Lecture 09

Date: 30 Jan 2024

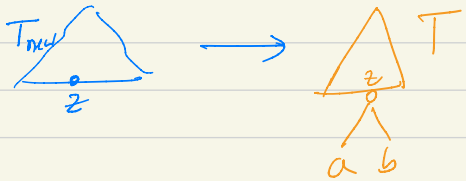
Lemma:- C alphabet, $x, y \in C$ have lowest frequencies.

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

Then T is optimal for C.



Proof:- Let T_0 be an optimal tree for C with x & y as siblings.

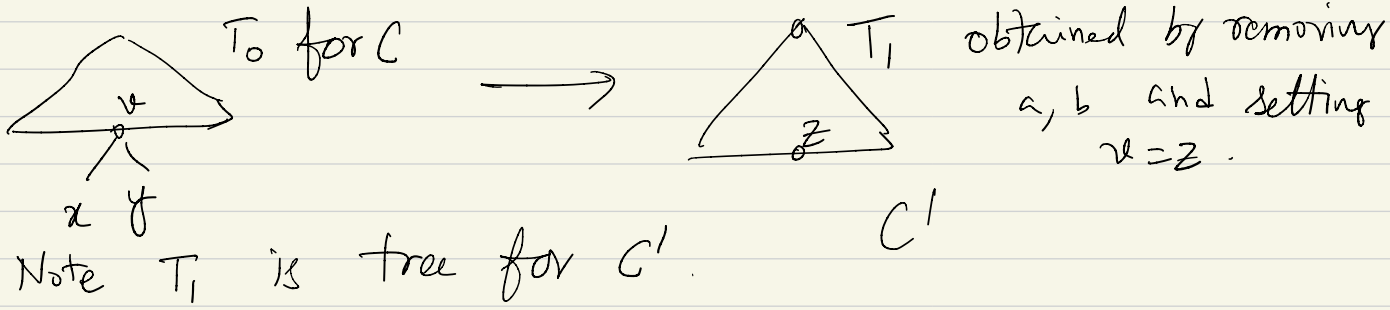
$cost(T_0) \leq cost(T)$ (since T_0 is optimal for C) — (1)

$$cost(T_{new}) - cost(T)$$

$$= depth_{T_{new}}(z) \cdot (f(x)+f(y)) - depth_T(x) f(x) - depth_T(y) f(y)$$

$$= -f(x) - f(y) \quad [\because depth_T(x) = depth_T(y)$$
$$depth_T(x) = depth_{T_{new}}(z) + 1]$$

$$\underline{cost(T_{new}) = cost(T) - f(x) - f(y)} \quad \text{— (2)}$$



Note T_1 is tree for C'.

$$\underline{\underline{cost(T_1)}} - \underline{cost(T_0)} = depth_{T_1}(z) f(z) - depth_{T_0}(x) f(x) - depth_{T_0}(y) f(y)$$

$$= -f(x) - f(y)$$

$$cost(T_1) = cost(T_0) - f(x) - f(y) \quad \text{— (3)}$$

$$cost(T_1) \geq cost(T_{new}) \quad - (4)$$

From (3)

$$cost(T_0) - \cancel{b(x)} - \cancel{b(y)} \geq cost(T) - \cancel{b(x)} - \cancel{b(y)}$$

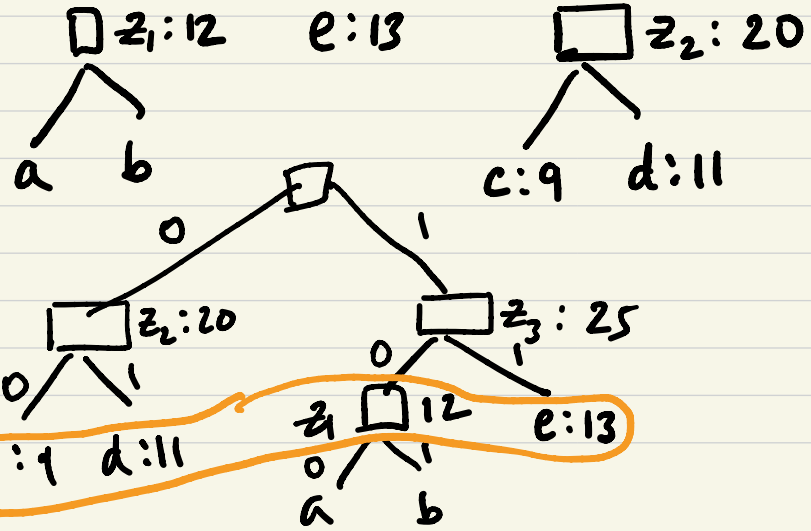
$$\Rightarrow \quad cost(T_0) \geq cost(T)$$

$\Rightarrow$ T is optimal for C.

a : 5　　b : 7　　c : 9　　d : 11　　e : 13

c : 9　　d : 11　　z_1 : 12　　e : 13 ←

a : 5　　b : 7 ←



a : 100

b : 101

c : 00

d : 01

e : 11

$n \leftarrow |C|$

$Q \leftarrow C$ // Q is a min priority queue. $O(n \log n)$

for $i \leftarrow 1$ to $n-1$

do allocate a new node z

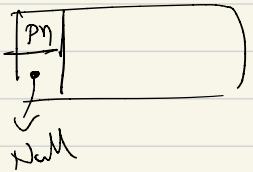
left $[z] \leftarrow x \leftarrow$ Extractmin (Q)

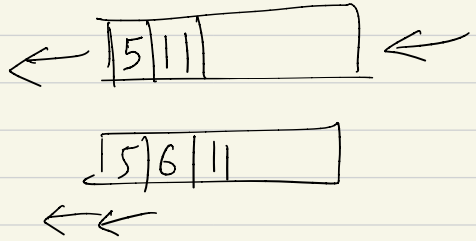
right $[z] \leftarrow y \leftarrow$ Extractmin Q.

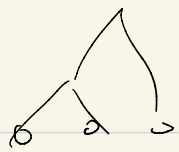
$f[z] \leftarrow f[x] + f[y]$

Insert (Q, z)

return Extract Min (Q)

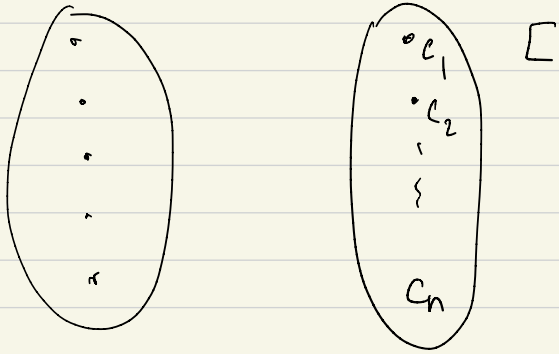






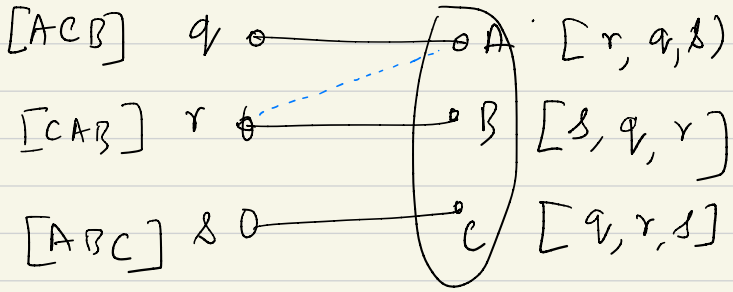
$n \longrightarrow n-1$

$O(n^2 \log n)$

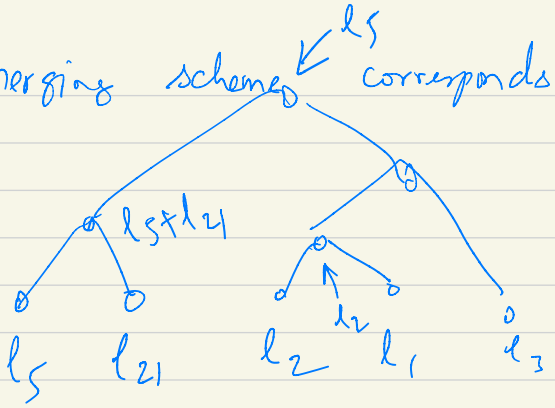


C

JE

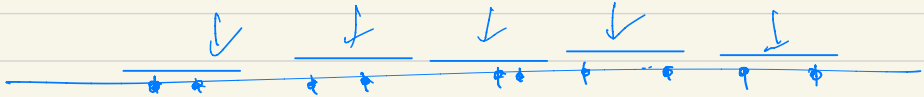


Any merging scheme corresponds to a binary tree.



$$\text{cost}(T) = \sum_{i=1}^{n} l_i \cdot \text{depth}_T(l_i)$$

Thus, this problem is equivalent to solving Huffman coding with freq $l_1, \dots, l_n$.



Let $f = [x_1, x_{1+1}]$. x_i be the first point not in f.

Ex
① Prove that there is an optimal sol^n with f as one of its interval. | S_{ij}

② Once we pick f then the subproblem to solve is S_{in}.

Proof :- Let Opt be a sol^n to S_{in}
Let g be the 1st interval in Opt.

Consider New $= (Opt \setminus \{g\}) \cup \{f\}$.

Show that $x_j \in g \Rightarrow x_j \in f$.