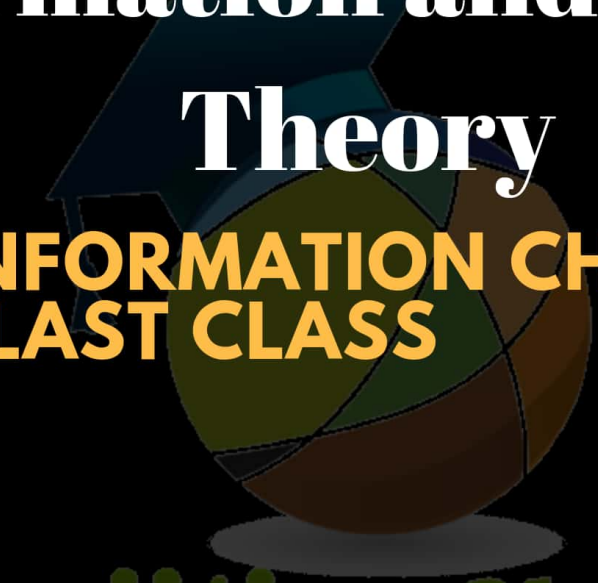
NOTES

Information and Coding Theory

UPTO INFORMATION CHANNEL
LAST CLASS



iitism2k16

Be Updated!

IIT-ISM 2K16 BATCH

BE UPDATED!!

iitism2k16@gmail.com

iitism2k16.webnode.com

Page No.
Date: / /

Information and Coding theory.

* Information theory
* Source coding
* Channel coding

Compression → Remove data redundancy
+
Retain Information.

9 11 11 11 14 13 - - - 21 → 5 bits for each element

$\hat{x}_n = n + 6$

$e_n = x_n - \hat{x}_n$

↓ only 1, 1, 0 → 2 bit for each element

Source coding : Uniquely + efficiently map Soure ⇌ binary

mappings must not be prefix of some other (codewords) - unique encoding + decoding

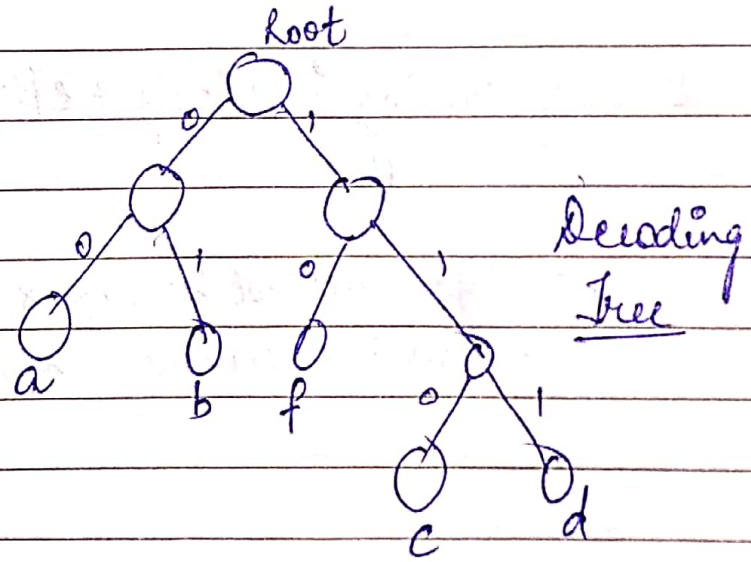
Fano Codes

a	b	c	d	e	f
0.3	0.2	0.1			

divide symbol set into 2 sets with almost equal cummulative probabilities Sum
[First sort in non increasing order]

a	0.3	00
b	0.3	01
f	0.2	10
c	0.1	110
d	0.1	111

Prefix-free code



Decoding Tree

Page No.
Date: / /

Kraft Theorem / Inequality : alphabet size r, q codewords, individual codeword lengths of $l_1 l_2 \dots l_n$

↓ can say if prefix-free code

$$K = \sum_{i=1}^{q} r^{-l_i} \leq 1$$

$r=2$, $q=4$:

0 $\Rightarrow l_1 = 1$

100 $\Rightarrow l_2 = 3$

110 $\Rightarrow l_3 = 3$

111 $\Rightarrow l_4 = 3$.

$2^{-1} + 2^{-3} + 2^{-3} + 2^{-3}$

$\frac{1}{2} + \frac{3}{8} = \frac{7}{8} \leq 1$. ✓

$s_1\ s_2\ s_3 \dots s_n$ binary code

$L_1 \leq L_2 \leq L_3 \leq \dots \leq L_n$

$N(S_1) = r^{L_1}$

$N(S_2) = r^{L_2} - r^{L_1} = r^{L_2} - r^{L_2 - L_1}$

$N(S_n) = r^{L_n} - r^{L_n - L_{n-1}} - r^{L_n - L_{n-2}} - \dots - r^{L_n - L_1}$

$N(S_i) \geq 1$

→ multiply by r^{-L_n}

$1 - r^{-L_{n-1}} - r^{-L_{n-2}} - \dots - r^{-L_1} \geq r^{-L_n}$.

$\Rightarrow r^{-L_n} + r^{-L_{n-1}} + \dots + r^{-L_1} \leq 1$

$\Rightarrow \boxed{\sum_{i=1}^{n} r^{-l_i} \leq 1}$

Compact Codes : Avg length ≤ Avg. length of all other uniquely decodable codes for all the same source & code Alphabet

Fano code is not Compact

$$\eta = \frac{H(S)}{\bar{R}}$$

$H(S)$ → Entropy

$\bar{R}$ → Avg. length

Huffman Codes :

a	b	c	d	e	f	g
0.2	0.2	0.25	0.15	0.1	0.1	0.1

(0.25) de (0.2) fg

Avg. min variance $= (\sigma^2) = \sum P_i (l_i - L)^2$

$$\bar{R} = 0.2 \times 2 + 0.4 \times 1 + 0.1 \times 4 \times 2 + 0.2 \times 3$$
$$= 2.2$$
$$\sigma^2 = 0.2 (2.2 - 2)^2$$

⇒ $\bar{R}$ for all Huffman trees are same ∴ efficiency is also same.

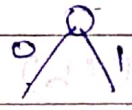
Q 3000 bits/s transfer. Every-sec-1000 symbols by source. Combine n-combined symbols together.

→ Huffman-tree with min variance is best

Canonical Huffman Coding :

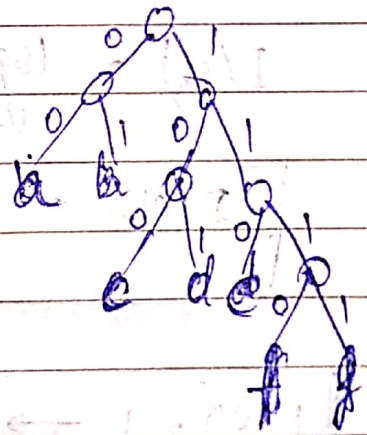
↓ [send codebook efficiently] (size).

well structured



Build tree from L to R in increasing order of length.

a	01
b	000
c	101
d	110
e	100
f	1111
g	1110



a	00
b	01
c	100
d	101
e	110
f	1110
g	1111

Send only 3 starting symbols of different length.

Codebook

a	01
c	101
f	1110

+ set of symbols abcdefg

Entropy :- Information $\propto \frac{1}{\text{occurence}}$.

$$I(S) \propto \frac{1}{P(S)}$$

$$\boxed{I(S) = \log_2\left(\frac{1}{P(S)}\right)} \text{ bits}$$

Information is additive. ∴ log used.

Events a, b

$$I(ab) = \log_2\left(\frac{1}{P(ab)}\right) = \log_2\left(\frac{1}{P(a)\cdot P(b)}\right)$$

$$I(ab) = \log_2\left(\frac{1}{P(a)}\right) + \log_2\left(\frac{1}{P(b)}\right)$$

$$\boxed{I(ab) = I(a) + I(b) = -\log_2 P(ab)}$$

$$\boxed{P(S) = 1 \longrightarrow I(S) = 0}$$

S_1	S_2	———	S_n
n_1	n_2	———	n_n

Total Information $= n_1 I(s_1) + n_2 I(s_2) + \cdots + n_n I(s_n)$

Avg. Info $= \frac{n_1}{N} I(s_1) + \frac{n_2}{N} I(s_2) + \cdots + \frac{n_n}{N} I(s_n)$

$= P(s_1)\, I(s_1) + P(s_2)\, I(s_2) + \cdots + P(s_n) I(s_n)$

$= \sum_{i=1}^{n} P(s_i)\, I(s_i)$

$\therefore$ $$H(S) = \sum_{i=1}^{n} P(s_i) \log_2 \left(\frac{1}{P(s_i)}\right)$$ bits/symbol

also $H(S) \geq 0$

Proof

$0 \leq P(s_i) \leq 1$

$\Rightarrow \frac{1}{P(s_i)} \geq 1$

$\Rightarrow \log_2 \left(\frac{1}{P(s_i)}\right) \geq 0$

$\Rightarrow P(s_i) \log_2 \left(\frac{1}{P(s_i)}\right) \geq 0$

Hence, $H(S) \geq 0$ Proved

If $p_1, p_2, \ldots, p_n$ and $q_1, q_2, \ldots q_n$ are all non-negative that satisfy

$$\sum_{i=1}^{n} p_i = 1 \qquad \sum_{i=1}^{n} q_i = 1$$

Then $$\sum p_i \frac{1}{\log_2 p_i} \leq \sum p_i \frac{1}{\log_2 q_i}$$

Proof

$$\ln x \leq x - 1$$

$$x = \frac{q_i}{p_i}$$

$$\Rightarrow \ln \frac{q_i}{p_i} \leq \frac{q_i}{p_i} - 1$$

$$\Rightarrow \sum p_i \ln \frac{q_i}{p_i} \leq \sum p_i \left(\frac{q_i}{p_i} - 1\right)$$

$$\Rightarrow \sum p_i \ln \frac{q_i}{p_i} \leq \sum q_i - \sum p_i$$

$$\Rightarrow \sum p_i \ln \frac{q_i}{p_i} \leq 0$$

$$\Rightarrow \sum p_i (\ln q_i - \ln p_i) \leq 0$$

$$\Rightarrow \sum p_i \ln q_i \leq \sum p_i \ln p_i$$

$$\Rightarrow -\sum p_i \ln q_i \geq -\sum p_i \ln p_i$$

$$\Rightarrow \sum p_i \ln \frac{1}{q_i} \geq \sum p_i \ln \frac{1}{p_i}$$

$$\Rightarrow \sum p_i \ln \frac{1}{p_i} \leq \sum p_i \ln \frac{1}{q_i}$$

$$\Rightarrow \frac{\sum p_i \log_2 \frac{1}{p_i}}{\log_2 e} \leq \frac{\sum p_i \log_2 \frac{1}{p_i}}{\log_2 e}$$

$$\Rightarrow \boxed{\sum p_i \log_2 \left(\frac{1}{p_i}\right) \leq \sum p_i \log_2 \left(\frac{1}{q_i}\right)}$$

Proved

$$\therefore \boxed{H(S) \leq \sum p_i \log_2 \left(\frac{1}{q_i}\right)}$$

$q_i = \frac{1}{n}$ ⇒ $H(S) \leq \sum p_i \log_2 n$

↓ all events are equal probable

$$\Rightarrow H(S) \leq \log_2 n \sum p_i$$

$$\Rightarrow \boxed{H(S) \leq \log_2 n}$$

$H(S) = 0 \Rightarrow p_i = 1$, rest $p_j = 0$.

$\geq$ One event is certain
no information gained

$$\Rightarrow \boxed{0 \leq H(S) \leq \log_2 n}$$

(SCT)

Source Coding Theorem :- The n of data Compression is based on efficient representation of the symbols.
SCT establishes the limit for possible data compression & condition is -

$$H(S) \leq \bar{R} < H(S)+1$$

$H(S)$ ← entropy. $\bar{R}$ → avg. coding length

$$\boxed{\eta = \frac{H(S)}{R} \times 100\ \%}$$

$$H(S) - \bar{R} = \sum p_i \log \frac{1}{p_i} - \sum p_i R_i$$

$$= \sum p_i \left(\log_2 \frac{1}{p_i} - R_i \right)$$

$$= -\sum p_i \left(\log_2 p_i + \log_2 2^{R_i} \right)$$

$$\Rightarrow H(S) - \bar{R} = -\sum P_i \left(\frac{\log_2 P_i \cdot 2^{R_i}}{\log_2 e} \times \log_2 e \right)$$

$$= -\sum P_i \log_e P_i \times 2^{R_i} \times \log_2 e$$

$$\log_2 e \left[\sum P_i \ln \left(\frac{1}{P_i} \right) - \sum P_i \times R_i \log_e 2 \right]$$

$$H(S) - \bar{R} = \log_2 e \left[\sum P_i \log_e \frac{1}{P_i \times 2^{R_i}} \right]$$

$$\leq \log_2 e \left[\sum P_i \left(\frac{1}{P_i 2^{R_i}} - 1 \right) \right] \quad [\ln x \leq x - 1]$$

use this

$$\leq \log_2 e \left[\sum 2^{-R_i} - \sum P_i \right]$$

$$\leq \log_2 e \left[\sum 2^{-R_i} - 1 \right]$$

≤ 1 [Kraft's inequality].

$$\leq 0.$$

$$\Rightarrow \boxed{H(S) \leq \bar{R}}$$

Now,

$$\bar{R} = \sum p_i R_i \qquad R_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil$$

$$= \sum p_i \left\lceil \log_2 \frac{1}{p_i} \right\rceil$$

$$\because \boxed{\lceil x \rceil < x + 1}$$

$$\Rightarrow \bar{R} < \sum p_i \left(\log_2 \frac{1}{p_i} + 1 \right)$$

$$\Rightarrow \bar{R} < \sum p_i \log_2 \frac{1}{p_i} + \sum p_i$$

$$\Rightarrow \boxed{\bar{R} < H(S) + 1}$$

$$\therefore \boxed{H(S) \leq \bar{R} < H(S) + 1}$$

Q

$P(a) = 0.8$ | 0

$P(b) = 0.2$ | 1

$$\bar{R} = 0.8 \times 1 + 0.2 \times 1 = 1$$

$$H(S) = \sum p_i \log \left(\frac{1}{p_i} \right)$$

$$\Rightarrow H(S) = 0.8 \log_2 \frac{1}{0.8} + 0.2 \log_2 \frac{1}{0.2}$$

$$\Rightarrow H(S) = 0.8\left[\log_2 10 - 3\right] + 0.2\left[\log_2 10 - 1\right]$$

$$= 0.8\left[3.322 - 3\right] + 0.2\left[3.322 - 1\right]$$

$$= 0.2575 + 0.4644$$

$$= 0.72 \text{ bits/symbol.} \quad \bigstar$$

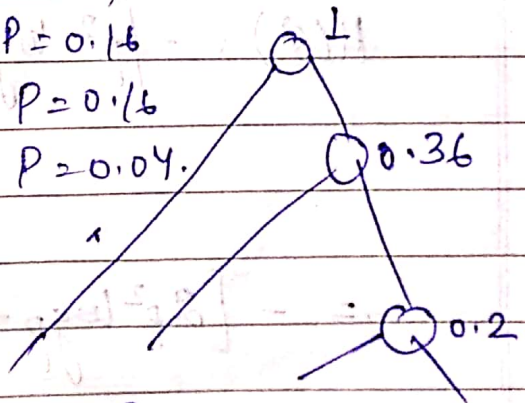
$$\eta = \frac{H(S)}{R} \times 100\% = 72\%.$$

Extension of source : 2nd order extension source (S^2)

(Efficiency (η))

$\sigma_{ij} = (S_i S_j)$

aa – A → P = 0.64
ab – B → P = 0.16
ba – C → P = 0.16
bb – D → P = 0.04.



A → 1
B → 00
C → 011
D → 010

$$\overline{R_2} = 0.64 \times 1 + 0.16 \times 2 + 0.16 \times 3 + 0.04 \times 3$$

$$= 1.56 \text{ bits.}$$

$$H(S^2) = -\left[0.64 \log_2 0.64 + 0.16(\log 0.16) \times 2 + 0.04 \log 0.04\right]$$

$$= 1.44 \text{ bits/symbol}$$

$$\eta = \frac{H(S^2)}{R_2} \times 100 = 92\%.$$

$$P(a) = p, \quad P(b) = 1-p.$$

$aa \rightarrow p^2$

$ab \rightarrow p(1-p)$

$ba \rightarrow (1-p)p$

$bb \rightarrow (1-p)^2$

$$H(S^2) = -\left[p^2 \log p^2 + (1-p)^2 \log (1-p)^2 + 2p \times (1-p) \times \log p \times (1-p)\right]$$

$$= -\left[2p^2 \log p + 2(1-p)^2 \log (1-p) + 2p(1-p)\log p + 2p(1-p)\log\right]$$

$$= -\left[2p \log p \left[p + 1 - p\right] + 2(1-p)\log(1-p)[p+1-p]\right]$$

$$= -\left[2p \log p + 2(1-p)\log(1-p)\right]$$

Page No.
Date: / /

$$\Rightarrow H(S^2) = 2\left[p \log \frac{1}{p} + (1-p) \log \frac{1}{1-p}\right]$$

$$\boxed{H(S^2) = 2H(S)}$$

Now, $H(S^n)$ $\quad S_1\ S_2\ S_3 - - - - S_q$

$\sigma_i = S_{i_1}\ S_{i_2} - - S_{i_n}$

$$H(S^n) = -\sum_{i=1}^{n} p(\sigma_i) \log_2 P(\sigma_i)$$

$$P(\sigma_i) = P(S_{i_1}) P(S_{i_2}) - - \cdot P(S_{i_n})$$

$$H(S^n) = -\left[\sum P(S_{i_1}) P(S_{i_2}) - - P(S_{i_n}) \log_2 (P(S_{i_n}) P(S_{i_2}) - P(S_{i_n}))\right]$$

$$= -\sum_{i=1}^{n} P(S_{i_1}) \log_2 P(S_{i_1}) \sum_{i=2}^{q} P(S_{i_2}) - -$$

$$\boxed{H(S^n) = nH(S)}$$

$\bar{R}_n$

Source Coding theorem

$$H(S^n) \leq \bar{R}_n < H(S^n) + 1$$

$$n H(S) \leq \bar{R}_n < n H(S) + 1$$

$$\Rightarrow \boxed{H(S) \leq \frac{\bar{R}_n}{n} < H(S) + \frac{1}{n}}$$

$$\boxed{\lim_{n \to \infty} \frac{\bar{R}_n}{n} = H(S)}$$

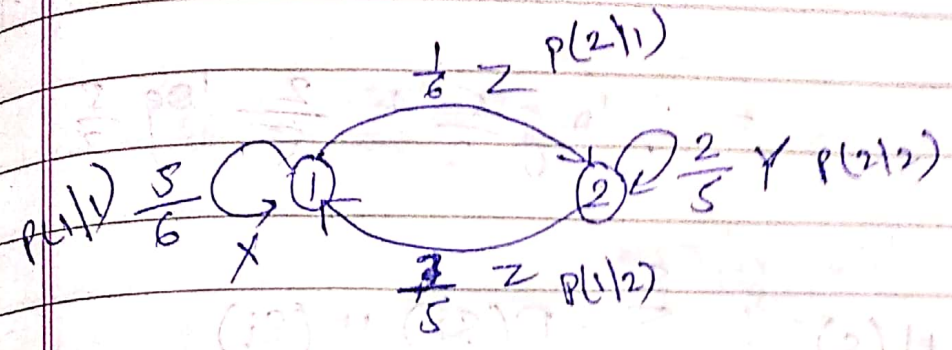
Huffman Coding → not suitable for large Symbols.

∴ higher order extension is not efficient to implement

↓ Solution.

Arithmetic Coding

Markov Model



$$P(1) + P(2) = 1.$$

$$P(1) = \frac{5}{6}P(1) + \frac{3}{5}P(2).$$

→ 2 eqn, 2 var.

$$P(2) = \frac{2}{5}P(2) + \frac{1}{6}P(1).$$

$$\Rightarrow \boxed{\begin{aligned} P(1) &= 18/23 \\ P(2) &= 5/23 \end{aligned}}$$

→ Entropy of each state

$$H(S_i) = \sum_{j=1}^{8} P(j|i) \log_2 \frac{1}{P(j|i)}$$

where, 8 → total no. of states.

$$H(S_1) = P(1|1) \log_2 \left(\frac{1}{P(1|1)}\right) + P(2|1) \log \frac{1}{P(2|1)}$$

$$= \frac{5}{6} \times \log \frac{6}{5} + \frac{1}{6} \times \log 6. = 0.65 \text{ bits/symbol}$$

$$H(S_2) = P(1/2)\log\frac{1}{P(1/2)} + P(2/2)\log\frac{1}{P(2/2)}$$

$$= \frac{3}{5}\log\frac{5}{3} + \frac{2}{5}\log\frac{5}{2}$$

$$H(S) = \sum_{i=1}^{S} P(S_i)\, H(S_i)$$

$$= P(S_1)\cdot H(S_1) + P(S_2)\, H(S_2)$$

$$= \frac{18}{23} \times H(S_1) + \frac{5}{23} \times H(S_2)$$

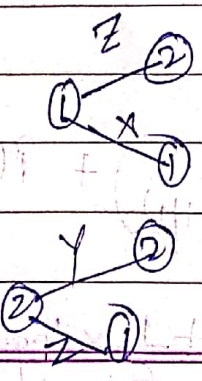
$$H(S) = 0.7192 \text{ bits/symbol}$$

Now, $\bar{R} = ?$

$|message| = 1$

$L \leftarrow$ length of message

entropy of message of length L / symbol $\leftarrow G_L = \frac{1}{L}\sum P(msg_L)\cdot\log\frac{1}{P(msg_L)}$



$$P(X) = P(1) \times P(1/1) = \frac{18}{23} \times \frac{5}{6} = \frac{15}{23}$$

$$P(Z) = P(1)\cdot P(2/1) + P(2) \times P(1/2)$$

$$= \frac{18}{23} \times \frac{1}{6} + \frac{5}{23} \times \frac{3}{5}$$

$$= \frac{6}{23}.$$

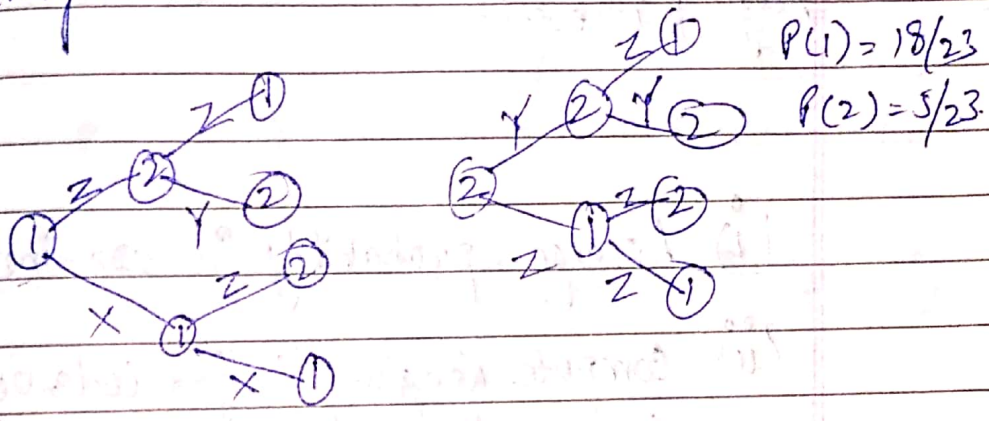
$$P(Y) = P(2) \times P(2/2)$$

$$= \frac{5}{23} \times \frac{2}{5} = \frac{2}{23}.$$

$$G_{L_1} = \frac{15}{23} \times \log \frac{23}{15} + \frac{6}{23} \log \frac{23}{6} + \frac{2}{23} \log \frac{23}{2}$$

$$= 1.2142 \text{ bits/symbol}.$$

$|msg| = 2$.



$P(ZZ) = 1/10$
$P(XX) = 25/46$
$P(YY) = 4/115$
$P(XZ) = 5/46$
$P(YZ) = 6/115$
$P(ZY) = 6/115$
$P(ZX) = 5/46$.

$G_{L_2} = 1.0597$ bits/symbol

$G_{L_1} > G_{L_2} > \cdots \rightarrow H$.

$$\lim_{L \to \infty} G_L = H(S)$$

Shannon Coding :

$$\text{Info} \propto \frac{1}{\text{occurence}}$$

$$l_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil \text{ bits}$$

l_i ↙ length of i^{th} event.

(i) Arrange probability in non-increasing order

(ii) Compute length l_i for codeword corresponding to each symbol s_i

$$l_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil \text{ bits}$$

(iii) Define following parameters.

(Cummulative Probability)

$$\begin{aligned} q_1 &= 0 \\ q_2 &= q_1 + p_1 \\ q_3 &= q_2 + p_2 = p_1 + p_2 \\ q_4 &= q_3 + p_3 = p_1 + p_2 + p_3 \\ &\vdots \\ q_{n+1} &= q_n + p_n = \sum p_i = 1 \end{aligned}$$

(iv) Expand q_i in binary till l_i no. of places after decimal point.

(v) The no. after decimal places in the binary representation of q_i are the codewords for corresponding symbol s_i

Q1

A	B	C	D
0.1	0.2	0.3	0.4

	P_1	P_2	P_3	P_4
D C B A	p_i:	0.4	0.3	0.2	0.1

$$l_i : \left\lceil \log_2 \frac{1}{0.4} \right\rceil, \left\lceil \log_2 \frac{1}{0.3} \right\rceil, \left\lceil \log_2 \frac{1}{0.2} \right\rceil, \left\lceil \log_2 \frac{1}{0.1} \right\rceil$$

$$l_i : [2, 2, 3, 4]$$

$q_1 = 0 = (0.00)_2$

$q_2 = 0 + 0.4 = 0.4 = (0.01)_2$

$q_3 = 0.4 + 0.3 = 0.7 = (0.101)_2$

$q_4 = 0.7 + 0.2 = 0.9 = (0.1110)_2$

$q_5 = 0.9 + 0.1 = 1$

Hence, Prefix code

D →	00	(0.4)
C →	01	(0.3)
B →	101	(0.2)
A →	1110	(0.1)

$$H(S) = \sum P_i \log_2 \left(\frac{1}{P_i}\right)$$

$$= -(0.4 \log 0.4 + 0.3 \log 0.3 + 0.2 \log 0.2 + 0.1 \log 0.1)$$

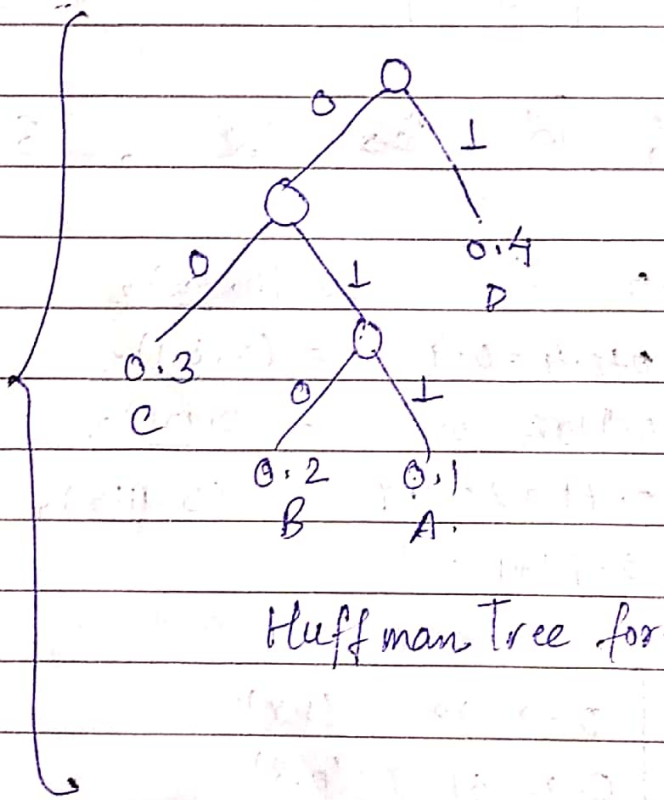
$$\bar{R} = \sum P_i R_i$$

$$= 0.4 \times 2 + 0.3 \times 2 + 0.2 \times 3 + 0.1 \times 4$$

$$= 0.8 + 0.6 + 0.6 + 0.4$$

$$= 2.4$$

$$\eta = \frac{H(S)}{\bar{R}} \times 100 = 76.93\%$$



Huffman Tree for the same.

Huffman coding :-

A → 011 (0.1)
B → 010 (0.2) H(S) is same
C → 00 (0.3)
D → 1 (0.4)

Now,

$\bar{R} = 3\times0.1 + 3\times0.2 + 2\times0.3 + 1\times0.4$

$= 0.3 + 0.6 + 0.6 + 0.4$

$= 1.9$

$\eta = 96.84\%$ $\left(\frac{H(S)}{\bar{R}} \times 100\%\right)$

Hence, $\bar{R}_H < \bar{R}_S$ [more compact coding]

n-ary Huffman code (non-binary)

n - children of a node
add dummy node with prob = 0
if needed

$\alpha = \frac{q - r}{r - 1}$ ← no. of original nodes; ← no. of children of (n-ary)

$q' = r + \lceil \alpha \rceil (r - 1)$.

$d = q' - q$ [no. of dummy nodes]

$s_1 = 0.16$

$s_2 = 0.14$

$s_3 = 0.13$

$s_4 = 0.12$

$s_5 = 0.10$

$s_6 = 0.10$

$s_7 = 0.06$

$s_8 = 0.06$

$s_9 = 0.05$

$s_{10} = 0.04$

$s_{11} = 0.04$

$s_{12} = 0.00$ } dummy nodes

$s_{13} = 0.00$ }

$q = 11$

$r = 4$

$\alpha = \frac{11-4}{4-1} = \frac{7}{3} = 2.33$

$q' = r + \lceil \alpha \rceil (r-1)$

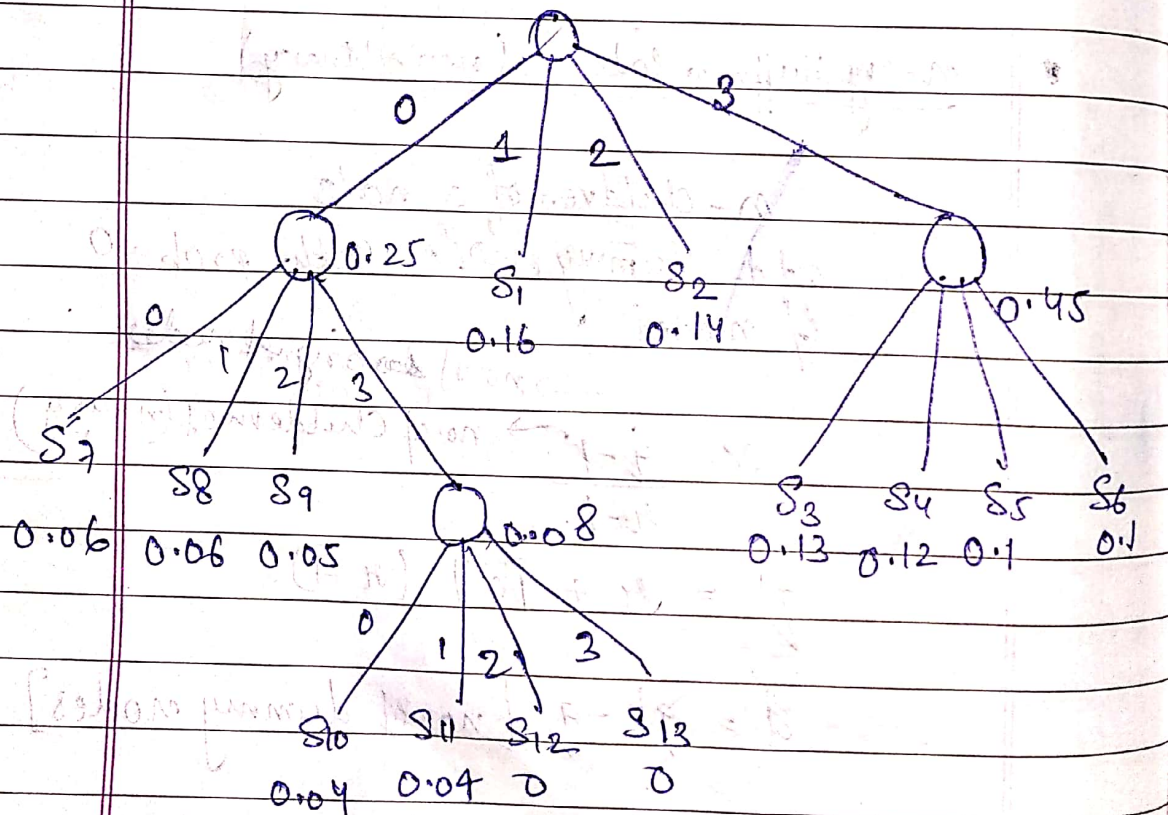
$= 4 + 3 \times 3$

$= 13$

$d = q' - q = 2$

↓

number of dummy nodes!



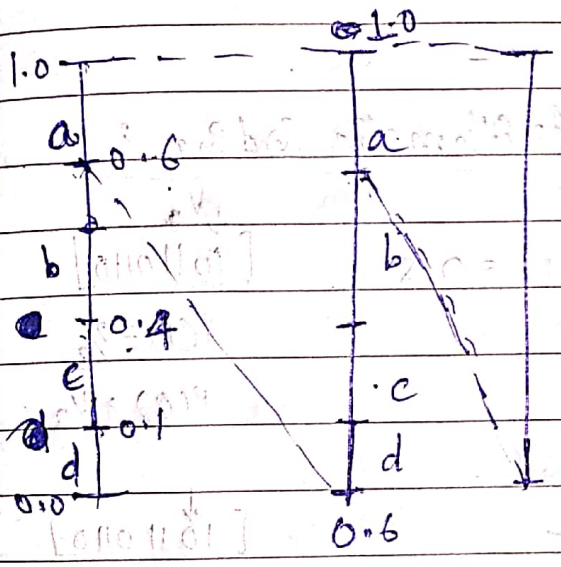
Arithemetic Coding:

- → No sorting required
- → Drawback → Floating pt ops.

(a, b, c, d) = (0.4, 0.2, 0.3, 0.1)

↑ no sorting

(aacbd)



→ difference

→ LL + d × Pj → Probability.

↓ lower limit

Range LL. < Codeword < Range UL

$$\text{tag} = \frac{\text{Range}(LL + UL)}{2}$$

aacbd = 0.871 → tagvalue

Inform = (0.879, 4)

make intervals where value falls
decoding done

0.8 → 0.(11)00400

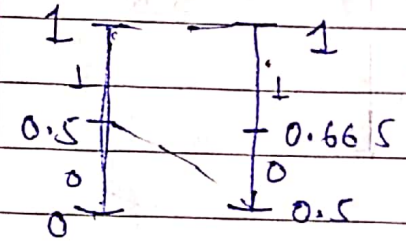
0.6 → 0.(10)011001

[0.11] → In tag value
upto

Select proper tag value → else Information loss occurs.

Adaptive Arithmetic Coding :

P(0) = P(1) = 0.5



0.5 + 0.5 × 1/3

[10110110]

P(i) = 2/3 C(1) = 2

P(0) = 1/3 C(0) = 1

[10110110]

C(1) = 2 P(1) = 1/2

C(0) = 2 P(0) = 1/2

Huffman Coding → Codebook needed at recieves end [Overhead].

Adaptive Huffman Coding → No Codebook needed (Source probability Calculated on the basis of bitstream).

~~Greedy Property~~ -

Visualizing Adaptive Huffman Coding -

* What is Huffman Coding ?

When we transmit infor^{m}, we generally need to convert some sort of data (text, pic, etc) into binary.
To do this, we assign codes to help us distinguish between different pieces of data.

For example - If we had a string of "abca", we might assign codes like -

a = 00
b = 01
c = 10.

a	b	c	a
00	01	10	00

But what if we wanted to encode "aabaacaa"?
If we used original encoding where 2 bits for each character were used, we would be treating "a" and "c" with equal importance, even though "a" appears much more.

Wouldn't it be more efficient if we used fewer bits for 'a' and more for 'c'?
This is where Huffman Coding comes in.

Page No.
Date: / /

Huffman coding is a "lossless data compression" algorithm that assigns variable-length codes based on the frequencies of our i/p characters.

In order to determine what code to assign to each character, we will first build a binary tree that will organise our chars based on frequency.

* **Building a Huffman Tree:-**

Encode → "bookKeeper"

first build Huffman tree for such string which will in turn show us what binary encoding we will use for each char.

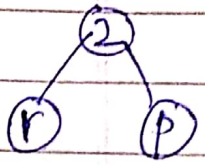
b	e	K	o	p	r
1	3	2	2	1	1

use this table to add nodes & edges that will build up the tree.

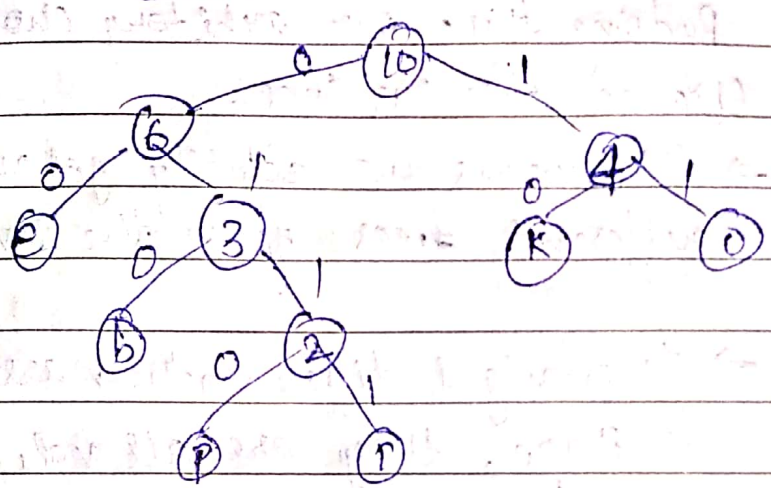
First, we start by adding leaf nodes for the two char. that occur the least. here, we have a tie b/w p, r, b so, arbitrary choose any two.

Repeat this process with remaing nodes.

b	e	k	o	pr
1	3	2	2	2



bpr	e	k	o
3	3	2	2



bpr	ko	e
3	4	3

→

bpre	ko
6	4

↓ merge all

	b	e	k	o	p	r
freq.	1	3	2	2	1	1
code	010	00	10	11	0110	0111

Page No.
Date : | |

Note :- As we were building our tree, when choosing our two least frequency characters in our table, we repeadly had ties b/w three or more characters. When this happened, we would choose two of our tied element arbitrarily.

By doing this, our arbitary choice will change the our tree.

→ This means we actually get multiple different trees from the same string.

→ Trees might differ in their arrangement & shape, they are all valid Huffman trees.

Adaptive Huffman Coding-

while traditional Huffman coding is very useful, we can sometimes be limited by the fact that we need to know what data we are going to be encoding before we can encoding. This might work in some scenarios, but there are many other applications where this is impractical.

for ex. if we wanted to transmit a live video stream, we could not possibly know exactly what is going to be transmitted before hand.

As an alternative, we can use adaptive Huffman coding.

With adaptive huffman coding, our purpose and goal is identical to traditional Huffoman coding - we want to build a tree that will give us an optimal binary encoded lysto scheme.

The major difference is that we will not pre-process our ip before we start encoding it. Instead, we will build a tree on the fly as we read in our input.

We will use FGK [Faller Gallager - Knuth] algorithm.

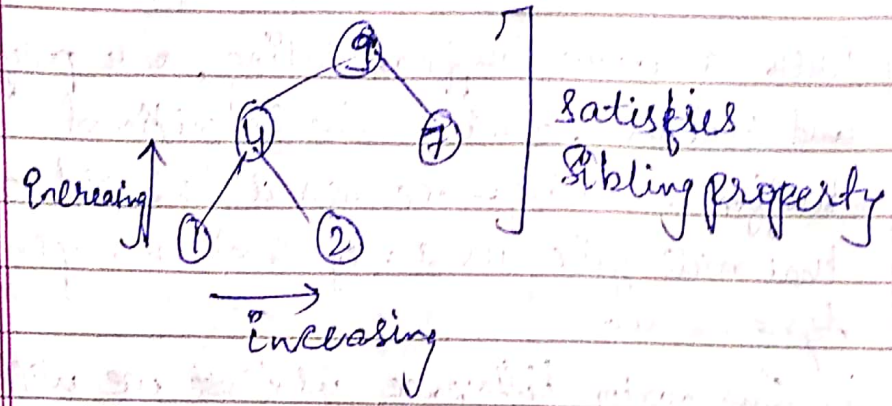
The two major difference b/w Huffmand & FGK tree -

→ First, out FGK tree must satisfy the sibling property. In order to do this our tree must meet following condition -

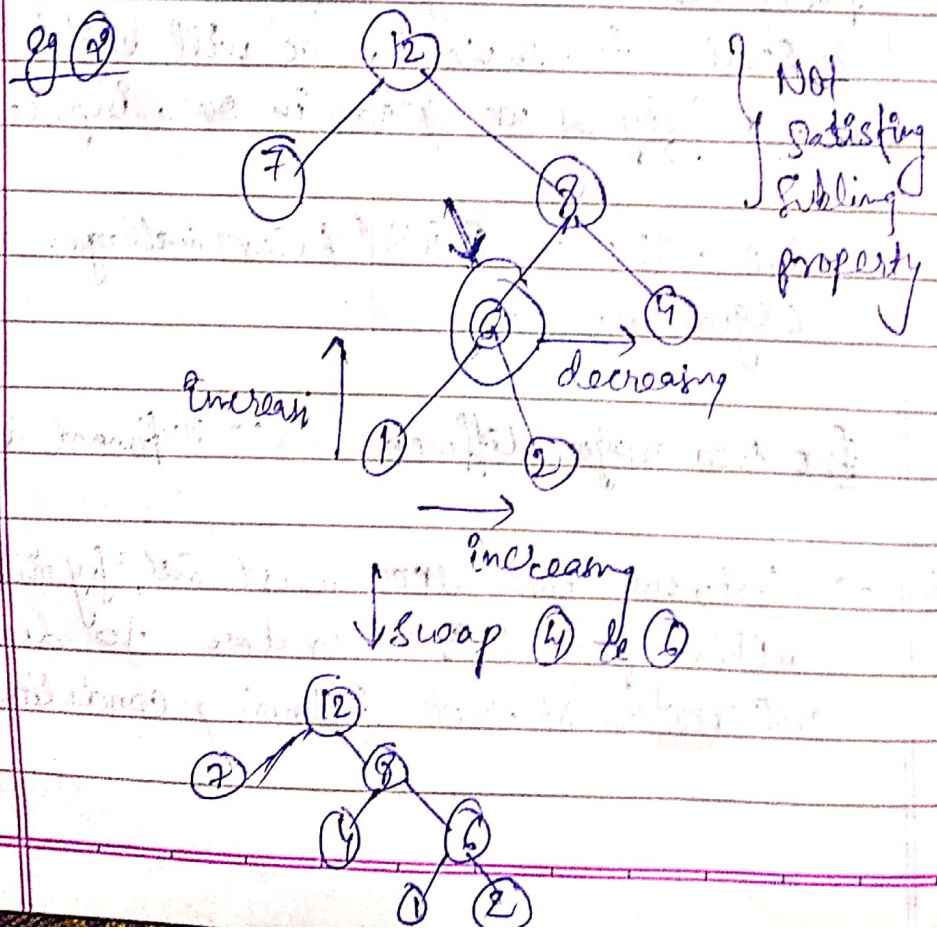
1. All nodes in our tree (except root) must have a sibling.

2. The nodes can be listed (from L-to-R, Bottom to top) in order of ↑ value.

eg (1)



eg (2)



Page No.
Date: / /

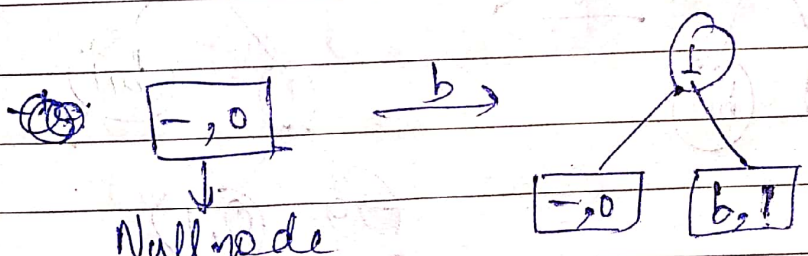
→ The second major difference from Huffman trees to FGK trees is our use of null node.

- In Huffman we build tree in bottom up fasion [as we have frequency] but in adaptive while reading i/p & building our tree at same time. As a result, we must build FGK top down.

we use our null node as a sibling for new character nodes we will add. This way we will still maintain the sibling property.

eg "bookkeeper". Build FGK

→ As 'b' is our char, we will have to add it into our tree. we can make our b node siblings with the null node & then make both of them children of a new internal node.

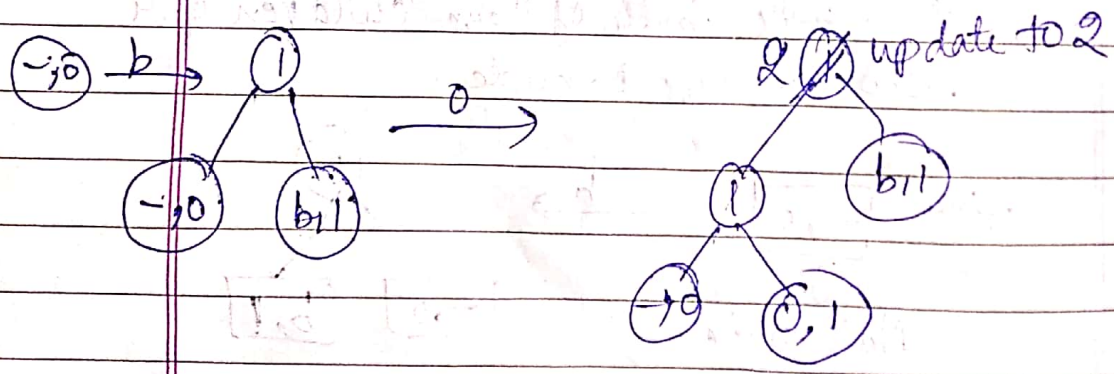


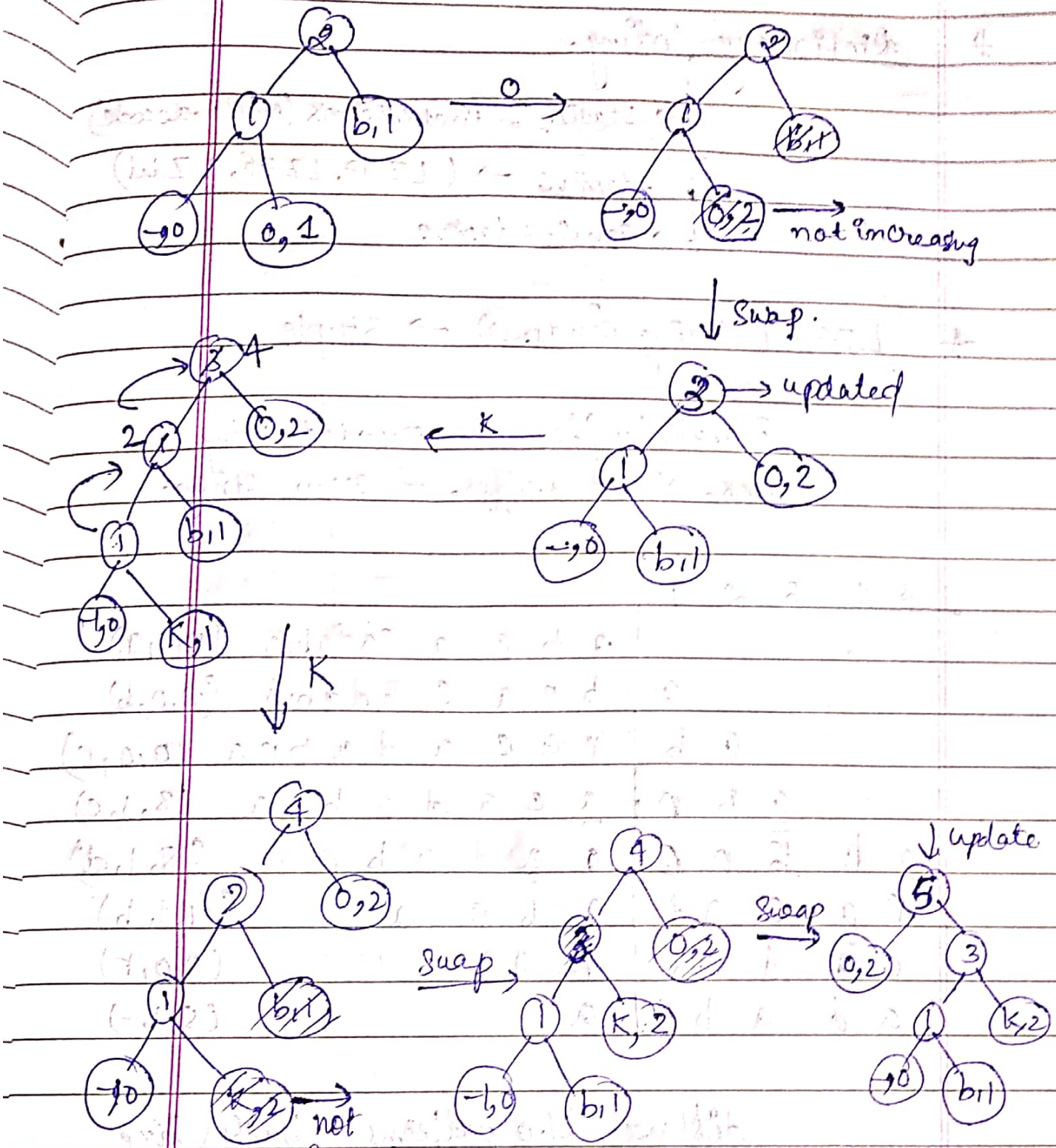
After inserting each node, we must sure that we don't violate the sibling property.

Since we will always insert new leaves as a sibling of the null node, we know that we will never violate the first sibling property.

However, we will need to make sure that our nodes are always in the right order.
(↑ value from L to R & ↑ value from B to T).

To do this, we will incrementally walk up the tree and update the parent chain b/w our new leaf & the root.
If we ever encounter conflicting nodes, we can swap them.





Dictionary coding

- → static : fixed codebook (encode-decode)
- → adaptive → (LZ-77, LZ-78, LZW)
- → semi-adaptive

LZ-77 (Sliding window) → 3tuple

Search buffer – encoded string
look-ahead buffer – rem. string.

5 4 3 2 1		
=	a b r a c a d a b r a	(0,0,a)
a	b r a c a d a b r a	(0,0,b)
a b	r a c a d a b r a	(0,0,r)
a b r	a c a d a b r a	(3,1,c)
a b r a c	a d a b r a	(5,1,d)
r a c a d	a b r a	(4,1,b)
c a d a b	r a	(0,0,r)
a d a b r	a	(5,1,-)

distance b/w patterns > buffer size
pattern might have been shifted.

Date: / /

LZ-78 (2-tuple).

if char not found
Insert in dictionary.

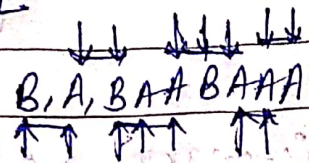
dictionary [A B B C B C A B A B C A A B C A B B]

Output	Index	String
(0, A)	1	A
(0, B)	2	B
(2, C)	3	BC
(3, A)	4	BCA
(2, A)	5	BA
(4, A)	6	BCAA
(6, B)	7	BCAAB

Patterns will not not be missed (not moved out of buffer).

Size of dictionary can be very large.

LZW Algo



Encoding			
Output	Representing	Index	entry
		1	A
		2.	B
2	AB	3	BA
1	A	4	AB
3	BA	5	BAA
4	AB	6	ABA
1	A	7	AA
7	AA	-	-

Now, send sample table

Index	Entry
1	A
2	B

and sequence 2 1 3 4 1 7

Decoding Decode first index to w.

Repeat {

Decode next index to S.

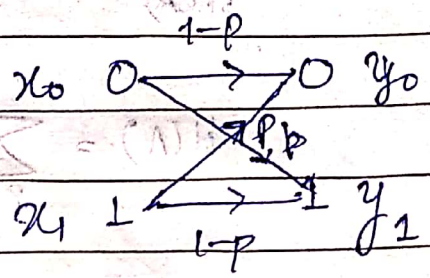
Add w with first char of string

update w by S

}.

Information Channel :-

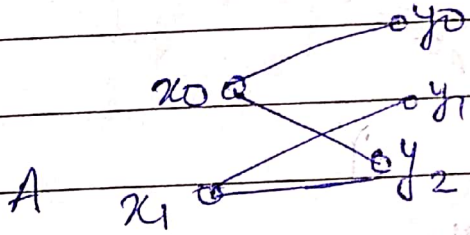
Binary Symmetric channel-



Content may change.

Cross-over probability = p

$P(y_0|x_0) = -$.

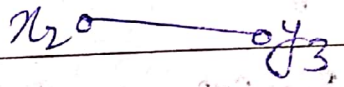


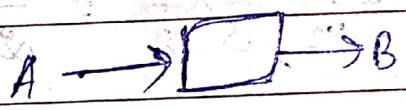
$P(y_i|x_i)$

↓

channel matrix

sum of row = 1





channel must be stationary → statistic same transmission.

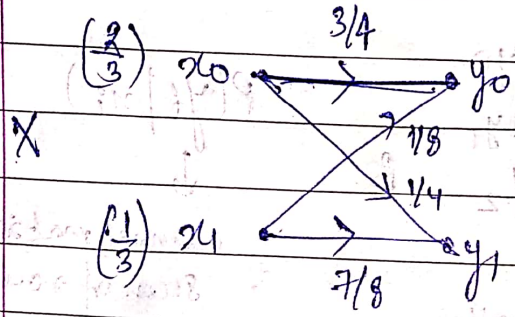
memory less → current o/p independent of history o/p.

$$\sum_j P(y_j|x_i) = 1$$

$$P(y_j) = \sum_i P(y_j|x_i)\, P(x_i) = \sum_i P(x_i, y_j)$$

$$H(A) = \sum_i p(x_i) \log_2 \frac{1}{p(x_i)}$$

$$H(B) = \sum_j p(y_j) \log_2 \frac{1}{p(y_j)}$$



$$p(y_0) = \frac{2}{3} \times \frac{3}{4} \times 2 + \frac{1}{3} \times \frac{1}{8} = \frac{13}{24}$$

$$p(y_1) = \frac{7}{8} \times \frac{1}{3} + \frac{1}{4} \times \frac{2}{3} \times \frac{2}{2} = \frac{11}{24}$$

$$H(X) = -\left[\frac{2}{3} \log_2 \frac{2}{3} + \frac{1}{3} \log_2 \frac{1}{3}\right] = 0.9182$$

$$H(Y) = -\left[\frac{13}{24} \log_2 \frac{13}{24} + \frac{11}{24} \log_2 \frac{11}{24}\right] = 0.9949$$

$H(X|Y)$

Conditional entropy

$$H(x|y_0) = -\sum p(x_i|y_0) \log_2 p(x_i|y_0)$$

Also, $p(y_0|x_0) \cdot p(x_0) = p(y_0)\, p(x_0|y_0) =$

$$\frac{3}{4} \times \frac{2}{3} = \frac{13}{24} \times p(x_0|y_0)$$

$$\frac{12}{24} = \frac{13}{24} \times p(x_0|y_0)$$

$$\boxed{p(x_0|y_0) = \frac{12}{13}}$$

$$\boxed{p(x_1|y_0) = \frac{1}{13}}$$

$$H(X|y_0) = -p(x_0|y_0) \log_2 p(x_0|y_0) - p(x_1|y_0) \log_2 p(x_1|y_0)$$

$$= 0.3912$$

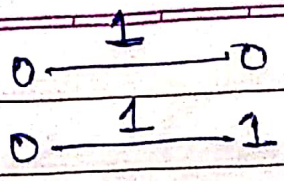
$H(X|y_1) =$ ✓

$p(y_0) > p(y_1)$

∴ less info^{n} expressed.

$$H(X|Y) = p(y_1) H(X|y_1) + P(y_0) H(X|y_0)$$

$$= \frac{11}{24} \times 0.9456 + \frac{13}{24} \times 0.3912$$



$$H(X|Y) = p(y_1) \sum p(x_i|y_1) \log_2 \frac{1}{p(x_i|y_1)} + p(y_0) \sum p(x_i|y_0) \log_2 \frac{1}{p(x_i|y_0)}$$

$$= \sum_{i} \sum_{j} p(y_j) \cdot p(x_i|y_j) \log_2 \frac{1}{p(x_i|y_j)}$$

$$H(X|Y) = 0.645$$

$$I(X,Y) = H(X) - H(X|Y)$$
$$= H(Y) - H(Y|X).$$

Types of channel-

1) Noise-free $H(X|Y) = 0$
2) Noisy $I(X;Y) > 0 \quad \therefore H(X) > H(X|Y)$
3) Ambiguous $H(X) = H(X|Y) \quad I(X,Y) = 0$