Class 12

General comment about engineering quantities [like length of compression or rate of comm] in info theory :-

"Achievability": ("a certain 'length' L is achievable" / "... 'rate' R ...")

↓

There exists some scheme in which we can show that the length of compression (or) 'rate' (or) any quantity of interest, happens to be equal to L (or) R (or) that value which is achievability

Converse:

No scheme exists which can improve upon some value.

10m/s speed is 'Achievable': (Optimal) Max speed among humans ≥ 10m/s.

Running race: ① ∃ some person who can run 10m/s.

Converse: Speed < 1 km/s.

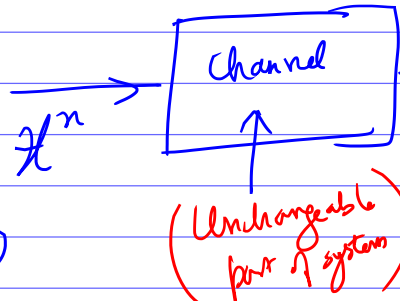
② ∄ no person who can run at 1 km/s.

"Matching converse" means "No human being can run at a speed $10+\epsilon$, for any $\epsilon > 0$".

Channel coding:

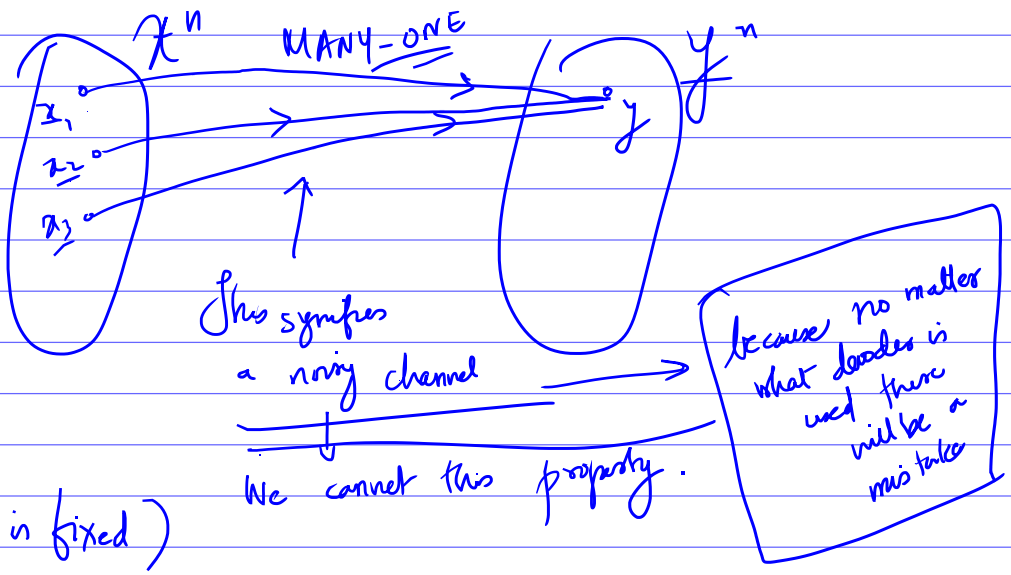
$x \in \mathcal{X}$ ($\mathcal{X}$ (i/p alphabet))

y ($\mathcal{Y}$ (o/p alphabet))



$x \in \mathcal{X}^n$ ($x_1, \ldots, x_n$)

$y \in \mathcal{Y}^n$ ($y_1, \ldots, y_n$)



(Assuming n is fixed)

To Make this channel one-one (& therefore ensure correct decoding),

we exist some sequences (n-length vectors from $\mathcal{X}^n$)

from set of all possible transmittable sequences.

This subset of transmittable sequences is call Channel Code (or simply Code). Denoted generally by $\mathcal{C}$

Note that $\mathcal{C} \subseteq \mathcal{X}^n$. Each seq/vector in $\mathcal{C}$ is called codeword.

No of bits that is required to represent $|\mathcal{C}|$ codewords

$$= \log_2 |\mathcal{C}| \text{ bits.}$$

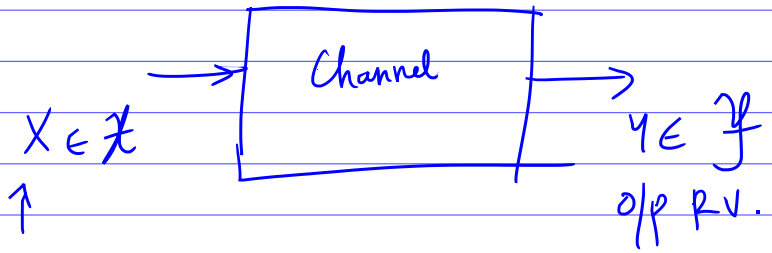
Rate of the code $\mathcal{C}$ $= \dfrac{\log_2 |\mathcal{C}|}{n}$ bits per channel use

bpcu

b/cu.

Intuitively: Higher the rate, more the chance for many-one kind of system, & thus more the chance of error.

Probabilistically Noisy Channel (or) Random Channel (or) Random Noise:



I/p RV

For $X = x \in \mathcal{X}$, there will be a probability distribution on the output RV. Y.

$$P_{Y|X=x} = \{ P(Y=y \mid X=x) : y \in \mathcal{Y} \}$$

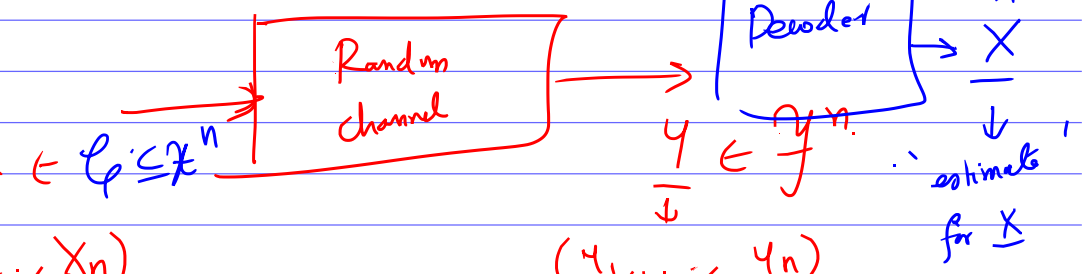
↳ represents the cond distribution on Y given $X = x$.

↓

For every $x \in \mathcal{X}$, we will this kind of distribution. $P(y|x=x)$

Those distributions $P_{Y|X}(y|x)$ $\forall x$ completely characterize or describe the random channel.

$P(\text{error})$ → how to think abt this?



$(X_1, \ldots, X_n)$ $\quad$ $(Y_1, \ldots, Y_n)$

X_i's are not independent in general

$\hat{X} = g(\underline{Y})$

g is decoding fn

Decoding Error event: $\hat{\underline{X}} \neq \underline{X}$

$P(\text{Decoding error}) = \text{Prob}(\hat{X} \neq X)$

Class no 13

$\underline{x} \in \mathcal{X}^3 \rightarrow \boxed{P_{y|x}(y|x)} \rightarrow \underline{y} \in \mathcal{Y}^3$.

$\mathcal{X} = \{0, 1\}$.

$\mathcal{C} = \{000, 111\}$ instead of all eight sequences.

Intuitively $P(\text{error})$ decreases, but Rate of $\mathcal{C} = \frac{\log_2 |\mathcal{C}|}{n} = 1/3$.

Intuitively, Prob of error can thus also not be reduced $\leftarrow$ [Non zero rate code with smaller rate is not possible for $n=3$]

To reduce $P(\text{error})$ further we have to increase n,

So if we pick $n=4$, then we can get codes of rate $< \frac{1}{3}$, & $P(\text{error})$ will be (intuitively) smaller

Intuitively, it seems like, if we want $P(\text{error}) = \epsilon$, for some extremely small ϵ, then we should expect should be very close to 0.

Surprisingly this is NOT the case (at least in the ideal sense) (assuming we have freedom in choosing n):

For ANY (small) $\epsilon > 0$, there exists a code $\mathcal{C}$ with $P(\text{error}) \leq \epsilon$, & rate of the code $R(\mathcal{C}) = \left[\max_{P_x} I(X;Y)\right] - f(\epsilon)$ → small for small ϵ.

(maximization is over all possible input distributions)

Achievability

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

Note that $I(X;Y)$ = channel depends on (a) $P_{Y|X=x}$ $\forall x \in \mathcal{X}$

(b) Distribution of P_X

Note:
X is not a 'natural source' upon which we have no control, but it is the output of some encoder which encodes the 'raw source'.

So $P_X(x)$ is generally assumed to be 'controllable' in the mathematical framework of Info. Theory.

P_Y also, but this is dependent on P_X & $P_{Y|X}$

$\hookrightarrow P_Y(y) = \sum_{x \in \mathcal{X}} P(x,y)$

$= \sum_{x \in \mathcal{X}} P(x)\, P_{Y|X}(y|x)$

The quantity $\max_{P_X} I(X;Y)$ is called the "Channel Capacity". Generally denoted by C.

Channel coding theorem:

Achievability: Goes in next page.

Converse: No matter what we do, we cannot get a code with rate $> C$ (channel capacity) & expect small probability of error.

Note: To make the rate very close to C, we have to have very high value of the code length n.

Eg: Binary Symmetric Channel:- ("Bit-flip channel")

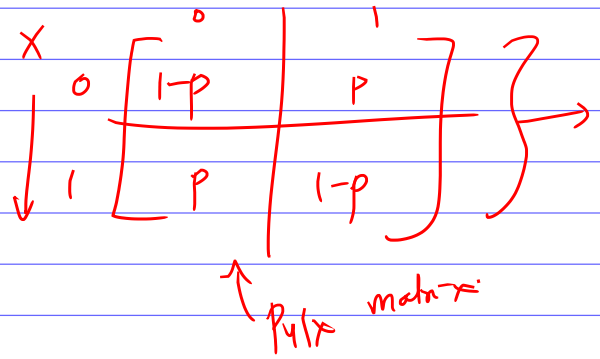
↓

Binary Input / Binary output : BSC(p)

$\mathcal{X} = \{0,1\} = \mathcal{Y}$.

$$P_{Y|X}(y|x=0) = \begin{cases} p & , \text{if } y=1 \\ 1-p & , \text{if } y=0 \end{cases}$$

$$P_{Y|X=1}(y|x=1) = \begin{cases} 1-p & , \text{if } y=1 \\ p & , \text{if } y=0 \end{cases}$$



Symmetric matrix

This is a 'symmetric' channel.

↑ $P_{Y|X}$ matrix

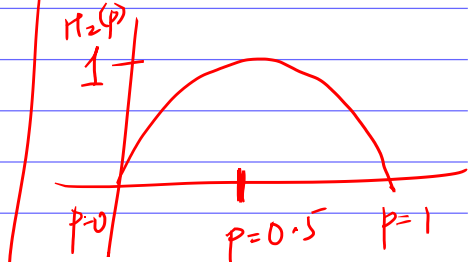
Note that this means $H(Y|X=0) = H(Y|X=1)$

$\Rightarrow H(Y|X) = H_2(p) = H_2(p)$

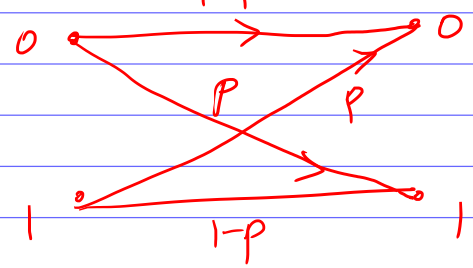
$H_2(p) \triangleq -p\log_2 p - (1-p)\log_2(p)$

↓

Binary entropy (for p)



Picture: $\mathcal{X}$ — $P(y|x)$ on the edges — $\mathcal{Y}$



We want to calculate $\max_{P_X} I(X;Y)$

$$I(X;Y) = \underbrace{H(Y)}_{?} - \underbrace{H(Y|X)}_{\hookrightarrow H_2(p)}$$

$\left[\max_{P_X} = H(X) - H(X|Y) \right]$ ↑ Exercise: compute this

$$H(Y) = -\sum_{y \in \mathcal{Y}} p(y) \log_2 p(y)$$

$$\max_{P_X} \Big(H(Y) - H(Y|X) \Big) = \Big(\max_{P_X} H(Y) \Big) - H_2(p)$$

↳ $H_2(p)$ which is not affected by choice of P_X as p is only dependent on the channel, which cannot be changed.

[it is given statement that channel transition prob = p]

$\max_{P_X} H(Y) \leq 1$ (this is already known to us)

Question : Can it ever attain 1 for any P_X ?

Answer : Yes, $H(Y) = 1$ for P_X being the uniform distribution

↳ Exercise. (Please try!)

↳ Show that If P_X is uniform then P_Y is also uniformly distributed

Channel capacity of BSC(p)

$$C_{BSC(p)} = 1 - H_2(p).$$

Class no 14 :-

We will show a converse for the BSC(p) capacity.

i.e.,

★ we will show, no matter what we do,

we cannot get a rate to be $> C_{BSC(p)} = 1 - H_2(p)$

& simultaneously have small $P(\text{error})$

Let $\mathcal{C}$ be the code which has rate R, & Perror very small.

We want to show that $R < C$.

$$R = \frac{\log_2 |\mathcal{C}|}{n} \Rightarrow |\mathcal{C}| = 2^{nR}$$

(n is very large : assumption)

Suppose that $\underline{c} \in \mathcal{C}$ is transmitted,

then

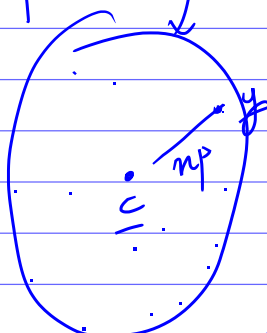
$$\underline{c} = (c_1, \ldots, c_n) \longrightarrow BSC(p) \longrightarrow \underline{y} = (y_1, \ldots, y_n)$$

→ There are $\overset{\text{(approx)}}{\sim} np$ positions in $\underline{c}$ which are flipped to get $\underline{y}$. { Channel is independently acting on each input bit }.

→ but decoder doesn't know which positions are flipped.

→ We can expect any seq in the set $S(\underline{c})$ as the o/p sequence, with high probability.

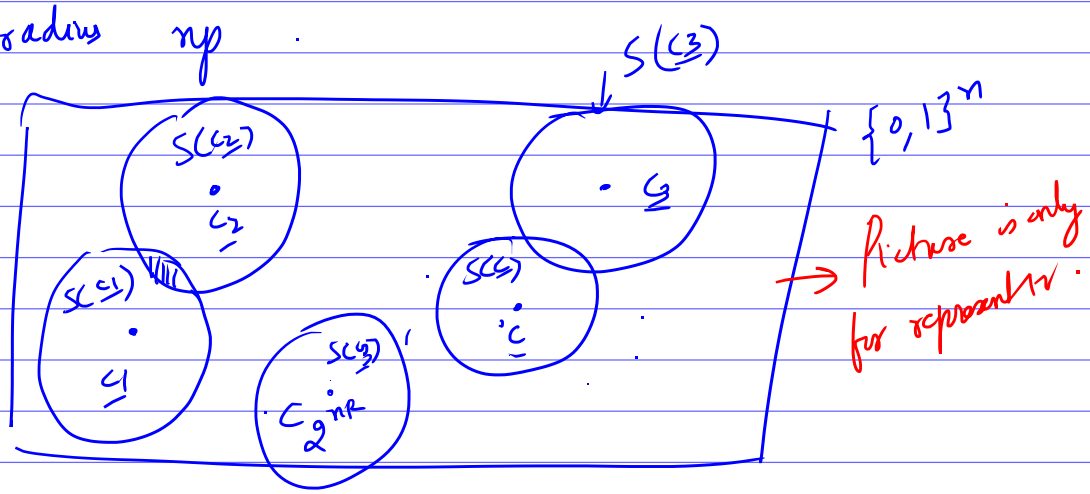
$$S(\underline{c}) = \{ \underline{z} \in \{0,1\}^n : d_H(\underline{c}, \underline{z}) = np \}$$

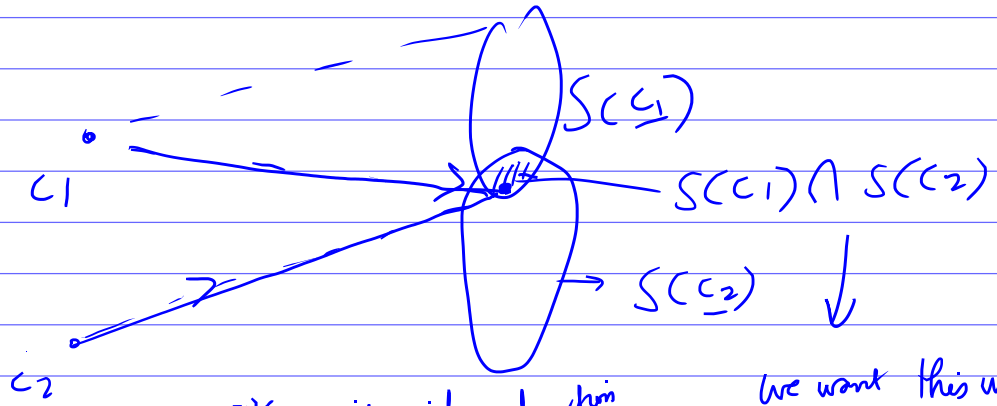


↓

Hamming distance bw $\underline{y}$ & $\underline{c}$ ≜ no of positions in $\underline{c}$ which we have to flip to get $\underline{y}$.

Now around every codeword we draw this 'Hamming ball' of radius np.





Otherwise, if intersection is non-empty, then there will be a many-one mapping from $\mathcal{C} \to$ Rx vectors, which will mean decoding error (or $P(\text{error})$ will be high.

← We want this intersection to be empty.

Because the code $\mathcal{C}$ has small P_{error}, this means that the balls around the codewords in $\mathcal{C}$ are non-intersecting

$$\Rightarrow |\mathcal{C}| \leq \frac{2^n}{|\text{No of vectors in any ball}|}$$

$\hookrightarrow |S(c)| = \text{same}$ for any $c \in \mathcal{C}$.

But $|S(c)| = |\text{No of vectors in any ball}|$

$$= \binom{n}{np}$$

$$|\mathcal{C}| \leq \frac{2^n}{\binom{n}{np}}$$

Recall from Source coding ↑

$$\log_2 |\mathcal{C}| \leq n - \log_2 \binom{n}{np} \approx n - nH_2(p)$$

$$\Rightarrow R = \frac{\log_2 |\mathcal{C}|}{n} \leq 1 - H_2(p) \longrightarrow \text{Converse is over.}$$

$$\boxed{R \leq C_{BSC}(p)}$$

Achievability to be done later:

Claude Shannon: "A mathematical theory of communication" Bells System Journal, 1948.

For a class of channels called Discrete Memoryless Channels (without feedback)

- Shannon's achievability result for Channel Coding was not constructive [not identifying a specific code which works];
- it was 'existence' proof ['We show that such a code exists but we don't know what it is precisely!].
- It took about 40-50 years to come up with candidate constructions which have rate close to capacity, & very small prob of error, which could actually be implemented in the real world

Theory & practice of Coding theory = Theory of Codes

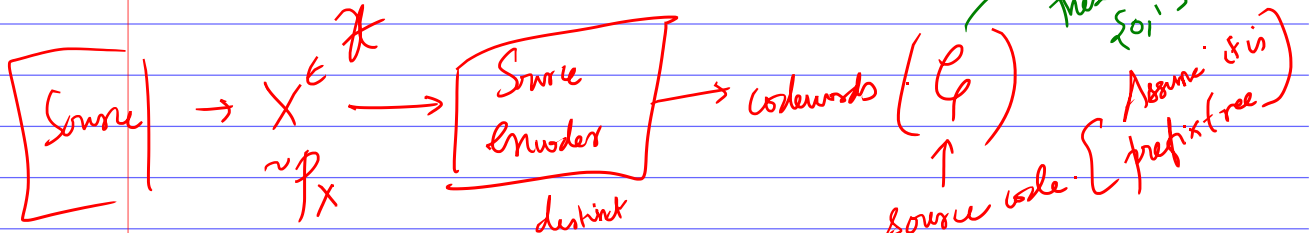
① Info theory = Used to model the comm systems & find out limits of communication (without actual constructions)

② Coding theory = Theory & practice of codes which are constructible/explicit constructions.
Synthesis & Analysis

③ Communication theory :— We have limits & codes from the above ① & ② → CT enables us to actually realize the comm systems in practice.

Class no 15 :-

Go back to Source coding:



contains codewords for each symbol in $\mathcal{X}$ which has non zero probability.

$C(x)$ be the codeword assigned to $x \in \mathcal{X}$.

$l(x)$ be the length of the codeword assigned to x.

(Fixed – Variable length source coding)

length ↑ length of some seq (here = 1)

$\varphi = \{C(x) : x \in \mathcal{X}\}$ ↑ source code.

Average length of code $L_C = \sum_{x \in \mathcal{X}} p(x) \ell(x)$

Goal: is to design a prefix-free code C which has minimum L_C.

Optimal length $L^* = \min_C L_C$

Lemma:

$$L^* \geq H(X) \cdot \text{for}^*$$

$\Rightarrow$ Any prefix free code has average length atleast $H(X)$

Proof:

To prove this we need the following claim

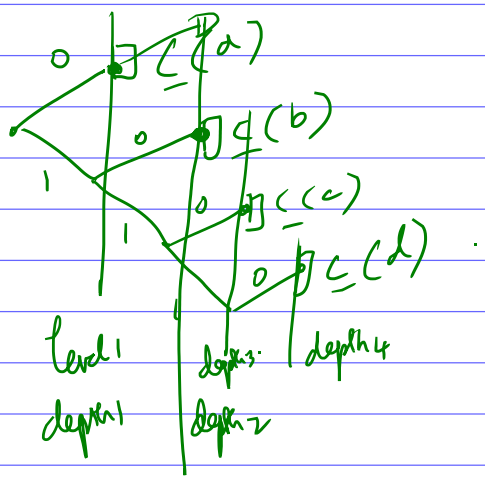
Claim: Let C be any pre-free binary code. Then

[Kraft's inequality] $\sum_{x \in \mathcal{X}} 2^{-\ell(x)} \leq 1.$

Proof:

We know that any prefix-free code can be represented using a binary tree which has the 'leaves' as the codewords.

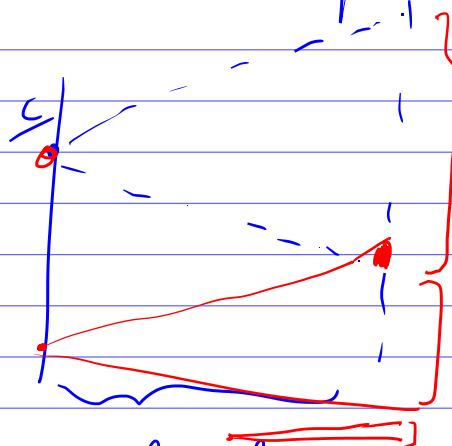
$\mathcal{X}$	$C(x)$
a	0
b	1 0
c	1 1 0
d	1 1 1 0
e	1 1 1 1



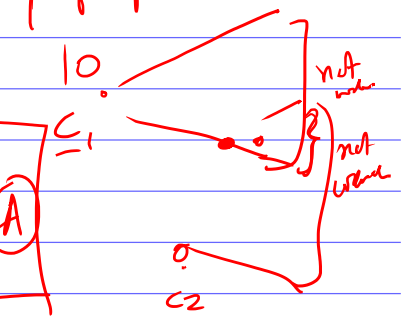
For the binary tree corresponding to the code, what is the depth of the tree? → = Length of the largest codeword in the code ($l_{max} = \max_{x \in \mathcal{X}} l(x)$)

Suppose there is a codeword $\underline{c}$ with length l. $(l \le l_{max})$

↓ represented by a node in the tree at depth l.



Is $\sum_{\underline{x} \in \mathcal{X}} 2^{l_{max} - l(\underline{x})} \le 2^{l_{max}}$? (A)



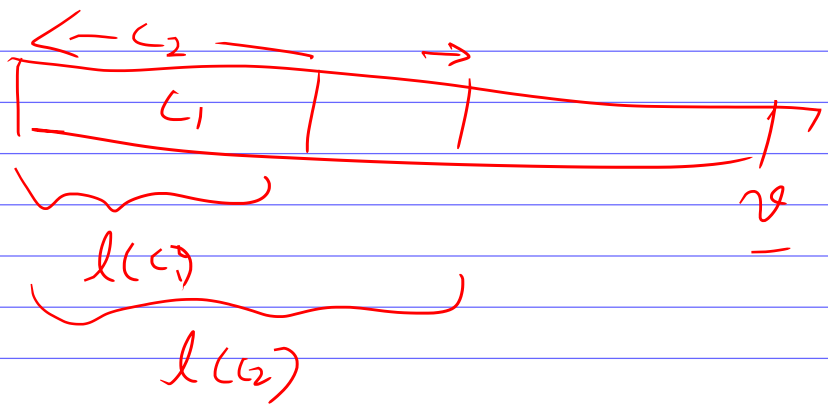
→ This is true if distinct codewords $\underline{c}_1, \underline{c}_2$ dont have common successors at l_{max} level.

→ Any successor of $\underline{c}_1$ at l_{max} level has c_1 as a prefix.

→ Illegal for c_2 as well.

So Suppose $l(\underline{c}_1) \le l(\underline{c}_2)$ there can be a common successor $\underline{w}$ at l_{max} level

Then we should have that



$\Rightarrow$ c_1 should be a prefix of c_2, which is not true as $\mathcal{C}$ is a prefix free code.

$\Rightarrow$ No pair of the codewords in $\mathcal{C}$ have any common successor.

$\Rightarrow$ (A) is true

$$\sum_{x \in \mathcal{X}} 2^{-l(x)} \leq 1$$

$\rightarrow$ This proves the claim (Kraft inequality for p-f code).

Now back to lemma proof. For any p-f code $\mathcal{C}$ we must have $L_{\mathcal{C}} - H(X) \geq 0$.

$$\sum_{x \in \mathcal{X}} p(x)\, l(x) - \sum_{x \in \mathcal{X}} p(x) \log \frac{1}{p(x)} \geq 0$$

Recall $D(p_X \| q_X) = \sum_{x \in \text{supp}(p_X)} p(x) \log \frac{p(x)}{q(x)}$.

Let $q_X(x) \triangleq \left(\frac{2^{-l(x)}}{\sum_{x' \in \mathcal{X}} 2^{-l(x')}} \right)$ (≥ 0) $\rightarrow$ Note that this satisfies $\sum_{x \in \mathcal{X}} q(x) = 1$

Hence this is a valid prob dist

Now $D(p_X \| q_X) = \sum_{x \in \text{supp}(p_X)} p(x) \log \frac{p(x)}{\frac{2^{-\ell(x)}}{\sum_{x' \in \mathcal{X}} 2^{-\ell(x')}}}$

$= -\sum p(x) \log \frac{1}{p(x)} + \sum_{x \in \text{supp}(p_X)} p(x) \log \left(\frac{1}{2^{-\ell(x)}} \sum_{x' \in \mathcal{X}} 2^{-\ell(x')} \right)$

$= -H(X) + \sum_{x \in \text{supp}(p_X)} p(x) \log 2^{\ell(x)} + \sum_{x \in \text{supp}(p_X)} p(x) \log \left(\sum_{x' \in \text{supp}} 2^{-\ell(x')} \right)$

$\leq -H(X) + L_\varphi$ (this is some nonpositive number) $\hookrightarrow \leq 1$ by Kraft.

$0 \leq D(p_X \| q_X) \leq -H(X) + L_\varphi$

$0 \leq$ known result

$\Rightarrow L_\varphi - H(X) \geq 0$

$\Rightarrow L_\varphi \geq H(X)$ //

Equality happens for $p_X = q_X$ & this term being 0.

Class 16

Kraft's ineq: For a gn prefix free code with length function $\ell(\cdot)$,

$$\sum_{x \in \mathcal{X}} 2^{-\ell(x)} \leq 1.$$

Lemma

Now suppose that we have a RV $X \in \{x_1, \ldots, x_K\}$ & +ve integers $\sum_{i=1}^{K} 2^{-l_i} \leq 1$.

Then there exists a p-f code for X with codeword lengths $l_1, \ldots, l_K$.

Proof:

We show that we can construct a binary tree with nodes at depths (levels) $l_1, \ldots, l_K$ such that none of these nodes are successors of each other

($\Leftrightarrow$ they are the leaves of some binary tree $\Rightarrow$ they represent a valid p-f code).

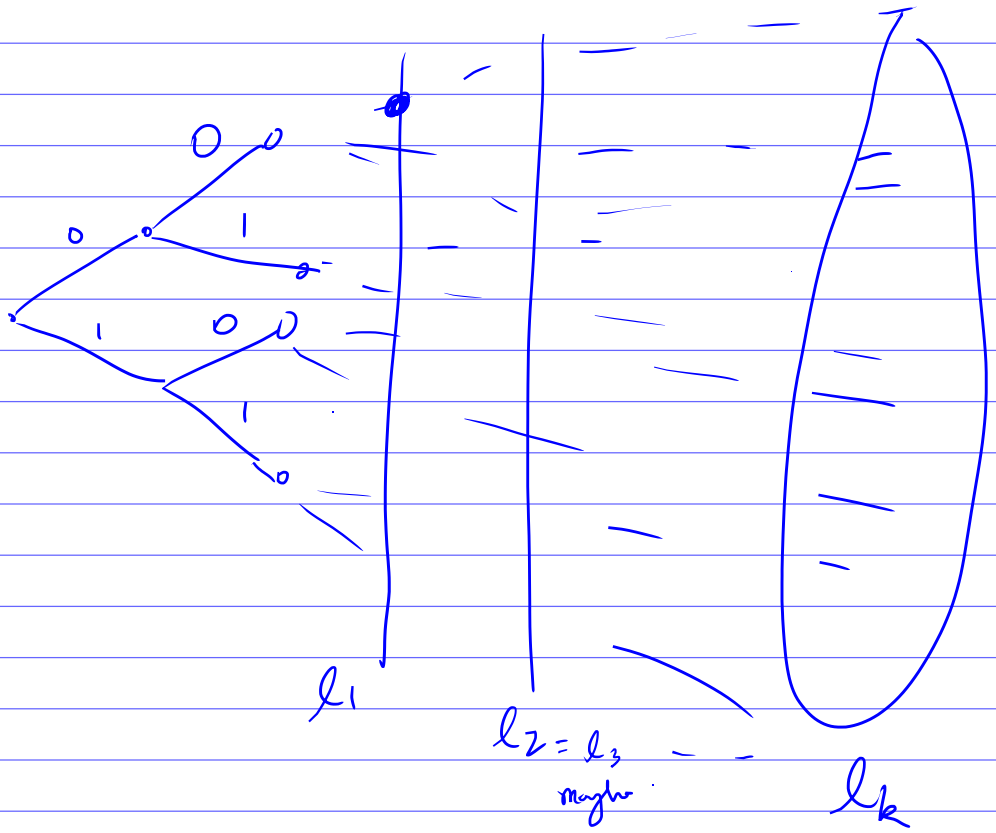
Assume that "Without loss of generality" $l_1 \leq l_2 \leq \cdots \leq l_K$.

Observation

For any $i \leq K$, $\sum_{j=1}^{i-1} 2^{-l_j} < 1 \rightarrow$ (A)

(by given statement)

Imagine that we take the full binary tree upto level l_K.

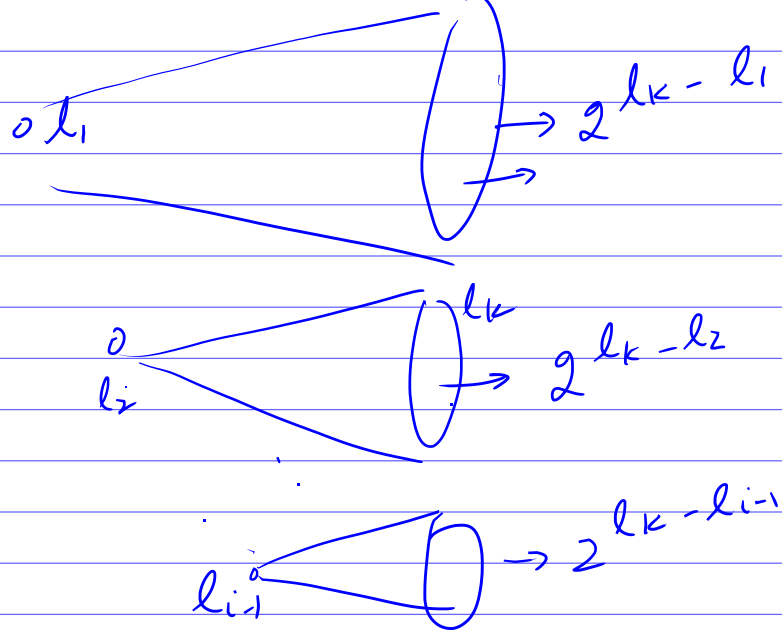


At each step i of the algorithm, we 'intend' to pick one available (undeleted) node from the above tree at level l_i & delete all its successors from the tree. Repeat this process for $i = 1, \ldots k$ we will then have a 'p-f tree'.

We have to show that at each step $i = 1, \ldots, k$, there is atleast one node left undeleted at level (depth) l_i. For this we will use Observation (A).

Base case: Clearly at step 1, there is a node at level l_1, $(l_1 \geq 1)$.

Induction: For any $i < k$, after $(i-1)$ steps, assume that we have picked & appropriately deleted nodes at levels $l_1, \ldots, l_{i-1}$. We want to show there is node at level i.



Total nodes at level l_K which are not in the tree after $(i-1)^{th}$ step $= \sum_{j=1}^{i-1} 2^{l_K - l_j}$

No of Nodes remaining at level l_K : $2^{l_K}\left(1 - \sum_{j=1}^{i-1} 2^{-l_j}\right) > 1$

< 1 by Kraft's (A)

$\Rightarrow$ No of nodes remaining at l_K in tree after $(i-1)^{th}$ step > 1

$\Rightarrow$ At least one survivor node should be present at level l_i also. So we can pick a node for the i^{th} step from level l_i also. //

This completes the proof.

Remark: We construct the tree from smallest length to largest length codewords. Contrast this with Optimal Source code construction that will follow later (Huffman codes).

Now, suppose that the source RV $X \sim p_X$. $(|\mathcal{X}| = k)$ $[\text{supp}(p_X) = \mathcal{X}]$

We want to obtain a collection of integers $l_1, \ldots, l_k$ (+ve) such that $\sum_{i=1}^{k} 2^{-l_i} \leq 1$. Then we know how to obtain a p-f code for X.

If we want small avg length

$$\sum_{i=1}^{k} p_i l_i$$

Then choose small l_i for larger p_i.

Idea: (behind Shannon-Fano codes as in McEliece book)

For $i = 1, \ldots, k$

Fix $l_i = \lceil \log_2 \frac{1}{p_i} \rceil \rightarrow (1)$

where p_i is the prob of X taking the i^{th} value in $\mathcal{X}$.

Clearly $l_i \geq 1$.

Checking K-inequality.

$$\sum_{i=1}^{k} 2^{-l_i} = \sum_{i=1}^{k} 2^{-\lceil \log_2 \frac{1}{p_i} \rceil} \leq \sum_{i=1}^{k} 2^{-\log_2 \frac{1}{p_i}}$$

$$= \sum_{i=1}^{k} p_i = 1$$

$\Rightarrow$ The lengths given by (1) satisfy K-inequality.

$\Rightarrow$ We can use the tree-growing algorithm to get a p-f code for X.

Suppose all codewords are of same length

$$\sum_{i=1}^{k} 2^{-l} \leq 1$$

$$k 2^{-l} \leq 1$$

$$2^l \geq k$$

$$l \geq \log_2 k$$

$\Rightarrow$ Fix $l = \lceil \log k \rceil$ will work ("ceil")

$\rightarrow$ But there is no guarantee that this code is 'good' i.e., it may not have small avg length L

This PF code obtained is called as the Shannon - Fano code for X.

Q: What is the L value for Shannon Fano code?

$$L_{\text{Shann. Fano}} = \sum_{i=1}^{k} p_i \left\lceil \log_2 \frac{1}{p_i} \right\rceil$$

$$< \sum_{i=1}^{k} p_i \left(\log_2 \frac{1}{p_i} + 1 \right)$$

$$L_{s-f} < H(X) + 1$$

↓ Great!

→ But SF code is not always an optimal (longest) p-f code

↓ is not always the smallest

L_{s-f} length among any p-f code for X.