ITC : Mod 2

(Tutorial 3)

→ Classification of codes
- Fixed length
- Variable length
- Distinct
- Prefix / Instantaneous — No cw should be prefix of another
- Uniquely decodable
- Optimal – instantaneous & has minimum L

→ Instantaneous code
- Constrution
- Decision Tree
- Kraft's Inequality

$$\boxed{\sum_{k=1}^{L} 2^{-n_k} \leq 1}$$

→ Shannon Fano

① Write P in descending order

② Divide equally (P) → Upper 0, Lower 1 } Do until last "repeatedly"

③ Write codeword & its length (n_i) for each symbol

④ Efficiency $\eta = \frac{H}{\hat{H}}$ $[0-1] / [\%]$

Entropy $H = -\sum_{i=1}^{n} P_i \log_2 P_i$ [bits/symbol]

$\hat{H} = \sum_{i=1}^{n} P_i n_i$ [bits/symbol]

Redundancy $R = 1 - \eta$

⑤ For ambiguity where equal division of P is not possible, we solve for both cases, find min $\hat{H}$, as η needs to be max.

→ Huffman Coding

① Write symbols & Ps in descending order
② Combine last two & move as high as possible. Hence, rearranging others.
③ Repeat until last
④ Write codeword & length (n_i) for each symbol
Trace left to right but cw, R → L
⑤ $H = -\sum P_i \log_2 P_i$ (Entropy) [bits/symbol]

$L = \sum P_i n_i$ (Avg Codeword) [bits/symbol]

$\eta = \frac{H}{L \times \log_2 r}$ (Efficiency) [%]/[0 –]

r: ① Binary ② Ternary ③ Quaternary

$\sigma^2 = \sum P_i (n_i - L)^2$ (Variance)

⑥ In step 2
Combine last 2 $\xrightarrow{\text{if}}$ binary $\xrightarrow{\text{at last}}$ Remain 2
Combine last 3 $\xrightarrow{\text{if}}$ ternary → Remain 3
Combine last 4 $\xrightarrow{\text{if}}$ quarternary → Remain 4
↓ if not
Add one symbol with probability 0 at the start

→ Arithmetic Coding

① Create a line with given probabilities 0 → 1
② Follow the pattern of code given to magnify
③ After magnification again create divisions
④ UL = LL + P of given code × total of that part
⑤ For repeated letters in code, repeat/(create) line.

A **discrete memoryless source (DMS)** in information theory is a source that generates a sequence of symbols from a finite set, where each symbol is independent of past symbols. This means the probability of producing a symbol depends only on its predefined probability distribution and not on previous symbols. Since it has no memory, past outputs do not affect future ones, making it simpler to analyze in communication and coding theory.

→ Arithmetic Decoding

1. Create line similarly (low to up)
2. Assign values from given Probabilities
3. Figure out the range in which value to be decoded falls
4. zoom in to it & calculate P 's again for newline
5. Final range / lower value based on decimal digits
6. The trail of zoomed in letters is the decoded value.

→ LZW Encoding

1. Given letters & starting dictionary (index, entry)
2. Create table containing encoded o/p, index, entry
3. Trace input & create new entries
4. The already existing part of new entry → encoded o/p (उसका index)
5. start new trace from last letter of previous

→ LZW Decoding

1. Given sequence of numbers & starting dictionary
2. Create decoded letters based on the given info
3. Keep adding new entries to dictionary based on live decoded letters
4. Both go simultaneously.

→ Run length encoding

a) If bits 0 & 1 given

1. count bits
2. bit value : count
3. Convert count to 8421 form
4. Concatenate everything
5. Compression ratio = $\frac{\text{new length}}{\text{old length}}$

$2^4 \geq$ ~~4 bits~~ 5 bits | std : 4 b
15
$2^5 \geq$ 6 bits | Gr than 15

b) If letters given

① Count

② Write finally in the form Num Letter . . . .

Remember : for CR , 1 is counted only once (either letter or number)

→ Source Coding Theorem

$$\left.\begin{aligned} H(X) &\leq \log_2 L \\ -\sum_{i=1}^{L} P(x_i) \cdot \log_2 P(x_i) &\leq \log_2 L \end{aligned}\right\} \text{Equality holds when the symbols are equally likely}$$