CURS 2

TEORIA INFORMAȚIEI

- Informația = date prelucrate
- Cantitatea de informație: $I(x_i) = \log_2\left(\frac{1}{p_i}\right)$

ex: pachet de cărți = 52

$I(\text{roșu}) = \log_2\left(\frac{\frac{1}{4}}{\frac{52}{52}}\right) = \log_2\left(\frac{52}{4}\right) = 3,7$ biți

unde p_i = probabilitate

- Cantitate informație primită: $I = \log_2\left(\frac{N}{M}\right)$

unde N = nr evenimente și M = nr. evenimente posibile

- $M \geq N$ => incertitudine
- $M = N$ => incertitudine la o singură variantă = perfect
- $M \gg N$ => incertitudine mare

ENTROPIA

- Entropie = val. medie de inf. primită despre o var. X

$$H(X) = E(I(X)) = \sum_{i=1}^{N} p_i \cdot \log_2 \frac{1}{p_i} = \sum -p_i \log_2 p_i$$

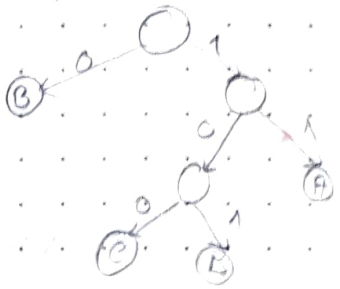
$H(X)$ = entropia lui X
$I(X)$ = inf. despre X
E = expected value

- Entropia este limita de compresie posibilă

CODAREA DATELOR

- Dacă $X = \{A, B, C, D\}$ cu probabilități $\{1/3; 1/2; 1/12; 1/12\}$ are entropia lui X, $H(X) = 1.626$
- Codarea trebuie să fie unică

- O variantă: ABBC = 00 01 01 10 → A=00, B=01, C=10, D=11
- O altă variantă mai eficientă: ABBC = 01 1 1 000
- O codare eficientă și unică se poate face printr-un arbore binar:
  - frunzele = coduri
  - stg/dr sunt 0/1



ex: B=0, A=11, C=100, D=101

- Cum se generează codarea eficientă?
  - algoritmul Huffmann
  - input: probabilitatea fiecărui eveniment
  - output: codurile se citesc de pe arbore
  - cheia: unele evenimente apar mai des → deci ele primesc o codare mai scurtă (dacă sunt egale nu face nimic)
- Calcularea eficienței: $2 \cdot \frac{1}{3} + 1 \cdot \frac{1}{2} + 3 \cdot \frac{1}{12} + 3 \cdot \frac{1}{12} = 1{,}667 > 1{,}626$ (de)
- Codare cu dimensiune fixă (indiferent de nr. → folosim N biți)
- Algoritmul Huffman:
  - luăm evenimentele cele mai improbabile
  - se creează un nou eveniment (CD), iar probabilitatea lui devine suma: $\frac{1}{12} + \frac{1}{12} = \frac{1}{6}$
  - se reiau evenimentele cele mai improbabile și se refac pașii de mai sus

DETECTAREA & CORECTAREA ERORILOR

1. Definim o distanță între șirul corect și cel corupt
2. Distanța Hamming: câți biți sunt diferiți / de pe aceleași poziții
3. Șirurile trb să aibă aceeași lungime

ex: 0 1 1 0 0 1 0 1 (inițial)
1 1 0 0 0 0 0 1 (cel corupt) ⟹ distanța = 3

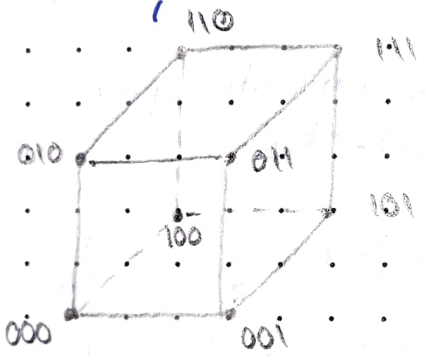
- Cu această metodă → nu toate șirurile vor fi valide (nu putem merge)

- O altă idee este adăugarea unui bit de paritate:
  - 0 dacă are 00 (la biți doi)
  - 1 dacă are 11

Dacă primim 01 sau 10 => s-a produs eroare

Problema: mai ineficient și erorile multiple nu se pot detecta

- Corecția erorilor:
- Se poate face tot cu distanța Hamming
- Pt. detectarea a „E" erori => distanță între coduri de E+1
  ex: 0 poate să fie 000 /=> se pot corecta 2 erori
  1 poate să fie 111
- O distanță Hamming de 2E+1 poate corecta E erori
- Pt. șiruri de 3 biți



- Singurele coduri valide sunt 000 care e 0 sau 111 pt 1
- Dacă primim 001 sau 010 sau 100 putem suspecta că e 0
- Dacă primim 110 sau 011 sau 101 — // — că e 1

VEZI ȘI LECTURĂ SUPLIMENTARĂ!