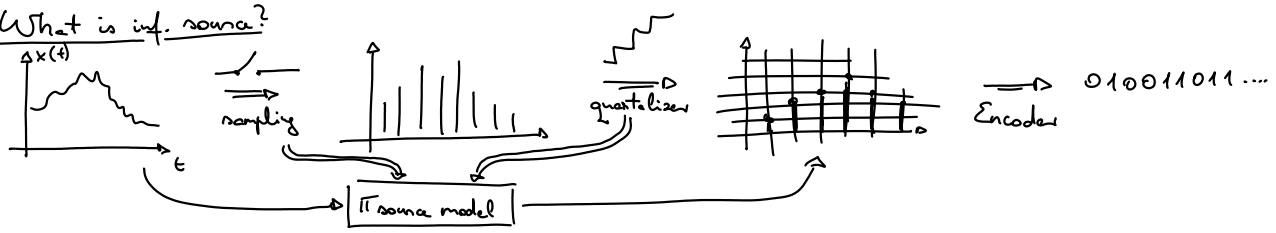
Data Compression - Source Coding

2016. november 9., szerda 16:20

a) [information source] (low speed) → [Source Coding] → shortest binary representation

b) high data speed → [Access ch.] (low speed)

What is inf. source?

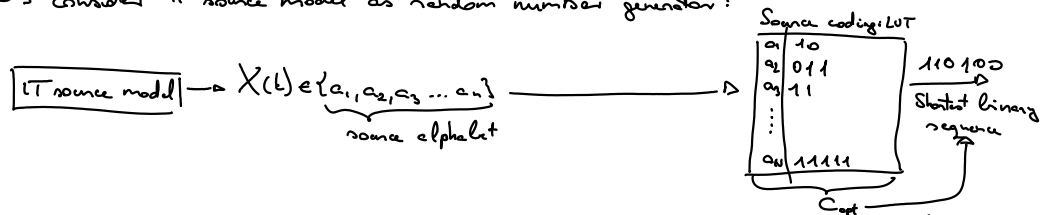


Data speed: $f_s \cdot n$ [bps]

- Speech: 8 kHz · 8 bit ⇒ 64 kbps
- Music: 44.1 kHz · 16 bit ⇒ 705.6 kbps
- Video: 84 Mbps

} Compression ⇒ GSM: 13.2 kbps

Let's consider π source model as random number generator:



$P(X(t) = a_i \mid X(t-1) = a_j, \ldots, X(0) = a_m) = P(X(t) = a_i)$ ⇒ Memoryless model

Stacionary

Model:

$p_1, p_2, \ldots p_N \to p(x)$ ← probability

$a_1, a_2, \ldots a_N \to X$ ← symbols

$c_1, c_2, \ldots c_N \to c(x)$ ← code

$l_1, l_2, \ldots l_N \to l(x)$ ← length of code

Average codelength: $L := \mathbb{E}(l(x)) = \sum_x p(x) l(x)$ (expected value) ⇒ Data speed: $f_s \cdot L$

→ Optimal source coding: $C_{opt}: \min_C L$

Intuitive idea: → if $p(x)$ is large, $l(x)$ is small

→ if $p(x)$ is small, $l(x)$ is large

Questions:

1) Unique decoding: how can I decide where the new codeword starts

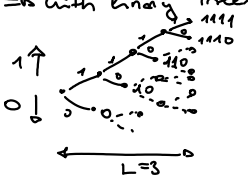
2) Optimal $l(x)$?

⇒ Example: Ineedacoffeeverymuch ⇒ no breaks or spaces, but we can divide it to words

Why? → because it is a prefix-free code ⇒ This is not: $c_1 = 01$, $c_2 = 0111$

→ How can we generate prefix-free codings?

⇒ with binary trees



number of missing nodes: $\sum_x 2^{L-l(x)}$

all possible nodes 2^L

→ missing nodes should be lower than all nodes: $\sum_x 2^{L-l(x)} \leq 2^L$

$\Rightarrow \boxed{\sum_x 2^{-l(x)} \leq 1}$

Kraft-McMillan

information

$I(x) = \ld \frac{1}{p(x)}$ ⇒ $H(x) = \mathbb{E} I(x) = \sum_x p(x) \ld \frac{1}{p(x)}$ ← entropy!!!

↓ random variable

↓ we multiply positive numbers,

$H(x) \geq 0$

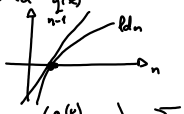
$\forall x \quad \pi(x) \geq 0$

$\Rightarrow 0 \leq H(x) \leq \ld N$

Los two extremities: 

I - divergence: $D(p \| q) := \sum_x p(x) \ld \frac{p(x)}{q(x)}$

$p(x)$ $q(x)$



$D(p,q) \geq 0$

$-D(p,q) = \sum_x p(x) \ld \frac{q(x)}{p(x)} \leq \sum_x p(x) \left(\frac{q(x)}{p(x)} - 1\right) = \sum_x q(x) - \sum_x p(x) = 1 - 1 = 0$

$D(p,q) - \sum_x p(x) \ld \frac{p(x)}{1/N} = \sum_x p(x) \ld (p(x) \cdot N) = \sum_x p(x) [\ld p(x) + \ld N] = \sum_x p(x) \ld p(x) + \sum_x p(x) \ld N =$

$p(x)$ $q(x) = \frac{1}{N}$

$= -H(x) + \ld N \geq 0 \Rightarrow H(x) \leq \ld N$

$\rightarrow p(x) = \frac{1}{N} \Rightarrow H(x) = \ld N$

Source code theorem: $L \geq H(x)$ (introduced by Shannon)

Proof: $p(x)$ source distribution

$q(x) = \frac{2^{-l(x)}}{\sum_y 2^{-l(y)}}$ artificial $\quad 0 \leq q(x) \leq 1 : \forall x$

$\sum_x q(x) = L$

$D(p,q) = \sum_x p(x) \ld \frac{p(x)}{q(x)} = \sum_x p(x) \ld \frac{p(x)}{2^{-l(x)} / \sum_y 2^{-l(y)}} = \sum_x p(x) \ld \left\{ p(x) \cdot 2^{l(x)} \cdot \underbrace{\left(\sum_y 2^{-l(y)} \right)}_{\leq 1} \right\} \leq \sum_x p(x) \ld \{ p(x) \cdot 2^{l(x)} \} =$

$= \sum_x p(x) \ld p(x) + \sum_x p(x) \ld 2^{l(x)} = -H(x) + L \geq 0 \Rightarrow H(x) \leq L$

Objective is to approach $H(x)$ as much as it is possible:

→ Shannon-Fano coding:

$l(x) := \lceil \ld \frac{1}{p(x)} \rceil$

$\sum_x 2^{-l(x)} \leq 1$

$\sum_x 2^{-\lceil \ld \frac{1}{p(x)} \rceil} \leq \sum_x 2^{-\ld \frac{1}{p(x)}} = \sum_x 2^{\ld p(x)} = \sum_x p(x) = 1$ ✓

$H(x) \leq L = \sum_x p(x) \lceil \ld \frac{1}{p(x)} \rceil \leq \sum_x p(x) \left(\ld \frac{1}{p(x)} + 1 \right) = H(x) + 1$

Algorithm: given $p(x)$

1) $l(x) = \lceil \ld p(x) \rceil$, $x \in \{a_1 \dots a_N\}$
2) Binary tree → C
3) LUT