ST06 LECTURE 08

Note Title 3/7/2006

Today

- Optimality of Huffman Codes
- "Universal Coding"
- Universal Coding for i.i.d. sources
- Universal Coding (theoretically) for Markovian Channels
- Lempel-Ziv Coding

Review of last lecture

Kraft's Inequality

DMC: $X = \{1 \dots n\}$

w.p. $p_1 \dots p_n$

$C : \{1 \dots n\} \to D^*$ with $C(i) \in D^{\ell_i}$

then $\sum D^{-l_i} \leq 1$ [for prefix free / uniquely decodable]

furthermore if $\exists\, l_1 .. l_n$ s.t.

$$\sum D^{-l_i} \leq 1$$

then $\exists$ C prefix free s.t

$$C(i) \in D^{l_i}$$

---

(1) Entropy l.b.

$\forall$ prefix / uniquely dec. code

$$E[L] \geq H(x)$$

$\forall$ non-singular code

$$E[L] \geq H(x) - O\left(\sqrt{H(x)}\right)$$

(2) Shannon Code : $l_i = \lceil \log p_i \rceil$ QED !

yields $\quad E[L] \leq H(x) + 1$

3 Huffman Code : (Binary version)

Code $(p_1 \ldots p_n)$:

- Sort so that $p_1 \geq p_2 \ldots p_n$;

- Let $C' \leftarrow$ Code $(p_1 \ldots p_{n-2}, p_{n-1} + p_n)$;

$$- \quad C[i] \begin{cases} = C'[i] & i \leq n-2 \\ = C'[n-1].0 & i = n-1 \\ = C'[n-1].1 & i = n \end{cases}$$

Return C ;

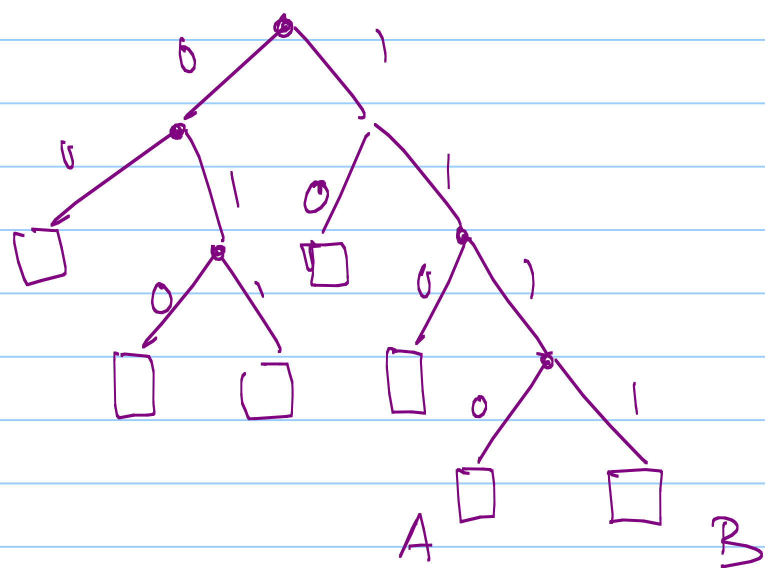
Huffman Code is optimal

$\forall$ prefix code $C : \{1 \ldots n\} \rightarrow \{0,1\}^*$

$$\sum_{i=1}^{n} p_i |C(i)| \geq \sum_{i=1}^{n} p_i \, |C_{Huff}(i)| \,.$$

Proof (by picture):

1. Prefix code looks like leaves of binary tree



2. For optimal if $p_i < p_j$ then $l_i \geq l_j$.

3. Tree is "full" (every node has two/zero children).

4. W.l.o.g. A, B have lowest probabilities

(5) $\Rightarrow$ length of C is optimal

$\Leftrightarrow$ length of C' is optimal.

Pro's & Cons of Shannon/Huffman Coding

(New).

+

- Optimal given what we know
- Should apply to $X^{(k)} = (X_1 \dots X_k)$

i.i.d $\sim X$.

Then the "+1" is amortized over k blocks.

-ve

- Needs to know the p_i's!
- Can we avoid this? Seems essential?
- if we code $p_1 \dots p_n$ according $q_1 \dots q_n$ then we $H(x) + D(p\|q)$.

· Memoryless Universal Encoders

$$C : \Omega^* \to \{0,1\}^*$$

is universal if

$\forall$ distribution p on Ω, $\epsilon > 0$

$\exists n_0 \, \forall n \geq n_0$ & $X_1 \dots X_t$ i.i.d $\sim p$

$$|C(X_1 \dots X_t)| \sim t \cdot H(X) \cdot (1+\epsilon)$$

Can such a C exist?

Easy:

- C first counts $\#_i = |\{j \mid X_j = i\}|$
- Outputs $(\#_1, \dots \#_n)$;

outputs which pattern in $\#_1 \dots \#_n$ it sees

Won't prove correctness ... but obvious it works.

Cons: - is Discrete Memoryless for real?

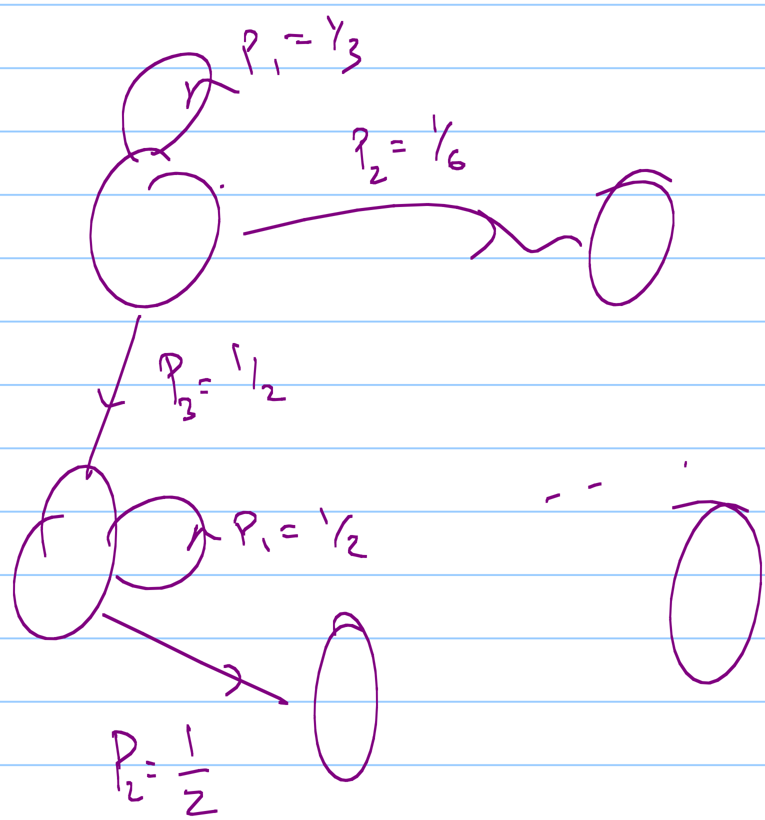
- Huge "Dictionaries" - Can we do better? → $|\Omega|^{t}$ ...

[Fax anecdote here]

Lempel-Ziv [LZ76, ZL77, ZL78]

- Nice, natural, efficient algorithm
- Universal coding for Markovian sources.

Markovian Source

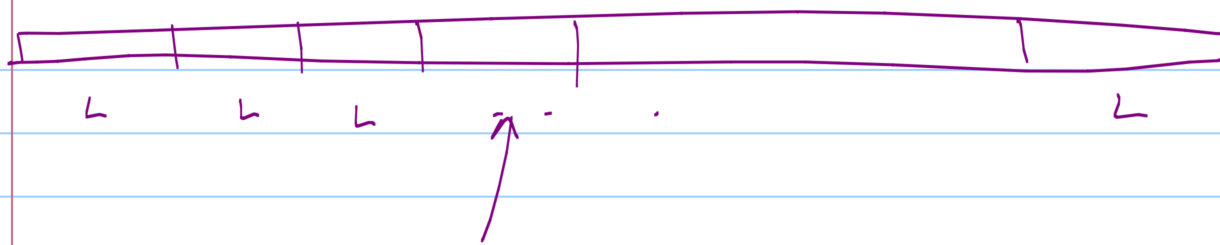


"Models Natural Languages" reasonably
[Shannon].

Compress efficiently?

Use AEP .....

A Shabby Algorithm



How many sequences do we expect to see?

$$2^{H(x)\cdot L}$$

- $H(x)$ = Entropy rate of Hidden Markov Chain.
- Other sequences very unlikely ... happen w.p. $\le \epsilon = \exp(-\Omega(L))$.

Encoding strategy. [SHABBY]

- Count # diff. patterns that appear more than k times. [k = parameter]

- Build dictionary of such patterns & send dictionary to receiver.

- For each block do:

  1. Send "typical" or "not typical";
  2. If typical, send index in dictionary
  3. Else send entire sequence.

$$\text{Cost} = \epsilon \cdot \left(L \cdot \lceil \log n \rceil + 1\right) \cdot t$$

$$\boxed{\text{Cost} \approx H(x) \cdot L \cdot t}$$

Dominates

$$\text{Cost} = 2^{H(x) \cdot L} \cdot L \cdot \lceil \log n \rceil$$

but no "t"!

vanishes as $L \to \infty$

vanishes as $t \to \infty$

- But how to choose L, k?
- Try all possibilities; use one which minimizes absolute length!

<u>Conclusion</u>: SHABBY is a universal encoder, but it is SHABBY

How can we make this more elegant?

<u>What is inelegant?</u>

1. "Explicit choices of L, k".
2. "Explicit & huge dictionary".

[ZL '78] :

Code as follows

$\sigma_i \in \Omega$

Step 1: $\sigma_1 \sigma_2 \ldots \sigma_t \ldots$

$$\Big\Downarrow \text{ Parse}$$

$B_1, B_2 \ldots B_{t'} \ldots$

B_i = sequence over Ω

has property that (1) $B_i \neq B_j$ $j < i$

(2) $B_i = B_j \cdot \sigma$ $j < i$

$\sigma \in \Omega$.

Encoding $B_1, \ldots B_{t'} \ldots$

$\Downarrow$

$

$B_i \longrightarrow (j, \sigma)$