187 - Day 15

Monday, May 7, 2018 9:56 AM

HW now due on Wednesdays 10pm.

Day 15 – Huffman Code and Random Cryptosystem

Recap - Decryption matrix for rectangular transposition

The following matrix was obtained from the applet for breaking rectangular transposition, using the ciphertext in the first problem of HW4.

	1	2	3	4	5	6	7
1		17	17	21	**34**	22	24
2	25		20	20	19	27	**35**
3	16	19		18	22	**34**	25
4	18	21	**33**		19	19	21
5	24	**32**	18	21		20	17
6	26	24	24	22	17		19
7	22	21	15	**32**	23	20	

Rule: If the big entry is on row i, col j then j follows i in decryption perm.

Find the decrypting and encrypting permutations.

. One row w/o big entry gives the last entry in perm.

. One col w/o big entry give the first perm entry.

$(a_1, a_2, a_3, a_4, a_5, a_6, a_7)$ = decrypting perm.

. Row 6 has no big entry $\Rightarrow$ $a_7 = 6$.

. To find a_6, look at Col 6 for the big entry.

On 6^{th} col, big entry is at row 3 (row 3, col 6)

$\Rightarrow a_6 = 3$

. To find a_5 $\Rightarrow$ look for big entry on col 3

$\Rightarrow a_5 = 4$

Decrypting perm: $(1, 5, 2, 7, 4, 3, 6)$

Encrypting perm. is the inverse of decrypting perm.

decrypting perm

$$(1, 5, 2, 7, 4, 3, 6) = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 1 & 5 & 2 & 7 & 4 & 3 & 6 \end{pmatrix} \quad (*)$$

To get the inverse, simply reverse the arrows in (*)

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ 1 & 3 & 6 & 5 & 2 & 7 & 4 \end{pmatrix}$$

$\Vert$

$(1, 3, 6, 5, 2, 7, 4)$: encrypting perm.

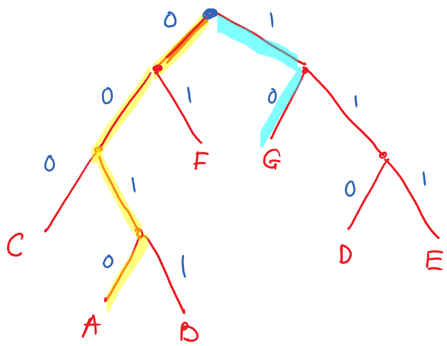
Example. Suppose a certain file contains only the letter with the following frequencies

frequency →

A	B	C	D	E	F	G
1	2	2	4	4	5	6

Construct the comma-free code that enables you to compress the file so that you can store it using the least number of bits.

- Sort frequencies in increasing order
- Group the smallest 2 entry, and replace w/ their sum.
- Re-sort, repeat ...



$A \to \underline{0010}$: need 4.

$\to G \to \underline{10}$: only 2 bits

$D \to \underline{110}$

More common letter
||
less binary bits needed to encode.

In general, if $\mathbb{P}(\alpha) = \dfrac{n_\alpha \;(\text{frequency of } \alpha)}{N \;(\sum n_\alpha = \text{length of plaintext})}$, then code length for α, is $\lceil \log_2 (1/\mathbb{P}(\alpha)) \rceil$

Letter	A	B	C	D	E	F	G
Frequency	1	2	2	4	4	5	6
Code	0010	0001	000	110	111	01	10
Bits	4	4	3	3	3	2	2

File length after encrypted is

$$\sum_\alpha \textit{code length}(\alpha) \cdot \textit{frequency}(\alpha) = 64$$

Average number of bits per letter is

$$\frac{\textit{ciphetext length}}{\textit{plaintext length}} = \frac{64}{24} \approx 2.66$$

Compare to the entropy of the file

$$\begin{aligned}\sum_\alpha \mathbb{P}(\alpha) \log_2\left(\frac{1}{\mathbb{P}(\alpha)}\right) &= \frac{1}{24}\log_2\left(\frac{1}{1/24}\right) + \frac{2}{24}\log_2\left(\frac{1}{2/24}\right) + \frac{2}{24}\log_2\left(\frac{2}{1/24}\right) \\ &+ \frac{4}{24}\log_2\left(\frac{1}{4/24}\right) + \frac{4}{24}\log_2\left(\frac{1}{4/24}\right) \\ &+ \frac{5}{24}\log_2\left(\frac{1}{5/24}\right) + \frac{6}{24}\log_2\left(\frac{1}{6/24}\right) \\ &\approx 2.62165\end{aligned}$$

Theorem 1. *Suppose the letter counts in the plaintext are n_1, n_2 ~~ldots~~ $\ldots$, n_k and let $N = n_1 + \ldots + n_k$. Then the best possible code length (in terms of bits per letter) is*

p/t length

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

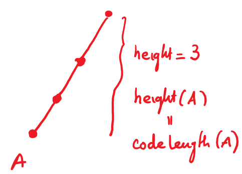
where $p_i = n_i/N$ for all $1 \le i \le k$.

prob. of each letter i in p/t.

Fact: If p_i's and q_i's are probability distributions

$$p_1 + p_2 + \ldots + p_k = 1 = q_1 + q_2 + \ldots + q_k.$$

then $$\sum_{i=1}^{k} p_i \cdot \log_2\left(\frac{1}{p_i}\right) \leq \sum_{i=1}^{k} p_i \cdot \log_2\left(\frac{1}{q_i}\right)$$ due to the fact that $\log_2(\cdot)$ is a concave fcn.



* Take any binary tree with leaf heights $h_1, h_2, \ldots, h_k$.

File length after encryption $= \sum_{i=1}^{k} n_i h_i$

$$= N \sum_{i=1}^{k} \frac{n_i}{N} \cdot \log_2\left(2^{h_i}\right)$$

$$= N \cdot \sum_{i=1}^{k} p_i \log_2\left(\frac{1}{q_i}\right) \geq N \cdot \sum_{i=1}^{k} p_i \cdot \log_2\left(\frac{1}{p_i}\right) = \left(\text{entropy of the p/t}\right) \cdot N.$$

where $q_i = \frac{1}{2^{h_i}}$

4

So $$\frac{\text{file length after encryption}}{N} \geq \text{entropy of p/t}$$

Theorem 2. *The Huffman tree constructed from the probabilities $p_1, p_2, \ldots, p_k$ yields an expected code length that is within **1 bit** of the entropy*

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

$$\text{Expected Codelength} = \sum_{i=1}^{k} p_i h_i = \sum_{i=1}^{k} p_i \left\lceil \log_2(1/p_i) \right\rceil$$

So,

$$\underbrace{\sum_{i=1}^{k} p_i \log_2\left(\frac{1}{p_i}\right)}_{H} \leq \sum_{i=1}^{k} p_i \left\lceil \log_2(1/p_i) \right\rceil \leq \sum_{i=1}^{k} p_i \left(1 + \log_2\left(\frac{1}{p_i}\right)\right)$$

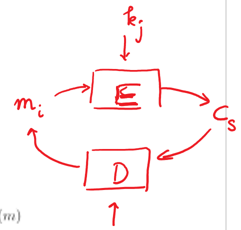
$$H \leq \text{expected code length} \leq \underbrace{\sum_{i=1}^{k} p_i}_{1} + \underbrace{\sum_{i=1}^{k} p_i \cdot \log_2\left(\frac{1}{p_i}\right)}_{H}$$

$$1 + H$$

Random Crypto-systems

The set up of cryptography

- Message/plaintext space $\mathcal{M} = \{m_1, m_2, \dots, m_N\}$
- Key space $\mathcal{K} = \{k_1, k_2, \dots, k_S\}$
- Ciphertext space $\cancel{\mathcal{M}} = \{c_1, c_2, \dots, c_Q\} = \mathcal{C}$
- The encrypting function corresponding to the key k: $c = E_k(m)$
- The decrypting function corresponding to the key k: $m = D_k(c)$



k_t (may or may not be the same as the encryption key)

In a random crypto-system:

random variables.

- $M \in \mathcal{M}$ is the chosen message $\quad \mathbb{P}(M = m_i) = p_i$
- $K \in \mathcal{K}$ is the chosen key $\quad \mathbb{P}(K = k_j) = q_j$
- $C \in \mathcal{C}$ is the resulting ciphertext

$C = E_{k_j}(m_i)$.

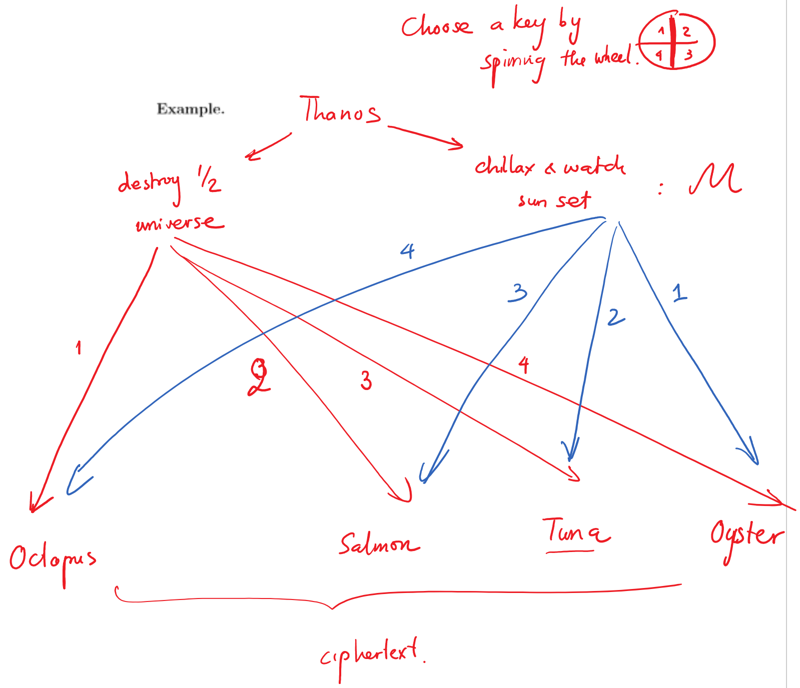
Remarks:

- We choose the key K independently of the message M. $\rightarrow$ $\mathbb{P}(M = m_i \cap K = k_j) = \mathbb{P}(M = m_i) \cdot \mathbb{P}(K = k_j)$ for all $m_i \in \mathcal{M}$, $k_j \in \mathcal{K}$
- $C = E_K(M)$ so the ciphertext C is a random variable which depends on M and K.
- $M = D_K(C)$ so $H(K, C) = H(K, M)$
- $H(K|C)$ is the remaining uncertainty about the key after we intercept the ciphertext.

$$H(K|C) = 0 \Leftrightarrow \text{the ciphertext determines the key} \quad : \text{very bad!}$$

uncertainty about K, C is the same as the uncertainty about K, M

Example.



ciphertext "Tuna"

good : if key = k_2

bad : if key = k_3

Theorem 3. *For* ***ciphertext only*** *attack:*

$$H(K|C) = H(K) + H(M) - H(C)$$

Theorem 4. *For **known plaintext** attack:*

$$H(K|C, M) = H(K) - H(C|M)$$

Definition 1. *A cryptosystem is said ti attain* ***perfect secrecy*** *if the* ***ciphertext gives no information about the plaintext****. That is,* M, C *are random variable, namely,*

$$\mathbb{P}(M = m_i \cup C = c_j) = \mathbb{P}(M = m_i) \cdot \mathbb{P}(C = c_j)$$

for all $m_j \in \mathcal{M}$ *and* $c_j \in \mathcal{C}$.