

Consider the code $\{0, 01\}$

(a) Yes, it is nonsingular. The two codewords are different.

(b) Yes, it is UD. Given a sequence of codewords, first isolate occurrences of 01 (i.e., find all the ones) and then parse the rest into 0's.

Remark: This code is suffix-free.

No codeword is a suffix of any other codeword.

Can you see why suffix-free codes are also UD?

(This fact can be seen by backward decoding.)

For any received sequence/string, we work backward from the end, and look for the reversed codewords. Since the codewords satisfy the suffix condition, the reversed codewords satisfy the prefix condition, and therefore we can uniquely decode the reversed sequence/string.)

(c) No, the code is not prefix-free. The first codeword, 0, is a prefix of the second codeword, 01.

Reminder: prefix-free code is the same as prefix code.

This is more accurate in terms of meaning.

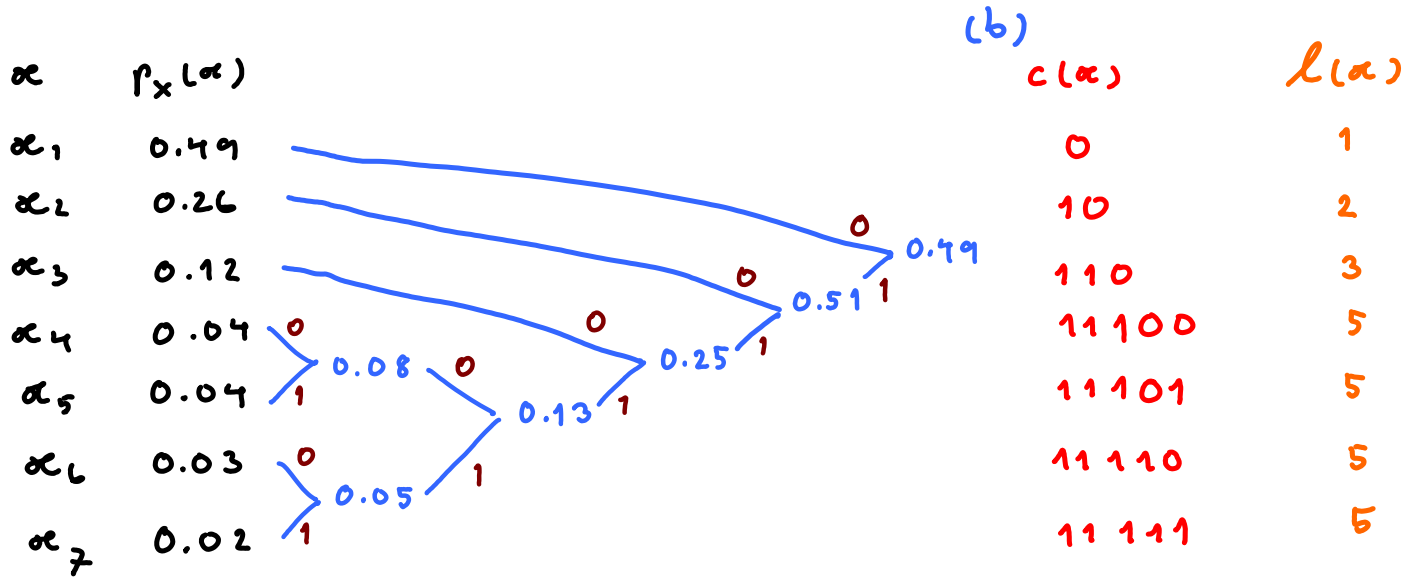
This is what has been traditionally used to refer to prefix-free code.

Remark: Observe that we can specify whether a code is nonsingular, uniquely decodable, or prefix-free even when there is no information about the shared alphabet and pmf of the DMS symbols.

Q2 Huffman Code

Sunday, August 28, 2011
9:38 PM

$$(a) H(x) = -\mathbb{E}[\log_2 p_X(x)] = -\sum_x p_X(x) \log_2 p_X(x) = 2.0128 \text{ bits}$$



$$(c) \mathbb{E}[l(x)] = 1 \times 0.49 + 2 \times 0.26 + 3 \times 0.12 + 5 \times (2 \times 0.04 + 0.03 + 0.02)$$

$$= 2.02 \text{ bits}$$

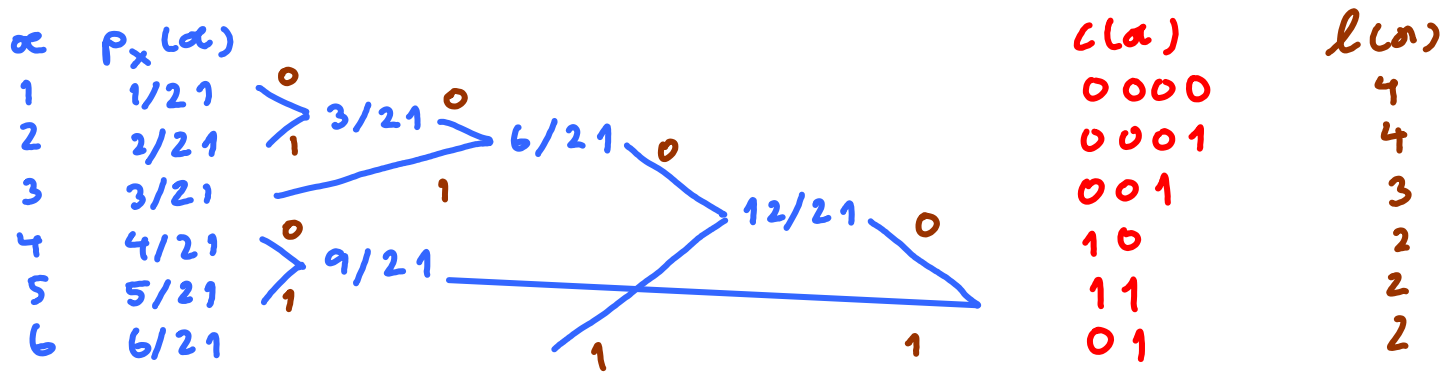
$\underbrace{2 \times 0.04 + 0.03 + 0.02}_{0.08}$
 $\underbrace{\hspace{1.5cm}}_{0.13}$

Q3 Huffman code

Sunday, August 28, 2011
10:13 PM

MATLAB

$$H(x) = -\sum_{\alpha} p_x(\alpha) \log_2 p_x(\alpha) \stackrel{\downarrow}{=} 2.3983 \text{ bits}$$



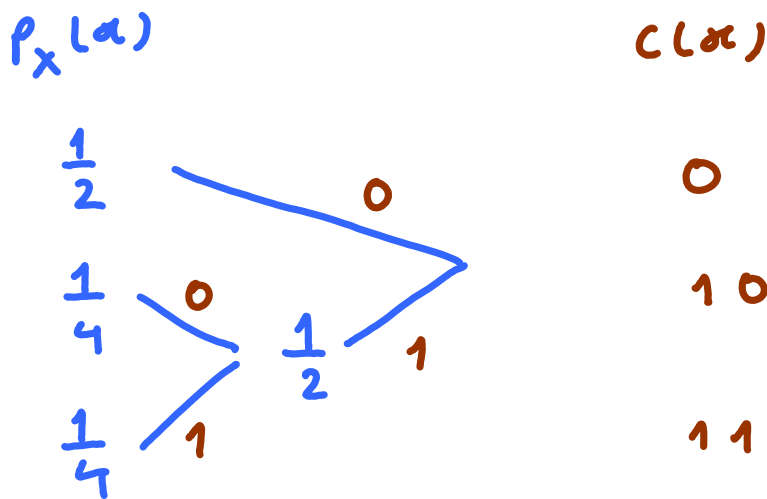
$$E[\mathcal{L}(x)] = \frac{17}{7} \approx 2.43 \text{ bits}$$

Q4 Inverse Huffman Problem

Sunday, August 28, 2011
10:07 PM

$\{0, 10, 11\}$ is a Huffman code for

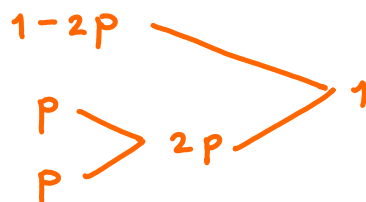
$$p_X(\alpha) = \begin{cases} 1/2, & \alpha = 1 \\ 1/4, & \alpha = 2, 3 \\ 0, & \text{otherwise.} \end{cases}$$



Note that the answer for this question is not unique. You may check that

$$p_X(\alpha) = \begin{cases} 1-2p, & \alpha = 1 \\ p, & \alpha = 2, 3 \\ 0, & \text{otherwise} \end{cases}$$

works when



this condition $1-2p \geq p$
is here to guarantee that $\alpha = 2$

$\iff p \leq \frac{1}{3}$

and $x=3$ are grouped first.

Note also that when $p = \frac{1}{3}$, we have uniform pmf.

Q5 Non-Huffman Codes

Thursday, September 27, 2012
9:08 PM

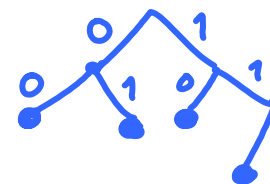
(a)

The code $\{00, 01, 10, 110\}$
can be shortened to

$\{00, 01, 10, 11\}$

without losing its prefix-free property.

Therefore the code $\{00, 01, 10, 110\}$ is not optimal and so it cannot be a Huffman code.



(b)

The code $\{01, 10\}$ can be shortened to $\{0, 1\}$
without losing its prefix-free property.

Therefore the code $\{01, 10\}$ is not optimal and so it cannot be a Huffman code.

Q6 Source Extension

Monday, August 29, 2011
9:35 AM

(a) $H(X) = - \sum_x p_X(x) \log_2 p_X(x) = 0.469 \text{ bits}$
 ↑
 MATLAB

(b) Third-order source extension

$\alpha_1, \alpha_2, \alpha_3$	$P_{X_1, X_2, X_3}(\alpha_1, \alpha_2, \alpha_3)$	$\ell(\alpha_1, \alpha_2, \alpha_3)$
Y Y Y	0.729	1
Y Y N	0.081	3
Y N Y	0.081	3
Y N N	0.009	5
N Y Y	0.081	3
N Y N	0.009	5
N N Y	0.009	5
N N N	0.001	5

Diagram showing merging of probabilities:
 - 0.081 + 0.081 = 0.162
 - 0.009 + 0.081 = 0.090
 - 0.009 + 0.009 = 0.018
 - 0.018 + 0.009 = 0.027
 - 0.027 + 0.162 = 0.189
 - 0.189 + 0.090 = 0.279
 - 0.279 + 0.027 = 0.306
 - 0.306 + 0.018 = 0.324
 - 0.324 + 0.018 = 0.342
 - 0.342 + 0.018 = 0.360
 - 0.360 + 0.018 = 0.378
 - 0.378 + 0.018 = 0.396
 - 0.396 + 0.018 = 0.414
 - 0.414 + 0.018 = 0.432
 - 0.432 + 0.018 = 0.450
 - 0.450 + 0.018 = 0.468

$E[\ell(X_1, X_2, X_3)] = 1.5980 \text{ bits [per three source symbols]}$
 $= 0.5327 \text{ bits [per source symbols]}$

(c)

Fourth-order source extension

To do this, we first need to list all the probabilities in the pmf.

There are $\binom{n}{k}$ source strings that has k Y and $n-k$ N.

The probability for each of these strings is

$p^k (1-p)^{n-k} = 0.9^k (0.1)^{n-k}$

*Y	k	*N	Probability
0	0	4	0.0001
1	1	4	0.0009
2	2	6	0.0081

1	7	0.0007
2	6	0.0081
3	4	0.0729
4	1	0.6561

we put all these 16 probabilities in one pmf vector.

Alternatively, the pmf can be generated using MATLAB "kron" command.

↑
Kronecker product

Defn. $\text{kron}(A, B) = \begin{bmatrix} a_{11}B & a_{12}B & \dots \\ a_{21}B & a_{22}B & \dots \\ \vdots & & \ddots \end{bmatrix}$

Observe that the pmf used for the n^{th} -order source extension is the Kronecker product of

the pmf used for the 1st-order extension

and

the pmf used for the $(n-1)^{\text{th}}$ -order extension

MATLAB is then used to find the expected length.

MATLAB's result: $\mathbb{E}[\ell(x_1, x_2, x_3, x_4)] = 1.9702$ bits per 4 source symbols

= 0.4925 bits per source symbol.

(d)

