Q1 Classes of Codes

Sunday, August 28, 2011 9:33 PM

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

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

(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.

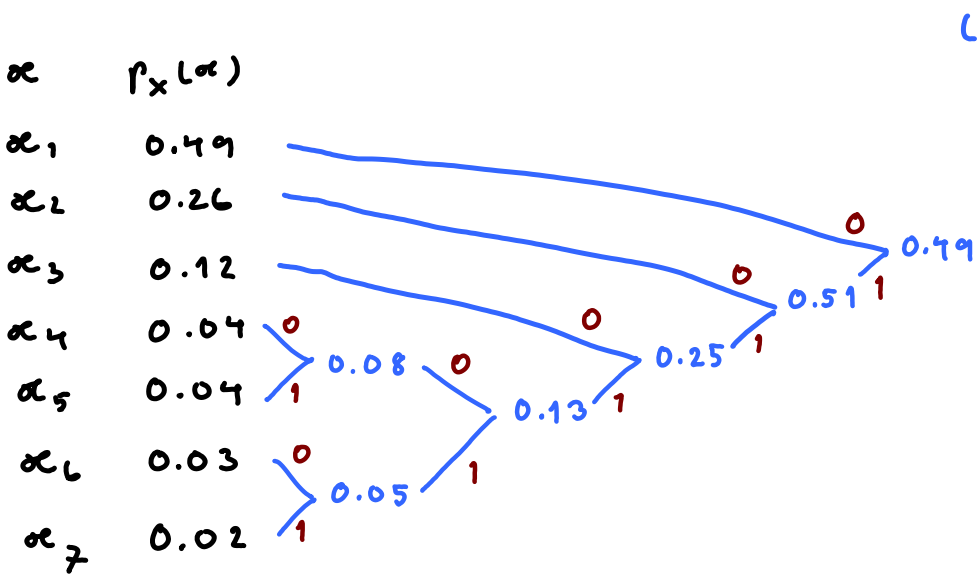
This is what has been traditionally used to refer to prefix-free code.

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) $H(X) = -\mathbb{E}[\log_2 p_X(X)] = -\sum_{\alpha} p_X(\alpha) \log_2 p_X(\alpha) = 2.0128$ bits per symbol.



(b)

α	$p_X(\alpha)$	$c(\alpha)$	$\ell(\alpha)$
α_1	0.49	0	1
α_2	0.26	10	2
α_3	0.12	110	3
α_4	0.04	11100	5
α_5	0.04	11101	5
α_6	0.03	11110	5
α_7	0.02	11111	5

(c) $\mathbb{E}[\ell(X)] = 1\times0.49 + 2\times0.26 + 3\times0.12 + 5\times(\underbrace{\underbrace{2\times0.04}_{0.08} + 0.03 + 0.02}_{0.13})$

$= 2.02$ bits per symbol

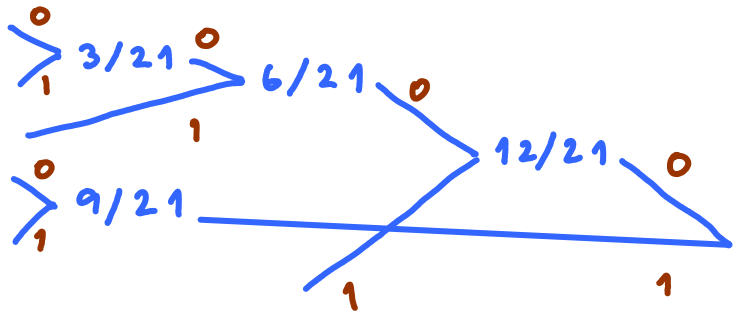
Q3 Huffman code

Sunday, August 28, 2011 10:13 PM

MATLAB

$$H(X) = -\sum_{\alpha} p_X(\alpha) \log_2 p_X(\alpha) \overset{\downarrow}{=} 2.3983 \text{ bits per symbol}$$

α	$p_X(\alpha)$	$c(\alpha)$	$\ell(\alpha)$
1	1/21	0000	4
2	2/21	0001	4
3	3/21	001	3
4	4/21	10	2
5	5/21	11	2
6	6/21	01	2



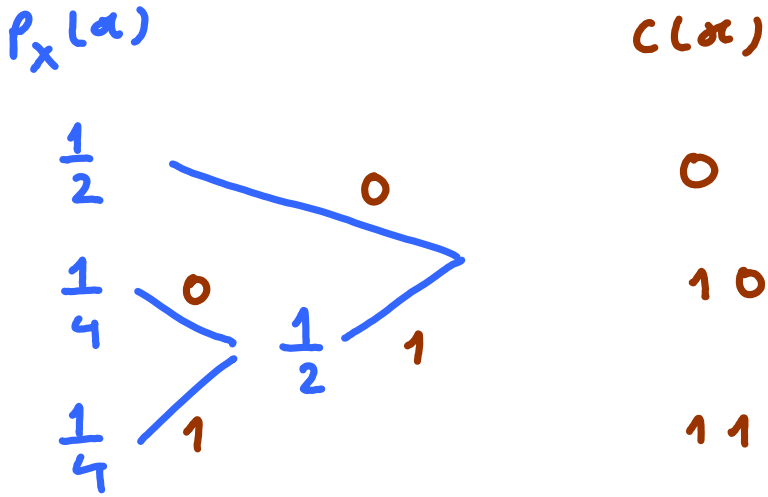
$$\mathbb{E}[\ell(X)] = \frac{17}{7} \approx 2.43 \text{ bits per symbol}$$

Q4 Inverse Huffman Problem

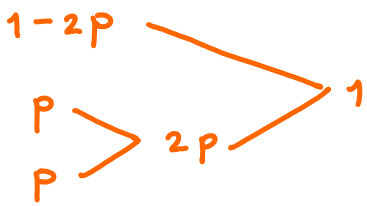
Sunday, August 28, 2011 10:07 PM

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

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



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



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

works when

$$1 - 2p \geq p \iff p \leq \frac{1}{3}$$

this condition is here to guarantee that $x = 2$

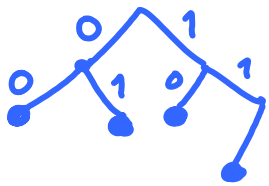
and $x=3$ are grouped first.

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.

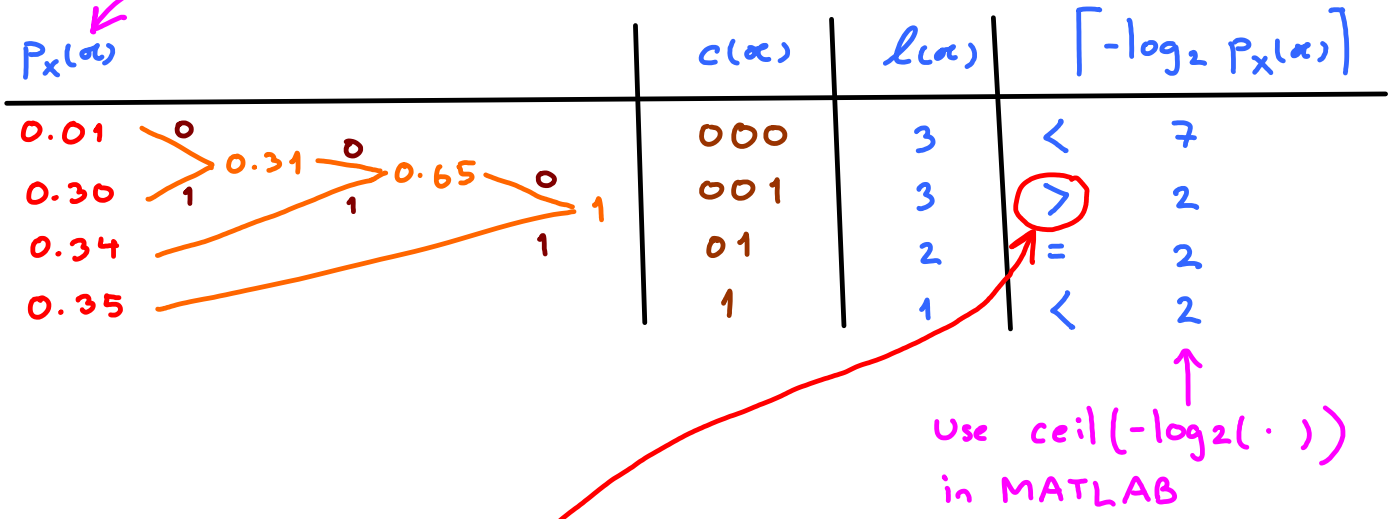
Therefore the code {01, 10} is not optimal and so it cannot be a Huffman code.

Q6 p(x) and ceil(-log2 p(x))

Monday, August 25, 2014 4:28 PM

Following the hint, we create a Huffman code from the provided probability values

We also add this column for comparison



The claim says $\ell(x) \leq \lceil -\log_2 p_X(x) \rceil$ for any x.

Here, we found a case where $\ell(x) > \lceil -\log_2 p_X(x) \rceil$.

Therefore, the claim is not true in general.

Q7 Source Extension

Monday, August 29, 2011 9:35 AM

(a) $H(X) = -\sum_{\alpha} p_X(\alpha) \log_2 p_X(x) = 0.469$ bits per symbol

↑ MATLAB

Notice that the solution does not attempt to find the corresponding codewords. Only the length values are important here because the question only asks us to find the expected length.

(b) Third-order source extension

α_1	α_2	α_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

0.162 = 0.081 + 0.081

0.018 = 0.009 + 0.009

0.028 = 0.018 + 0.01

0.109 = 0.081 + 0.028

0.271 = 0.162 + 0.109

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

$L_3 = 0.5327$ bits [per source symbol]

(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

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

k	#	
0	1	0.0001
1	4	0.0009
2	6	0.0081
3	4	0.0729
4	1	0.6561

(# = number of Y)

we put all these 16 probabilities in one pmf vector.

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

"kron" → Kronecker product

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

Observe that the pmf used for the n^{th}-order source extension is the Kronecker product of
the pmf used for the 1^{st}-order extension
and
the pmf used for the $(n-1)^{th}$-order extension

MATLAB is then used to find the expected length.

MATLAB's result : $\mathbb{E}[\ell(X_1X_2X_3X_4)] = 1.9702$ bits per 4 source symbols

$L_4 = 0.4925$ bits per source symbol.

(d)