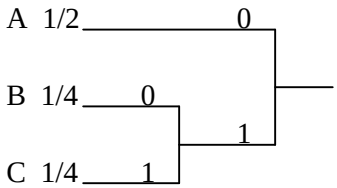
TCOM 370 **March 30, 1999**

Solutions to Homework #6

Problem 1

(a) $H = -\sum_{n=1}^{3} P_n \log P_n = -(0.5\log 0.5 + 0.25\log 0.25 + 0.25\log 0.25) = 1.5$ bits/symbol

(b)



A	→	0		1
B	→	10	or	01
C	→	11		00

(c) (i) Can't do any better since the number of bits per symbol equals the entropy of the source producing independent symbols.

(ii) In light of (i), simply use code of (b) for each symbol, i.e. AA → 00, AB → 010, … , CC → 1111. This is uniquely decodable because the original single-letter code is such.

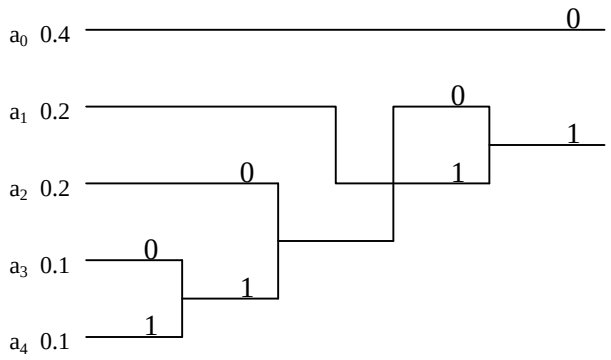
(iii) We expect the average codeword length to be the same as (i) since it is already optimal.

Symbol	Probability	Codeword
AA	1/4	00
AB	1/8	010
AC	1/8	011
BA	1/8	100
BB	1/16	1010
BC	1/16	1011
CA	1/8	110
CB	1/16	1110
CC	1/16	1111

Average codeword length = $\dfrac{2\times 1/4 + 4\times 3\times 1/8 + 4\times 4\times 1/16}{2} = 1.5$ bits/symbol, as expected.

Problem 2

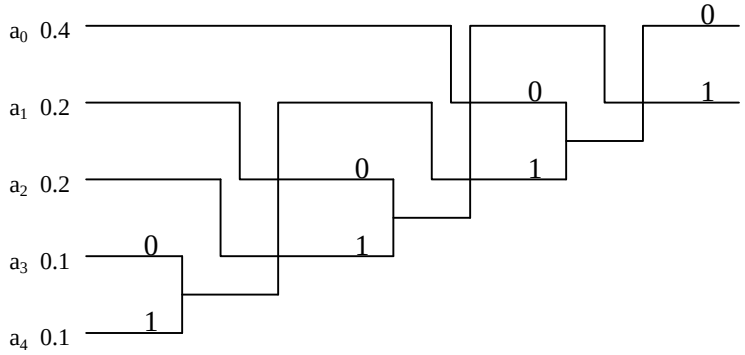
(a)



0	0.4
11	2×0.2
100	3×0.2
1010	4×0.1
1011	4×0.1

Average = 2.2 bits/symbol

(b)



00	2×0.4
10	2×0.2
11	2×0.2
010	3×0.1
011	3×0.1

Average = 2.2 bits/symbol

(c) Both have the same average length per symbol. Code (b) has lower variance of codeword lengths, hence a smaller buffer size for constant transmission rate of 2.2 bits per symbol can be used. Therefore, (b) is preferable over (a).

Problems 3 and 4 : Handout

Problem 5

(a) $H = -\sum P \log P = -0.9 \log 0.9 - 0.1 \log 0.1 = 0.469$

(b) Since there are only 2 letters in the alphabet, one code is:

$A \to 0$

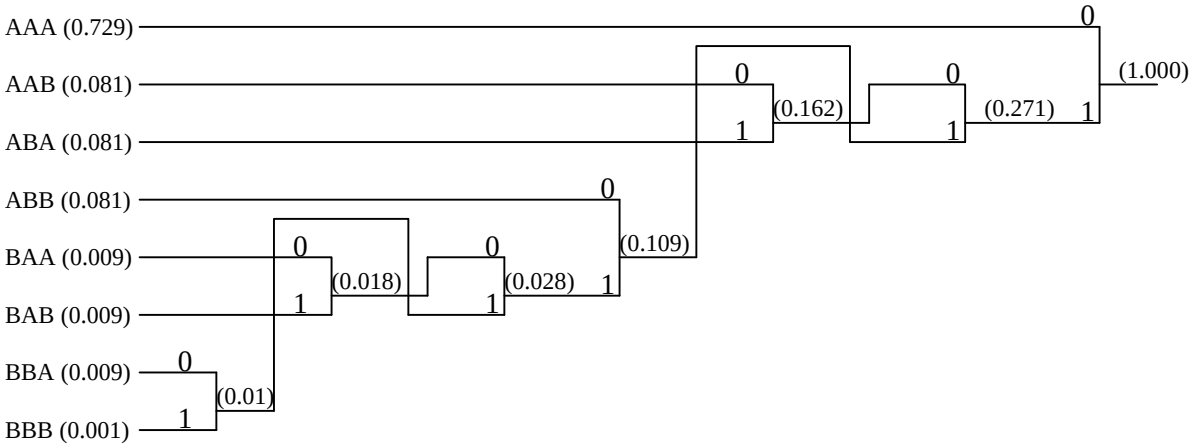
$B \to 1$

Clearly, the average number of bits per symbol is 1.

(c) Compute probabilities of the new symbols (blocks of 3 originals). Assuming independent generation of symbols, the joint probability is simply the product of individual probabilities.

Symbol	Probability
AAA	0.729
AAB	0.081
ABA	0.081
ABB	0.009
BAA	0.081
BAB	0.009
BBA	0.009
BBB	0.001

Now design the Huffman code using the above information.



From the above graph, we can read off the codewords for each symbol (sequence).

AAA	0
AAB	100
ABA	101
ABB	110
BAA	11100
BAB	11101
BBA	11110
BBB	11111

The average number of bits per symbol is:

$$l_{avg} = \frac{1 \cdot 0.729 + 3 \cdot 3 \cdot 0.081 + 3 \cdot 5 \cdot 0.009 + 5 \cdot 0.001}{3} = 0.5327$$

Problem 6

(a) Recall that standard telephone-quality PCM scheme uses 8 KHz sampling with 8 bits/sample. Since the entropy of the source is only 6 bits/sample, in principle we only need $8000 \times 6 = 48000$ bps rather than 64000 bps. So it is possible to achieve a bit rate of 50 Kbps for the speech. One way to achieve such loss-less compression is Huffman coding (by working out the probability distribution of the input). We may need to code blocks of samples in order to achieve the desired level of compression.

(b) One such coding scheme is DPCM discussed in class.

Problem 7 (3.16, Halsall)

(a) $H = -\sum_{i=1}^{7} P_i \log_2 P_i = 2.37$

(b) Use the tree method to generate the codewords.
D=11 B=10 G=00 A=010 F=0111 C=01101 E=01100

(c) Average number of bits per codeword =

$$\sum_{i=1}^{7} N_i P_i = 2\times0.32 + 2\times0.25 + 2\times0.2 + 3\times0.1 + 4\times0.07 + 5\times0.05 + 5\times0.01 = 2.44$$

Fixed-length binary requires 3 bits per character, and ASCII requires 7 bits per character. Compare these values to the entropy of 2.44.

Problem 8 (3.17, Halsall)

(a) Prefix property – A codeword set derived using Huffman coding has the property that a shorter codeword in the set will never form the start of a longer codeword. By inspection, the codeword set derived in Problem 5 (3.16, Halsall) is a prefix code. More formally, one can use Kraft inequality to show the prefix property.

(b) The flowchart of an algorithm to decode a received bit string is given in Fig. 3.23 (of the text) and the text associated with this explains how it operates.

(c) Assuming the use of the codeword set constructed above and the following received bit stream: 010 10 01101 11 01100 0111 00 we find the decoded symbols written below.
A B C D E F G

Problem 9

The bit sequence is parsed (uniquely) in the following way:

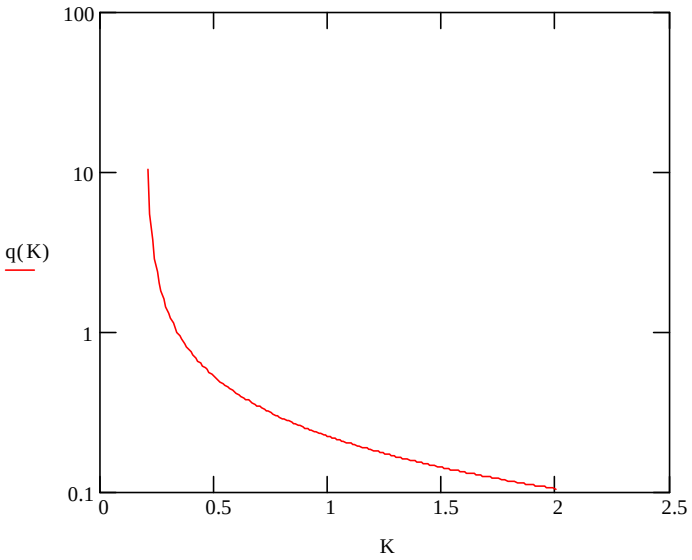
00000000001	00110101	00011	00011011	0010	011010100	1100	00000000001
EOL	0 white	7 black	31 white	6 black	960 white (make-up code)	5 white	EOL

Instead of having 7+31+6+960+5=1010 bits to represent the pixels, only 60 bits are needed using the T4 standard. Thus the compression is (1010-60)/1010=94%

Problem 10

(a) For a useful system, $a < K < 1$. If $K < a$, then the multiplexed transmission line cannot handle the aggregate traffic even with perfect statistical multiplexing because the average required rate is larger than that available. $K = 1$ corresponds to the case in which $M = NR$, i.e. all terminals are transmitting simultaneously. $K > 1$ results in waste of transmission resources; one can use K=1 with synchronous multiplexing.

(b)



As intuition suggests, faster transmission lines (higher K and thus M) results in an average smaller number of inputs in the queue (and thus smaller average delay). Note that in above plot values of K beyond 1 would not be used, and that as K gets closer to α the queue length and delay increase very rapidly. A reasonable design might use K=0.4.