EE 121: Introduction to Digital Communication Systems
Homework 2 Solutions

February 1, 2008

1. The generalized Kraft's inequality will be: Every D-ary prefix-free code for an alphabet $\mathcal{X} = \{a_1, a_2, \dots, a_M\}$ with codeword lengths $\{l(a_j), j = 1, \dots, M\}$ satisfies

$$\sum_{j=1}^{M} D^{-l(a_j)} \leq 1.$$

To outline the proof, first note that since the code is prefix free we know that we can construct a D-ary tree where the codewords in the code are all leaf nodes in the tree. Therefore we can view each codeword as representing a subinterval in [0.1] with width $D^{-l(a_j)}$ where $l(a_j)$ is the corresponding length of the codeword. With the same argument as the book, as the codewords are prefix free, their corresponding intervals are disjoint. Therefore the sum width of the intervals should be less than 1 or,

$$\sum_{j=1}^{M} D^{-l(a_j)} \leq 1.$$

2. (a) First look at the following useful inequality known as the log-sum inequality:

Lemma 0.1. *(Log-sum inequality) For non-negative numbers $a_1, a_2, \dots, a_n$ and $b_1, b_2, \dots, b_n$,*

$$\sum_{i=1}^{n} a_i \log \frac{a_i}{b_i} \geq \left(\sum_{i=1}^{n} a_i \right) \log \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i}$$

with equality if and only if a_i/b_i=constant.

Now note that

$$\overline{L} - H(X) = \sum_{i=1}^{M} p_i (l_i - \log \frac{1}{p_i}) = \sum_{i=1}^{M} p_i \log \frac{p_i}{2^{-l_i}}.$$

By the log-sum inequality we have

$$\overline{L} - H(X) \geq \left(\sum_{i=1}^{M} p_i\right) \log \frac{\sum_{i=1}^{M} p_i}{\sum_{i=1}^{M} 2^{-l_i}} = \log \frac{1}{\sum_{i=1}^{M} 2^{-l_i}} \geq 0$$

where the last inequality is true by Kraft's inequality. Therefore by the equality condition of log-sum inequality we have

$$\overline{L} - H(X) \Leftrightarrow p_i = c \cdot 2^{-l_i}, \quad i = 1, \ldots, M$$

for some constant c. Now since for the equality we have $\sum_{i=1}^{M} 2^{-l_i} = \sum_{i=1}^{M} p_i = 1$ therefore $c = 1$ and the proof is complete.

(b) First note that the first bit is going to be 0 or 1 with probability 0.5, because the probability of upper sub-tree (starting after first branch 1) is equal to the probability of the lower sub tree (starting from the second branch 0) and is equal to 0.5. With the same argument at any node the probability of going up is equal to the probability of all leaves at that sub-tree and is also equal to the probability of going down and is independent of the probability of all other bits. Therefore at any time the probability of 0 and 1 are the same and independent of the output at other times. Therefore the sequence of encoded binary digits is a sequence of iid equiprobable binary digits.

3. (a) Group the symbols occurring with probabilities 1/9 and 2/9 first. This creates a supersymbol occurring with probability 1/3. So now group this symbol and one of the other symbols occurring with probability 1/3 into a second supersymbol and finally group this with the remaining symbol. The resulting codebook is $\{0, 10, 100, 101\}$.

(b) Swap the first and second codewords, i.e. $\{10, 0, 100, 101\}$.

(c) The above codes have expected length 2. Thus the fixed-length code $\{00, 01, 10, 11\}$ is also optimal, but cannot be generated by the Huffman algorithm.

4. (a) Start with a tree with three leaves. Each time we add a branch to the tree, we add three leaves and subtract one. Thus it is only possible to have a complete ternary tree for M odd and ≥ 3.

(b) We typically group symbols into triples, but if M is even, one of the groupings will have to be only a pair. The code with the shortest average length will have its pair occurring at the deepest level of the tree. Thus the Huffman algorithm for ternary symbols with M even involves first grouping the two symbols with the lowest probabilities, and then grouping triples with increasing probability in a manner entirely analogous to the Huffman algorithm for binary symbols.

(c) The codewords are $0, 1, 20, 21, 220, 221$, respectively.

5. (a) As the zero probability symbol will be placed at the deepest level of the tree, it's inclusion in the code will cause the leaf with minimum probability to be one branch longer. Let $i^* = \arg\min_i p_i$. Then $L = p_{i^*} l_{i^*} + \sum_{i \neq i^*} p_i l_i$ whereas $\overline{L} = p_{i^*}(l_{i^*} + 1) + \sum_{i \neq i^*} p_i l_i$. Thus $\overline{L} = L + \min_i p_i$.

(b) The n zero probability symbols will ultimately all be grouped into a single supersymbol with zero probability. Thus the answer is again $\overline{L} = L + \min_i p_i$.