ECE 130 - TREES - INVESTIGATION 21
HUFFMAN CODES

WINTER 2004 **A.P. FELZER**

To do "well" on this investigation you must not only get the right answers but must also do neat, complete and concise writeups that make obvious what each problem is, how you're solving the problem and what your answer is. You also need to include appropriate graphs and tables.

One of the main goals in data storage and transmission is to compress information as much as possible - to represent the data with the fewest possible number of bits - without losing any or at most a minimal amount of information. Some common examples of compression codes are MP3 for music, JPEG for graphics and MPEG for movies. The objective of this problem is to introduce Huffman codes for reducing the memory required - the number of bits required - to store and transmit data.

1. Suppose a temperature gauge monitoring a given chemical process classifies the temperature as low, medium or hot. And that for every 90 times the temperature gauge reads medium it reads low 5 times and hot 5 times
   a. How much memory space is required - how many bits are required - to store 1000 readings if we use two bit binary codes of 1's and 0's to represent the data as follows

Temperature	Low	Medium	Hot
Code	00	01	10

   b. How much memory space is required if we change the code as follows

Temperature	Low	Medium	Hot
Code	10	0	11

      where we now use a one bit code for the reading that occurs most often.
   c. What's the advantage of using a coding scheme like the one in part (b)
   d. What's a disadvantage of a coding scheme like the one in part (b)
   e. What kind of coding scheme is used by ASCII code

2. From Problem (1) we see that we can reduce the amount of storage necessary when we use codes that are shorter for data that occurs more frequently. This of course is great but raises the question of how to differentiate one data word from another in a stream of data like the following

   100101101110010100

   One way to deal with this problem is to use **prefix codes** - codes like the one in Problem (1b) in which we can tell from the first bit or bits how many bits are in a data word being decoded
   a. Explain how we can tell that the code in Problem (1b) is a prefix code
   b. Explain how the following code is a prefix code

Character	a	b	c	d
Code	0	100	110	1110

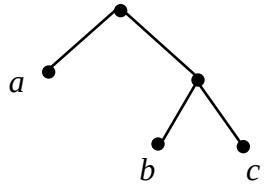
      In particular explain how a decoder can tell how many bits are in a data word being decoded
   c. Find your own example of a prefix code
   d. Find an example of a code that is not a prefix code. Explain

3. Given the following prefix code from Problem (2)

Character	a	b	c	d
Code	0	100	110	1110

   a. Find the data stream for abcadab
   b. Decode 01100111010000100110

4. One convenient way to represent codes is with **binary character trees** as follows



   with characters at the leaves. The code for any given character is determined by its path from the root with 0 for an edge to the left and 1 for an edge to the right. In particular for the character *b* we have the code 10

   a. Find the codes for *a* and *c*
   b. How is the depth of a character - its distance from the root - related to its code
   c. Explain why codes from binary character trees are prefix codes

5. Suppose d(i) is the depth of character *i* in a binary character tree and f(i) is the relative number of times character *i* occurs. Why do we want d(i) to be small when f(i) is large

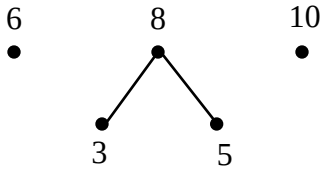
6. Generalizing on the result in Problem (5) we see that in order to minimize the total number of bits required to store data we need to find a character tree T - a **Huffman tree** - that minimizes

$$\sum_{all\ leaves i} d(i) f(i)$$

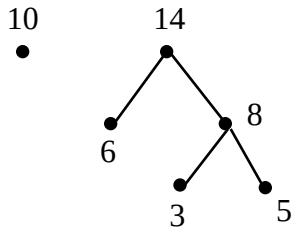
   Suppose in particular that we have four characters with relative frequencies arranged in increasing order as follows

   Then the algorithm for forming a Huffman tree is as follows

   (1) Add the two lowest frequencies to obtain a new vertex at 3 + 5 = 8 as follows



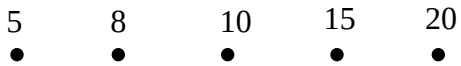
   (2) Again add the two lowest frequencies to obtain a new vertex at 6 + 8 = 14



(3) Continue until there is just a single root

a. Complete the algorithm above to obtain the Huffman tree
b. Find the corresponding codes
c. Find the number of bits required to store $C = 3 \cdot 5 \cdot 6 \cdot 10 = 900$ characters

7. Find the Huffman tree and corresponding codes for



8. Make up and solve your own Huffman tree problem