

## 18.310 Homework # 3

**1a:** For the relative frequency assignment

A	C	E	H	I	M	T	S
.10	.07	.03	.05	.07	.22	.16	.30

what does Shannon's entropy formula give us for the minimum number of bits per letter required to compress this message?

**1b:** For the above frequency assignment, find the associated Huffman code. If compressing a message that has these letter frequencies, what would the resulting number of bits per letter be?

**1c:** For the above frequency assignment, find the associated Hu-Tucker code. If compressing a message that has these letter frequencies, what would the resulting number of bits per letter be?

**2:** What is a length 30 sequence of 0's and 1's which would require the most bits to encode using compressed by the Lempel-Ziv compression algorithm discussed in class? What is a length 30 sequence which would require the fewest bits?

**3:** Suppose that Danny and Peter are playing mini-bingo. There are two boards which look like

Peter		Danny	
1	2	4	2
3	5	1	5

A random permutation of the numbers 1 through 5 is called out, and the winner is the first person to get two numbers in the same row or column. Diagonals don't count. If the same number gives a bingo to both of them, then they tie.

**a.** Show that Danny and Peter have an equal probability of winning. (Note: this can be done using a symmetry argument, although you can also do it the long way, by computing these probabilities.)

**b.** Suppose the 5 card is lost, so only the numbers 1 through 4 get called out (all permutations are equally likely). Now what are Peter and Danny's probabilities of winning? What is the probability of a tie?

**4:** Compute the expected number of comparisons that the quicksort procedure makes as follows:

Consider the following sorting algorithm. Start with an array containing the keys in any order. Choose a random permutation  $\pi$  of the keys (uncorrelated with the initial order of the keys in the array). Now, go through the keys in order given by  $\pi$ , with one step (involving several comparisons) associated with each key. After each step, all the keys that have already been handled will be in their correct positions in the array, and the unhandled keys will be sorted with respect to the keys that have already been handled. This is done by, at each step, comparing the key you are currently handling with all keys between the next lowest and next highest ones you have already handled.

**4a:** Argue that this algorithm gives exactly the same number of comparisons as quicksort.

**4b:** Consider an indicator variable  $I_{i,j}$  for every pair of keys  $(i, j)$  with  $i < j$  which is 1 if the  $i$ th key and the  $j$ th key are compared during the sort, and 0 otherwise (here  $i$  and  $j$  refer to the final sorted order of the keys). Show that the expected value of this variable

$$\langle I_{i,j} \rangle = \frac{2}{j-i+1}$$

**4c:** Use 4b to show that the expected number of comparisons in quicksort is approximately  $2n \ln n$ , where  $\ln$  is the logarithm base  $e$ . The formula

$$\ln k \leq \sum_{j=1}^k \frac{1}{j} \leq \ln k + 1,$$

(which can easily be derived with calculus) may be useful.

**5** Consider the algorithm for finding the maximum of  $n$  keys where you choose a random key, compare it to all the other keys, and throw away everything less than it, then repeat.

**5a:** Show that for this algorithm, with the indicator variable  $I_{i,j}$  defined in 4b, we have

$$\langle I_{i,j} \rangle = \frac{2}{n-i+1}$$

**5b:** Use 5a to approximately compute the expected number of comparisons in the randomized maximum-finding algorithm above. You should find the constant on the leading term.

**6a:** On a spreadsheet or otherwise compute  $w(n) = n!/(n/e)^n$  for  $n$  from 1 to 128 (or 256, if you'd prefer).

**6b:** Square the results and divide by  $2n$ . Then extrapolate. Put  $w^2/2n$ , for  $n = 1, 2, 4, 8, 16, 32, \dots$ , in cells  $C1, \dots, C8$ , say, on a spreadsheet and let

$$D1 = 2C2 - C1$$

(and copy this down column C),

$$E1 = \frac{4}{3}D2 - \frac{1}{3}D1, \quad F1 = \frac{8}{7}E2 - \frac{1}{7}E1, \quad G1 = \frac{16}{15}F2 - \frac{1}{15}F1.$$

What happens?

**6c:** Assume  $w(n)^2/2n = \gamma + \delta/n + \eta/n^2$ . Figure out what  $\delta$  and  $\eta$  are using the spreadsheet

**7:** Extra credit. Consider the variant of quicksort that finds the largest  $k$  keys by only solving a subproblem with recursion if any of the keys in the subproblem are among the largest  $k$ . Use the methods of problems 4 and 5 to find the expected number of comparisons this algorithm requires. You should get an answer with leading terms of the form  $cn + dk \log n$  where  $c$  and  $d$  are constants.