CSCI 503B:
HOMEWORK 2

Each problem is worth 25 points. Good luck!

1. Consider activity selection. It might be helpful to draw pictures for this problem.

   (a) All n activity requests have the following property: they intersect with exactly one other activity. Argue that you can find out how many activities you can schedule without even knowing the requests. So how many can you schedule?

   (b) Now consider requests where except for two activities a and b, all activities overlap with exactly two other activities. a and b overlap with one other activity each. Argue that you cannot solve this problem without seeing the actual requests, unlike the above.

2. This is the hardest problem in this set. Work hard on this one! Consider the following variant of our activity selection problem: You are given a set of n activities, each of which will take one hour each. They will all start on the hour, so, go, for instance, from 2pm to 3pm, etc. They all have an integer deadline: if an activity i is not run and finished by deadline d_i, the activity charges the room owner a penalty of $p_i \geq 0$. As before, the activities cannot overlap. The goal is to schedule *all* activities (notice how this is different from our version where we did not schedule all activities) so that the total penalty is minimized.

   (a) Is it true that, if the deadlines d_i are the same for all activities (and equal to k), then the optimal solution will have a penalty equal to the sum of the penalties of the k lowest penalty jobs. Argue.

   (b) Now assume all activities have the same penalty. Then I can sort the activities by their deadlines, and place them into the slots, starting from the earliest deadline into the earliest slot, and so on, placing them as ordered by their deadlines. Prove that this algorithm is optimal.

   (c) Argue that, even if we have more than n time slots, we never need those beyond n.

   (d) Here is the algorithm for the general case:

Sort the activities according to their penalties, so

$$p_1 \geq p_2 \geq \cdots \geq p_n$$

Then, add each activity to the schedule in turn. When adding activity i, if any time slot between 1 and d_i is available schedule i in the latest such slot. Otherwise schedule it in the latest available slot $\leq n$. Prove that this algorithm is optimal.

3. Find the frequencies of the letters in the English language (search on the web; there is not a single answer, but any reasonable one will do) and draw a Huffman tree for it. Write the length of the code for each letter, then compare to a code where each letter is the same length. What is the percentage gain in length from using Huffman code compared to the fixed length encoding? You can compare the bit length for encoding a long text by the two different codes.

4. Analyzing Huffman.

   (a) Is it possible in Huffman code for letters with the same frequency to end up with different length codes? Argue your answer either by giving an example if YES, or by giving a proof if NO.

   (b) Given a Huffman Tree on 8 characters, what is the maximum possible difference in the frequencies of the highest and lowest frequency letters if we know that the tree is a complete binary tree? You should assume that we have used the algorithm discussed in class to form the Huffman Tree.