Problem Set 4: Greedy Algortihms

Handed out Friday, September 13. Due at the start of class Friday, September 20.

Problem 1. (20 points) In class we presented a greedy algorithm for scheduling a set of n tasks, in which each task is given a duration t_i and deadline d_i. We showed that scheduling the tasks in increasing order of deadline minimizes the maximum lateness. (Recall that if task i is scheduled at time $s(i)$, then its lateness is $\ell_i = max(0, s(i) + t_i - d_i)$.) Define the *average lateness* to be $(1/n)\sum_{i=1}^{n} \ell_i$.

(a) Provide a counterexample to show that scheduling tasks in increasing order of deadline does not minimize average lateness. Briefly explain your example.

(b) Provide a counterexample to show that scheduling tasks in increasing order of duration does not minimize average lateness. Briefly explain your example.

(c) Suppose that we redefine lateness to be $\ell_i = s(i) + t_i - d_i$ (ignoring the "max"). Thus, if the task finishes before the deadline, its lateness is negative. (Intuitively, this can be thought of as a bonus, which can be applied to reduce the positive lateness of some other task.) Prove that if tasks are scheduled in increasing order of duration, then average lateness (under this modified definition) will be minimized. (Hint: Use the same sort of exchange argument we used in class to prove that earliest deadline first minimizes maximum lateness.)

Problem 2.(10 points) The input to this problem consists of an ordered list of n words. The length of the ith word is w_i, that is the ith word takes up w_i spaces. (For simplicity assume that there are no spaces between words.) The goal is to break this ordered list of words into lines, this is called a *layout*. Note that you are *not* allowed to reorder the words. The length of a line is the sum of the lengths of the words on that line. The ideal line length is L. No line may be longer than L, but it may be shorter. The penalty for having a line of length k is $L - k$. The *total penalty* is defined to be –**fill in later**– (see below). The problem is to find a layout that minimizes the total penalty. Prove or disprove that the following greedy algorithm correctly solves this problem.

```
for (i = 1 to n) {
    Place the ith word on the current line if it fits
    else {
       start a new line and place the ith word on this line
    }
}
```

(a) Suppose that we set "–fill in later–" to "the **sum** of the individual penalties". Is the greedy algorithm optimal? Either give a proof or present a (short) counterexample.

(b) Suppose that we set "–fill in later–" to "the **maximum** of the individual penalties". Is the greedy algorithm optimal? Either give a proof or present a (short) counterexample.

(Hint: For one of the above alternatives, the greedy algorithm is optimal and for the other it is not.)

Problem 3.(10 points) Professor X drives from Memphis to Chicago. He starts with a full tank and can go for 100 miles on a full tank. Let $x_1 < ... < x_n$ denote the locations of the various gas stations along the way, measured in miles from Memphis. Present an algorithm that determines the fewest number of gas stations he needs to stop at to make it to Chicago without running out of gas along the way. Give a short proof of correctness, and analysis of the run-time of your algorithm.

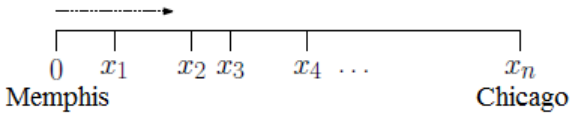

Figure 1: Example for Problem 3.