CS 5633: Analysis of Algorithms

Homework 6

1. This problem has to do with dynamic programming.

   Suppose that you are managing the construction of a toll road which is M miles long, and you are faced with the task of picking locations for toll booths from a set of n possible locations. The possible locations for toll booths are given as positive integers $x_1, x_2, \ldots, x_n$ which denote the distance of the location from the beginning of the road (assume that the x_i's are in increasing order). Further suppose that for each location x_i, you are given a value t_i which denotes the amount of money you would make in tolls from this location per day. Finally suppose that regulations require that each of the toll booths be at least 10 miles apart. What is the optimal choice of toll booths that makes as much money as possible without violating the constraint on the distance between the booths?

   Example: Suppose that $M = 50$ and $n = 4$ such that $\{x_1, x_2, x_3, x_4\} = \{10, 15, 23, 30\}$ and $\{t_1, t_2, t_3, t_4\} = \{50, 60, 50, 10\}$. Then the optimal solution would be to place toll booths at x_1 and x_3 for a total gain of 100.

   (a) Consider a brute force algorithm that checks every possible placement of toll booths. What is the running time of this algorithm? Justify your answer.

   (b) Let $a[j]$ denote the optimal solution when considering only the first j possible toll booth locations. Using the example above, we would have $a[1] = 50$, $a[2] = 60$, $a[3] = 100$, and $a[4] = 100$. Write a recursive definition of $a[j]$ (do not forget the base case).

   Hint: let $\ell(j)$ denote the possible toll booth location that is to the left of x_j and is the farthest down the road without being within 10 miles from x_j (i.e. of all of the locations to the left of x_j which can be included in a feasible solution with x_j, $\ell(j)$ is the one that is farthest to the right). In our example, $\ell(3) = x_1$ and $\ell(4) = x_2$.

   (c) Describe a dynamic programming algorithm based on your recursive definition. Be sure to avoid computing the value for a subproblem more than once.

   (d) What is the running time of your algorithm? Justify your answer.