CS314: Algorithms
Greedy Algorithms
In class, September 25

Problems

1. Suppose your friends have decided to go camping. They want to hike as much as possible during the day, but (since they've watched *The Blair Witch Project* too often) they don't want to do any hiking after dark. On the map, they've identified a large number of good stopping points for camping, and they're proposing the following system. Every time they come to a good stopping point, they'll determine whether they can make it to the next one before nightfall. If they can make it, they keep hiking; otherwise, they stop. They claim that this system must be good, since it minimizes the number of stops they need to make.

   Being a good algorithms student, you immediately recognize a greedy algorithm and become skeptical. Does this really minimize the number of stopping points?

   To formalize the problem, we'll model the trail as a line segment of length L, and assume your friends can hike d miles per day. We'll assume potential stopping points are labeled $x_1, x_2, \ldots, x_n$ along the segment, starting with 0 at the beginning of the segment. We'll also assume (quite generously) that your friends are correct in their estimation of whether they can make it to the next good stopping point before dark. Finally, we'll say a set of the stopping points is valid if the distance between each adjacent pair is at most d, the first is at most d from 0, and the last is at most d from the end of the segment.

   We now state the question formally: is your friends greedy algorithm optimal, in the sense that it finds a valid set that is as small as possible?

2. In the US, we have a standard set of coins: 1, 5, 10, and 25 cents. Given an amount M, we wish to make change using as few coins as possible. Formalize the greedy algorithm mentioned in class to solve this, and prove that it is optimal (for U.S. currency).

3. Consider a weighted version of the class scheduling problem from class, where different classes offer different number of credit hours (totally unrelated to the duration of the class, since we love those 0 or 1 credit hour labs that last for 4 hours). Your goal is now to choose a set of non-conflicting classes that give you the largest possible number of credit hours, given an array of start times, end times, and credit hours as input.

   (a) Prove that the greedy algorithm from lecture - choosing the class that ends first and recursing - does NOT always return an optimal schedule.

   (b) Describe an algorithm to compute the optimal schedule in $O(n^2)$ time.