CS:3330 Homework 6, Spring 2017

Due in class on Thursday, March 9th

This homework refers to problems from exercises in Prof. Jeff Erickson's lecture notes, specifically Lecture 7 on Greedy Algorithms and Lecture 21 on SSSP problem and algorithms. Links to these lectures notes are posted on the course website.

1. Consider the problem of making change for n cents (for positive integer n) using the *fewest* number of coins. So the input to the problem is a positive integer (e.g., 83) and as output we seek the fewest number of quarters, dimes, nickels, and pennies that add up to n (e.g., for 83, the optimal change is 3 quarters, 1 nickel, and 3 pennies). Describe a *greedy* algorithm that uses quarters, dimes, nickels, and pennies to make change.

   Now comes the more interesting question: how do you *prove* that your algorithm yields an optimal solution? The next few parts of this problem will lead you through a proof.

   (a) Let O be an optimal change for n. In other words, O is a smallest possible set of coins using quarters, dimes, nickels, and pennies, whose total value is n. We will first prove the following claim:

   > **Claim:** O contains exactly as many quarters as produced by the greedy algorithm.

   To prove this claim, answer the following questions: what is the maximum number of pennies that O can contain? what is the maximum number of nickels that O can contain? what is the maximum number of dimes that O can contain? what are all the possible combinations of dimes and nickels that O can contain? Using the answers to these questions, determine the maximum value of dimes, nickels, and pennies in O. Use this value to prove the claim.

   (b) Use the above strategy to prove that O contains exactly as many dimes as produced by the greedy algorithm and then exactly as many nickels as produced by the greedy algorithm.

   (c) It turns out that the greedy strategy works only for certain denominations. For example, it works for the set of coin denominations in the U.S. (quarters, dimes, nickels, pennies), but it does not work for all denominations. Come up with a specific set of denominations $\{1, d_1, d_2, \ldots, d_k\}$, $1 < d_1 < d_2 < \cdots < d_k$, and a positive integer n such that the greedy algorithm with input n and using these denominations does not return optimal change.

2. Problem 1(h), Lecture 7, Prof. Jeff Erickson's notes.

3. Problem 2(a), Lecture 7, Prof. Jeff Erickson's notes.

4. Problem 5, Lecture 7, Prof. Jeff Erickson's notes (see link posted on course website). The motivation for this problem is similar to the motivation for the *Interval Scheduling* problem we studied in class. Suppose that we are given a bunch of events (e.g., courses) we need to schedule, then finding the fewest number of rooms we can schedule all of these events in, is the *Interval Coloring* problem described here.

   Here is a brief description of a greedy algorithm, you should consider. Let $\mathcal{P} = \{1, 2, 3, \ldots\}$ be the set of "colors" we want to use for the interevals. Consider the intervals one-by-one in left-to-right order of start times to to each interval, assign the smallest color from $\mathcal{P}$ that is "available."

   You'll have to think pretty carefully about how to prove the correctness of this algorithm and how to present your proof. Your proof will be graded for correctness as well as clarity.

5. Problem 6, Lecture 21, Prof. Jeff Erickson's notes.