
- For this and all future homeworks, groups of up to three students can submit (or present) a single common solution. Please remember to write the names of all group members on every page.
- Students in **Cluster 3** will present their solutions to Jeff or one of the TAs, on Tuesday or Wednesday of the due date (February 16 or February 17), instead of submitting written solutions. **Each homework group in Cluster 3 must sign up for a 30-minute time slot no later than Monday, February 15.** Signup sheets will be posted at 3304 Siebel Center ('The Theory Lab') later this week. Please see the course web page for more details.

---

1. You saw in class a correct greedy algorithm for finding the maximum number of non-conflicting courses from a given set of possible courses. This algorithm repeatedly selects the class with the earliest completion time that does not conflict with any previously selected class.

   Below are four alternative greedy algorithms. For each algorithm, either prove that the algorithm constructs an optimal schedule, or give a concrete counterexample showing that the algorithm is suboptimal.

   (a) Choose the course that *ends latest*, discard all conflicting classes, and recurse.

   (b) Choose the course that *starts first*, discard all conflicting classes, and recurse.

   (c) Choose the course with *shortest duration*, discard all conflicting classes, and recurse.

   (d) Choose a course that *conflicts with the fewest other courses* (breaking ties arbitrarily), discard all conflicting classes, and recurse.

2. You have been given the task of designing an algorithm for vending machines that computes the smallest number of coins worth any given amount of money. Your supervisors at The Area 51 Soda Company are anticipating a hostile takeover of earth by an advanced alien race that uses an unknown system of currency. So your algorithm must be as general as possible so that it will work with the alien system, whatever it turns out to be.

   Given a quantity of money x, and a set of coin denominations $b_1, \ldots, b_k$, your algorithm should compute how to make change for x with the fewest number of coins. For example, if you use the US coin denominations (1¢, 5¢, 10¢, 25¢, 50¢, and 100¢), the optimal way to make 17¢ in change uses 4 coins: one dime (10¢), one nickel (5¢), and two pennies (1¢).

   (a) Show that the following greedy algorithm does *not* work for all currency systems: If $x = 0$, do nothing. Otherwise, find the largest denomination $c \leq x$, issue one c-cent coin, and recursively give $x - c$ cents in change.

   (b) Now suppose that the system of currency you are concerned with only has coins in powers of some base b. That is, the coin denominations are $b^0, b^1, b^2, \ldots, b^k$. Show that the greedy algorithm described in part (a) does make optimal change in this currency system.

   (c) Describe and analyze an algorithm that computes optimal change for *any* set of coin denominations. (You may assume the aliens' currency system includes a 1-cent coin, so that making change is always possible.)

3. Suppose you have just purchased a new type of hybrid car that uses fuel extremely efficiently, but can only travel 100 miles on a single battery. The car's fuel is stored in a single-use battery, which must be replaced after at most 100 miles. The actual fuel is virtually free, but the batteries are expensive and can only be installed by licensed battery-replacement technicians. Thus, even if you decide to replace your battery early, you must still pay full price for the new battery to be installed. Moreover, because these batteries are in high demand, no one can afford to own more than one battery at a time.

   Suppose you are trying to get from San Francisco to New York City on the new Inter-Continental Super-Highway, which runs in a direct line between these two cities. There are several fueling stations along the way; each station charges a different price for installing a new battery. Before you start your trip, you carefully print the Wikipedia page listing the locations and prices of every fueling station on the ICSH. Given this information, how do you decide the best places to stop for fuel?

   More formally, suppose you are given two arrays $D[1..n]$ and $C[1..n]$, where $D[i]$ is the distance from the start of the highway to the ith station, and $C[i]$ is the cost to replace your battery at the ith station. Assume that your trip starts and ends at fueling stations (so $D[1] = 0$ and $D[n]$ is the total length of your trip), and that your car starts with an empty battery (so you must install a new battery at station 1).

   (a) Describe and analyze a greedy algorithm to find the minimum number of refueling stops needed to complete your trip. Don't forget to prove that your algorithm is correct.

   (b) But what you really want to minimize is the total *cost* of travel. Show that your greedy algorithm in part (a) does *not* produce an optimal solution when extended to this setting.

   (c) Describe a dynamic programming algorithm to compute the locations of the fuel stations you should stop at to minimize the cost of travel.