# Greedy

1. Often a chashier wants to give change to a customer. Unless the customer requests otherwise, the cashier wants to use the fewest number of bills. In a more general setting, say that the cashier has the denominations $\{a_1, a_2, \ldots, a_k\}$, with $a_i < a_{i+1}$; he has an unlimited supply of bills of each denomination.

   (a) A greedy algorithm to solve this problem is to recursively choose the largest bill that is not more than the amount of change we want to make. Construct an example where the natural greedy algorithm does not give an optimal solution.

   (b) The denominations of United States currency are \$1, \$2, \$5, \$10, \$20, \$50 and \$100. Does the greedy algorithm work for this set of denominations?

2. Suppose you are running a business and you get, on day one, a set of $n$ jobs $1, 2, \ldots, n$. For each job, you will get \$100 less one dollar per day it takes you to complete the job. You can only work on one job at a time and once you start a job, you must complete it. You know, on day one, how many days each job will take. You must complete all the jobs. You want to work on the jobs in an order that will maximize your profit.

   Formally, job $j$ takes $t_j$ days to complete. An ordering of the jobs will define a completion time $C_j$ for each job $j$. That is, if the first job in the ordering is $j$, then the completion time will be $C_j = t_j$. The completion time of any other job $j'$ is given by the completion time of the job before it in the ordering plus $t_{j'}$. You want to maximize $100n - \sum_{j=1}^{n} C_j$ which is equivalent to minimizing $\sum_{j=1}^{n} C_j$.

   (a) Design a greedy algorithm to find an ordering (and so define completion times) that minimizes $\sum_{j=1}^{n} C_j$. Give pseudocode for this algorithm.

   (b) Prove that your algorithm in (a) correctly minimizes $\sum_{j=1}^{n} C_j$.

   (c) Now suppose that each job $j$ has a penalty $p_j$: your profit for job $j$ is now \$100 $-p_j C_j$. Design a greedy algorithm to find an ordering (and so define completion times) that minimizes $\sum_{j=1}^{n} p_j C_j$. Give pseudocode for this algorithm.

   (d) Prove that your algorithm in (c) correctly minimizes $\sum_{j=1}^{n} p_j C_j$.

   (e) What is the running time of your algorithms for part (a) and (c)?

3. Let $X$ be a set of $n$ intervals on the real line. A proper coloring of $X$ assigns a color to each interval, so that any two overlapping intervals are assigned different colors. Describe and analyze an efficient algorithm to compute the minimum number of colors needed to properly color $X$ . Assume that your input consists of two arrays $L[1 \ldots n]$ and $R[1 \ldots n]$, where $L[i]$ and $R[i]$ are the left and right endpoints of the $i^{th}$ interval.

4. Let $X$ be a set of n intervals on the real line. A subset of intervals $Y \subseteq X$ is called a tiling path if the intervals in $Y$ cover the intervals in $X$ , that is, any real value that is contained in some interval in $X$ is also contained in some interval in $Y$ . The size of a tiling path is just the number of intervals. Describe (*cough* give pseudocode) and analyze (*cough* prove correctness and running time) an algorithm to compute the smallest tiling path of $X$ as quickly as possible. Assume that your input consists of two arrays $X_L[1..n]$ and $X_R[1..n]$, representing the left and right endpoints of the intervals in $X$.

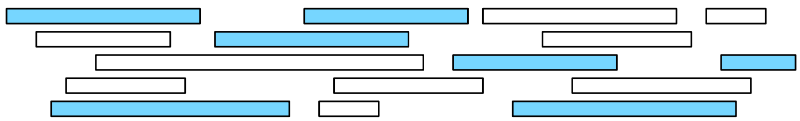

Figure 1: A set of intervals. The seven shaded intervals form a tiling path.