Algorithm Design and Analysis
CSE222 Winter '20

Tutorial 6

Problem 1 Consider the problem of making change for n rupees using the fewest number of coins. Suppose that the available coins are in denominations that are powers of c, i.e., the denominations are $c^0, c^1, \cdots, c^k$, for some integer $c > 1$ and $k \geq 1$. Show that the following greedy algorithm always yields an optimal solution – pick as many coins of denomination c^k as possible, then pick as many coins of denomination c^{k-1} and so on.
Solution. We will prove this using exchange argument. Let σ denote an ordering of the coins as picked by the Greedy solution - that is the denominations of the coins are non-increasing. Assume for contradiction that there exists a different optimal solution which uses stricly lesser number of coins. Let $\sigma^\star$ be the ordering of the coins in non-decreasing order of denomination in this optimal solution.

Let i be the first index such that coin i in σ has a larger denomination - say c^j compared to coin i in $\sigma^\star$. Here is the crucial claim. There exists an $i' > i$ such that the sum of the denominations of coins i to i' in $\sigma^\star$ is *exactly* equal to c^j. Before seeing the proof of this fact, let us see why this helps. We exchange all the coins i to i' in $\sigma^\star$ with the single coin coin i from σ. It is easy to see that then we have a solution which has a stricly lesser number of coins compared to $\sigma^\star$ contradicting the fact that it is optimal.

Now on to the proof of the fact. We are going to prove the following by induction. For any $j' \geq 1$, consider a sequence of numbers, each of the form $c^\ell, \ell \leq j'$ such that the numbers are in non-increasing order and they sum up to at least $c^{j'}$. Then there exists a prefix of these numbers that sum up to exactly $c^{j'}$. The base case is $j' = 1$ which is easy to prove - either we have c many 1s or just one c. Now suppose this is true for all $j' = 1, 2, \cdots j-1$ and consider c^j. Consider c^j as a sequence of c many copies of c^{j-1} - crucially note that c is an integer here. Now apply induction hypothesis on each of c^{j-1}.

Problem 2 [KT-Chapter 4] Let us consider a long, quiet country road with houses scattered very sparsely along it. (We can picture the road as a long line segment, with an eastern endpoint and a western endpoint.) Further, let's suppose the residents of all these houses are avid cell phone users. You want to place cell phone base stations at certain points along the road, so that every house is within 4 kilometers of one of the base stations. Give an efficient algorithm that achieves this goal, using as few base stations as possible. Prove the correctness of your algorithm.
Solution: There is a simple greedy algorithm here. Let h denote the left-most house. Then we place a base station 4 km to the right of h. Now remove all houses which are covered by this base station, and repeat.

The proof follows the 'greedy stays ahead' strategy. It is easy to show (by induction) that if the algorithm locates base stations at $b_1, \cdots, b_k$ and some other algorithm (which could be the optimal algorithm) places base stations at $b'_1 \cdots b'_k$ (from left to right), then $b_1 \geq b'_1, b_2 \geq b'_2$ and so on. Therefore $k \leq k'$.

Problem 3

You are given two sets X and Y of n positive integers each (assume all distinct for simplicity in each individual list). You are asked to arrange the elements in each of the sets X and Y in

some order. Let x_i be the i th element of X in this order, and define y_i similarly. Your goal is to arrange them such that $\prod_{i=1}^{n} x_i^{y_i} = x_1^{y_1} \times x_2^{y_2} \times \cdots x_n^{y_n}$ is maximized.
Solution. The algorithm is simple and intuitive. Just sort X and Y in non-increasing order. Now for the proof, first of all assume optimal sorts X in exactly this order. Why is this fine ? Well, if opt does not do that, we can simply change the ordering of X and generate a new-ordering of Y such that x_i is still matched to the same y_j as in the original optimal ordering.

Now we need to prove that there exists an optimal solution which orders Y in non-increasing order as well. This can be easily done by an exchange argument. Assume that this is not true. Then there exists an inversion pair in Y. Perform an exchange argument and show that you can strictly increase the objective function value in this process.

Problem 4 (Vertex Cover.) Although we have not started with graphs yet, I guess all of you know what graphs are. So suppose you have an undirected connected graph $G(V, E)$. Your task is the following - pick a subset of vertices such that every edge in E is 'covered' - this means that you must pick at least one end-point of *every* edge in your set. This set is then known as a *vertex cover.* The problem of course has trivial solution - just pick all vertices ! As usual, the catch is - we want to pick the vertex cover of minimum possible size.

Consider now the following greedy algorithm - pick the vertex with maximum degree in the remaining graph. Remove that vertex and all the edges incident upon it. Continue the process until all edges have been covered. Is this algorithm optimal ? Prove that, otherwise find an example where greedy is not optimal.
Solution. Consider just a path of 5-vertices : v_0, v_1, v_2, v_3, v_4. The greedy algorithm might choose v_1 - this will cover edges $(v_0, v_1), (v_1, v_2)$. After those are removed, greedy might choose v_2 and then it still needs to choose either v_3 or v_4. The optimal solution here is clearly v_1, v_3.