ShanghaiTech University

CS240 Algorithm Design and Analysis

Spring 2023

Problem Set 2

Due: 23:59, March 17, 2023

1. Submit your solutions to the course Blackboard.
2. If you want to submit a handwritten version, scan it clearly.
3. Late homeworks submitted within 24 hours of the due date will be marked down 25%. Homeworks submitted more than 24 hours after the due date will not be accepted unless there is a valid reason, such as a medical or family emergency.
4. You are required to follow ShanghaiTech's academic honesty policies. You are allowed to discuss problems with other students, but you must write up your solutions by yourselves. You are not allowed to copy materials from other students or from online or published resources. Violating academic honesty can result in serious penalties.

Problem 1:

Consider a rural road (assumed to be a long line segment with a left and right endpoint) with a set of houses located along the road. Unfortunately, there is no access to piped water in this area and the residents of the houses can only get water by digging wells. As the community leader, you want to dig wells at certain locations on the road so that each house is within at most four miles from a well.

Give an efficient algorithm to achieve this goal while minimizing the number of wells to be dug, and prove the correctness of your algorithm.

Problem 2:

Suppose you have a single machine on which you want to schedule a set of jobs for n customers. You want to find a schedule to finish all the jobs which will make the customers the happiest. Let t_i denote the length of customer i's job, and let C_i be the job's finishing time in the schedule. For example, if jobs i and j are the first and second jobs in the schedule, then $C_i = t_i$, and $C_j = t_i + t_j$. Each customer i also has a weight w_i representing her job's importance, and her happiness is determined by both her job's importance and its finishing time. Specifically, you want to order the jobs to minimize the weighted sums of the jobs' completion times, defined as $\sum_{i=1}^{n} w_i C_i$. Design an efficient algorithm for this task, and prove that your algorithm is correct.

As an example, suppose there are two jobs, with $t_1 = 1$ and $w_1 = 100$, and $t_2 = 5$ and $w_2 = 4$. Doing the jobs in the order 1, 2 yields a weighted completion time of $100 \times 1 + 4 \times (1 + 5) = 124$, while doing them in the order 2, 1 gives a weight completion time of $100 \times (1 + 5) + 4 \times 5 = 620$. Thus, your algorithm should output the ordering 1, 2.

Problem 3:

Suppose you are given two sets A and B , each containing n positive integers. You can choose to reorder each set however you like. After reordering, let a_i be the i'th element of set A and b_i be the i'th element of set B. You then receive a payoff of $\prod_{i=1}^{n} {a_i}^{b_i}$. Give an algorithm that will maximize your payoff. Prove that your algorithm maximizes the payoff, and state its running time.

Problem 4:

We are given a gray-scale picture consisting of an $m \times n$ array A of pixels, where each pixel specifies a single intensity value. Suppose that we wish to compress this picture slightly. Specifically, we wish to remove one pixel from each of the m rows, so that the whole picture becomes one pixel narrower. To avoid disturbing visual effects, however, we require that the pixels removed in two adjacent rows be in the same or adjacent columns. A 'seam' is defined as the set of removed pixels.

(a) Show that the number of such possible seams grows at least exponentially in m, assuming that $n > 1$.

(b) Suppose that for each pixel $A[i, j]$, we have calculated a real-valued disruption measure $d[i, j]$, indicating how disruptive it would be to remove pixel $A[i, j]$. Intuitively, the lower a pixel's disruption measure is, the more similar the pixel is to its neighbors. Suppose further that we define the disruption measure of a seam to be the sum of the disruption measures of its pixels. Find the best seam, i.e. the seam with the smallest disruption measure.

Problem 5:

Jack wants to invite his friend John to have a meal in a restaurant, but Jack only has M yuan of money. The restaurant has n different kinds of dishes. The i'th dish costs a_i yuan. John is picky and doesn't want to eat the same dish more than once. John is also greedy, and wants to use up all of Jack's money. John wants to know how many ways there are to order dishes so that each dish is ordered at most once, and the total price of the dishes adds up to M. For example, if there are 4 dishes with price $\{1, 2, 3, 4\}$ and Jack has 6 yuan, then there are two ways John can order, namely the dishes with costs 1, 2, 3, or with costs 2, 4, and so the algorithm should output 2.

Design an efficient algorithm which takes Jack's money M and the prices of n dishes as input, and outputs the number of the ways to order the food which satisfy John's requirements. Analyze the complexity of your algorithm.

Problem 6:

Given two strings s and t, an *interleaving* of s and t is a string formed by first dividing s and t into n and m substrings respectively, for n and m differing by at most 1, and then concatenating the substrings in an alternating fashion. In other words, let $\|$ denote the concatenation operation, and write $s = s_1 \parallel s_2 \parallel ... \parallel s_n$ and $t = t_1 \parallel t_2 \parallel ... \parallel t_m$, for some strings $s_1, ..., s_n, t_1, ..., t_m$ and $|n - m| \leq 1$. Then the interleaving of s and t is any of the strings $s_1 \parallel t_1 \parallel ... \parallel s_n \parallel t_m$, or $s_1 \parallel t_1 \parallel ... \parallel t_m \parallel s_n$, or $t_1 \parallel s_1 \parallel ... \parallel t_m \parallel s_n$ or $t_1 \parallel s_1 \parallel ... \parallel s_n \parallel t_m$, depending on the sizes of n and m. For example, "hodtog" and "doghot" are both interleavings of the words "hot" and "dog".

Suppose you are given three strings r, s and t. Give an algorithm to determine if r can be formed by some interleaving of s and t. Analyze the complexity of your algorithm.