Introduction to Algorithms April 2, 2004
Massachusetts Institute of Technology 6.046J/18.410J
Professors Erik Demaine and Shafi Goldwasser Handout 17

Problem Set 6

This problem set is due **in recitation** on **Friday, April 16**.

Reading: Chapter 15, Chapter 17, 16.1-16.3, 22.1-22.2, Chapter 23

There are **four** problems. Each problem is to be done on a **separate sheet** (or sheets) of paper. Mark the top of each sheet with your name, the course number, the problem number, your recitation section, the date, and the names of any students with whom you collaborated. As on previous assignments, "give an algorithm" entails providing a description, proof, and runtime analysis.

Problem 6-1. Danny's Daemon

Suppose there are m bins containing a total of n balls, where $m > n$. Initially, n of the bins contain one ball and the other $m - n$ bins are empty. Sitting on top of the bins is a daemon who rearranges the balls by a series of *moves*. Each move, the daemon will select a bin b_i containing k_i balls, and redistribute each ball to a unique bin. In other words, the bin b_i will lose all k_i balls and k_i other bins will each gain exactly one ball. We define the cost of this move to be k_i. The total number of balls in the system remains constant, i.e. $n = \sum k_i$.

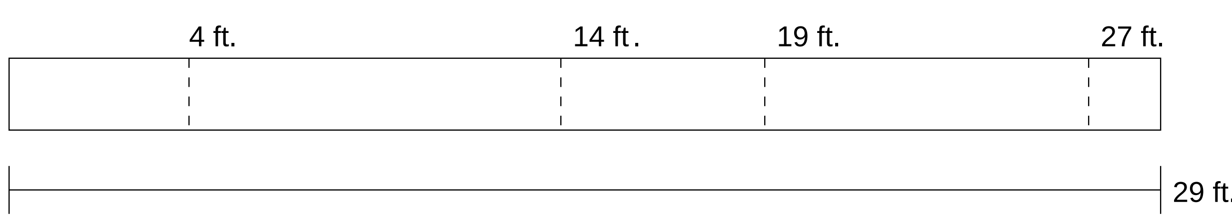
(a) Define an infinite sequence of moves such that after some finite start-up period the cost of each move is $\Theta(\sqrt{n})$. Hint: Your solution may be exactly $\lfloor \sqrt{2n + 1/4} - 1/2 \rfloor$.

(b) Prove that the amortized cost of a move is at most $\lfloor 2\sqrt{n} \rfloor$. Hint: Consider a potential function in which a bin with k_i balls contributes $\max\{0, k_i - t\}$, where t is a constant of your choice and is the same for all bins.

Problem 6-2. Cutting Wood

Your favorite sawmill charges by length to cut each board of lumber. For example, to make one cut anywhere on an 8 ft. board of lumber costs \$8. The cost of cutting a single board of wood into smaller boards will depend on the order of the cuts.

As input, you are given a board of length n marked with k locations to cut. The input to part (b) represents the following board:



(a) Give an algorithm that, given an input length n of wood and a set of k desired cut points along the wood, will produce a cutting order with minimal cost in $O(k^c)$ time, for some constant c.

(b) Suppose you have a 29 ft. board and you want to cut it at points 4, 14, 19, and 27 ft. from the left end. Use your solution from part (a) to determine the minimal cutting cost and illustrate the execution of your algorithm.

(c) Paul Bunyan suggests: Always cut a piece as close as possible to the center. Does this produce an optimal solution? Why or why not?

Problem 6-3. Minimum Spanning Tree in the Plane

Consider the problem of finding the minimum spanning tree connecting n distinct points in the plane, where the distance between two points is the ordinary Euclidean distance. In this problem we assume the distances between all pairs of points are distinct. For each of the following procedures, either argue that it constructs the minimum spanning tree of the n points or give a counterexample.

(a) Sort the points in order of their x-coordinate. (You may assume without loss of generality that all points have distinct x-coordinates; this can be achieved if necessary by rotating the axes slightly.) Let $(p_1, p_2, \ldots, p_n)$ be the sorted sequence of points. For each point p_i, $2 \leq i \leq n$, connect p_i with its closest neighbor among $p_1, \ldots, p_{i-1}$.

(b) Draw an arbitrary straight line that separates the set of points into two parts of equal or nearly equal size (i.e. within one). (Assume that this line is chosen so it doesn't intersect any of the points.) Recursively find the minimum spanning tree of each part, then connect them with the minimum-length line segment connecting some point in one part with some point in the other (i.e. connect the two parts in the cheapest possible manner).

(c) Begin with each point as an isolated tree with no line segments. Consider each segment e in *decreasing* order by length. If the segment e connects two distinct trees, add it to the set of segments in the current spanning forest, and merge the trees together. If the segment e connects two points x, y in the same tree of the forest, add this segment to the spanning forest while removing the longest segment on the path between x and y in the tree.

Problem 6-4. Shortest Paths

(a) We are given a directed graph $G = (V, E)$ on which each edge $(u, v) \in E$ has an associated value $r(u, v)$, which is a real number in the range $0 \leq r(u, v) \leq 1$ that represents the reliability of a communication channel from vertex u to vertex v. We interpret $r(u, v)$ as the probability that the channel from u to v will not fail, and we assume that these probabilities are independent. Give an efficient algorithm to find the most reliable path between two given vertices.

(b) Let $G = (V, E)$ be a weighted, directed graph with weight function $w : E \to \{0, 1, \ldots, W\}$ for some nonnegative integer W. Modify Dijkstra's algorithm to compute the shortest paths from a given source vertex s in $O(WV + E)$ time.