CS 577: Introduction to Algorithms Fall 2020

Homework 2

Instructor: Dieter van Melkebeek TA: Nicollas Mocelin Sdroievski

This homework covers the divide and conquer paradigm. **Problems 1 and 3 must be submitted for grading by 11:59pm on 9/21.** Problem 1 will be discussed in your discussion section on Friday. Please refer to the homework guidelines on Canvas for detailed instructions.

Warm-up problems

1. [Graded, 2 points]

   (a) Design an algorithm to compute the binary representation of 10^n in $O(n^{\log_2 3})$ time.

   (b) Design an algorithm that converts a given n-digit decimal number to binary in $O(n^{\log_2 3})$ time.

   Hint: Use the integer multiplication algorithm from class as a blackbox.

2. Given an array $A[1,\ldots,n]$ of integers and a positive integer k, we want to find a rearrangement B of A such that the subarrays $B_1 \doteq B[1,\ldots,k], B_2 \doteq B[k+1,\ldots,2k],\ldots,$ satisfy the following property: For every $i < j$, every element of B_i is less than or equal to every element of B_j.

   Design an $O(n\log(n/k))$ algorithm for this problem. You can assume that all elements in the array are distinct.

   Hint: First solve the case where $n = 2k$.

Regular Problems

3. [Graded, 3 points] In the knapsack problem, you are given a weight limit W and arrays $w[1,\ldots,n]$ and $v[1,\ldots,n]$ of nonnegative integers for some $n \geq 1$, where $w[i]$ represents the weight of the i-th item and $v[i]$ its value. You want to know the maximum total value that you can achieve by a subset of the items such that the total weight does not exceed W.

   Consider the knapsack problem with the following additional restriction: if you include an item with value v then you need to also include every item with value greater than v.

   For example, for $W = 10$ and $(w[1], v[1]) = (1, 7)$, $(w[2], v[2]) = (4, 11)$, $(w[3], v[3]) = (6, 9)$, and $(w[4], v[4]) = (3, 10)$, the answer is 21, which is realized by the subset of items $\{2, 4\}$. Note that the subset $\{1, 2, 4\}$ is not a valid solution because of the additional restriction, even though its total weight does not exceed W.

   The simple solution to this problem is to sort the items by value and then place items in the knapsack until you can't fit more. As we saw in class, sorting the items would take $\Theta(n \log n)$ time. Your goal is to design an algorithm that solves this problem in $O(n)$ time irrespective of the order in which the items are given. You can assume that all values in the array $v[1,\ldots,n]$ are distinct.

4. You are given an $n \times n$ grid, and a procedure $V(i, j)$ that assigns an integer value to each position (i, j) in the grid, where i and j are integers such that $1 \leq i, j \leq n$. Your goal is to find a local minimum (or sink) in the grid (i.e., integers i^*, j^* with $1 \leq i^*, j^* \leq n$ such that for all neighbors (i, j) of (i^*, j^*) in the grid, $V(i^*, j^*) \leq V(i, j)$). The neighbors of (i, j) are $(i-1, j)$, $(i+1, j)$, $(i, j+1)$, $(i, j-1)$; the elements along the diagonals do not count as neighbors.

   Design an algorithm that makes $O(n)$ calls to V. Note that the grid has n^2 nodes.

5. You are given a topographical map that provides the maximum altitude along the direct road between any two neighboring cities, and two cities s and t. Design a linear-time algorithm that finds a route from s to t that minimizes the maximum altitude. All roads can be traversed in both directions.

Challenge problem

The following is one of the nicest introductory algorithm problems I know. Give it a try!

6. You are given n coins, at least one of which is bad. All the good coins weigh the same, and all the bad coins weigh the same. The bad coins are lighter than the good coins.

   Design an algorithm that makes $O((\log n)^2)$ weighings on a balance to find the exact number of bad coins. Each weighing tells you whether the total weight of the coins on the left side of the balance is smaller than, equal to, or larger than the total weight of the coins on the right side.

Programming problem

7. SPOJ problem Aggressive Cows (problem code AGGRCOW).