CS 3511: Algorithms Honors, Homework 2

Instructor: Prof. Merrick Furst
TAs: Karthik Gopalakrishnan, Pranathi Reddy Tupakula, Jiachen Li

Please submit a LaTeX-formatted PDF containing your solutions on T-Square, by Feb 20, 3:00 pm.

1 Sorting in Linear Time!

(10 points) Show that an array A containing N integers can be sorted in $O(N + D)$ time, where

$$D = \max_{1 \le i \le N} A[i] - \min_{1 \le i \le N} A[i]$$

If $D = O(N)$, this implies we could sort in time linear in the number of elements!

But shouldn't the time taken for sorting be in $\Omega(N \log N)$, you ask? Why is that lower bound for sorting inapplicable here?

2 Divide and Conquer: Tap Two Tags!

(10 points) You're given a set of N high-tech tags. All tags look exactly the same, but embedded in each tag i is a chip with an ID t_i unknown to you. If two tags i and j have the same ID $t_i = t_j$, they light up when tapped against each other (and turn back off once they're separated).

Your task is simple: devise an efficient algorithm to find out if there is a collection of more than $N/2$ tags with the same ID in the given set of N tags. Write down the pseudocode for the algorithm, briefly explain how and why it works, and formally derive its time-complexity.

Note that there's no way of figuring out the tag's actual ID. Your only option is to tap two tags against each other and see whether they light up. Each tap costs one "operation", and you can only tap two tags at a time. Your algorithm will be considered efficient if the number of taps it performs is in $O(N \log N)$.

(10 points) Devise a divide-and-conquer algorithm that performs $O(N)$ taps and formally prove its correctness.

3 Divide and Conquer on Multiple Inputs

(20 points) Given two sorted lists A and B of sizes m and n, give an $O(\log K)$ time algorithm to find the K^{th} smallest element in the list that would result from merging A and B. Formally argue that the algorithm is correct.