Foundations of Algorithms Homework 4 SU24

Strategies for solving ***Collaborative Problem*** problems are to be formed by your assigned group, with each member *participating and contributing fully*. All answers must be your own work, in your own words, even on collaborative problems—collaborate on strategies only. Individual participation will be evaluated through collaborative assessments, submitted when the homework is due.

1. [20 points total] CLRS (4th ed) 9.3-7: Professor Olay is consulting for an oil company, which is planning a large pipeline running each to west through an oil field of n wells. The company whats to connect a spur pipeline from each well directly to the main pipeline along a shortest route (either north or south), as shown in Figure 9.2 in the textbook. Given the x- and y-coordinates of the wells, how should the professor pick the optimal location of the main pipeline, which would be the one that minimizes the total length of the spurs? Explain how to determine the optimal location in linear time.

   HINT: Consider two points y_1 and y_2 on a vertical line, and a middle point m in between. What is the sum of the distances between y_1 and m, and y_2 and m? Think of a simple clothesline with pulleys at y_1 and y_2, and let m be a clothespin on the line. If you move m around, does it change the sum of the distances? What about several clotheslines in parallel? Finally, what about the case of an odd, freestanding pulley added to the mix? See the figure below.

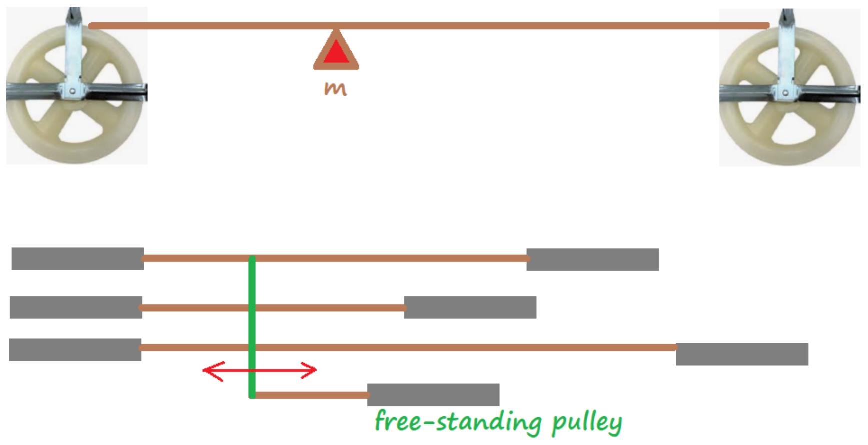

Figure 1: A set of pulleys.

2. [20 points total] Write a $\Theta(m+n)$ algorithm that prints the in-degree and the out-degree of every vertex in an m-edge, n-vertex directed graph where the directed graph is represented using adjacency lists. Prove the running time by giving a talk-through of your algorithm. Hints: See Figure 20.2 on page 550 of CLRS (4th ed) for a diagram and description of adjacency list representations of graphs. Like a lot of these questions, the Θ expression should suggest the implementation to you. You may assume the use of a dictionary data structure D, also called a *map*, where $D[i] = j$ means to associate item j with item i. Last, don't overthink this.

3. [30 points total] Given the task of finding the closest pair of points in a two-dimensional plane, some of Dr. Fink's students were tasked with implementing a brute force algorithm, but making at least one change to improve its runtime down from $\Theta(n^2)$. Usually, brute force iterates over all pairs of points by way of a doubly nested for loop. They were also asked to implement the closest points divide-and-conquer approach in the CLRS textbook that has a runtime of $O(n \log n)$.

   To improve the brute force method, two clever students independently had the idea to sort the list of points beforehand by x value. When processing point i, and in the loop that pairs i with all other points j, if the difference of the x coordinates between i and j is greater than the current minimum distance, then stop processing j and subsequent points. Instead, *skip ahead* to the next i. In this way, since all points following j have a greater x value, there is no need to consider them.

Students were directed to do testing against random points in a list of increasing size, from 3 to N points, and plotting/comparing brute force skip-ahead performance against divide-and-conquer.

Now, the *paradox* is that the comparison plot in figure 2 showed very similar asymptotic performance of our revised skip-ahead "Brute" to CLRS. It seemed that no matter how the points were randomized, the skip-ahead program always performed about the same as the divide-and-conquer. This seemed crazy; we are always taught that $n \log n$ beats n^2, so why is the plot telling us otherwise? Also, the plot seemed unstable - repeated trials showed near equal performance to divide-and-conquer, but sometimes the performance spiked a little bit high. What is going on here?

For these questions, look for the accompanying Jupyter Notebook (with student-authored code, nice job!) and use that as you care to help explore this situation. *Do not turn in the notebook* with your homework—it is only a tool for you to use.

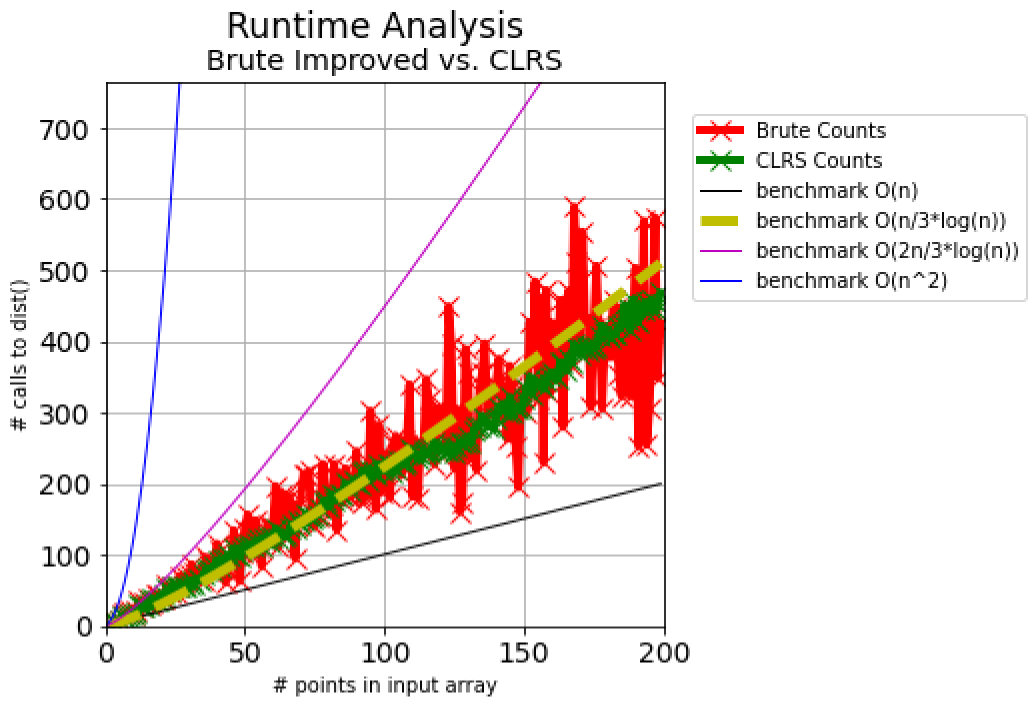

Figure 2: Can one simple tweak achieve $n \log n$ behavior?

(a) [12 points] Describe an input that "breaks" the skip-ahead algorithm, bringing it closer to its expected n^2 bound. Describe the general properties of the input; you do not need to supply code that generates it, although you may want to use the notebook to try out your theories.

(b) [12 points] Would this same bad input affect CLRS negatively as well? Why or why not?

(c) [6 points] Comparing the skip-ahead brute force optimization vs. the CLRS code, what is one reason to use the skip-ahead algorithm instead of CLRS? HINT: one reason has nothing to do with asymptotic behavior.

4. [30 points total] ***Collaborative Problem:*** In this problem we consider two stacks A and B manipulated using the following operations (n denotes the size of A and m the size of B):

   - $PushA(x)$: Push element x on stack A.
   - $PushB(x)$: Push element x on stack B.
   - $MultiPopA(k)$: Pop $min(k, n)$ elements from A.
   - $MultiPopB(k)$: Pop $min(k, m)$ elements from B.
   - $Transfer(k)$: Repeatedly pop an element from A and push it on B, until either k elements have been moved or A is empty.

   Assume that A and B are implemented using doubly-linked lists such the $PushA$ and $PushB$, as well as a single pop from A or B, can be performed in $O(1)$ time worst-case.

   (a) [15 points] What is the worst-case running time of the operations $MultiPopA$, $MultiPopB$, and $Transfer$? Hint: Consider each operation separately, think about its worst case and answer in terms of $O()$.

   (b) [15 points] Define the specific potential function as $\Phi(n, m) = 3n + m$ and use it to prove that each one of the operations have amortized running time $O(1)$. We must see the algebra for each operation. Then - use a different potential function (still with m and n) and redo your analysis on PushA, then answer does it matter what the functional coefficients are? Hint: think of the definition of potential: $\hat{c_i} = c_i + \Phi(D_{i+1}) - \Phi(D_i)$ and work this algebraically.