CS 3000: Algorithms & Data — Spring '20 — Jonathan Ullman

Homework 2
Due Friday January 24 at 11:59pm via Gradescope

Name:
Collaborators:

- Make sure to put your name on the first page. If you are using the LaTeX template we provided, then you can make sure it appears by filling in the `yourname` command.
- This assignment is due Friday January 24 at 11:59pm via Gradescope. No late assignments will be accepted. Make sure to submit something before the deadline.
- Solutions must be typeset in LaTeX. If you need to draw any diagrams, you may draw them by hand as long as they are embedded in the PDF. I recommend using the source file for this assignment to get started.
- I encourage you to work with your classmates on the homework problems. *If you do collaborate, you must write all solutions by yourself, in your own words.* Do not submit anything you cannot explain. Please list all your collaborators in your solution for each problem by filling in the `yourcollaborators` command.
- Finding solutions to homework problems on the web, or by asking students not enrolled in the class is strictly forbidden.

Problem 1. *Recurrences*

Suppose we have algorithms with running times $T(n)$ given by the recurrences:

$$T(n) = 36T(n/6) + n^2$$
$$T(n) = 8T(n/3) + n$$
$$T(n) = 7T(n/7) + n$$
$$T(n) = 11T(n/5) + n^2$$
$$T(n) = 3T(n/3) + 1$$

Solve each recurrence using the master theorem and put these running times in ascending order $T_1, T_2, \ldots, T_5$ so that $T_i = O(T_{i+1})$.

Solution:

Problem 2. *Improve the MBTA*

You have been commissioned to design a new bus system that will run along Huntington Avenue. The bus system must provide service to n stops on the eastbound route. Commuters may begin their trip at any stop i and end at any other step $j > i$. Here are some naïve ways to design the system:

1. You can have a bus run from the western-most point to the eastern-most point making all n stops. The system would be cheap because it only requires $n-1$ route segments for the entire system. However, a person traveling from stop $i = 1$ to stop $j = n$ has to wait while the bus makes $n-1$ stops.

2. You can have a special express bus from i to j for every stop i to every other stop $j > i$. No person will ever have to make more than one stop. However, this system requires $\Theta(n^2)$ route segments and will be expensive.

Using divide-and-conquer, we will find a compromise solution that uses only $\Theta(n \log n)$ route segments, but with the property that a user can get from any stop i to any stop $j > i$ making at most two stops in total. That is, it should be possible to get from any i to any $j > i$ either by taking a direct route $i \to j$ or by taking two routes $i \to m$ and $m \to j$.

(a) For the base cases $n = 1, 2$, design a system using at most 1 route segment.

Solution:

(b) For $n > 2$ we will use divide-and-conquer. Assume that we already put in place routes connecting the first $n/2$ stops and routes connecting the last $n/2$ stops so that if i and j both belong to the same half, we can get from i to j in at most 2 segments. Show how to add $O(n)$ additional route segments so that if i is in the first half and j is in the second half we can get from i to j making only two stops.

Solution:

(c) Using part (b), write (in pseudocode) a divide-and-conquer algorithm that takes as input the number of stops n and outputs the list of all the route segments used by your bus system.

Solution:

(d) Write the recurrence for the number of route segments your solution uses and solve it. You may use any method for solving the recurrence that we have discussed in class.

Solution:

(e) **(Challenge Question! You are not required to submit an answer.)** Suppose the MBTA is willing to compromise even further by designing a solution where users may need *three* segments to get from i to j. Design a solution that uses as few route segments as possible. The number of route segments should be asymptotically strictly smaller than the previous solution. That is, it should use $o(n \log n)$ route segments. For an extra challenge, what if we allow *four* segments to get from i to j?

Problem 3. *Babysitting*

You are babysitting your niece and before she will go to bed she insists on playing the following guessing game:

1. She picks a number x in $1, 2, \ldots, n$.
2. You make a guess y_1, and she simply says *correct* or *incorrect*.
3. You make a sequence of guesses $y_2, y_3, \ldots$. If your guess $y_i = x$ then your neice says *correct* and goes to bed. If your guess y_i is closer to x than the previous guess y_{i-1}, then she says *warmer* and if y_{i+1} is farther than the previous guess, then she says *colder*.

Your goal is to find x with as few guesses as possible so that your niece will go to bed. Design a divide-and-conquer algorithm that guesses your niece's number using $O(\log n)$ guesses.[1]

(a) Describe your algorithm in pseudocode.

Solution:

(b) Prove by induction that your algorithm correctly guesses the number.

Solution:

(c) Write a recurrence describing the number of guesses made by the algorithm, and solve the recurrence. You may solve the recurrence using any method you like.

Solution:

[1] **Hint:** You are allowed to guess negative numbers and numbers larger than n, although there are also correct solutions that do not use this option.