CS4800: Algorithms — S'18 — Jonathan Ullman

Homework 2
Due Friday January 26 at 4:59pm 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 26 at 4:59pm 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 prohibited.

Problem 1. *Recurrences*

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

$$\begin{aligned}
T(n) &= 25T(n/5) + n^2 \\
T(n) &= 7T(n/3) + n \\
T(n) &= 13T(n/4) + n^2 \\
T(n) &= 2T(n/2) + 1 \\
T(n) &= 4T(n/4) + n
\end{aligned}$$

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:

(b) Karatsuba's algorithm multiplies two n-digit numbers using three multiplications of $(n/2)$-digit numbers and $O(n)$ additional work, leading to the running time $\Theta(n^{\log_2 3})$. In general, for every $k \in \mathbb{N}$, we can multiply two n-digit numbers using $2k-1$ multiplications of (n/k)-digit numbers and $O(n)$ additional work. Thus, for any value of k we can obtain a divide-and-conquer multiplication algorithm in the spirit of Karatsuba's algorithm. Using the master theorem, find the running time of this algorithm for an arbitrary choice of k. Show that, for every $\varepsilon > 0$, there is some value of k so that the running time is $O(n^{1+\varepsilon})$.

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 (we'll ignore the Westbound 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 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 only 2 total stops.

(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 segments used by your bus system.

Solution:

(d) Write the recurrence for the number of routes your solution use and solve it using the Master Theorem to determine the total number of routes.

Solution:

Problem 3. *Babysitting*

You are babysitting your niece and before she will got bed she insists on playing the following game: First, she picks a number x in $1, 2, \ldots, n$. You get to make guesses $y_1, y_2, y_3, \ldots$. If your guess $y_i = x$, then your niece 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*. (For the first guess, she simply says *correct* or *incorrect*.)

Design a divide-and-conquer algorithm that correctly 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 running time of the algorithm, and use the Master Theorem to solve the recurrence.

Solution:

[1] **Hint:** You are allowed to guess negative numbers.