CS3000: Algorithms & Data — Fall '18 — Jonathan Ullman

Homework 2
Due Tuesday September 25 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 Tuesday September 25 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*

Karatsuba's algorithm is not the only divide-and-conquer algorithm for integer multiplication. There are, in fact, algorithms that can multiply n-digit numbers recursively by solving:

1. 3 multiplications each of $\frac{n}{2}$-digit numbers (Karatsuba's Algorithm),
2. 5 multiplications each of $\frac{n}{3}$-digit numbers,
3. 7 multiplications each of $\frac{n}{4}$-digit numbers, or
4. 9 multiplications each of $\frac{n}{5}$-digit numbers,

and then doing $O(n)$ work to combine the solutions.

Determine the asymptotic running time of each of these four algorithms and rank them in ascending order of their asymptotic running time. You do not need to justify your answers.

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:

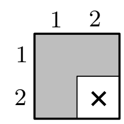
(e) **Challenge Question! For pride, not credit!** Suppose the MBTA is willing to compromise even further by designing a solution where users may need *three* segments to get from i to j. Design a solution that uses as few route segments as possible. The number of route segments should be asymptotically smaller than the previous solution. That is, it should use $o(n \log n)$ route segments.

Problem 3. *Tiling*

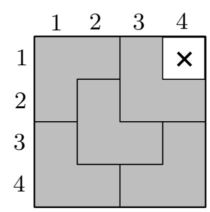
We are retiling the floor of ISEC, which has dimension $2^n \times 2^n$, and one square is reserved to be occupied by the statue of a wealthy donor. The location of the statue can be at any location (a, b) with $1 \le a, b \le 2^n$ that the donor tells us. The rest of the squares are tiled by L-shaped pieces, each covering three squares.

Here are two examples:

- If $n = 1$, and $(a, b) = (2, 2)$, we could cover the floor using a single L-shaped piece:



- If $n = 2$, and $(a, b) = (1, 4)$, we could cover the floor using five L-shaped tiles:



(a) Design a divide-and-conquer algorithm that takes as input the values n, a, b and outputs a list of the locations and orientations of $(4^n - 1)/3$ tiles.[1] Give pseudocode for your algorithm.

Solution:

(b) Write a recurrence for the running time and solve it using any method that we've covered in class.[2]

Solution:

[1] **Hint:** divide into four quadrants, each of dimension $2^{n-1} \times 2^{n-1}$, place a single tile so that each quadrant has one square covered, and then recurse.

[2] **Hint:** You may find it helpful to introduce a new variable $m = 2^n$ and solve for the running time as a function of m, then substitute to get the running time as a function of n.