CS 577: Introduction to Algorithms — Homework 1

Out: 02/02/21 — **Due: 02/09/21**

Name: _____ Wisc ID: _____

Ground Rules

- Answer the questions in the boxes provided on the question sheets. If you run out of room for an answer, add a page to the end of the document.
- The homework is to be done and submitted individually. You may discuss the homework with others in either section but you must write up the solution *on your own*.
- You are not allowed to consult any material outside of assigned textbooks and material the instructors post on the course websites. In particular, consulting the internet will be considered plagiarism and penalized appropriately.
- The homework is due at 11:59 PM CST on the due date. No extensions to the due date will be given under any circumstances.
- Homework must be submitted electronically on Gradescope.

Problem 1:

1. You are given 2^s arrays of integers that are already sorted, and each one has n elements. You want to combine these into a single sorted array of $2^s n$ elements.

   (a) One way of doing this is to repeatedly apply the MERGE subroutine from MERGESORT. That is, first merge the 1st and 2nd array, then merge the resulting array with the 3rd array, and so on. Asymptotically, how many times will this algorithm perform a comparison of two integers? Briefly justify your answer. (*Hint*: Merging two sorted arrays of size m and n requires $O(m+n)$ pairwise comparisons of integers.)

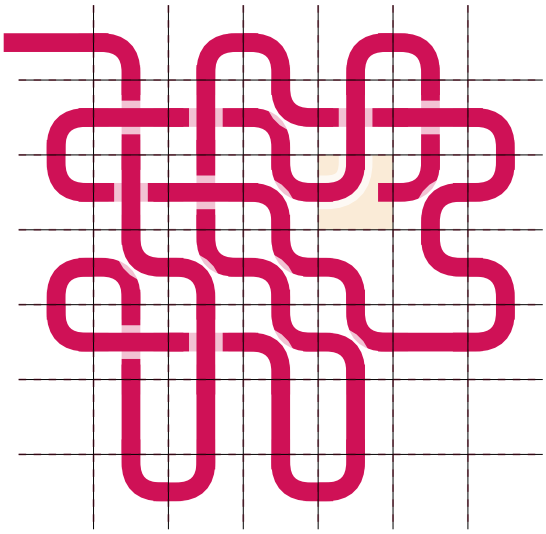
(b) Devise a divide and conquer algorithm that uses $O(2^s sn)$ comparisons of integer pairs to combine the arrays into one sorted array.

(c) Use induction to prove that your algorithm correctly combines the 2^s arrays into one sorted array.

(d) Write a recurrence as a function of n and s for the number of times your algorithm performs a comparison of two integers. State the solution of your recurrence in asymptotic notation. No explanation is required.

Problem 2:

2. You are given an image depicting a rope that is wrapping around itself. The image is split into $n \times n$ cells. You are given that the top-left cell contains one end of the rope and you want to find the cell that contains the other end. When the rope comes in from one of the edges of a cell, it leaves out of another edge (unless it is the end of the rope). The rope never goes through the same edge of a cell twice. An example of such an image split in cells is given below for $n = 7$.



Your goal is to identify the cell that contains the other end. Every time you can ask to see how the image looks like at a cell of your choice. Find an algorithm that lets you identify the cell containing the other end of the rope while looking at as few of the cells of the image as possible.

(a) Give an example showing that following the rope around requires looking at $\Omega(n^2)$ cells.

(b) Suppose you iterate over the middle column of cells. How can you tell if the endpoint is to the left of the middle column, to the right of this column, or in this column?

(c) Devise a divide and conquer algorithm that is more efficient than following the rope in terms of the number of cells it queries. Describe your algorithm in pseudocode. Aim for a total of $O(n \log n)$ queries.

(d) Prove the correctness of your algorithm.

(e) How many cells does your algorithm need to query in the example image?
For general $n \times n$ images, write down a recurrence as a function of n for the number of queries that suffice for your algorithm to find the other end of the rope. State the solution of your recurrence in asymptotic notation. No explanation is required.