CMPSC 465 Data Structures & Algorithms
Fall 2024 Mingfu Shao and Debarati Das

HW 3

Due September 20th (Friday), 11:59 pm

Formatting: Each problem should begin on a new page. When submitting in Gradescope, try to assign pages to problems from the rubric as much as you can. Make sure you write all your group members' names. For the full policy on assignments, consult the syllabus.

0. (0 pts.) Acknowledgements. List any resources besides the course material that you consulted in order to solve the assignment problems. If you did not consult anything, write "I did not consult any non-class materials." The assignment will receive a 0 if this question is not answered.

1. (15 pts.) Problem 1.

You are given a 2D matrix of integers, where each row and each column is sorted in non-decreasing order. Your task is to search for a target integer within the matrix using an efficient search strategy. The matrix is defined as follows:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix}$$

where m is the number of rows and n is the number of columns.

The goal is to find the position of the target integer in the matrix, or determine that it does not exist. You must propose a search strategy that leverages the properties of the matrix to minimize the number of comparisons.

Input

- A 2D matrix A of size $m \times n$, where each row and column is sorted in non-decreasing order.
- An integer value x, the target value to be searched for in the matrix.

Output

- If the target is found, return its position in the matrix as a tuple (i, j).
- If the target is not found, return -1.

Objectives

1. **(5 pts.)** Assume that $m = 1$, i.e., the input matrix A is essentially a sorted array of size n, and the required output is the position j where $A[j] = x$ or -1 is x is not in A. Design an algorithm for this special case and analyze its time complexity. (*Hint*: consider binary search.)
2. **(10 pts.)** Design an algorithm for this problem (the general case where A is of size $m \times n$) and analyze its time complexity. (*Hint*: consider how you can efficiently divide the search space (the matrix) by focusing on sections where the target is most likely to be, while eliminating areas where the target cannot possibly be located.)

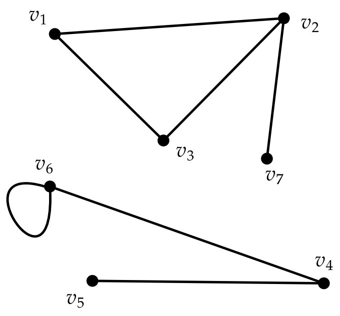
2. **(16 + 4 bonus pts.) Problem 2.** In class, we have learned how to find a pair of points that minimizes the distance between them from a given set of N points on a 2D plane. The brute force solution for this problem runs in $O(n^2)$, and we can improve the solution to $O(n \log n)$ using the divide-and-conquer approach.

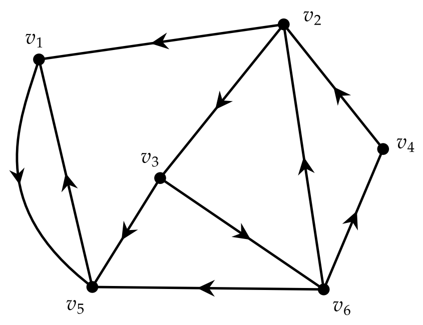
   Now, however, you are more interested in finding the "smallest triangle". Specifically, you want to find a set of three points from the given N points, where point i has coordinates (x_i, y_i), that forms a triangle with the smallest perimeter. Let $\langle i, j, k\rangle$ represent the triangle formed by points i, j and k, and $d(i, j, k)$ denote its perimeter. To simplify the problem, you may assume that no two points share the same x-coordinate or y-coordinate, and no three points are collinear.tm

   a. **(3 pts.) Brute force approach:** Write pseudocode for a brute force method. The time complexity of this approach should be $O(n^3)$.
   b. **(8 pts + 4 bonus pts.) Merging step**: We now apply the divide and conquer approach. Suppose we have divided all points into a left half P_L and a right half P_R, and have already identified the "smallest triangle", denoted as $\langle \ell_0^*, \ell_1^*, \ell_2^*\rangle$ in P_L and $\langle r_0^*, r_1^*, r_2^*\rangle$ in P_R. Let d^* be the minimum of $d(\ell_0^*, \ell_1^*, \ell_2^*)$ and $d(r_0^*, r_1^*, r_2^*)$. Let point q be the rightmost point in P_L.
      (a) (2 pts.) Prove that if a triangle with a perimeter smaller than d^* exists, then all three points must lie within the band between the lines $x = x_q - d^*/2$ and $x = x_q + d^*/2$.
      (b) (2 pts.) Prove that, for each point i in the band, if there exist points j and k that can form a smaller triangle with i, then points j and k must satisfy $y_i - d^*/2 \le y_j, y_k \le y_i + d^*/2$.
      (c) (4 bonus pts.) Suppose that we divide the band into $(d^*/2) \times (d^*/2)$ boxes, prove that the number of points in each box is at most a constant c.
      (d) (4 pts.) Describe how you would determine if there are three points in the band that form a triangle with a perimeter smaller than d^*. You can assume that B, the list of points in the band sorted in ascending order by their y-coordinates, is given. You can also assume that the constant c above is given. Your algorithm should run in $O(n)$ time.
   c. **(5 pts.) Complete algorithm and time complexity.** Describe the complete algorithm, and analyze its time complexity.

3. **(12 pts.) Problem 3.**

   1. **(3 pts.)** Draw an undirected graph with 4 vertices and 4 edges, and its adjacency matrix contains exactly 6 1s, or show that such graph does not exist.
   2. **(3 pts.)** Draw a directed graph with 4 vertices and 4 edges, and its adjacency matrix contains exactly 6 1s, or show that such graph does not exist.
   3. **(3 pts.)** Show all connected components in the undirected graph given below (left).
   4. **(3 pts.)** Show all connected components in the directed graph given below (right).


v_1
v_2
v_3
v_7
v_6
v_4
v_5


v_2
v_1
v_4
v_3
v_5
v_6