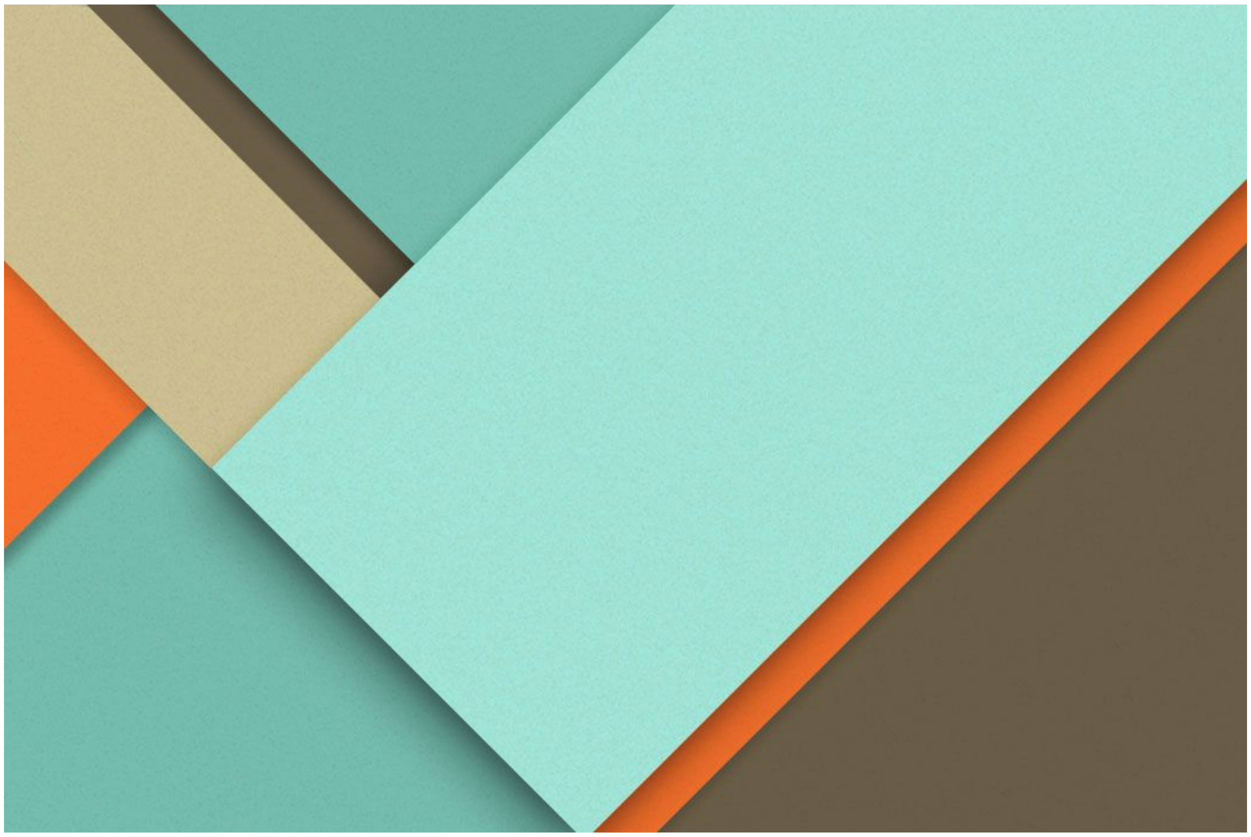


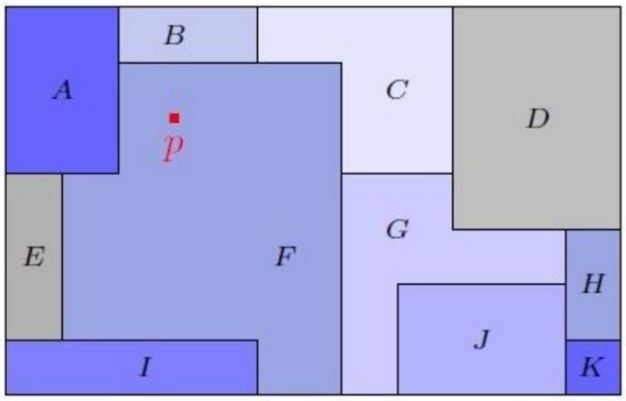
DAA Project

Region Locator (Problem 10)

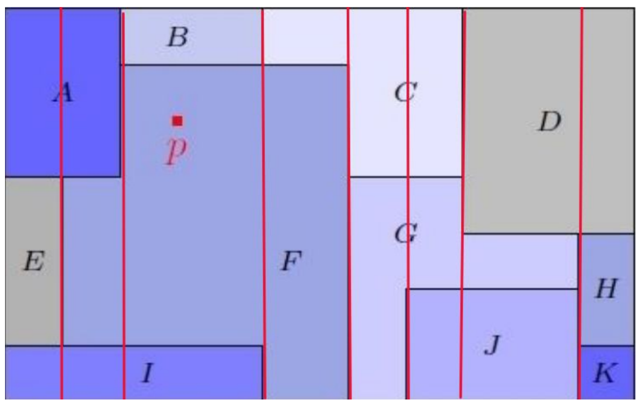
Gurleen Kaur and Swasti Shreya Mishra
(IMT2017020 and IMT2017043)

Problem Statement

A rectangle is partitioned into a number of disjoint regions. All regions have axis-parallel boundaries. Build a data structure such that the following query can be answered in O(log n) time: Given a point p, which region does it belong to?
n refers to the number of boundary line segments. Assume any reasonable representation of the input. You need not explain how the data-structure is built, just describe what you will store and how you would answer a query.

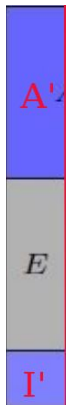


Input Format:



x1 x2 x3 x4 x5 x6 x7 x8 x9

fig (i)



If the point lies on A' we can say the point lies on A and if a point lies in I' it lies in I

x1 x2 Strip(x1,x2)

fig (ii)

- **Definition:**
    - Strip: A strip is defined by two coordinates x1 and x2 where x1 is the left vertical boundary of the strip and x2 is the right vertical boundary of the strip. It doesn't contain any other vertical boundary of any region and therefore the plane can be divided into disjoint and exhaustive set of strips. Moreover, a strip consists of horizontal lines which divides the strip into regions. (namely A',E and I' for fig (ii))

    - Region: A region lies inside a strip which is defined by two coordinates y1 and y2 where y1 is the bottom horizontal boundary of the region and y2 is the top horizontal boundary of the region. A region can span over multiple strips.

- Given 'n' strips and 'q' queries.
        - Each strip is described by two coordinates of x-axis x1, x2 and the number of regions it contains. Each region is described by two coordinates of y-axis y1, y2 and the name of the region. **Note** - the name of the region for A' will be A - refer fig (ii)
        - Each query comprises of point p which is described by coordinates x,y.
- **Constraints:**
    - All the given strips are non-overlapping.
    - All the regions specified inside a strip are non-overlapping.
    - The point given in a query always lies on the plane (i.e the union of all the regions).
    - The point given in a query never lies on the boundary of any region.
    - The strips given in the input are in the sorted order (ascending).
    - The regions given in the input are in the sorted order (ascending).

Data Structures Used:

- Strip :

  struct Strip consists of two coordinates left_x and right_x which denotes the left and right vertical boundary of the strip respectively. It consists of a vector of consecutive y-coordinates of the regions present in the strip. It also consists of a map which maps two consecutive y-coordinates to its corresponding region.

- Plane:

  map plane maps two consecutive x-coordinates to its corresponding strip.
- y_map:

  map y_map maps two consecutive y-coordinates to its corresponding region.

Pseudo code of the algorithm used

Terminologies used in code :

Point P(x,y) - given for querying

Plane - maps tuple (x1,x2) to a strip whose left and right vertical boundaries are x1 and x2 respectively.

Region - maps tuple (y1,y2) to a region whose upper and lower boundaries are y1 and y2 respectively.

```
findStrip(x){          // function takes the x coordinate of point P(x,y) as parameter
    Binary search over all the strips (the strips are given in sorted order)
    Let at some iteration the strip be S(left_x,right_x)
        If ( left_x < x and x < right_x )
            return the strip S
}
findRegion(y){          // function takes the y coordinate of point P(x,y) as parameter
    Binary search over all the regions in the above returned strip S specified by the y
    coordinates array (y_coordinates[ ]) in the structure of the strip
    Let at some iteration the y-coordinates be bottom_y and top_y such that bottom_y
    and top_y are consecutive
        If ( bottom_y < y and y < top_y )
            return the region R mapped by (bottom_y,top_y)
}
```

The region R returned above, is the region in which the point P(x,y) lies.

Proof Of Correctness

To prove the correctness of the algorithm we need to prove that the point lies in the region returned by our algorithm.

For that we'll need to prove the following -

The point P(x,y) lies only on the returned strip - proving correctness of findStrip(x)

- Lemma 1 : If the point lies on the plane defined by the union of all regions it definitely lies on a **single** strip.
    - The union of regions = the union of strips = the entire plane. Therefore the set of strips is nothing but the exhaustive set of the plane. This implies that the point will definitely lie on at least one of the strips given it lies on the plane.
    - The strips do not overlap. This implies that the set of strips are mutually exclusive.
    - From a) and b) we can conclude that the point lies on exactly one strip.

  From Lemma 1 we know that findStrip(x) will definitely return a single strip and now we need to prove that the strip returned by findStrip(x) is the strip on which the point lies.

- **Proof** :
    - For any strip S(x1,x2), the point P(x,y) either lies on the strip, to the left of the strip or to the right of the strip i.e
        - x < x1 && x < x2 - to the left of the strip
        - x > x1 && x > x2 - to the right of the strip
        - x1 < x < x2 - on the strip
    - Let x_coordinates[x1...xn] be the array of all the vertical boundaries of the strips

      l = 0, r = n-1 and mid = (l+r)/2.

      Loop Invariant - We always maintain the loop invariant, that the x coordinate of the point P(x,y) lies between x_coordinates[l] and x_coordinates[r].

      l, r and mid keep getting updated within the loop depending upon the conditions stated above.

- If x_coordinates[mid] < x and x_coordinates[mid+1] < x, then, the point P(x,y) lies to right of the strip S(x_coordinates[mid],x_coordinates[mid+1]) . Hence, we set l = mid and retain r as it is, maintaining the invariant that the point lies in the range (x_coordinates[l]........x_coordinates[r])

- If x_coordinates[mid] > x and x_coordinates[mid+1] > x, then, the point P(x,y) lies to left of the strip S(x_coordinates[mid],x_coordinates[mid+1]) . Hence, we set r = mid and retain l as it is, maintaining the invariant that the point lies in the range x_coordinates[l]...x_coordinates[r]

- If x_coordinates[mid] < x and x_coordinates[mid+1] > x, then the point lies on the strip and the loop invariant is maintained hence we return the corresponding strip.

  The loop runs as long as l < r. Once l = r = mid, the point can lie only on the strip S(x_coordinates[l],x_coordinates[l+1]).

The point P(x,y) lies only on the returned region - proving correctness of findRegion(y)

The proof of correctness of findRegion(y) follows from the proof of correctness of findStrip(x) the only difference being that instead of strips we have the regions contained within the strip and instead of x_coordinates we use y_coordinates.

Time and Space Complexity

Time complexity:

N : Total number of strips

- Within each strip there can be at most N regions

Q : Total number of queries

We binary search over all the strips and since the total number of strips are N the total time taken will be O(log N)

Once we find the strip we binary search over all the regions and since the total number of regions can be at most N, the total time taken will be O(log N)

Time taken for each query is : (log N + log N) = **O(** log N **)**

Total time taken = Q * (log N + log N) = **O(** Q * log N **)**

[Time complexity of binary search is O(log N) as it has a recurrence relation

T(n) = T(n/2) + 1]

Space complexity:

N : Total number of of strips

- Within each strip there can be at most N regions

We need to store N strips and inside each strip there's an array of regions whose size can be at most N

So, the space complexity is **O(** N **)** in the number of strips (input)

Values of N and Q:

If we expect our output within 10 seconds:

N can be at most 1000000 and Q can be at most 1000000

Sources and Contributions

- Sources : DAA class notes
- Contributions :
    - Idea & Algorithm : discussed together
    - Documentation : discussed together, but Gurleen documented Proof of Correctness and Time and Space Complexity and Swasti documented Input Format and Pseudo Code
    - Code : struct Strip written by Gurleen and findStrip and findRegion by Swasti.
    - Proof reading of document : done by both

Compilation and Running the code

Algorithm/Code file : regLoc.cpp

Test case generation file : testCaseGen.py

[gcc version 7.3.0 and Python version 2.7.15]

- Bash script:

  ```
  g++ regLoc.cpp
  python testCaseGen.py > t1.txt
  ./a.out < t1.txt > a1.txt
  ```