Advanced Track Project

Main Topics Covered:

Convex Hull using divide and Conquer
Point in Polygon query
Maintaining dynamic Convex hull
Line polygon intersection
Half plane query : Number of points on a given side of a line, Using convex hulls
Fractional Cascading : Improvement in Half Plane Query
Rectangular query : Number of points in a rectangle
Ham Sandwich theorum, Cuts
Half Plane query : Using Ham sandwich cuts
Simplex problem : First Approach
Simplex problem : Second Approach

Algorithms Implemented:

Convex Hull using divide and Conquer
Point in Polygon query
Line polygon intersection
Half plane query : Number of points on a given side of a line, Using convex hulls
Rectangular query : Number of points in a rectangle

Introduction:

Finding the Convex Hull of points in a plane :

We tackle the problem using a divide and conquer approach. The dividing step, as usual involves breaking the problem into two sub problems. The conquer part (trickier) involves making a convex hull using two smaller convex hulls. The whole procedure and implementation is explained in the main report.

Point in Convex Polygon:

This is a fundamental problem used in other algorithms and is therefore discussed in the report.
It uses binary seach trees or sorted arrays, for storing the polygon. The query is a point, and the result states whether it's in the polygon or not.

Efficiently maintaining Convex Hull under addition of points:

This was an extra question solved by us during the project. It made us think a lot, and also helped us to think about solutions to the future problems. Looking at the solution, we also realised the importance of data structures i.e. representing data in some specific format can make the problem quite easier. It is discussed in more detail in the main report.

Line Polygon Intersection:

This is another fundamental problem used in other algorithms and is discussed in the main report. The approach mainly comprises of binary searches on our hulls (or polygons).

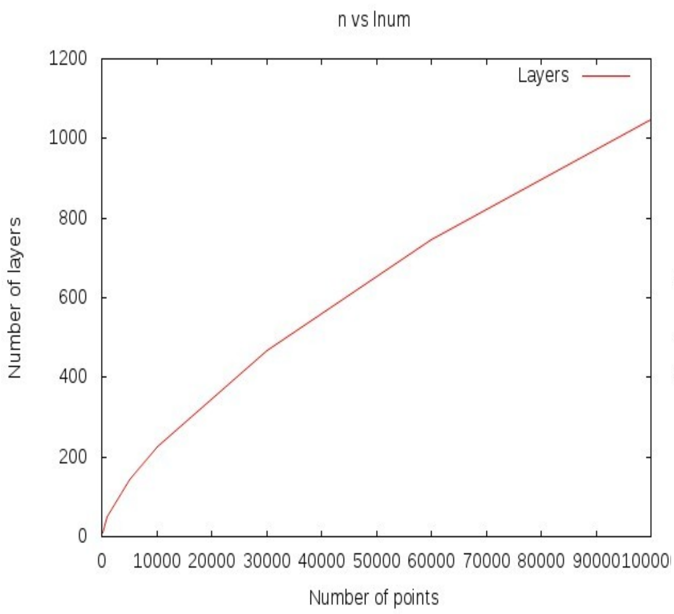
Half plane query - Number of points on a given side of a line:

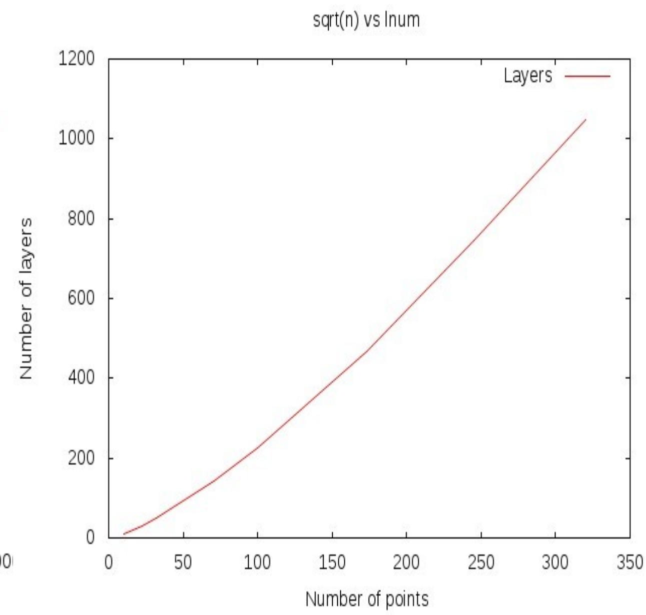
This is the first major problem which was solved by us and required quite a lot of thinking. The data structure used is quite unobvious, but it has it's own advantages and disadvantages. We have discussed the data structure used and the related algorithms in the main report.

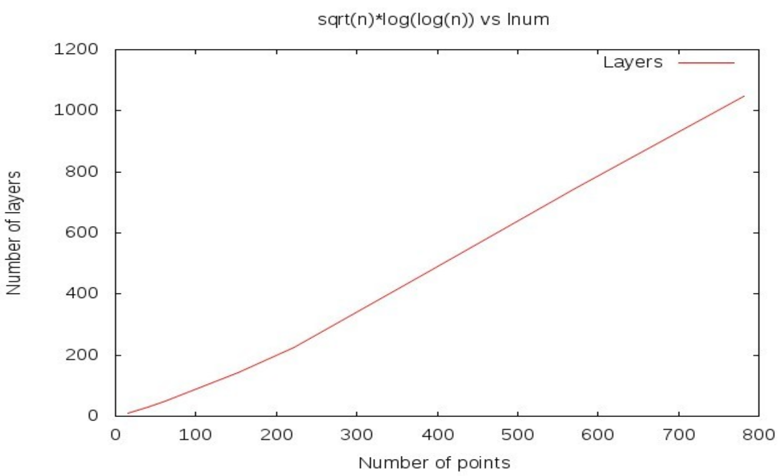
Experimental Results:

Number of layers in the onion:

We noticed that the number of hulls is almost constant for a given 'n'. So the plotted graph looks like,



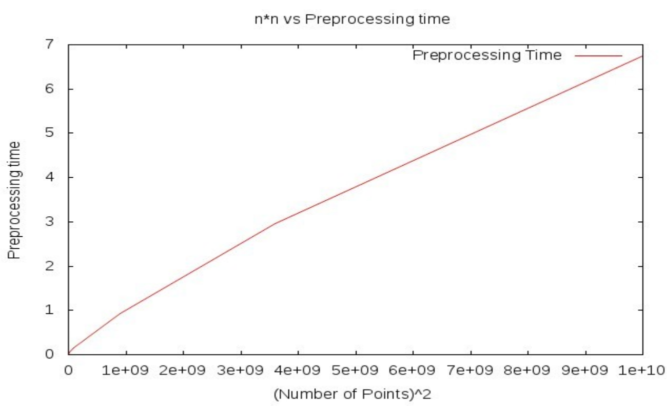




The above graphs are lnum vs n, and lnum vs sqrt(n) respectively. So, after another approximation, we felt that sqrt(n)*log(log(n)) is quite close to lnum. The graph is,

Preprocessing Time:

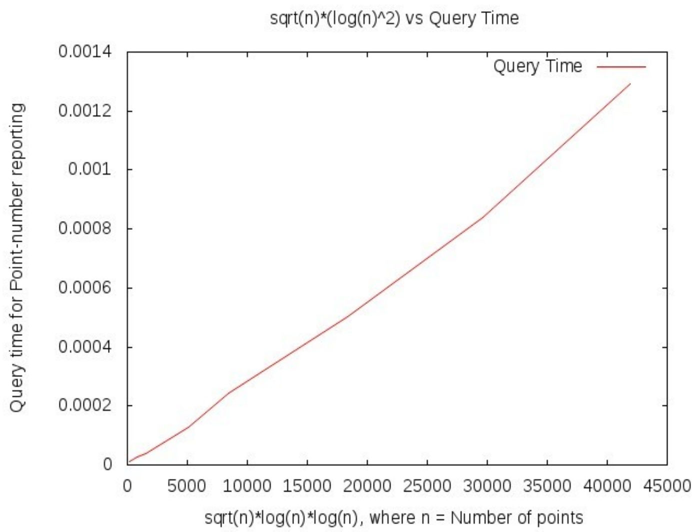
The graph of Preprocessing time vs n*n looks like,



So, due to the concave nature of the graph we can say that it is bounded above by n*n.

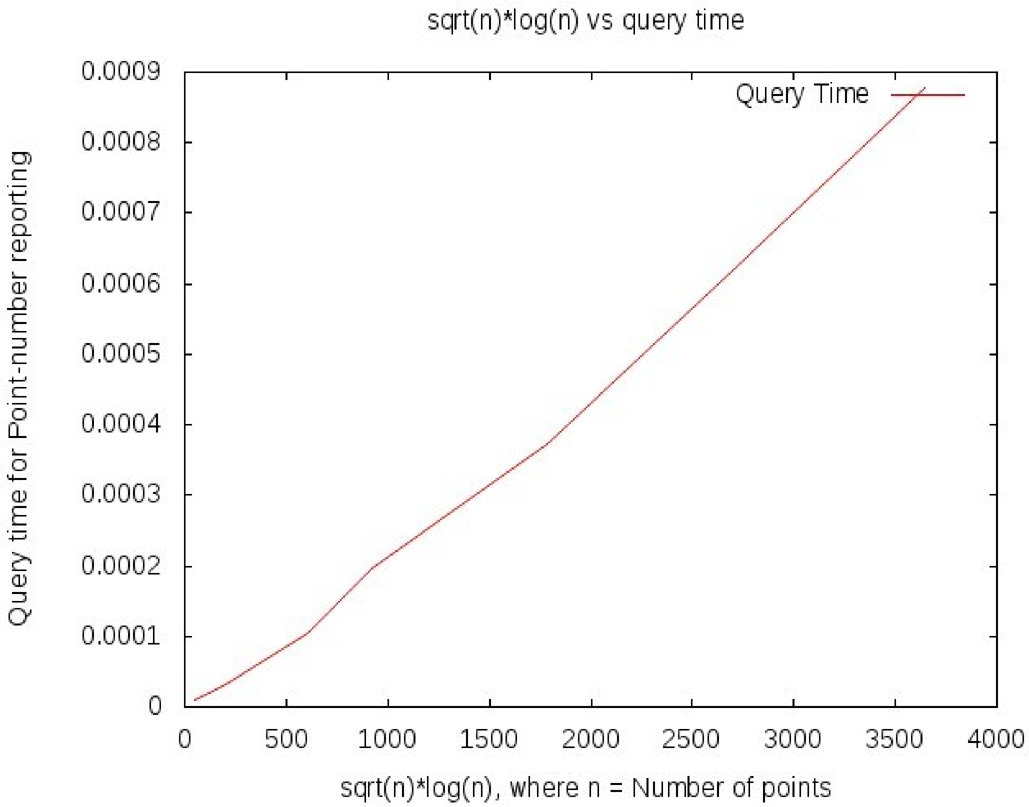
Query Time:

For point reporting the graph of time vs sqrt(n)*(log(n)*log(n)) looks like,



So we can say that the time is proportional to,
((number of points) * log(n)).

For point number reporting, the graph of time vs sqrt(n)*log(n) looks like,



Rectangular query : Number of points in a rectangle:

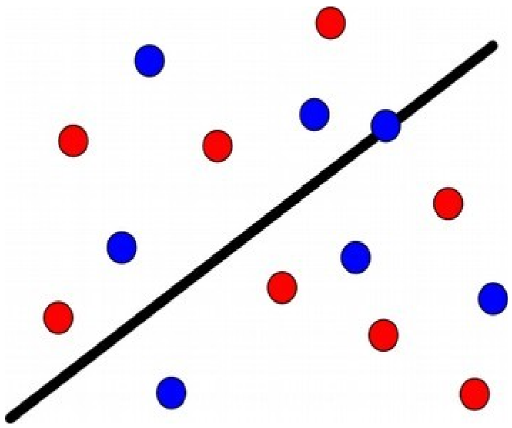
This is an extra problem solved by us during the pursuit of an efficient solution for the half plane problem. So we have just discussed some details of our solution.
The data structure created consists of many sorted arrays, each array has the elements in some specific range in sorted order. We have many such ranges, and thus the query can be decomposed into these sub ranges and computing it becomes quite easier.

Ham Sandwich Theorum:

Ham Sandwich theorum is actually a general for 'n' dimensions. But, we will be using only the two dimension version, which is,

For a finite set of points in the plane, each colored "red" or "blue", there is a line that simultaneously bisects the red points and bisects the blue points, that is, the number of red points on either side of the line is equal and the number of blue points on either side of the line is equal.

It is depicted in the following figure,



The line which divides it in equal parts is called a Ham Sandwich cut. We use the HS (Ham Sandwich) theorum to prove the existence of two lines, which divide the plane into four quarters, each of which contains the same amount of points.

Half Plane query : Using Ham sandwich cuts:

This is an alternative solution to the half plane problem discussed earlier. This solution involves Ham Sandwich cuts which we've just discussed. This is a sub prolem used to solve the main problem of our project. We have discussed the data structure used and the related algorithms in the main report.

Simplex problem : First Approach:

This was the first solution we were able to think of to answer a simplex query efficiently.
It uses a mixture of approaches which we thought of while solving the points in rectangle problem, and half plane problem. The details are given in the main report.

Simplex query : The Second Approach:

This is an improved solution to solve the simplex problem. This is the main problem in the project, and took a considerable amount of effort. This solution involves an approach which is similar to the one used to solve half plane query. We have discussed the data structure used and the related algorithms in the main report.