Assignment 2

Computational Geometry (CS60064)

Suahas Jain	19CS30048	suhasjain142@gmail.com
Sarthak Johnson Prasad	18CS10049	sarthakjohnsonprasad@gmail.com

Question 1

Show a partition with minimum number of y-monotone polygons. Justify the minimality of partition.

Answer:

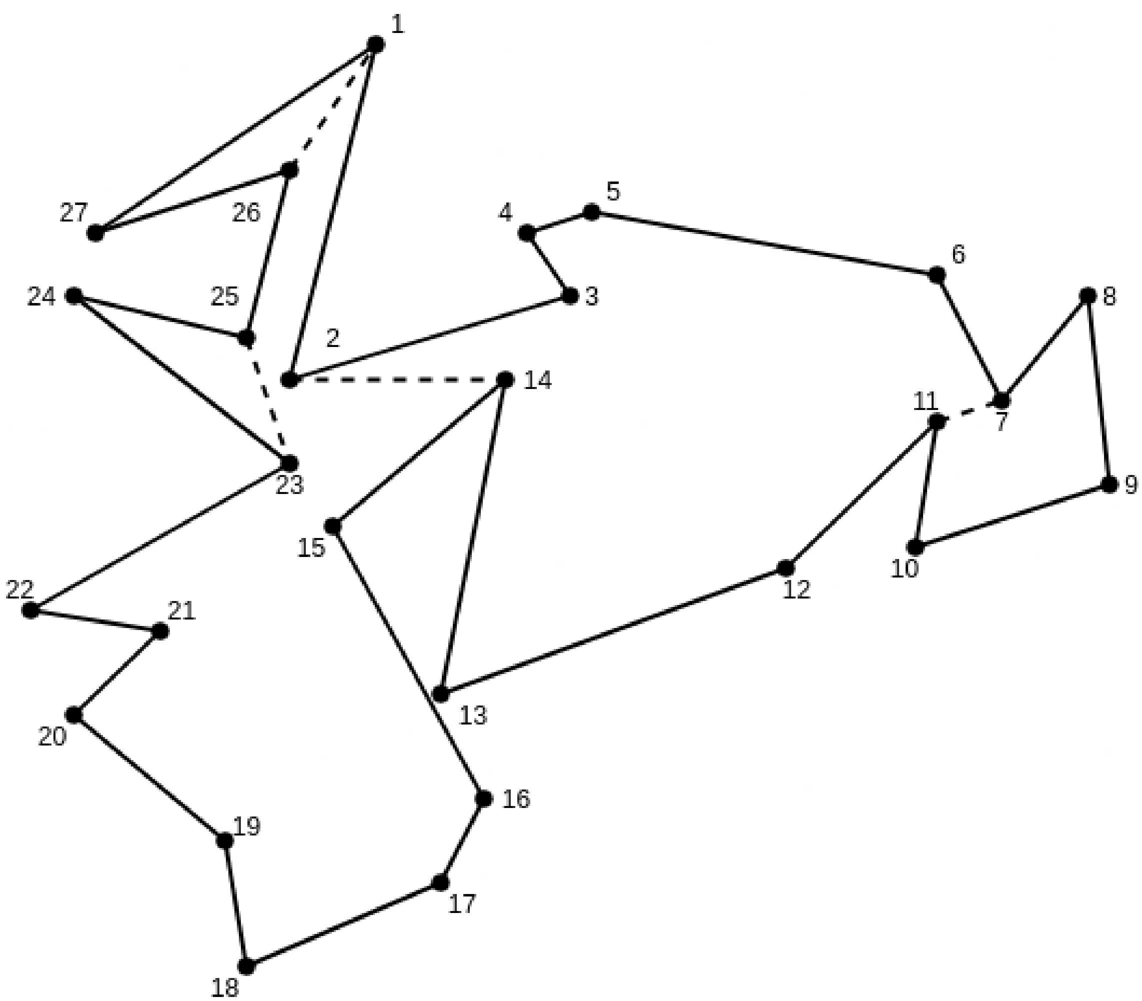

Figure 1: Division into y-monotone polygons

Minimum number of y-monotone polygons the given polygon can be divided into: 5

Justification

There are 4 top vertices: 1, 5, 8, 24
There are 4 bottom vertices: 10, 13, 18, 27
We know for a fact that a single y-monotone polygon cannot have more than one top vertex and one bottom vertex. So we can say that we need at-least 4 y-monotone polygons to accommodate all the top and bottom vertices. As we can observe from the figure that there is only one top vertex (1) which is above the bottom vertex 27. So it becomes necessary to draw a diagonal from split vertex 26 to 1. When we draw this diagonal we can see that vertex 1 will now act as top vertex of 2 different y-monotone polygons. We also need 3 more y-monotone polygons to accommodate rest of the top vertices. So, the minimum comes down to 5, which is possible as we have shown in the figure.
To make sure that there are no split or merger vertices we have made 3 more optimum diagonals.
-We join the merge vertex 2 with the split vertex 14.
-We join the split vertex 11 with the merge vertex 7.
-We join the merge vertex 25, we join it to 23.
Hence, when we draw these 4 optimum diagonals our total y-monotone polygons comes out to be 5.

Question 2

Given n points in general positions in the 2D-plane, sketch an O(n log n)-time algorithm for determining the Tukey depth of a query point.

Answer:

Algorithm

- Let us call the query point as ***Q*** and let the total number of points be ***n***. We take a reference horizontal line from which we calculate our polar angles with respect to ***Q***. We calculate angles made by all the points from this line for all the ***n-1*** remaining points and sort these points according to these angles. Let us call the sorted sequence of these remaining points as P_1, P_2, ..., P_{n-1}. We will call the angles the sorted angles of these n-1 points as A_1, A_2, ..., A_{n-1}.
  Note: As the points lie in general position $A_1 < A_2 < .... < A_{n-1}$ and $0 \leq A_i < 2\pi$.

- For a point P_i we find the point P_j with the largest angle such that $A_j \leq A_i + \pi$. Now if we have ***i*** and ***j*** we can say that we have 2 halves of a plane. One half contains points from ***i*** to ***j*** i.e. total ***(j-i+1)*** points and the other half contains ***(n-1) - (j-i+1)*** points. We take a minimum of ***(j-i+1)*** and ***(n-1) - (j-i+1)*** across all ***i*** such that $A_i < \pi$ (since the other half is just symmetric). We add 1 to this minimum value (to account for point **Q**) to get our answer.

- If we do the previous step in a naive brute force manner it will take $\mathbf{O(n^2)}$ time which is not ideal. So, we follow the two-pointer approach. Notice that when we increase $\boldsymbol{i}$ by 1 the value of $\boldsymbol{j}$ either increases by some amount or just remains constant. We can keep two points, one for $\boldsymbol{i}$ and one for $\boldsymbol{j}$ which will move in the forward direction by atmost ***n-1*** amount. Hence, this can be accomplished in $\mathbf{O(n)}$ time.

Complexity Analysis

- Calculating the polar angles for all the points take $\mathbf{O(n)}$ time.
- Sorting all the polar angles and storing all $\boldsymbol{P_i}$ and $\boldsymbol{A_i}$ takes $\mathbf{O(n * log(n))}$ time.
- Following the two pointer approach, calculating points in both halves and taking minimum on each step takes a total of $\mathbf{O(n)}$ time.

In total the whole algorithm takes $\mathbf{O(n * log(n))}$ time and $\mathbf{O(n)}$ space.

Question 3

You are given a simple polygon P with n sides and two points s and t in P, and let T denote a triangulation of P. Show that the Euclidean shortest path (ESP) between s and t is unique. Also show that the minimal set of triangles containing the ESP forms a path in the dual tree of T.

Answer:

Proof 1

Euclidean Shortest Path between two points in a polygon is unique

Let us denote the ESP between s and t as $\pi(s,t)$ and shortest distance between s and t as $d(s,t)$. If $\pi(s,t)$ is not unique, then we can say that there will exist at least two different paths $\pi_1(s,t)$ and $\pi_2(s,t)$ from s to t. Let a and b be two points of $\pi_1(s,t) \cup \pi_2(s,t)$ such that the two paths are disjoint between a and b. What we mean by the paths being disjoint is, $\pi_1(a,b) \cup \pi_2(a,b) = \{a,b\}$. We have $\|\pi_1(a,b)\| = \|\pi_2(a,b)\| = d(a,b)$. The two paths $\pi_1(s,t)$ and $\pi_2(s,t)$ enclose some region of P's interior which is free of obstacles. We can say so as it has been given that polygon P is simple. At least one of the two paths has a convex corner and cutting off that convex corner will always shorten the path. This is a contradiction since the path was chosen to have minimum length. Hence, we can say there exist only one unique path $\pi(s,t)$.

Proof 2

Minimal set of triangles containing the ESP forms a path in the dual tree of T
To prove this we need these two lemmas to be correct:

- **Lemma 1:** *$\pi(s,t)$ crosses each diagonal of the triangulation at most once*
  This can be said so because if $\pi(s,t)$ crosses the diagonals in the triangulation at 2 distinct points a and b then we can just directly connect a to b and make the path shorter, hence it is a contradiction.

- **Lemma 2:** *It only crosses those diagonals which separate s and t into two parts*
  If we say that $\pi(s,t)$ crosses a diagonal d_k which does not divide s and t into two different parts of P then we can say that in one part there will be both s and t and in other part there will be none of these two points. So for a path to start at s and finish at t, if it goes into the part where both s and t are not there then it will have to cross d_k once again to reach to t which is a contradiction of lemma 1.

If $\pi(s,t)$ crosses a set of diagonal which divide s and t only once then it will cross these diagonals in a unique sequence hence forming a path in the dual tree of T.

Question 4

4(a)

Let L be an arbitrary line segment interior to a convex polygon P with n vertices. Does there exist a triangulation such that the number of intersections of L with all diagonals become $O(log\, n)$? If so, provide a method for constructing such a triangulation.

Answer: Yes, there does exist such an algorithm to triangulate convex polygons such that number of intersections of L with all diagonals become O$(\log n)$ as given below.

Algorithm

- To triangulate the convex polygon we add some diagonals in each iteration and reduce the number of sides in our polygon by almost half. We keep adding diagonals to smaller and smaller polygons in each step till the whole polygon is completely triangulated.

- Now we will talk about what we actually do in each iteration. In each iteration we loop through the vertices and we add a new diagonal between every alternate vertex. Now, our new polygon (for the next iteration) will be the polygon formed by these newly added diagonals. If the number of vertices is odd we also take the last side in the polygon for the next step.

- Since we are creating a new triangle every time we draw a diagonal and are cutting off every alternate vertex from the polygon which will be used in the next iteration we can say if the number of diagonals in the $i^(th)$ step are K_i then, $K_i = \lceil K_{i-1}/2 \rceil$.

- In the worked out example below diagonals added in each iteration have been colored.
  -Diagonals added in the first iteration are red. (11 sides → 7 sides)
  -Diagonals added in the second iteration are blue. (7 sides → 4 sides)
  -Diagonals added in the third iteration are orange. (4 sides → Triangulation done)

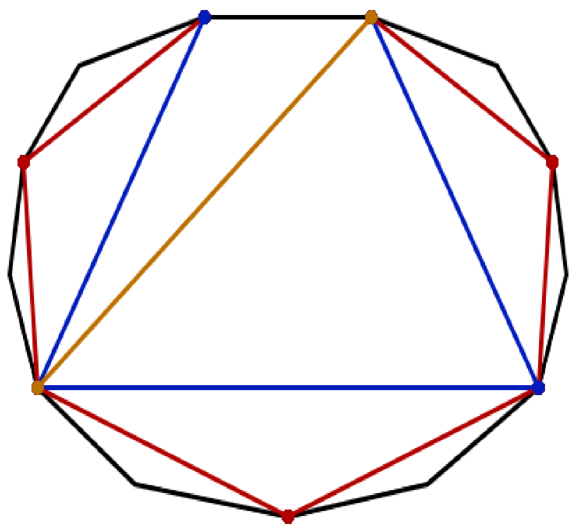

Figure 2: Worked out example for polygon with 11 vertices

Correctness

Let us say we are going through the $i^(th)$ iteration and let us say that number of diagonals added during this iteration is K_i. We can say that out of these K_i new diagonals, maximum number of diagonals which can intersect with L is two, as a straight line cannot re-enter a convex region once it has exited it. Now, we claim that there will be $O(log\, n)$ iterations as in each iteration number of edges of the polygon gets almost halved. Hence, L can intersect with the diagonals atmost $O(log\, n)$ times.

4(b)

Are there any polygons such that for any triangulation, such a line L will have Ω(n) intersections with diagonals? If so, show an example.

Answer: No, there does not exist a polygon such that for any triangulation line segment L will have Ω(n) intersections with the diagonals.
The reason is that we can always draw a line segment L which lies completely inside one of the triangles of the triangulation, hence it will have 0 intersections with the diagonals in that particular triangulation.