Computational Geometry (CS60064)

Assignment 4

Ashutosh Kumar Singh - 19CS30008

Vanshita Garg - 19CS10064

Question 1

We are given a set S of n triangles in the plane such that the boundaries of the triangles are disjoint, but one may completely enclose another and a set P of n points in the plane. We need an $O(n\ log\ n)$ time algorithm that reports all points in P that lies outside all triangles in S.
For this, we will use a sweep line algorithm as explained below.

Algorithm

1. Sort the vertices of all the triangles and also the points in set P w.r.t. to their x-coordinates and store them in an event queue Q. There are two types of events, viz., the occurrence of a vertex of the triangle and occurrence of a query point from P. Mark them differently in Q. For each vertex type event, store the two line segments of the triangle such that this vertex is one of the end-points of the line segments.

2. Sweep a vertical line from left to right and maintain a binary search tree that stores the sweep line status. The sweep line status will store the active line segments in the sorted order, sorted w.r.t. to their y-coordinates of the point of intersection of this line segment and the sweep line. A line segment of a triangle is considered to be active if it has an intersection with the current position of the sweep line.

3. Let the event that is currently at the front of Q be a point p. The two types of events are handled as:

   - If the extracted event is of type vertex, then check for both the line segments associated with this vertex. If a segment lies to the left side of the sweep line, delete it from the sweep line status and if a segment lies to the right side of the sweep line, insert it into the sweep line status.
     But before doing this, find the two line segments from the sweep line status such that the current vertex point lies in between these two line segments. If these two line segments belong to the same triangle, then this means that we have encountered a vertex of a triangle that is inside another triangle. All such triangles that are inside another triangle are discarded from consideration as they do not affect the end result.
   - If the extracted event is a query point in P, then find the two line segments from the sweep line status tree such that the query point lies in between these line segments, using binary search. There can be two cases:
     - All the line segments lie on one side of the query point. In this case, this point lies outside all the triangles in S.
     - The two line segments found belong to the same triangle. In this case, the query point lies inside the triangle.

4. Repeat step - 4 until the event queue Q is non-empty.

5. Output all those points from P that lies outside the triangle as marked in step - 4.

Time Complexity Analysis

- The total number of end-points for the n line segments of set S is $2n$ and the total number of query points in set P is n, hence the sorting in step - 1 takes $O(n\ log\ n)$ time.
- Handling a vertex type event in step - 3 is either a insertion or a deletion in the binary search tree which can take a maximum of $O(logn)$ time for each event. Hence, the total time for $3n$ such events will be bounded by $O(n\ log\ n)$.
- Handling a query point in step - 3 requires searching for two line segments in the binary search tree which can take a maximum of $O(log\ n)$ time for each event. Hence, the total time for n such events will be bounded by $O(n\ log\ n)$.

Hence, the total time complexity of the algorithm is bounded by $O(n\ log\ n)$.

Question 2

We are given a simple arrangement $A(L)$ of a set L of n lines. We need an $O(n\ log\ n)$ time algorithm to construct the convex hull of all the intersection points. Let S be the set of all the intersection points.
If W is any set of points, then we use $CH(W)$ to denote the convex hull of W. A point $p \in W$ is a corner iff it belongs to $CH(W)$ and it does not belong to the straight-line segment which joins the point preceding it on $CH(W)$ to the one succeeding it on $CH(W)$. The result of deleting all the non-corner points from $CH(W)$ is called the edge-hull of W and is denoted by $ECH(W)$. If all the points of W are in general position (i.e. if no three of them are aligned) then every point of the convex hull of W is a corner, and therefore in that case $ECH(W) = CH(W)$. However, the points of the set S under consideration are certainly not in general position, and therefore $ECH(S)$ and $CH(S)$ differ.
We first compute $ECH(S)$ and then obtain $CH(S)$ from $ECH(S)$.

Computing $ECH(S)$ - Algorithm Edge-Hull

1. Sort the n input straight lines by decreasing slope. Let $L_0, ..., L_{n-1}$ be the n lines listed by decreasing slope, i.e. if the equation of L_1, is $y = a_1x + b_1$, then we have $a_0 > a_1 > ... > a_{n-1}$. This step takes $O(n\ log\ n)$ time.
2. Let point q_i denote the intersection of lines L_i and $L_{(i+1) mod\ n}$ Find the set $Q = \{q_0, ..., q_{n-1}\}\cdot$ This takes $O(n)$ time.
3. Compute $ECH(Q)$ using any of the known $O(n\ log\ n)$ time convex hull algorithms like Graham's Scan.

It is obvious that the above algorithm runs in $O(n\ log\ n)$ time. Correctness of the algorithm would immediately follow if we could prove that $ECH(Q) = ECH(S)$. To prove this, it suffices to show that if p is a corner point of S then $p \in Q$. So let p be a corner point of S. Let L_i and $L_j, j > i$ be the two lines whose intersection is p. To prove that $p \in Q$, we must show that either $j = i+1$ (i.e. $p = q_i$), or $j = n-1$ and $i = 0$ (i.e. $p = q_{n-1}$). We prove this by contradiction. Suppose to the contrary that $p \neq q_i, q_j$. Since $p \neq q_i$, there is at least one line L_k whose slope is between the slopes of L_j and L_i, i.e. $a_i > a_k > a_j$. Let v and w be the points of intersection of L_k with L_i and L_j respectively (see Figure 1).

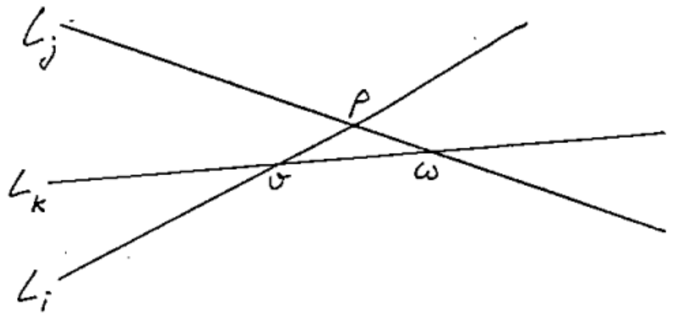

Figure 1

We continue the proof assuming v is to the left of w. as in the figure (the argument when w is to the left of v is entirely symmetrical). Since $p \neq q_{n-1}$ one (or both) of the following is true:
(i) $j \neq n-1$, or
(ii) $i \neq 0$

If (i) holds then we obtain a contradiction as follows. Line L_{n-1} must cross line L_i, somewhere, say at point s. If s is to the right of p then p is on the straight-line segment joining s to v, which contradicts the fact that p is a corner point. If, on the other hand, s is to the left of p. then L_{n-1} intersects L_j at a point t which is to the left of p. This implies that p is on the straight-line segment joining t to w, which contradicts the fact that p is a corner point. Therefore (i) cannot occur.
The argument for finding a contradiction when (ii) holds is similar to the above argument for (i), except that L_0 plays the role of L_{n-1}. Since either (i) or (ii) leads to a contradiction, it follows that our original assumption (that $p \notin Q$) was wrong.
This completes the proof that every corner point belongs to Q, from which the correctness of the algorithm follows.

Computing $CH(S)$
The algorithm for computing $CH(S)$ works as follows. We compute $ECH(S)$ in time $O(n \, log \, n)$ using the algorithm of the previous section. Then we mark every edge of $ECH(S)$ which is along one of the input lines as being specia1. This takes $O(n)$ time. Then for every line L_i, we do the following. First we find the intersection of L_i with the convex polygon $ECH(S)$. This takes $O(log \, n)$ time by using the algorithm for intersecting a line with a convex polygon. Let s_i and t_i be the points of intersection of L_i with $ECH(S)$. If s_i (respectively t_i) is a corner, or the edge of $ECH(S)$ to which s_i (respectively t_i) belongs is special, then add s_i (respectively t_i) to a set H (H is initially empty). Since this is done for every L_i, the total cost of creating H is $O(n \, log \, n)$ time, and $|H| \leq 2n$. The set H now contains all the points of S that appear on $CH(S)$, whether they are corners or not, and therefore $CH(H) = CH(S)$. Computing $CH(H)(= CH(S))$ can now be done in time $O(n \, log \, n)$ by using any of the known $O(n \, log \, n)$ time convex-hull algorithms.

Since the time complexity of every step is atmost $O(n \, log \, n)$, the total time complexity is $O(n \, log \, n)$.

Question 3

We are given a set of n points on the plane. We need to prove that checking for the collinearity of any three points among these n points is $3SUM$-hard.

Proof: In order to prove that the test for collinearity of three points in a set of n points on the plane is $3SUM$ hard, we make a reduction from $3SUM$ to this problem. For this, we map from x to (x, x^3) for each x in the $3SUM$ instance, assuming the three chosen numbers are distinct. Now, we can show that three points on this curve will lie on a straight line if and only if there are three integers summing to 0 in the original set of $3SUM$ instance. Let the three points on this curve be $A(x_1, x_1^3)$, $B(x_2, x_2^3)$ and $C(x_3, x_3^3)$. For these points to lie on a straight line, we will have:

$$\tfrac{x_3^3-x_2^3}{x_3-x_2} = \tfrac{x_2^3-x_1^3}{x_2-x_1} \Rightarrow x_3^2 + x_2^2 + x_3x_2 = x_2^2 + x_1^2 + x_2x_1 \Rightarrow (x_3 - x_1)(x_1 + x_2 + x_3) = 0$$
$$\Rightarrow x_1 + x_2 + x_3 = 0 \text{ (since } x_3 \neq x_1)$$

Hence, we can say that checking for the collinearity of any three points among n points is $3SUM$-hard.

Question 4

We are given a visibility graph $G(V, E)$ of a set of n disjoint line-segments on the plane such that no three end-points of line-segments are collinear, and no line segment is horizontal or vertical. We need to prove that G admits a cycle through all its vertices.

Proof: In order to prove that G admits a cycle through all its vertices, it is sufficient to prove that there exists a Hamiltonian Polygon whose vertices are exactly the endpoints of the line segments and whose sides correspond to the edges of G.

Given a set S of disjoint line segments in the plane, denote by $V(S)$ the set of segment endpoints from S. A simple polygon P is defined as a closed region in the plane enclosed by a simple closed polygonal curve ∂P consisting of a finite number of line segments. Let $V(P)$ denote the set of vertices of P.

A simple polygon P is a Hamiltonian polygon for S, if $V(P) = V(S)$ and the sides of P correspond to edges of $Vis(S)$, where $Vis(S)$ is the visibility graph.

We say that a finite set $\mathcal{D}$ of pairwise non-overlapping simple polygons is a dissection of P , if $P = \cup_{D\in\mathcal{D}} D$. (Two polygons overlap, if there is a common point in the relative interior of both.)

We also state a few properties as follows:
For a set S of disjoint line segments, not all in a line, and a side yz of $conv(S)$, there is a simple polygon P whose sides correspond to edges of $Vis(S)$ and a dissection $\mathcal{D}$ of P satisfying the following lemmas:

- (L1) yz is a side of P
- (L2) for every $s = pq \in S$, either $s \subset int(P)$ or $\{p, q\} \subset V(P)$
- (L3) for every $s \in S$, if $s \subset int(P)$ then there is a $D \in \mathcal{D}$ such that $s \subset int(D)$, otherwise $s \cap int(D) = \phi$ for all $D \in \mathcal{D}$
- (L4) every polygon $D \in \mathcal{D}$ is convex
- (L5) every polygon $D \in \mathcal{D}$ has a common side with P which is different from yz.

The outline of the proof is as follows. We start with $P = conv(S)$ and $\mathcal{D} = \{P\}$ which together satisfy already (L1) and (L5). In the following, the polygon P and the set D are modified such that these properties are maintained and $V(P)$ never decreases. In a first phase, property (L2) is established by including the second endpoints of those segments for which one endpoint is already in $V(P)$. Then a simple dissection by diagonal segments assures (L3). Finally, during a second phase the dissection $\mathcal{D}$ is refined until all sets in D are convex, as demanded in (L4).

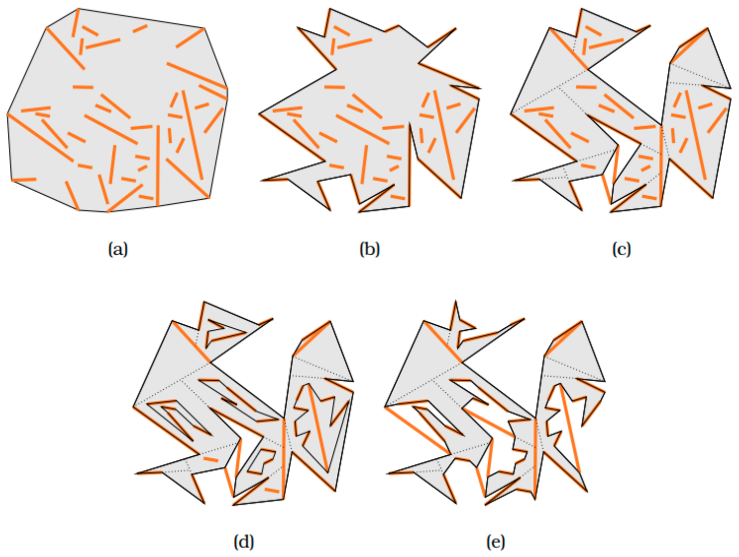

Figure 2: Steps in the proof

Proof

We prove by induction the following statement. For a set S of disjoint line segments, not all in one line, and for any fixed side yz of the polygon $conv(S)$, there is a Hamiltonian polygon H for S such that yz is a side of H.

The statement holds for $|S| = 2$. Suppose it holds for all S' with $1 < |S'| < |S|$.

Consider the simple polygon P and the set $\mathcal{D}$ of polygons described in the lemma. If both endpoints of every segment are in $V(P)$, then the statement holds. If there is a segment s whose neither endpoint is in $V(P)$, then by properties (L2) and (L3), s is in the interior of some $D \in \mathcal{D}$. By property (L5), D has a common side $ab \neq yz$ with P. By (L3) and (L4), $C(D) = conv(S \cap int(D)) \subset int(D)$. Moreover, $C(D)$ has a side cd such that both ac and bd are visibility edges. If $c_1d_1, c_2d_2, ..., c_md_m,\ m \geq 1$ are the segments in $int(D)$ and they are all collinear in this order, then replace the side ab of P by the path $ac_1d_1c_2d_2...c_md_mb$. Otherwise there is, by induction, a Hamiltonian polygon $H(D)$ for $S \cap int(D)$ such that cd is a side of $H(D)$. Replace the side ab of P by the path $(a, c) \oplus (\partial H(D) \setminus cd) \oplus (d, b)$. Doing so for each $D \in \mathcal{D}$ that contains segments from S results in a Hamiltonian polygon (see Figure 2). (For two polygonal arcs $A = (a_1, a_2, ..., a_k)$ and $B = (b_1, b_2, ..., b_l)$ with $a_k = b_1$, we denote by $A \oplus B$ the concatenation $(a_1, a_2, ..., a_k, b_2, ..., b_l)$ of A and B.)