Assignment 4

Computational Geometry (CS60064)

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

Question 1

Let S be a set of n triangles in the plane as shown in the figure below. The boundaries of the triangles are disjoint, but one may completely enclose another. Let P be a set of n points in the plane. Outline an $O(n\,log(n))$ algorithm that reports all points in P that lie outside all triangles in S.

Answer:

Algorithm

We use a plane sweep algorithm for this purpose which goes as follows:

- Make a set of all the points in the plane which include all three vertices of all the triangles in S and all the points in P. Sort all of these points according to their horizontal position. Also maintain markers for all the points which indicate if the point lies in P or not and which triangle a vertex belongs to.
- We will have different events associated with each type of points. These events are as follows:
  - **Test Point**: For a point which lies in P.
  - **Insert Triangle**: For the leftmost vertex of a triangle.
  - **Modify Triangle**: For the middle vertex of the triangle.
  - **Remove Triangle**: For the rightmost vertex of the triangle.
- Between two neighboring events, when we passed the beginning of triangle and not passed the ending we can characterise it by two lines, one for top and one for bottom edge. If the point lies between these two lines then it lies inside the triangle otherwise it lies outside.
- During the plane sweep, we keep the active objects (lines) sorted by their vertical coordinate in a data structure, let us call it T, allowing the Search, Insert and Delete operations in $O(logn)$ time (for example, a dynamically balanced search tree). Since triangles do not intersect, the vertical ordering is well defined. If a triangle vertex

is found to be inside another triangle, the lines belonging to the inside triangle can be discarded.

- When we encounter different kinds of events that we talked about earlier, we'll handle them in the following manner:
  - **Test Point**: We determine the position of this point w.r.t the currently active line. We find the pair of lines between which the point lies and then check if they belong to the same triangle or not. If they do not belong to the same triangle that means the point does not lie inside any triangle hence we report the point.
  - **Insert Triangle**: We insert a new two new lines in T which are the two leftmost edges of the triangle.
  - **Modify Triangle**: Instead of the edge between the leftmost vertex and the middle vertex, activate the edge between the middle and rightmost vertex. This involves one deletion and one insertion operation in T.
  - **Remove Triangle**: Remove the two rightmost edges of the triangle from T.

Time Complexity

The steps in the algorithm take the following time:

- Sorting all the points initially according to their horizontal position takes $O(n\,log(n))$ time.
- Checking the position of a point w.r.t a a line takes $O(1)$ time. Finding the two closest lines by binary search takes $O(log(n))$ time. So for n points it takes at most $O(n\,long(n))$ time.
- Inserting, searching and deleting a triangle in T takes $O(log(n))$ time so for n triangles it takes at most $O(n\,long(n))$ time.

In total the whole algorithm reports all points in P which lie outside all triangles in $O(n\,long(n))$ time.

Question 2

Let $A(L)$ denote a simple arrangement of a set L of n lines. Describe an $O(n\,log(n))$-time algorithm for constructing the convex hull of all intersection points.

Answer:

Algorithm

Let $L_0, L_1, ..., L_{n-1}$ be the n lines in $A(L)$.
Before proceeding with the algorithm we define $CH(S)$ and $EH(S)$.
Let S be any set of points. We know that $CH(S)$ is defined as the convex hull of S. A point $p \,\epsilon\, S$ is called a *corner* iff it belongs to $CH(S)$ and it does not belong to the straight-line segment which joins the point preceding it on $CH(S)$ to the one succeeding it on $CH(S)$. The result of deleting all the non-corner points from $CH(S)$ is called the edge hull of S and is denoted by $EH(S)$. If we consider general set of points, $EH(S)$ will be equal to $CH(S)$.
We denote S as the set of intersection points of lines in $A(L)$ The algorithm goes as follows, first we calculate $EH(Q)$ and then $CH(S)$:
To calculate $EH(S)$:

- Sort the n lines in order of decreasing of slope, , i.e., if the equation of L_1 is $y = a_1x + b_1$ then we have $a_0 > a_1 > ... > a_{n-1}$.
- Let q_i denote the intersection point of L_i and $L_{i+1 mod n}$. We then generate $Q = q_0, ..., q_{n-1}$.
- Compute $EH(Q)$ using any algorithm to find convex hulls.

To prove: that $EH(Q) = EH(S)$, it suffices to show that if p is a corner point of S then $p \,\epsilon\, Q$. So let p be a corner paint of S. Let L_i and L_j, $j > i$, be the two lines whose intersection is p. To prove that $p \,\epsilon\, Q$, 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 can 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. Without loss of generality, lets assume v is left of w. Since $p \neq q_{n-1}$, any one or both of the following is true:

1. $j \neq n - 1$
2. $i \neq 0$

If 1. holds then we can obtain a contradiction. Lets say L_{n-1} crosses L_i somewhere, say 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. Else, s is to the left of p, then L_{n-1} intersects L_j at point, say t which is to the left of p. This implies thatp is on the straight-line segment joining t to w. Contradiction!

For 2., we can give a similar argument. Here instead of L_{n-1}, we use L_0.

This completes the proof that every corner point belongs to Q, from which correctness of the algorithm follows. Hence $EH(Q) = EH(S)$.

Now, to calculate $CH(S)$.

Lets first mark every edge in $EH(S)$ which is along one of the lines in $A(L)$ as $black$. This takes $O(n)$ time. Then for every line L_i, we do the following:

- Find the intersection of L_i with the convex polygon $EH(S)$. Let s_i and t_i be those point of intersections.
- Initiate an empty set R.
- Add s_i to R, if s_i (or t_i) is a corner, or an edge $EH(S)$ to which s_i (or t_i) belongs is $black$, then add s_i (or t_i) to R.

The set R now contains all the points of S that appear on $CH(S)$, whether they are corners or not, and therefore CH(R) = $CH(S)$. Computing $CH(H)$ will take $O(n\,log(n))$ time by any well known convex hull algorithm.

Time Complexity

To calculate $EH(S)$:

- Sorting n lines by their slope takes $O(n\,log(n))$ time.
- Generating Q takes $O(n)$ time.
- Computing $EH(Q)$ takes $O(n\,log(n))$ time.

To calculate $EH(S)$:

- Finding the intersection of L_i with the convex polygon $EH(S)$ takes $O(logn)$ time as taught in class.
- Initialising an empty set takes $O(1)$ time.
- The third step takes $O(n\,log(n))$ time.

In total the whole algorithm constructs convex hull of all intersection points in $O(n\,log(n))$ time.

Question 3

Prove that the test for collinearity of three points in a set of n points on the plane, is as hard as checking whether there exist three distinct integers a, b, c among a set of n integers, such that $(a+b+c)=0$.

Answer: We can prove that the problem of finding if 3 points coincide in a line, is 3SUM-hard. We can do the construction as follows (please refer the figure), we map from x to (x, x^3) for each x in the 3SUM instance, where we assume the three chosen numbers have to be distinct.

We can show that three points on the curve will lie on a line if and only if there are three integers summing to 0 in the original set. The three numbers a, b, c follows the following

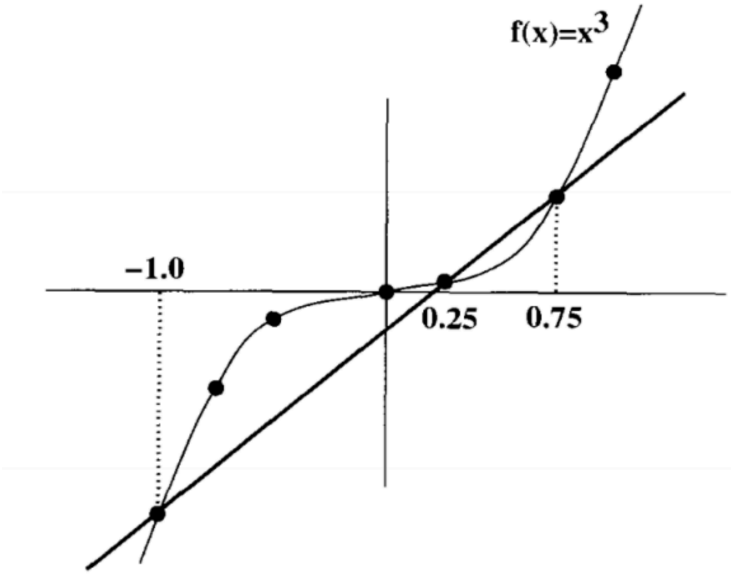

Figure 1: Cubic curve for 3SUM equivalence proof

For three points (a, b, c) on the curve to be collinear the slope of line connecting a and b has to be equal to slope of line connecting b and c.

$$\frac{b^3 - a^3}{b - a} = \frac{c^3 - a^3}{c - a} \iff b^2 + ba + a^2 = c^2 + ca + a^2 \iff (b - c)(b + c + a) = 0$$

Since a, b, c are distinct, hence the equation results in $a + b + c = 0$.

Hence, proved.

Question 4

Consider the visibility graph $G(V,E)$ of a set of n disjoint line-segments on the plane. Assume for simplicity that no three end-points of line-segments are collinear, and no line segment is horizontal or vertical. Each end-point defines a vertex in V. An edge (v_1, v_2) appears in E when the two end-points corresponding to v_1 and v_2 either belong to the same segment, or are visible on the plane. Show that G admits a cycle through all vertices.

Answer: Given a visibility graph $G(V,E)$ of disjoint line segments in the plane, denote by $V(S)$ the set of segment endpoints from $G(V,E)$. 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.

Definition: 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).

We say that a finite set $\mathscr{D}$ of pairwise non-overlapping simple polygons is a dissection of P, if $P = \bigcup_{D \in \mathscr{D}} D$. (Two polygons overlap, if there is a common point in the relative interior of both.) The following lemma is crucial in our argument, as it establishes this theorem by a simple induction.

Lemma: 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 V is (S) and a dissection $\mathscr{D}$ of P satisfying the following properties.

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

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 $\mathscr{D}$ of polygons described in 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 $D \in \mathscr{D}$. By property ($L5$), D has a common side $ab \neq yz$ with P. By (L3) and (L4), $C(D) := \text{conv}(S \cap \text{int}(D)) \subset \text{int}(D)$. Moreover, $C(D)$ has a side cd such that both ac and

bd are visibility edges. If $c_1d_1, c_2d_2, \ldots, c_md_m, m \geq 1$, are the segments in int(D) and they are all collinear in this order, then replace the side ab of P by the path $ac_1d_1c_2d_2 \ldots c_md_mb$. Otherwise there is, by induction, a Hamiltonian polygon $H(D)$ for $S \cap \text{int}(D)$ such that cd is a side of $H(D)$. Replace the side ab of P by the path $(a,c) \oplus (\partial H(D) \backslash cd) \oplus (d,b)$. Doing so for each $D \in \mathscr{D}$ that contains segments from S results in a Hamiltonian polygon. Hence, proved.

Note: For two polygonal arcs $A = (a_1, \ldots, a_k)$ and $B = (b_1, \ldots, b_\ell)$ with $a_k = b_1$, we denote by $A \oplus B$ the concatenation $(a_1, \ldots, a_k, b_2, \ldots, b_\ell)$ of A and B.

For proof of lemma refer to: https://people.inf.ethz.ch/hoffmann/pub/pre/ht-sevgh-03.pdf