Nipissing University

Department of Computer Science & Mathematics

MATH 2216 / COSC 2216 - Computational Geometry I

Instructor: A. Karassev

MIDTERM EXAMINATION - SOLUTIONS

1. Let P be a simple polygon (not necessarily convex) and n be the number of vertices of P. Describe in details $O(n)$ time algorithm that for a given point in the plane determines whether the point is inside, outside, or on the boundary of P.

Solution

Let p be a given point. It is easy to check whether p is on the boundary of P in $O(n)$ time by checking whether p belongs to each of the sides of P, so assume that p is not on the boundary of P. Let q be a point outside of P. To find q we can, for instance, look at the vertex (x, y) of P with minimal x-coordinate (i.e. the leftmost vertex), and let $q = (x - 1, y)$. Now we count the number of intersections of the segment pq with the boundary of P. This can be done in $O(n)$ time, again by checking for intersection of pq with each of the sides of P. Assuming that pq does not contain vertices of P, it is easy to see that if the number of intersections is odd then p lies inside, and if the number of intersections is even, then p lies outside of P. We have to be careful if pq passes through a vertex of P. In this case, we do not count the vertex as an intersection, if both sides of P, incident to this vertex, lie in the same half-plane with respect to the line pq, and count it otherwise. Alternatively, we can pre-compute the slopes of all lines, joining p with the vertices of P (it takes $O(n)$ time) and select q so that pq has different slope and thus does not contain any vertex of P.

2. Suppose that a doubly-connected edge list of a connected subdivision is given. Give pseudocode for an algorithm that lists all faces with vertices that appear on the outer boundary.

<u>**Solution**</u>

For each face record f **Do**
  Return f
  If $OuterComponent(f) \neq \text{nil}$
    Return $Origin(OuterComponent(f))$
    $e := Next(OuterComponent(f))$
    While $e \neq OuterComponent(f)$ **Do**
      Return $Orining(e)$
      $e := Next(e)$
    End Do
  End if
End For

3. Is the following statement true: a polygon is convex if and only if it is monotone with respect to any line? Give a detailed proof if the answer is "Yes", and a counterexample if the answer is "No".

<u>**Solution**</u> The statement is true. Here is the proof:

($\Rightarrow$) Suppose that a polygon P is convex. We will show that intersection of P with any line is connected. Let l be a line. Let also points a and b belong to $P \cap l$. Then, since a and b are in P and P is convex, the segment $ab \subset P$. Clearly, $ab \subset l$, so we have $ab \subset P \cap l$.

($\Leftarrow$) Suppose that a polygon P is monotone with respect to any line in the plane. We will show that P is convex. Let a and b be two distinct points from P. Consider a line l that is perpendicular to the line ab. Since P is monotone with respect to any line, it is monotone with respect to l. Therefore the intersection of P with the line ab is connected. Since this intersection contains points a and b, it must contain the segment ab. Therefore $ab \subset P$, as required.

4. Let S be a finite set of points in the plane. Recall that we define the diameter of S as $\max\{d(p,q) \mid p \in S,\ q \in S\}$. Consider the following algorithm to find the diameter of the set of vertices of P:

- Start with any vertex p of the polygon P.
- Traverse the vertices of P in clockwise direction from p to find the vertex q, which is furthest away from p.
- Traverse the vertices of P in clockwise direction from q to find a point r which is furthest away from q. If the distance between q and r is less than or equal to the distance between p and q, then p and q achieve the diameter. Otherwise, repeat the same procedure for points q and r, and so on.

Construct a convex polygon for which the algorithm **incorrectly** calculates the diameter.

Solution An example of a quadrilateral of certain type will do, but the following picture represents slightly more interesting example. Here points p and s are symmetric with respect to the center of the circle, and so are q and r. The point c is in the middle of the upper semicircle, so $|ac| = |bc| = 1/\sqrt{2}$. If points p and r are sufficiently close to c, all the distances $|ap|$, $|bp|$, $|ra|$, $|rb|$ are very close to $1/\sqrt{2}$. On the other hand, in this case $|pq|$, $|rq|$, $|ps|$, and $|rs|$ are very close to 1 (but less than 1!). Finally, we may assume that $|ps| < |pq| < |qr|$. If all these conditions are satisfied, and we start at p, the algorithm will return $|rq| < 1$ as the diameter, while the diameter is obviously equal to 1.