CSE355/AMS345 Fall 2014 Computational Geometry

Final

- This is a **OPEN book, OPEN notes** exam. You may bring a calculator or laptop. But no Internet is allowed.
- For all problems, unless specified otherwise, assume that there is no degeneracy (i.e., no three points co-linear, no four points co-circular, no three lines having a common intersection).
- There are 5 problems and total of 28 points.
- Exam starts at 5:30am and ends at 8pm sharp on Dec 10th, 2014.
- Be brief. But if you wish, you may attach additional pages if you need.

By signing below I declare that I follow the rule of academic integrity and finish the exam on my own, without the help of others.

Name ____________________________

ID ____________________________

Signature ____________________________

1	2	3	4	5

1. **Delaunay Triangulation.** (7pts)

   (a) The degree of a point in a triangulation is the number of edges incident to it. Give a point set of n points such that in the Delaunay triangulation one point has degree $n-1$. (2pt)

   (b) What about the *average* degree of a Delaunay triangulation? Give the best upper bound you can find. (1pt)

(c) Prove that any two triangulations can be turned into one another by using edge flips. (2pts)

(d) A Gabriel graph of a point set P in the plane places an edge between two points p, q if the disk with pq as the diameter has no other points inside. Prove that a Gabriel graph is a subgraph of the Delaunay triangulation on P. (2pts)

2. *kd*-**tree** The following figure shows a recursive, alternating partitioning of the bounding box of n points, during a *kd*-tree construction (6pts)

   (a) Construct a tree representing the shown partitioning. For example, the top two levels of the tree is shown. For a vertical cut, the left/right box is placed as the left/right child. For a horizontal cut, the top/bottom box is placed as the left/right child of the tree. Finish building this *kd*-tree. For each leaf node, mark which point is represents. (3pt)

   (b) Show on your *kd*-tree how to answer the given range query, with the range shown in dark rectangle in Figure 1. In particular, highlight the vertices of the kd-tree that is visited by the query algorithm. (2pt)

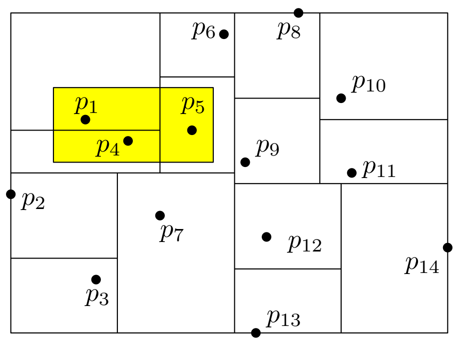

Figure 1: A *kd*-tree.

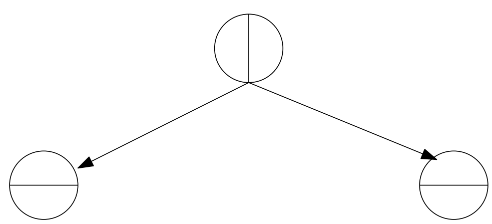

Figure 2:

(c) We know that on a kd-tree of n points, a rectangular range query takes time $O(\sqrt{n}+k)$, where k is the number of points inside the range. Now suppose we want to answer LookUp query, i.e., use the kd-tree to decide if there is a point at coordinate (a, b). We can formulate this query as a (degenerated) range $[a, a] \times [b, b]$ and perform the range query. What is the running time for this query? (1pt)

3. **Range Tree.** For the same point set as in Figure 1, you are asked to build a range tree. The points, sorted in x-coordinates, are in the order of

$$p_2, p_1, p_3, p_4, p_7, p_5, p_6, p_9, p_{13}, p_{12}, p_8, p_{10}, p_{11}, p_{14}.$$

Sorted in y coordinates the points are in the order of

$$p_{13}, p_3, p_{14}, p_{12}, p_7, p_2, p_{11}, p_4, p_5, p_1, p_{10}, p_6, p_8.$$

(6pts)

(a) Build a range tree. Show the entire tree for the main level (on x-coordinates). And for each internal node of this tree, only show the subset of nodes used for the secondary tree on y-coordinates. Figure 3 shows the top two levels of a range tree. (3pts)

(b) Show on your range-tree how to answer the given range query, with the range shown in dark rectangle in Figure 1. In particular, highlight the vertices of *main level* range tree for which you will query the secondary level tree. (2pt)

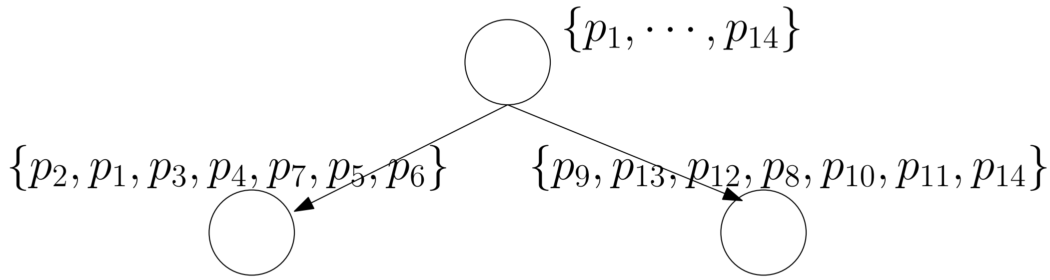

Figure 3: Range tree.

(c) For the same question as in 2(d), what is the running time for a 2D range tree? You may assume that no two points have the same x coordinates or y coordinates.(1pt)

4. **Interval Tree** (4pts) We would like to solve the following problem: Given a set S of n disjoint horizontal line segments in the plane, determine those segments that intersect a vertical ray running from a point (q_x, q_y) vertically upwards to infinity. Describe a data structure for this problem that uses $O(n \log n)$ storage and has a query time $O(\log n + k)$, where k is the number of reported answers.

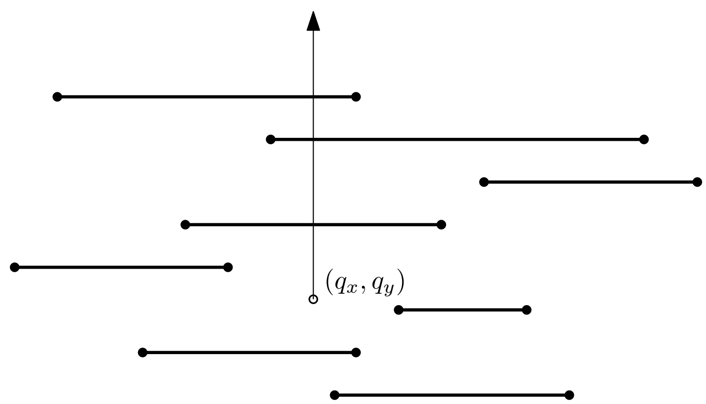

Figure 4: Example.

5. **Point Location.** (5pts)

(a) Build the traperzoid subdivision of the input planar subdivisions in Figure 5. How many traperzoid faces are there? (2pts)

(b) Build the point location search structure for this traperzoid subdivision by inserting the three segments in the order of s_1, s_2, s_3. (3pts)

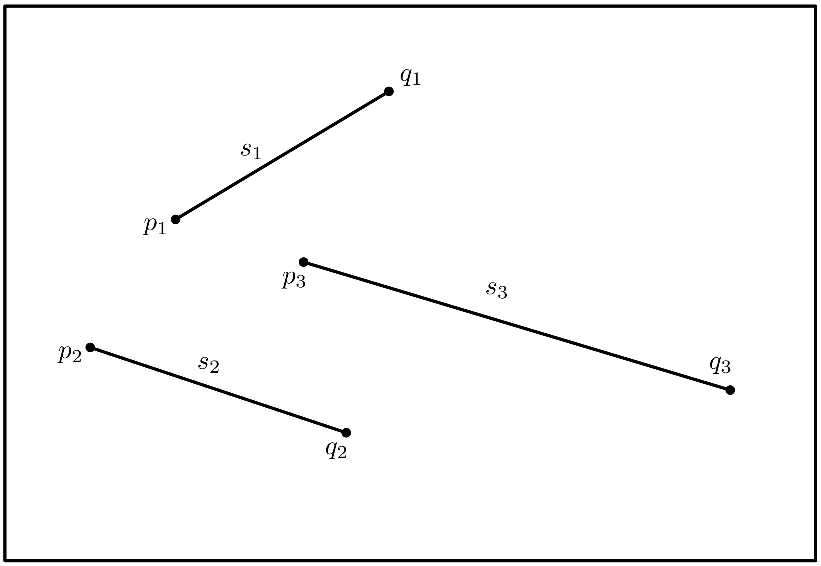

Figure 5: Example.