CS6050/7050 Computational Geometry

Spring Semester, 2018

Assignment 5 (Geometric Data Structures)

Due Day: 5:00 p.m., Monday, April 23, 2018

Note: The due day is not our class time. So if you want to turn in a hard copy, then you may either leave it in my mailbox in the CS main office or bring it to my office (slide it under the door if I am not in office).

1. Let S_1 be a set of n disjoint **horizontal** line segments and S_2 be a set of n disjoint **vertical** line segments.

   (a) Design a plane sweeping algorithm to compute the number of intersections of the segments in $S_1 \cup S_2$ in $O(n \log n)$ time. Since no two horizontal segments intersect and no two vertical segments intersect, an intersection only happens between a horizontal segment and a vertical segment.

   Note that the total number of intersections could be as large as $\Theta(n^2)$, but your algorithm should compute this number in $O(n \log n)$ time. This implies that you cannot report these intersections one by one. **(15 points)**

   (b) Design a plane sweeping algorithm to **report** all intersections of the segments in $S_1 \cup S_2$ in $O(n \log n + k)$ time, where k is the total number of intersections of all segments. So this is an output-sensitive algorithm because the running time is not only a function of the input size but also a function of the output size. **(15 points)**

2. **(30 points)** Given a set P of n points in the plane, design an efficient data structure to answer the following *segment dragging queries*: for each query, we are given a horizontal line segment s, and the goal is to report the first point of P that will be hit by s if we drag s vertically downwards (e.g., see Fig. 1). Each query segment s is specified by the coordinates of its two endpoints $(x_1(s), y(s))$ and $(x_2(s), y(s))$, with $x_1(s) \leq x_2(s)$.

   Please describe your data structure and the preprocessing time and space. Please also describe how your query algorithm works and the query time. You will receive full points if your data structure can achieve the following performance: The preprocessing time (i.e., the data structure construction time) is $O(n \log n)$, the space of the data structure is $O(n \log n)$, and the query time is $O(\log^2 n)$. For simplicity, you may make a general position assumption that no two points of P have the same x- or y-coordinate. (Hint: use range trees.)

3. **(10 points)** Let S be a set of n points in the plane. Let T be a k-d tree built on S, as described in class. Consider the following queries: Given a point p, determine whether p is in S (so the answer is either "Yes" or "No"). Using the k-d tree T, give an algorithm that can answer the queries in $O(\log n)$ time each. Please explain how your algorithm works and argue why the time is $O(\log n)$.

Total Points: 70

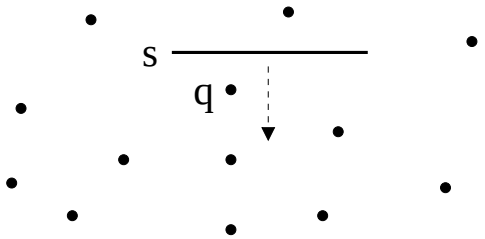

Figure 1: The point q is the first point that will be hit by dragging the segment s vertically downwards.