Homework 3 (due Nov 3 Friday 10am)

Instructions: see previous homework.

1. [25 pts] We are given a set $S$ of $n$ points in 2D, where each point is assigned a color. Present an efficient data structure so that a given query (axis-aligned) rectangle $q$, we can quickly decide whether all points inside $q$ have the same color. (You may assume that $q$ contains at least one point.)

   (Hint: modify the 2D range tree.)

2. [20 pts] Consider the following problem: store a set $S$ of $n$ (axis-aligned) rectangles in 2D so that for a given query horizontal line segment $q_1 q_2$, we can quickly count the number of rectangles $r \in S$ that $q_1 q_2$ completely cuts across (i.e., $q_1 q_2$ intersects both the left and right side of $r$). Design an efficient data structure for this problem.

   (Hint: directly reduce to orthogonal range searching. How many dimensions?)

3. [30 pts] In this question, you will explore a different method for solving the 2D planar point location problem.

   (a) [15 pts] Let $s_1, \ldots, s_n$ be a set of $n$ disjoint line segments, such that the left endpoints all lie on a common vertical line $\ell$. Give an $O(n)$-space data structure for this special case that can find the segment immediately above and the segment immediately below a query point in $O(\log n)$ time.

   Hint: sort $s_1, \ldots, s_n$ are sorted by the $y$-coordinates at $\ell$. For $i = 1, \ldots, n/2$, let $s'_i$ be the segment from $\{s_{2i-1}, s_{2i}\}$ whose right endpoint’s $x$-coordinate is larger. Recursively build a data structure for $s'_1, \ldots, s'_{n/2}$.
(b) [15 pts] Using (a) and divide-and-conquer, present a data structure for 2D planar point location with $O(n)$ space and $O(\log^2 n)$ time.

4. [25 pts] We are given a set $S$ of $n$ horizontal line segments in 2D, with integer coordinates in $[U]$. We want a data structure to answer the following type of queries: given a vertical line segment $q$ with integer coordinates in $[U]$, report all segments in $S$ that intersects $q$. Describe a solution with $O(n \log \log U)$ space and $O(\log \log U \log \log n + k \log \log n)$ query time (or better), where $k$ denotes the number of reported segments.

(Hint: persistence, and vEB trees.)