6.897 Advanced Data Structures (Spring'05)

Prof. Erik Demaine TA: Mihai Pătraşcu

$\boxed{\text{Problem 8}}$ *Due: Friday, Apr. 8*

In this problem and the next one, we look more closely at orthogonal range queries in *two dimensions*. The simplest query, and the one we will consider, is the *existential query*: given a set S of points in the plane, and a rectangle $[a, b] \times [c, d]$, is $S \cap ([a, b] \times [c, d]) = \emptyset$? Solutions to this problem usually generalize to reporting queries (find all points in the given rectangle).

Coordinates can come from any ordered set. However, we will assume they are integers in $\{1, \dots, n\}$; in this case, the problem is said to be in *rank space*. Given a different coordinate space, one can run four predecessor queries (for a, b, c, d), and convert the problem to rank space.

Range queries come in three degrees of generality. The general case allows the rectangle to be arbitrary. *Three-sided queries* fix one side of the rectangle; by symmetry, we can fix $a = 0$. *Dominance queries* fix one corner of the rectangle, usually by setting $a = b = 0$.

For the rest of the problem, assume that we are talking about static (S is not changing), existential range queries in rank space.

Argue that the RMQ (range minimum query) problem from class solves three-sided queries.

Since we can solve RMQ with linear space and constant-time queries, we have an optimal solution for three-sided queries. Such an optimal solution is not known for general queries, though one can get pretty close.

Prove that given a solution for three-sided queries in rank space with space $n \cdot \sigma$ and query time τ, one can obtain a solution for general queries with space $O(n \lg n \cdot \sigma)$ and query time $O(\tau + \lg \lg n)$. Thus, we can obtain $O(\lg \lg n)$ query time and $O(n \lg n)$ space for general queries.

Hint: consider a perfect binary tree over one axis; each node represents a vertical strip of space. Consider building two three-sided data structures for each such strip. Note that you need a predecessor query to convert to the rank space of each three-sided structure.