Weekplan: Range Reporting

Philip Bille

References and Reading

[1] Scribe notes from MIT.

[2] Computational Geometry: Algorithms and Applications, M. de Berg, O. Cheong, M. van Kreveld and M. Overmars,

[3] Fractional cascading: I. A data structuring technique, B. Chazelle and L. Guibas, Algoritmica, 1986

[4] Analysis of range searches in quad trees, J. L. Bentley and D. F. Stanat, Inf. Process. Lett., 1975

We recommend reading [1] in detail. [3] and [4] provide background on range trees and kD trees.

Exercises

1 [w] **2D Range Tree Example** Draw a 2D range tree for the set of points

$$P = \{(1,3),(3,8),(4,1),(7,5),(6,6),(9,6),(15,4),(20,17)\}.$$

Draw all stored arrays. Run a report$(2,2,10,10)$ query by hand and write all 1D queries on arrays.

2 Preprocessing for 2D Range Trees Give a fast algorithm that constructs a 2D range tree from a set $P \subseteq \mathcal{R}^2$ of n points.

3 [w] **2D Range Tree with Bridges Example** Convert the above example for 2D range tree to use fractional cascading. Simulate a report$(2,2,10,10)$ query and show how to traverse bridges.

4 Tight Bound for 2D Range Queries and Predecessor in Nested Sets Solve the following exercises.

4.1 A friend suggests that the $O(\log^2 n)$ term in the query time for 2D range tree is not tight. Specifically, the $O(\log^2 n)$ term is from $O(\log n)$ binary search the arrays stored at the node of the tree. However, as the lengths of the arrays decrease geometrically as we proceed down the tree and are thus much faster near the leaves. Clarify the analysis. Is your friend correct?

4.2 Another friend suggest that the $O(k \log n)$ bound for predecessor queries in solution 1 for predecessor in nested sets is not tight. Specifically, the $O(k \log n)$ bound is from k binary searches. However, since the total length of the arrays is n they cannot each be of length n. Clarify the analysis. Give concrete instances that show that your friend not correct.

5 Fractional Cascading Let $S_1, \ldots, S_k$ be a family of sets from a universe U (not necessarily nested). Solve the following exercises.

5.1 Suppose we store the sets as sorted arrays, add bridges, and implement the query exactly as we did with nested sets. Give an example that shows that this does not solve the problem.

5.2 [$*$] Can you modify the data structure to make it work? *Hint:* start by building and storing *augmented sets* $A_1, \ldots, A_k$ such that $A_k = S_k$ and A_i, $i < k$ is S_i and *every other element* of A_{i+1}. Add suitable pointers. By construction every set propagates a constant fraction of it's elements to higher sets in a cascading manner. This is where the name *fractional cascading comes from*.

6 Interval Trees Let $I = [l_1, r_1], \ldots, [l_n, r_n]$ be a set n of intervals. Give an efficient data structure that supports the following operation.

- intersect(x): return the set of intervals that contain the point x.

Hint: Start with a complete binary tree over the endpoints.

7 Skyline Range Reporting Let P be a set of n points with integer coordinate from a universe of size u. Give an efficient data structure that supports the following operation.

- report$_3(x_1, x_2, y_1)$: return the set of points in P whose x-coordinate is in the range $[x_1, x_2]$ and whose y-coordinate is in the range $[-\infty, y_1]$. *Hint:* use range minimum queries.

8 *k*D Tree Analysis Let T be a kD tree for a set of n points P. Consider a query for a range R. We want to bound the number of regions in T intersected by R to get a bound on the query time for R. The number of regions intersected by any rectangle is at most 4 times the number of regions intersected by a vertical or horizontal line (why?). We bound the number of regions intersected by a vertical in the following exercises and use that to prove the bound on the query time. Solve the following exercises.

8.1 Let $Q(n)$ denote the number of regions intersected by a vertical line in a kD tree for n points. Assume that the first split in kD tree is on the x-axis. Show that $Q(n)$ satisfies the following recurrence.

$$Q(n) = \begin{cases} 2Q(n/4) + O(1) & n > 1 \\ O(1) & n \leq 1 \end{cases}$$

8.2 Show that $Q(n) = O(\sqrt{n})$. *Hint:* draw recursion tree.

8.3 Conclude that the query time for a kD tree is $O(\sqrt{n} + \text{occ})$.

8.4 Show that for some points set P of size n and some range R, the regions of the kD tree intersects with R in $\Omega(\sqrt{n})$ regions. Conclude that the upper bound analysis is tight up to constant factors.

9 [$*$] **Fast 1D Range Reporting** Give a data structure for a set of integers $S \subseteq U = \{0, \ldots, u-1\}$ of n values that supports the following operation:

- report(x, y): return all values in S between x and y, that is, the set of values $\{z \mid z \in S, x \leq z \leq y\}$.

The data structure should use $O(n \log u)$ space and report queries should take $O(1 + \text{occ})$ time. *Hint:* x-fast tries and lowest common prefix in complete binary trees.