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** Construct 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 1D range trees used in the construction. Simulate a report$(2,2,10,10)$ query and show all queries to 1D range trees.

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

3 Query Bounds for 2D Range Trees A friend suggest that the $O(\log^2 n)$ analysis of reporting queries in 2D range trees is not tight. Specifically, the query time for the 1D range tree is $O(\log m)$, where m is the number of points stored in the 1D range tree. The 1D range trees over y-coordinate store different size subsets of P and hence the total query time is in fact asymptotically faster than $O(\log^2 n)$. Clarify the analysis. Is your friend correct?

4 [w] **2D Range Tree with Fractional Cascading 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 follow predecessor pointers.

5 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 the query time for R. The number of regions intersected by any rectangle is at most 4 times the number of regions intersected by any vertical or horizontal line (why?). We bound the number of region intersected by a vertical in the following exercises and use that to prove the bound on the query time. Solve the following exercises.

5.1 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.

$$Q(n) = \begin{cases} 2Q(n/4) + O(1) & n > 1 \\ O(1) & n = 0 \end{cases}$$

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

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

5.4 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.

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 $P \subseteq \mathscr{R}^2$ be a set of n points. 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 $[y_1, -\infty]$. *Hint:* range maximum queries.

8 Fractional Cascading for General Arrays Let A_1 and A_2 be two sorted arrays. Solve the following exercises.

8.1 A fellow student wants to compactly store A_1 and A_2 to support efficient range reporting queries on both arrays using a single binary search. He suggest using fractional cascading (as described in the lecture). Explain why this will not work.

8.2 [$*$] Can you modify the data structure to make it work? *Hint:* Add more elements to A_1. This is where the name *fractional cascading* comes from.

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 ancestors on complete binary trees.