6.851 Advanced Data Structures (Spring'14)

Prof. Erik Demaine TAs: Timothy Kaler, Aaron Sidford

$\boxed{\text{Problem 2}}$ *Sample solution*

This problem came from part 1 of the papers by Chazelle and Guibas on fractional cascading [1].

Let S denote the distinct a_i and b_i values of the intervals, i.e. $S = \{a_i \mid i \in [n]\} \cup \{b_i \mid i \in [n]\}$. Create a balanced binary search tree where the keys are S. For each node d in the binary search tree let $d_k \in S$ denote its key. Assign each interval $[a_i, b_i]$ to the least common ancestor of the nodes for a_i and b_i in the binary search tree. Note that interval $[a_i, b_i]$ is assigned to node d if and only if $a_i \le d_k \le b_i$.

For every node d in the binary search tree create two sorted lists, d_A and d_B. Let d_A contain the intervals $[a_i, b_i]$ assigned to d sorted by a_i and let d_B contain the intervals $[a_i, b_i]$ assigned to d sorted by b. Note that if some x satisfies $x \in [a_i, b_i]$ then $[a_i, b_i]$ is assigned to one of the nodes on the path from the root to either the successor or the predecessor of x in the binary search tree. Furthermore if for some node d we have $x \le d_k$ then if interval $[a_i, b_i]$ is in d_A and $a_i \le x$ then $x \in [a_i, b_i]$ (since $x \le d_k \le b_i$ for every $[a_i, b_i] \in A$). Similarly if $x \ge d_k$ then if $[a_i, b_i]$ in d_B and $b_i \ge x$ then $x \in [a_i, b_i]$ (since $x \ge d_k \ge a_i$ for every $[a_i, b_i] \in d_B$).

From the reasoning in the preceding paragraph we see that to compute $\texttt{stab}(x)$ it suffices to find the successor and predecessor of x in the binary search tree, follow the paths to the root for these two nodes, and for each node on the path find the position of x in d_A and d_B, and then walk these lists depending on whether $x \le d_k$ or $x \ge d_k$. Since the balanced binary search tree has at most $O(n)$ elements we see that its height is at most $O(\log n)$ and therefore we look up x in $O(\log n)$ lists connected in a graph of degree at most 3 (since a binary search tree is a graph of degree 3). Therefore, using fractional cascading we can find these elements in $O(\log n + \log n) = O(\log n)$ time (since each list has at most $O(\log n)$ elements).

Putting this all together, we see that it takes $O(n \log n)$ time to create the binary search tree and all the lists. Furthermore, since each interval is assigned to exactly one node, the total size of all the d_A and d_B lists is $O(n)$. Therefore, using fractional cascading to link all these lists takes $O(n)$ time and $O(n)$ space. Given a query, $\texttt{stab}(x)$ we can then find the paths for the successor and predecessor of x in $O(\log n)$ time since the binary search tree is balanced and using fractional cascading we can find x in each d_A and d_b lists in $O(\log n)$ time. Once we find x in each of these $O(\log n)$ lists we can output the results of $\texttt{stab}(x)$ in an additional $O(\log n + k)$ time by just walking each of the $O(\log n)$ lists depending on whether $x \le d_k$ or $x \ge d_k$. Therefore, this data structure is as desired.

Another Solution: Some students noted that for an interval $[a_i, b_i]$ it is the case that $x \in [a_i, b_i]$ if an only if the two dimensional point point $p = (a_i, -b_i)$ satisfies $p \in (-\infty, x) \times (-\infty, x)$ and therefore you could use the result we saw in Lecture 5. This is a valid answer as well and no points were deducted if you did this; though we may have preferred you figure out how to use fractional cascading directly ☺.

References

[1] B. Chazelle and L. J. Guibas. Fractional cascading: I. a data structuring technique. *Algorithmica*, 1:133–162, 1986.