Class Review

January 11, 2010

1 Introduction

The report is the documentation of the problems that we have solved in the advanced algorithm class dated 11/01/2010. As of yet, we have studied a lot about binary searches and so on. The problem that we consider now is a slight variation of the binary search itself. Let us first make a note of how the problem is different from the standard binary search tree problem that you have seen so long. Remember that in your earlier college classes, you have solved the following problem

Problem 1.1 *Given a set S (or an array) of n values all in $\mathbb{R}$, preprocess them into a data structure such that given two values a and b we can report all the values that are present in the array between a and b.*

The above problem means that you have to report all the values in the array that are present in the range $[a, b]$. A standard way to solve the problem is to construct a (height) balanced binary search tree and then answer the query. Now the first question is how much time do you need to answer the query ? Most of us think that its $O(\log n)$ time. But that is wrong. First of all the search time of binary search tree is never $\log n$, it is $\log n + 1$ (because you have to return the answer). But remember that in asymptotic notations we drop the constant terms and hence we say $O(\log n)$. But the trouble with that approach is that if we ask you to answer the number of points present in the range $[a, b]$, then what should be your answer ? Well, notice that the number of points in the range may vary from 0 to n. Hence its a variable and not a constant any more. Therefore the query time is $O(\log n + k)$ where k is the number of values that you return and $0 \leq k \leq n$.

2 How our problem differs from standard search problem

Consider that the points of our array are colored now. So each point in the array now has got a color along with a value. We now give you a range $[a, b]$. We can now ask you the following two questions:

1. Find the values in the range $[a, b]$.
2. Find the distinct color of points in the range $[a, b]$.

You know how to solve the first problem. All we are now left is to show you that you have the knowledge of all the data structures that you need to solve the second problem. The meaning of the distinct has been explained more elaborately in the subsequent sections.

3 Problem Formulation

Our main problem can be formulated as follows:

Problem 3.1 *We are given a set S of n colored points in $\mathbb{R}$. We need to preprocess S into a data structure such that given a query range $q = [a, b]$ we can efficiently report the distinct colors in $q \cap S$.*

But we will start our discussion by solving a much simpler problem. Then we will extend the simple solution to solve our main problem. Finally we will present the solution of [1]. So we start by solving the following problem

Problem 3.2 *We are given a set S of n colored points in $\mathbb{R}$. We need to preprocess S into a data structure such that given a semi-infinite query range $q = [a, \infty]$ we can efficiently report the distinct colors in $q \cap S$.*

The problem was first studied in [2]. Let us first try to understand why the problem is interesting. Notice that you are being asked to report the distinct colors in a query range and not the points. Suppose there are five red colored points in your query range. Then you are going to report red only for once and not for five times. Next we try to understand why this notion is important. In the worst case, it is quite possible that $\frac{n}{2}$ points are lying in your query range but all the points are of colored red. If you answer red for $\frac{n}{2}$ times, you are actually wasting your precious time and the computation resource. Assuming that the reader has understood why the problem is important, we next discuss a very simple solution for the problem.

Consider the following figure. Suppose your point set be as shown in the Figure 1 and your query $q = [3.5, \infty]$.

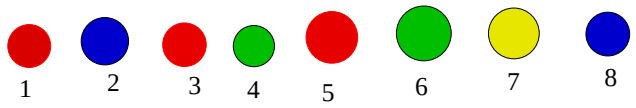

Figure 1: Your point set

Now notice the next figure carefully. Notice that if the red point having the value 5 is not in the semi-infinite range $[3.5, \infty]$, no other red point can be. Also notice that 5 is the maximum value for any red point.

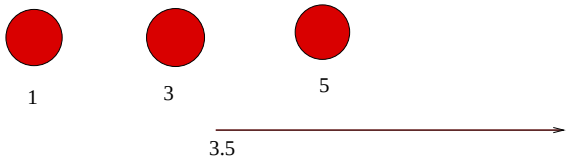

Figure 2: The point 5 has to be in $[3.5, \infty]$ for any red color point to be in the semi-infinite query range
.

We therefore got the hint that in case of any semi-infinite query range that is extending towards $+\infty$ (that is $[a, \infty]$, finding distinct colors in the query range is very simple. For a particular color to be in the query range, its maximum value should be inside the query range. We therefore do the following:

We group all the points of red colors. We sort the group. We then group all the points of blue color and sort the group and so on. Finally for each color we select the maximum value and construct a link list in descending order. See Figure 4.

Given a query interval $q = [a, \infty]$, we start walking along the link list and report the colors. We continue to walk until we find a value x in the link list such that $x < a$. We conclude the discussion with the following lemma

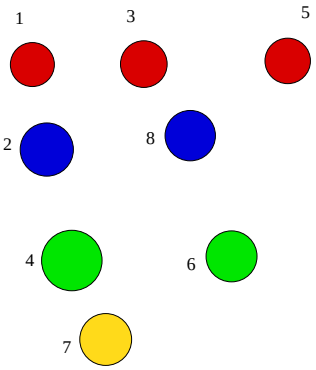

Figure 3: Group all the points in terms of their color. Then sort each group

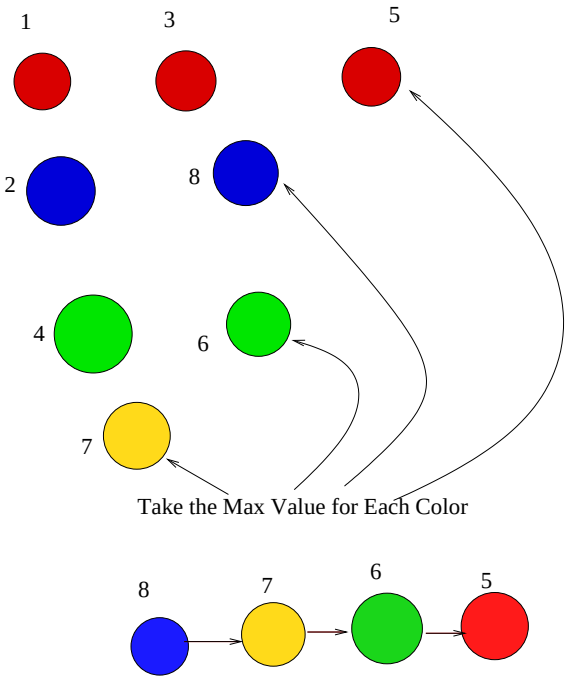

Figure 4:

Lemma 3.1 *Given a set of n colored points, we can preprocess them into a data structure of size $O(n)$ such that given a semi-infinite query $q = [a, \infty]$ we can efficiently report all the colors q in $O(k)$ time where k is the number of colors in q.*

Now let us see how can we extend this technique to solve the Problem 2.1. Build a binary search tree on the coordinates of the points. Call the tree to be T_x.

To each internal node $u \in T_x$, we do the following,

- If u is a left child of its parent, we associate the data structure of Lemma 2.1 to u. The data structure is built on the leaf nodes of the subtree rooted at u.

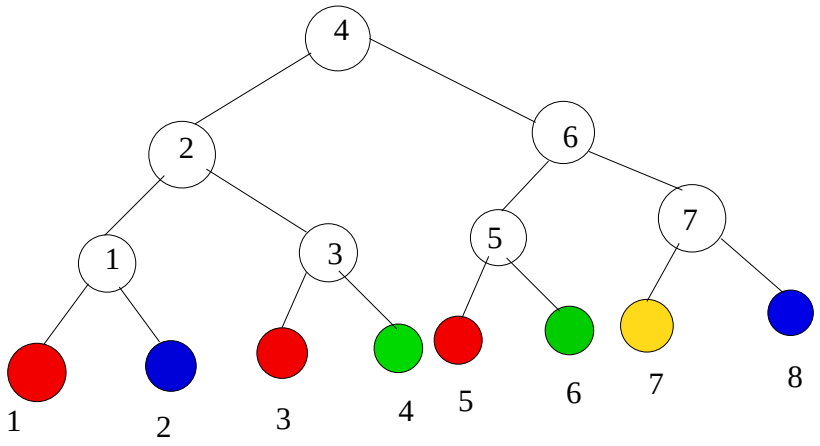

Figure 5:

- If u is a right child, build a variant of data structure of Lemma 2.1. The variant data structure should be able to support queries of the form $q' = [-\infty, a]$. (That is variant data structure will be used for deciding the distinct colors in the range $[-\infty, a]$).

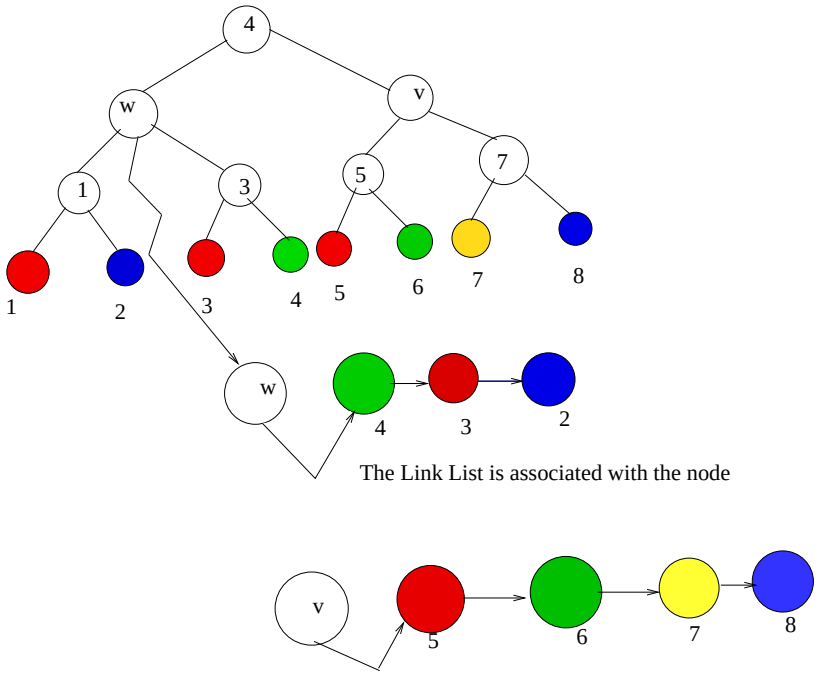

Figure 6:

3.1 Query Algorithm

Given a query $q = [a, b]$, find the split node $m \in T_x$. The split node m is a node such that the value m_x it stores is in between a and b that is $a \leq m_x \leq b$. See Figure 7.

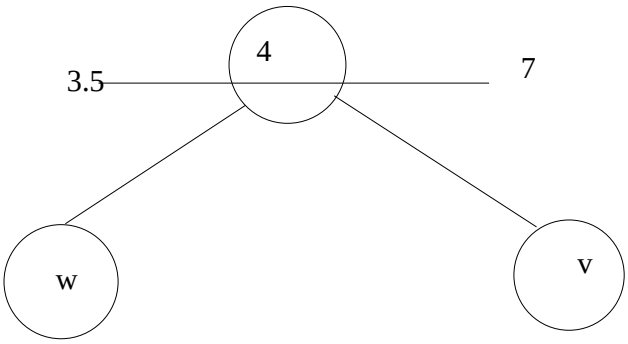

Figure 7:

Visit the left child and right child of the split node m. Let w and v be the left child an right child of the split node m. At the node w, find the distinct colors present in the query $[a, \infty]$. At node v find the distinct colors present in the query $[-\infty, b]$.

We therefore conclude the Problem 2.1 with the following lemma.

Lemma 3.2 *Given a set S of n colored points in $\mathbb{R}$, we can preprocess S into a data structure of size $O(n \log n)$ such that given a query range $q = [a, b]$ we can efficiently report the distinct colors in $q \cap S$ in time $O(\log n + k)$.*

You can understand easily that finding the split node takes $O(\log n)$ time. Then reporting distinct colors take $O(k)$ time and hence in total it takes $O(\log n + k)$ time.

4 Assignment Questions

Submission Deadline: 18/01/2010.

1. Why the storage space requirement is $O(n \log n)$?
   (Hint: Think in the lines of bottom up approach of the Merge-Sort technique)
2. Devise a way to find the split node for the range $[a, b]$.
   (Hint: Think if the concept of Least Common Ancestor can be used)
3. Write five to ten critical comments about the above specified solution strategy.

Any comments (feedback) on the documentation (god or bad) are welcome. Please mail your feedback at anandaswarup.das at mail.iiit.ac.in

References

[1] N. Madhusudhanan, P. Gupta, A. Mitra: *Efficient algorithms for range queries in protein sequence analysis* In Proceedings of the 17th Canadian Conference on Computational Geometry (CCCG'05), pages 146-149, 2005.

[2] P. K. Agarwal, S. Govindarajan, S. Muthukrishnan: *Range Searching in Categorical Data: Colored Range Searching on Categorical Data: Colored Range Searching on Grid* In Proceedings of European Symposium on Algorithms (ESA) pages 17-28,2002.