[LeetCode] 220. Contains Duplicate III

June 27, 2017

Given an array of integers, find out whether there are two distinct indices *i* and *j* in the array such that the **absolute** difference between **nums[i]** and **nums[j]** is at most *t* and the **absolute** difference between *i* and *j* is at most *k*.

Thought process:
Iterate through the array. Use a binary search tree (TreeSet in Java) to keep track of the most recent k numbers. For each number, get the ceiling (smallest number that's greater than num) and floor (greatest number that's less than num) of it, and check if either's difference with num is at most t.
If the tree's size exceeds k, remove the least recent number.
Note: there may be integer overflow when subtracting a negative number from a positive number.

Solution 1:

```
public class Solution {
    public boolean containsNearbyAlmostDuplicate(int[] nums, int k, int t) {
        TreeSet<Integer> set = new TreeSet<>();

        for (int i = 0; i < nums.length; i++) {
            Integer ceiling = set.ceiling(nums[i]);
            Integer floor = set.floor(nums[i]);
            if ((ceiling != null && (long) ceiling - (long) nums[i] <= t) || (flo
                return true;
            }
            set.add(nums[i]);
            if (set.size() > k) {
                set.remove(nums[i - k]);
            }
        }

        return false;
    }
}
```

Time complexity:
for loop takes O(n). Adding and removing numbers takes O(logk). The overall time complexity is O(nlogk).

There is also a O(n) solution:

1. Divide numbers into buckets of size t + 1. For example, numbers between 0 and t are in bucket 0, and numbers between t + 1 and 2t + 1 are in bucket 1.
2. Iterate through the array. Use a hash map to store bucket -> number mappings. Maintain a window of size k, i.e. there are at most k mappings in the map.
3. This gives me three cases as I iterate:
    1. Map contains the same bucket: return true, because I know the numbers satisfy both requirements.
    2. Neighboring buckets contain a number whose difference with the current number <= t: return true.
    3. If neither of 1 and 2 happens, put the mapping of current bucket -> number into the map. If current index is larger than k, evict the mapping of the bucket of array[index - k]. It is safe to remove this mapping because its index will never be within k difference with a future number.

Note: there may be three cases of integer overflow:

1. Bucket size: if t = Integer.MAX_VALUE.
2. Neighboring buckets: if t = 0, then size of bucket = 1. If there is Integer.MAX_VALUE in the array, its neighbor bucket will overflow.
3. Difference between numbers.

Solution 2:

```
public class Solution {
    public boolean containsNearbyAlmostDuplicate(int[] nums, int k, int t) {
        if (t < 0) {
            return false;
        }

        long size = t + 1;
        Map<Long, Long> map = new HashMap<>();

        for (int i = 0; i < nums.length; i++) {
            long bucket = getBucket(nums[i], size);
            if (map.containsKey(bucket)) {
                return true;
            }
            if (map.containsKey(bucket - 1) && (long) nums[i] - map.get(bucket -
                return true;
            }
            if (map.containsKey(bucket + 1) && map.get(bucket + 1) - (long) nums[
                return true;
            }
            map.put(bucket, (long) nums[i]);
            if (i >= k) {
                map.remove(getBucket(nums[i - k], size));
            }
        }

        return false;
    }

    private long getBucket(int number, long size) {
        return number < 0 ? number / size - 1 : number / size;
    }
}
```

Time complexity: O(n).

Binary Search Tree LeetCode

Location: San Jose, CA, USA

Popular posts from this blog

[LeetCode] 269. Alien Dictionary

June 29, 2017

There is a new alien language which uses the latin alphabet. However, the order among letters are unknown to you. You receive a list of non-empty words from the dictionary, where words are sorted lexicographically by the rules of this new language . Derive the order of letters in this language. …

READ MORE

[LeetCode] 261. Graph Valid Tree

July 29, 2017

Given n nodes labeled from 0 to n - 1 and a list of undirected edges (each edge is a pair of nodes), write a function to check whether these edges make up a valid tree. For example: Given n = 5 and edges = [[0, 1], [0, 2], [0, 3], [1, 4]] , return true . Given n = 5 and edges = [[0, 1], [1, 2], [2, 3] …

READ MORE

[LeetCode] 253. Meeting Rooms II

March 11, 2017

Given an array of meeting time intervals consisting of start and end times [[s1,e1],[s2,e2],...] (s i < e i), find the minimum number of conference rooms required. For example, Given [[0, 30],[5, 10],[15, 20]] , return 2 . Thought process: The idea is to first sort the array based on start time, so we can …

READ MORE