CSE 5311 Quiz 2 (Fall 2022)

89

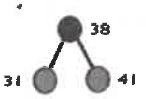
NETID: 1002070724

1:05-1:45pm, 11/15 (Tuesday)

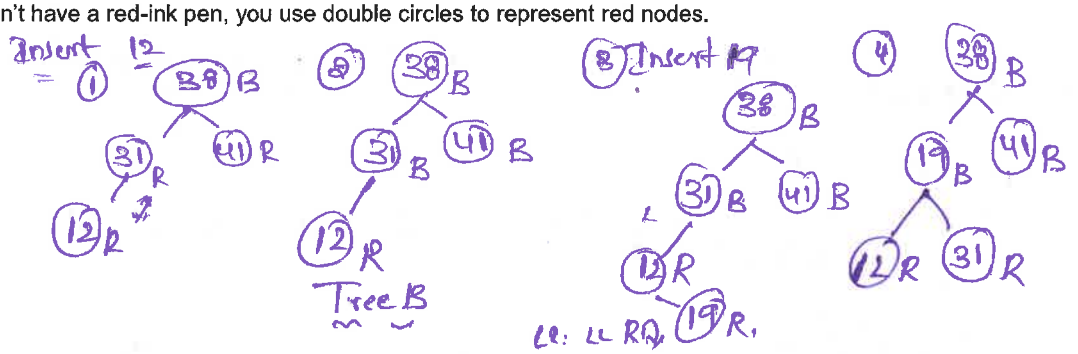
Name: Rohit kalyan Gandham. (30 points for writing your name)

1. No.
For algorithm to take right subtree, left[x] = NIL or low[i] > m[left[x]]
If left[x] = NIL, then there can be any overlapping interval in left[x] i.e., left subtree
If low[i] > m[left[x]]. Any interval in the left subtree cannot extend beyond m[left[x]] therefore there cannot be any interval in the left subtree that will overlap with i
2. Yes.
For algorithm to take left subtree, left[x] != NIL or low[i] <= m[left[x]]
There may be a condition in which the smallest low internal of the right[x] subtree <= low[i] <= m[left[x]]. So, there is the possibility of an overlapping interval in the left subtree.

1. [20 points] Show the red-black trees that result after successively inserting the keys12 and 19 into the following red-black tree (Tree A): one red-black tree (Tree B) after you insert 12 to Tree A, and another red-black tree (Tree C) after you insert 19 to Tree B. You may choose to draw additional intermediate trees before you obtain B and C, so that you can receive partial credits even if there are mistakes in the final results. If you don't have a red-ink pen, you use double circles to represent red nodes.



(Tree A)



2. [20 points] In the interval search algorithm we have studied, (1) if a search goes to the right subtree, is it likely that an overlapping interval exists in the left subtree? why? (2) if a search goes to the left subtree, is it likely that an overlapping interval exists in the right subtree? why?

-10

```
INTERVAL-SEARCH(i)
  x ← root
  while x ≠ NIL and (low[i] > high[int[x]]
                     or low[int[x]] > high[i])
    do ▷ i and int[x] don't overlap
      if left[x] ≠ NIL and low[i] ≤ m[left[x]]
        then x ← left[x]
        else x ← right[x]
  return x
```

Sol) as it is travel down the tree it will check the current (1) Node with the node (x) in the interval Node overlaps with the currently interval Node(x). when current Node overlaps with the supplied interval, instead of reaching to the. it will keep a track of the current Node to the left maxim in the left subtree ~~and~~ which having ~~[illegible]~~ smaller(or) lower endpoints and the left subtree should have a condition of the. lower Bound interval say i is less than the current match. with the current match if we go right subtree it will ~~[illegible]~~ Overlap at left subtree ~~and the [illegible]~~.

3. [15 points] The minimized search costs in a structure consisting of multiple sorted linked lists for speeding up searches are as follows (in Lecture 12: Skip Lists).

- 2 sorted lists ⇒ $2 \cdot \sqrt{n}$
- 3 sorted lists ⇒ $3 \cdot \sqrt[3]{n}$
- k sorted lists ⇒ $k \cdot \sqrt[k]{n}$
- $\lg n$ sorted lists ⇒ $\lg n \cdot \sqrt[\lg n]{n} = 2 \lg n$

lev 3 = 4/2 = 2
lev 2 = (n/2) = 8/2 = 4
lev 1 : (n/2) = 16/2 = 8
lev 0 16 Elem.

If we have 16 elements in the bottom (the longest) list, and there are 4 sorted lists, what are the sizes of the top three lists to obtain the minimal search cost?

Sol) Here No. of levels in skip list = $\log_2(n) = \log_2 16 = 4\log_2(2) =$ ④ = 4 sorted lists, To obtain the minimum search cost the skiplist should be perfect list as level increasing No. of Elements = (n/2), will cost the minimal search cost ($\log(n)$).

level = 0 ⇒ 16 Elements
level = 1 ⇒ 16/2 = 8 Elements
level = 2 = 8/2 = 4 Elements
level = 3 = 4/2 = 2 Elements.

level 3 = 4/2 = 2 Elements

-1

4. (15 points) Use descriptive language to describe how a key x is inserted in a skip list currently containing n keys.

Sol) Given skip list containing (n) keys as we know the perfect skip list is always have Elements (n/2) in levels.

Insertion in Skiplist's

1. * first Search the Boundaries of the keys in the Express line's

* if key "k" is greater than Elements in Highest Express line, proceed to next node.

3 * if key "k" is lesser than Elements in Highest Express line, Come down and check.

* Repeat Step 2 and 3 until you found the currect Boundaries of the "K".

* Now insert key "k" in right position, and check the (Coin) condition

* if coin condition true, copy, Element to next level. until the condition becomes false.

(2Q)

(2) When current Node overlap's with the Supplied interval (i), instead of Reaching the lower leaf we will keep a track of the interval and when the search goes to the left Subtree and keep a match to again to the left of tree of left Subtree which is having Smaller (or) lower endpoints and it will fail the condition lower Bound less than the Current match, in this case the overlapping may (or) not go to the overlapping interval Exists in right Subtree.