CS2233 – Data Structures

In addition to the properties of a binary search tree, red-black tree has the following properties:

i) Every node is either red or black
ii) Root is always colored black
iii) A red node can only have black children
iv) Every external node has the same black depth.

External node: NIL node appended to a leaf
Black depth: No. of black ancestors of a node.

i) We have the definition of a relaxed red-black tree i.e root can be colored red or black.

We have an event: the color of the root of a red black tree (relaxed) changes from red to black.

Claim: This event results in a valid red-black tree.

a) It was satisfied originally and is satisfied even after the event.
o property (i) satisfied.
Due to the event property (ii) is also satisfied

This property is satisfied by default for both of the original red-black tree & the relaxed version.

c) The relaxed red-black tree satisfied this property and the event doesn't cause any violations. ∴ property 3 & (iii) satisfied ✓

d) Case 1: The node is a root node
By property (iv) each external node has the same black depth.

Case 2: The node is a leaf node
If the leaf is colored black, the external node has a black height of (h+1) and h otherwise
h - # black nodes above the leaf node.

Case 2: The node is a leaf node
If the leaf is irrespective of the color of leaf node all paths from leaf node to the external node contain one black node (i.e the external node itself)

Case 3: The node is an internal node
All external nodes have the same black height (say h) by property (iv)

And the paths from the descendant of an internal node to the root have in common the nodes between root and the ~~node~~ internal node in consideration. Say there are n black nodes on this path.
∴ all descendants of an internal node are (h-n) black nodes away from it.

2) Proof by contradiction -

Suppose for a non-leaf node x, we have
a - closest descendant of x
b - furthest descendant of x

∴ The longest simple path - $(b_1, b_2 \ldots b_t)$
& the shortest simple path - $(a_1, a_2 \ldots a_s)$

By 1) d), we know that both contain the same number of black nodes ——①

By property (iii), we know that a red node cannot be preceded or followed by a red node.

This means that atmost $t - \left\lfloor \frac{t-1}{2} \right\rfloor$ of the nodes in the longest path are red, irrespective of x's color.

atleast $\left\lceil \frac{t+1}{2} \right\rceil$ are black.

∴ By ① $s \geq \left\lceil \frac{t+1}{2} \right\rceil$ ——②

Now, if we let $t > 2 \times s$, say $t = 2s+1$

In ② $s \geq \left\lceil \frac{(2s+1)+1}{2} \right\rceil$

$s \geq \left\lceil \frac{2s+2}{2} \right\rceil$

$s \geq s+1$ i.e a contradiction.

∴ Assuming $t > 2s$ led us to a contradiction.
⟶ the actual constraint is $t \leq 2s$
i.e the problem statement
H.P

□

Coding Assignment (Question 5) -

Proof of Correctness -

Base Case - The function starts by checking if the current node's data falls within the specified range ensuring that the function covers all values.

Recursion - The function then recursively calls for the left & right subtrees, if they exist. This ensures that all nodes in the ~~subt~~both subtrees are considered.

Handling Subtrees - If the nodes data is less than min, then only the right subtree is explored and if the nodes data is greater than max, only the left subtree is explored. This is due to the properties of a BST.

Termination - Recursion terminates when a leaf node is reached or when a subtree isn't explored because its data is outside the specified range. Therefore, all nodes are visited based on the defined conditions.