Algorithms and Data Structures: Homework #9

Due on April 14, 2020 at 23:00

Henri Sota

Problem 9.1

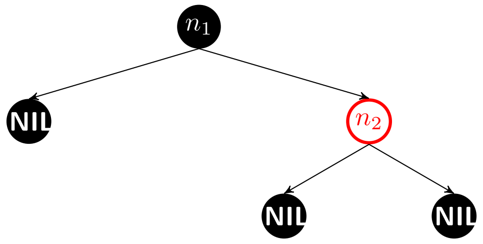
c) Consider a red-black tree formed by inserting n nodes with the algorithm described in the lecture slides. Prove that if $n > 1$, the tree contains at least one red node.

Proof. We use induction. The induction hypothesis, $\mathcal{P}(n)$, will be:

$$\textit{If } n > 1, \textit{ the tree contains at least one red node.} \tag{1}$$

Base Case: Our base case is $\mathcal{P}(2)$ when the tree has 2 elements ($n = 2$).

In this case, when inserting the second node onto the tree, the node will be a child of the root node (which is black due to property 2). In order to maintain property 5 of red black trees, the 2^{nd} inserted node will have to be red.

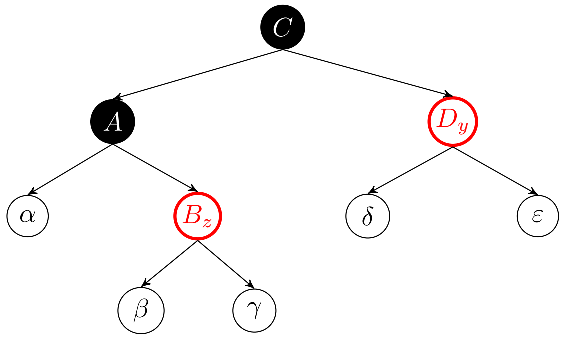


Induction Step: Assume that theorem 1 holds. When inserting a new node onto the tree, on insertion $n + 1$, there will be 2 cases:

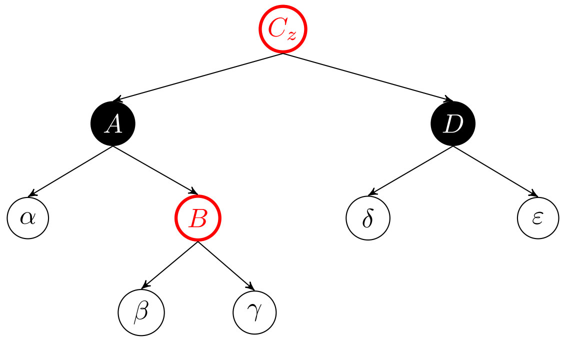
Case 1: The new node, colored red when inserted, will be the child of a black node. This case is the same as the base case and this represents no violation of the red black tree properties.

Case 2: The new node, colored red when inserted, will be the child of a red node. This case will be divided into 3 subcases that all 3 subcases of the RB-Insert-Fixup procedure.

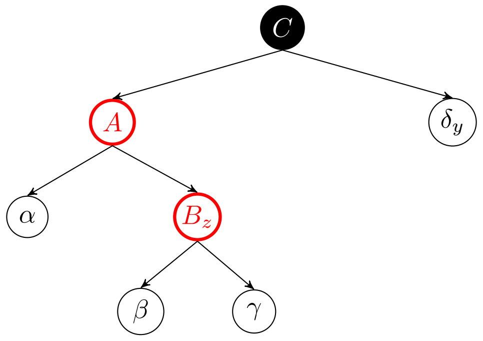
Case 2.1: In this case, the inserted node's uncle is also red as the inserted node's parent. The inserted node will stay red and both, the parent and the uncle, will be turned into black. The inserted node's grandparent will be turned into black and the violation will be propagated up the tree until it is fixed. Although this case will expand up until the root in the worst case, the inserted node will remain red and therefore the hypothesis is proved to be true.
Inserted new node:



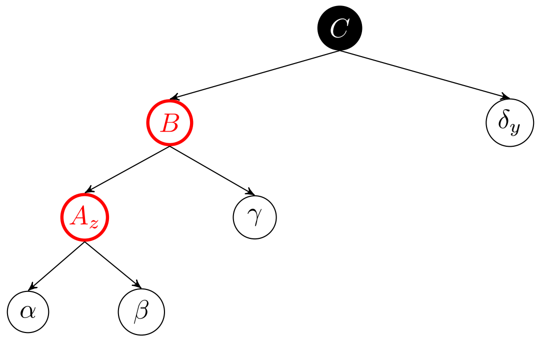
After Insert Fix Up procedure:



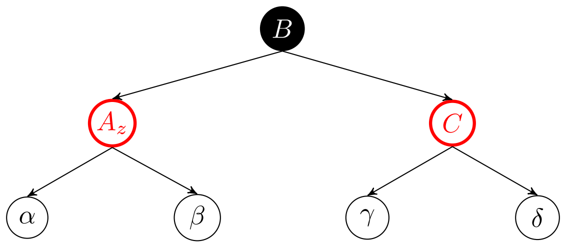
Case 2.2: In this case, the procedure will perform a rotation depending if the inserted node's parent is the left child or the right child of the inserted node's grandparent. The inserted node's and its parent's color will still remain red and the problem will be propagated to Case 3 of the Insert Fixup, or Case 2.3 in this proof. Therefore, the proof of this subcase depends on the proof of subcase 3.



Case 2.3: In this case, the procedure will perform a rotation on the inserted node's grandparent depending if the violation is on the left subtree or on the right subtree. In order to correct the violation of property 4, the parent of the inserted node now (the grandparent of the inserted node before the rotation in Case 2.3) will change color to red and drop to the right subtree and the inserted node will go into its place, while changing color to black. The violation of property 4 disappears and the number of red nodes is still atleast 1. Therefore the hypothesis is proved to be true.
Inserted new node:



After Insert Fix Up procedure:



* Each of the subtrees α, β, γ, δ and ε have a black root, and each have the same black-height.

■