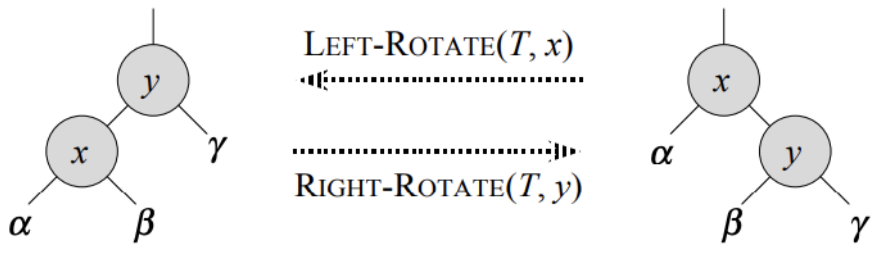
Justin Ciocoi

Nov. 27, 2023

CSCI 377 Textbook Notes

Chapter 13: Red-Black Trees

- **13.1: Properties of Red-Black Trees**
    - A red-black tree is a binary search tree which has one extra bit of storage per node which stores the color of the node - either red or black
    - Red-black trees ensure that no such path from root to a NIL leaf is more than twice the length of any other, something we refer to as having a *balanced tree*
    - A red-black tree must satisfy the following properties
        1. Every node is either red or black
        2. The root is black
        3. Every leaf (NIL) is black
        4. If a node is red, both its children are black
        5. For each node, all simple paths from the node to descendant leaves contain the same number of black nodes
- **13.2: Rotations**



    - Above is a visual illustration of both left and right rotations, and below is the pseudo-code for a left rotation

```
Left-Rotate(T, x)
    y = x.right
    x.right = y.left
    if y.left != T.NIL
        y.left.p = x
    y.p = x.p
    if x.p == T.NIL
        T.root = y
    else if x == x.p.left
        x.p.left = y
    else
        x.p.right = y
        y.left = x
        x.p = y
```

- **13.3: Insertion**
    - We will insert node z, which is assumed to have a key already, into the red-black tree T

      ```
      RB-Insert(T, z)
          y = T.NIL
          x = T.root
          while x != T.NIL
              y=x
              if z.key < x.key
                  x = x.left
              else
                  x = x.right
          z.p = y
          if y == T.NIL
              T.root = z
          else if z.key < y.key
              y.left = z
          else
              y.right = z
          z.left = T.NIL
          z.right = T.NIL
          z.color = RED
          RB-Insert-Fixup(T, z)
      ```

        - Here, the `RB-Insert-Fixup(T, z)` function is used to fix any violations of the red-black properties in the resulting tree after an insertion
    - `RB-Insert-Fixup()` will have the following properties
        - Insert node z as a red node

- Next, we re-color and rotate nodes in order to fix any violations of the rules of a red-black tree
- There are four possible cases here
    1. If z is the root
        - All we need to do is insert the z and color it black instead of red
    2. If z has a red uncle
        1. You must recolor the parent, grandparent, *and* uncle of node z
    3. If z has a black uncle and forms a triangle with the parent (i.e. if the parent is a right child and z is a left child or if the parent is a left child and z is a right child)
        - We rotate z's parent with z
            - if z is the right child of A, now A will be the left child of z
    4. If z has a black uncle and forms a line with the parent (i.e. if the parent is a right child and z is a right child or if the parent is a left child and z is a left child)
        - First, rotate z's grandparent
        - Then, recolor z's *original* parent and grandparent after rotation