CSCE 221
Problem Set 12

Jacob Purcell, Texas A&M, Student

I.

Deletion in a redblack tree goes as,

Algorithm 1 Void Delete

Require: *Red Black Tree*
  Pointer C = Find X
  if *both of C children are nullptr* **then**
    DeleteC
  end if
  Pointer B = C left child
  while *B right child is not nullptr* **do**
    Assign B to be its right child
  end while
  set value at C to be B
  C takes over right subtree of B if needed
  delete B
  Flip colors of C and children
  Rotate unbalanced nodes to maintain the 4th rule

II.

To get the maximum height, we qill use

$$N_h = N_r + N_b$$

which states that the height is determined by the maximum number of red and black nodes allowable per the rules of Red Black Trees. In an AVL tree, it is known that the total number of nodes is

$$N = 2^{h+1} - 1$$

Solving for the height plus 1(k) to account for the root node,

$$k = log(N + 1)$$

this number represents the total number of nodes in the longest subtree of an AVL Tree, but since in a Red Black Tree only black nodes are counted, the longest subtree may be doubled to give the total number of nodes,

$$N_h = \boxed{2log(N + 1)}$$

III.

Figure 1 is shown with a balance of 0 which meets the requirements of an AVL Tree while also being a Red Black Tree. The root node in Figure 2 has a balance of 2 which violates the balance condition for an AVL Tree, however, this tree still fulfills all of the necessary conditions for a Red Black Tree.

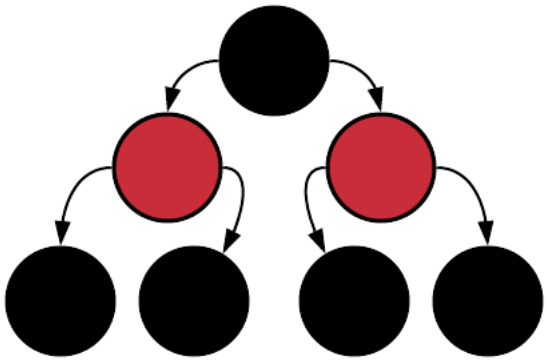

Fig. 1. AVL balanced Red Black Tree.

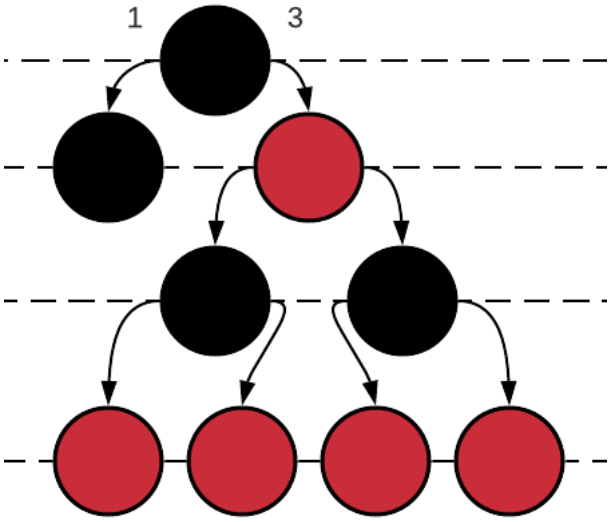

Fig. 2. Non-AVL Red Black Tree.