Implementation of Splay tree

Hrishikesh V. *Department of Computer Science*
PES University
PES1201700276
hrishi.vish@outlook.com

***Abstract*—The first assignment of Advanced Algorithms course comprises the implementation of a Splay Tree.**

***Index Terms*—C++, splay tree, Binary Search Tree**

I. Introduction

A Splay Tree is a binary search tree, with the additional property that recently accessed elements are easier to access in the future. The amortized time of operations such as insertion, deletion and look-up is O(log n). If a node is accessed, it is splayed, such that the node becomes the root of the tree.

II. Implementation of Splay Tree

A. Implementation

Each node of the splay tree belongs to the class node with private members data (value) and the child pointers. The root of the tree is initialized as a pointer to the class "node".

The insertion is done in the usual way. The nodes are accessed in an iterative manner. The value is inserted as the child of the appropriate node and splayed to the top.

When a value is accessed, it is first splayed to the root. If the node is found, the find function returns 1 and if it is not, then the find function returns 0 after splaying the last seen node before falling off the tree.

Deletion is done in the same manner as that of a Binary Search Tree. Upon deletion, the parent of the node deleted is splayed.

III. Traversal

A. Pre-Order

The node is accessed first, followed by its left sub tree and finally its right sub tree. This is done in a recursive manner. The values are appended to a vector.

B. Post-Order

The left sub tree is accessed recursively, followed by the right sub tree and finally the parent, whose value is appended to a vector.

IV. Implementation

A. Node

A node is defined as a class with private members data, left class and right class pointers. To construct a tree, pointers first are created for each node (class) and linked by assigning the left/right pointer of the node to pointers of other nodes.

B. Insert

The tree is first splayed with the element as the parameter. This ensures that if the node is present, it is pushed to the root of the tree. If it isn't, then the last node encountered is pushed to the root. If the element isn't present, a new node is created, with the element as its value and made the root of the tree.

C. Remove

After checking if the tree is NULL, it is splayed, with the element as the splay parameter. If the root's value on splaying isn't equal to the element to be deleted, the function is terminated.
If the root's value on splaying is equal, then it is removed and nothing else is done. Otherwise, the node is deleted in the same manner as that of the BST and the parent of the node to be deleted is splayed to the root.

D. Other Functions

1) Find: The find function splays the element and returns 1 if the element is present and 0 if it isn't.

2) Rotate: There are two rotate functions, left rotate and right rotate. In case of left rotate, The right child of the node becomes the parent and the node becomes the left child.

In the right rotate function, the left child becomes the parent of the node, which becomes the right child.

V. Splay

There are three major steps in splaying.
Zig - If the parent of the node to be splayed is the root of the tree, then the tree is right-rotated about the parent such that the child becomes the root.
Zig-Zig - If both the node to be splayed and the parent are both either left children or right children and if the parent is not the root, the tree is right rotated twice about the root.
Zig-Zag - This rotation is done if the parent is not the root and the parent is the left child and the node is the right child or vice versa. An L-R or an R-L rotation is performed to move the node to the root of the tree.