Splay Tree

Cheruku Ravi Teja

November 14, 2011

Introduction

- Splay trees are self branching binary search tree which has the property of reaccessing the elements quickly that which are recently accessed.
- The performance of the Splay trees depends on the self balancing and self optimizing.
- The worst case with this splay tree algorithm is that this will sequentially access all the elements of the tree which makes tree unbalanced.

Real Time Applications

- It is used to implement caches.
- It has the ability of not to store any data, which results in minimization of memory requirements.
- It can also be used for data compression, e.g.dynamic huffman coding.

Advantages

- It is easy to implement than other self branching binary search trees, such as Red black trees or AVL trees. Much simpler to code than AVL, Red Black trees.
- It requires less space as no balance information is required.

Disadvantages

- More local adjustments during Search operations.
- Invidual operations can be expensive, drawback for real-time applications.
- The main disadvantage of Splay trees is the height. After accessing all $'n'$ elements in the tree, the height of the tree corresponds to worst case access time.

Algorithm

Splay trees are self adjusting binary search trees which performs basic operations such as

- **Search**
- **Insertion**
- **Deletion**

Search,Insert,Delete operations are like in Binary Search trees, except at the end of each operation, a special step called Splaying is done.Splaying the tree rearranges the tree so that element is placed at the root of the tree.It uses tree rotations to bring the element to the top.

Operations

The main basic operations of the Splay tree are

- **Search**
- **Insert**
- **Delete**
- **Splaying**

When to Splay

- **Search**: Splay node where key was found.
- **Insert**: When an item is inserted , a Splay is performed. As a result, the newly inserted node becomes the root of the tree.
- **Delete**:Splay parent of removed node which is either the node with the deleted key or its successor.

Splaying

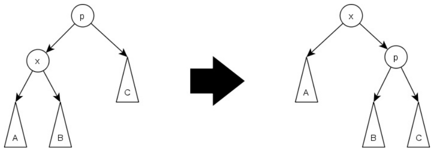
To perform a splay operation , there is need to carry out sequence of Splay steps, which moves the node closer to the root. The recently accessed nodes are kept closer to the root so that tree remains balanced.
Each Splay step depends on three factors

- X is left or right child of its parent node P.
- Check P is root node or not, if not.
- P is left or right child of its parent G.

The three types of Splay steps are

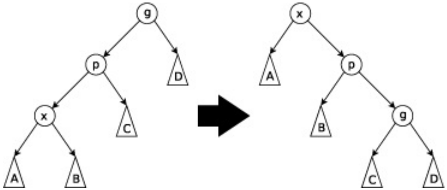
Zig Step

- Let X be a non root node on the access path on which we are rotating.
- If the parent P of X is the root of the tree, we merely rotate X and the root.
- This is same as single rotation.



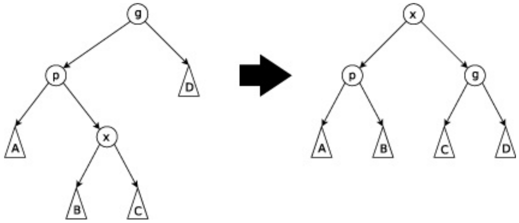
Zig-zig step

- Here X and P are either both left children or both right children.
- The Zig-Zig splay rotates between P and G and X and P.



Zig-Zag step

- In this case, X and both a parent P and a grandparent G. X is a right child and P is a left child or vice versa.
- This is same as the double rotation.



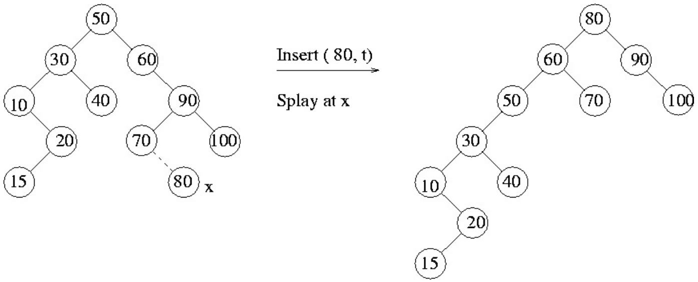
Results of Splaying

- The result is a binary tree, with the left subtree having all keys less than the root, and the right subtree having keys greater than the root.
- The resulted tree is more balanced than the original tree.
- If an operation near the root is done, the tree can become less balanced.

Insertion

To insert a node in to the tree

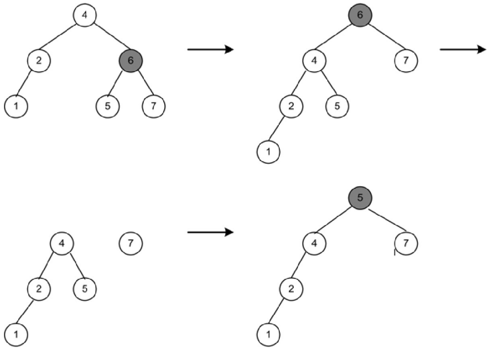
- Insert a node normally in to the tree.
- Splay the newly inserted node to the top of the tree.



Deletion

- Access the node to be deleted bringing it to the root.
- Delete the root leaving two subtrees L left and R right.
- Find the largest element in L, thus the root of L will have no right child.
- Make R the right child of L's root.

Delete element 6



TIME COMPLEXITY

For $'m'$ operations to be performed on splay tree it requires O(mlogn) steps where n is the size of the tree.On an average the time complexity of each operation on the splay tree is

- **Search**:Searching for a node in the tree would take O(logn).
- **Insert**:Inserting a node in to the tree takes O(logn).
- **Delete**:Deleting a node from the tree takes O(logn).

References

- Data Structures and Their Algorithms, Lewis and Denenberg, Harper Collins
- Algorithms by S Dasgupta U.V Vazirani
- wikipedia splaytree http://en.wikipedia.org/wiki/Splay_tree
- Self-Adjusting Binary Search Trees DANIEL DOMINIC SLEATOR AND ROBERT ENDRE TARJAN