Skip list insertion, deletion searching - https://youtu.be/WIqFEESJ5_Q

SPLAY TREE

Advantages of Splay tree

In the splay tree, we do not need to store the extra information. In contrast, in AVL trees, we need to store the balance factor of each node that requires extra space, and Red-Black trees also require to store one extra bit of information that denotes the color of the node, either Red or Black.

It is the fastest type of Binary Search tree for various practical applications. It is used in Windows NT and GCC compilers.

It provides better performance as the frequently accessed nodes will move nearer to the root node, due to which the elements can be accessed quickly in splay trees. It is used in the cache implementation as the recently accessed data is stored in the cache so that we do not need to go to the memory for accessing the data, and it takes less time.

Drawback of Splay tree

The major drawback of the splay tree would be that trees are not strictly balanced, i.e., they are roughly balanced. Sometimes the splay trees are linear, so it will take O(n) time complexity.

Algorithm for Insertion operation

```
Insert(T, n)
temp= T_root
y=NULL
while(temp!=NULL)
y=temp
if(n->data <temp->data)
temp=temp->left
else
temp=temp->right
n.parent= y
if(y==NULL)
T_root = n
else if (n->data < y->data)
y->left = n
else
y->right = n
Splay(T, n)
```

B-Tree of Order m has the following properties...

Property #1 - All leaf nodes must be at same level.

Property #2 - All nodes except root must have at least [m/2]-1 keys and maximum of m-1 keys.

Property #3 - All non leaf nodes except root (i.e. all internal nodes) must have at least m/2 children.

Property #4 - If the root node is a non leaf node, then it must have atleast 2 children.

Property #5 - A non leaf node with n-1 keys must have n number of children.

Property #6 - All the key values in a node must be in Ascending Order.

https://www.youtube.com/watch?v=94ErZ5K8XZg

B Tree Operations: http://www.btechsmartclass.com/data_structures/b-trees.html

2-3 Trees: Properties:

Properties of 2-3 Trees

A 2-3 tree follows the below mentioned properties.

Every internal node in the tree is a 2-node or a 3-node i.e it has either one value or two values.

A node with one value is either a leaf node or has exactly two children. Values in left sub tree < value in node < values in right sub tree.

A node with two values is either a leaf node or has exactly 3 children. It cannot have 2 children. Values in left sub tree < first value in node < values in middle sub tree < second value in node < value in right sub tree.

All leaf nodes are at the same level.

You can notice that the previous example given satisfies all the above mentioned properties.It is important to note from the above properties that all data in a 2-3 tree is stored in a sorted manner which makes search operations fast and effective.

2-3 Tree Operations- https://iq.opengenus.org/2-3-trees/

Simple Search : https://www.geeksforgeeks.org/searching-algorithms/

Range Search tree: Range SEARCH MODULE 5 PDF

PRIORITY SEARCH TREE CONSTRUCTION & INSERTION:
https://www.youtube.com/watch?v=toqAuY8K0Rg