DATA STRUCTURES AND ITS APPLICATIONS

Splay Tree

Kusuma K V

Department of Computer Science
& Engineering

DATA STRUCTURES AND ITS APPLICATIONS

Splay Tree

Kusuma K V

Department of Computer Science & Engineering

Splay Tree

- A splay tree is a binary search tree with the additional property that recently accessed elements are quick to access again.
- Example: Hospital records
- Like self-balancing binary search trees, a splay tree performs basic operations such as insertion, look-up and removal in O(log n) amortized time.
- All normal operations on a binary search tree are combined with one basic operation, called splaying.

Splay Tree

- Splaying the tree for a certain element rearranges the tree so that the element is placed at the root of the tree.
- One way to do this with the basic search operation is to first perform a standard binary tree search for the element in question, and then use tree rotations in a specific fashion to bring the element to the top.

Splaying

- Each splay step depends on three factors:
  - whether x is the left or right child of its parent node, p,
  - whether p is the root or not, and if not
  - whether p is the left or right child of its parent, g (the grandparent of x).

Splaying

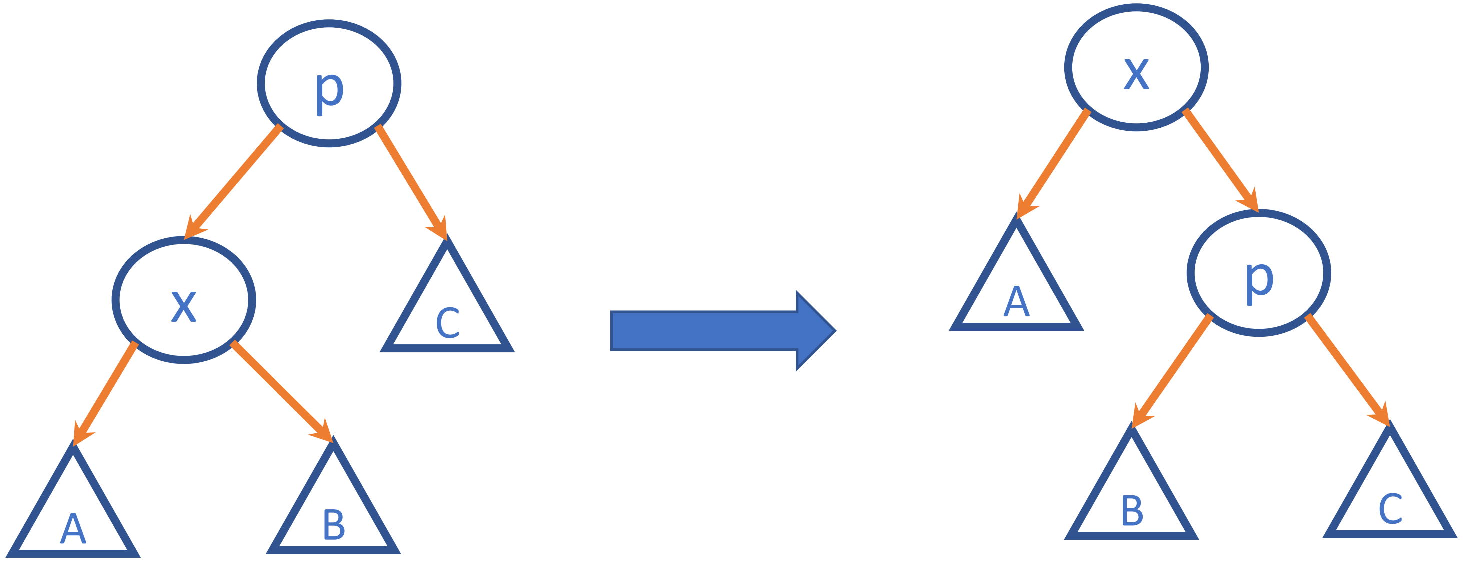
- There are three types of splay steps, each of which has two symmetric variants: left- and right-handed.

- Zig step: this step is done when p is the root.

- Zig - Zig step: this step is done when p is not the root and x and p are either both right children or are both left children.

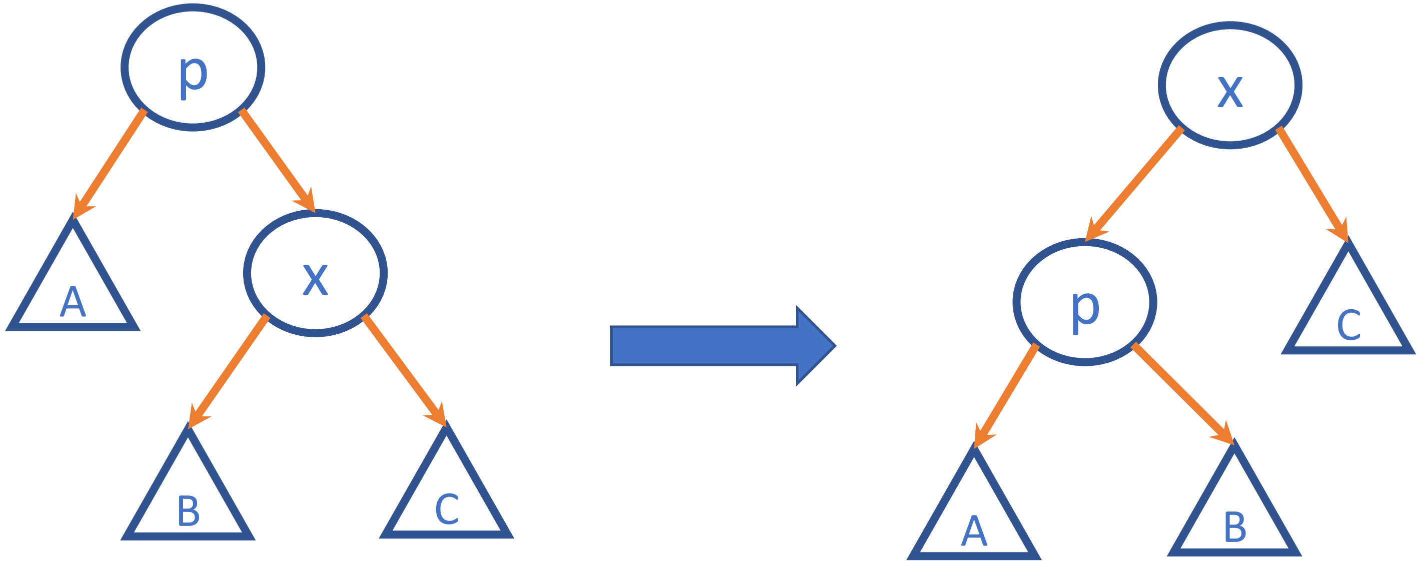
- Zig - Zag step: this step is done when p is not the root and x is a right child and p is a left child or vice versa (x is left, p is right).

Zig step : this step is done when *p* is the root. The tree is rotated on the edge between *x* and *p*.



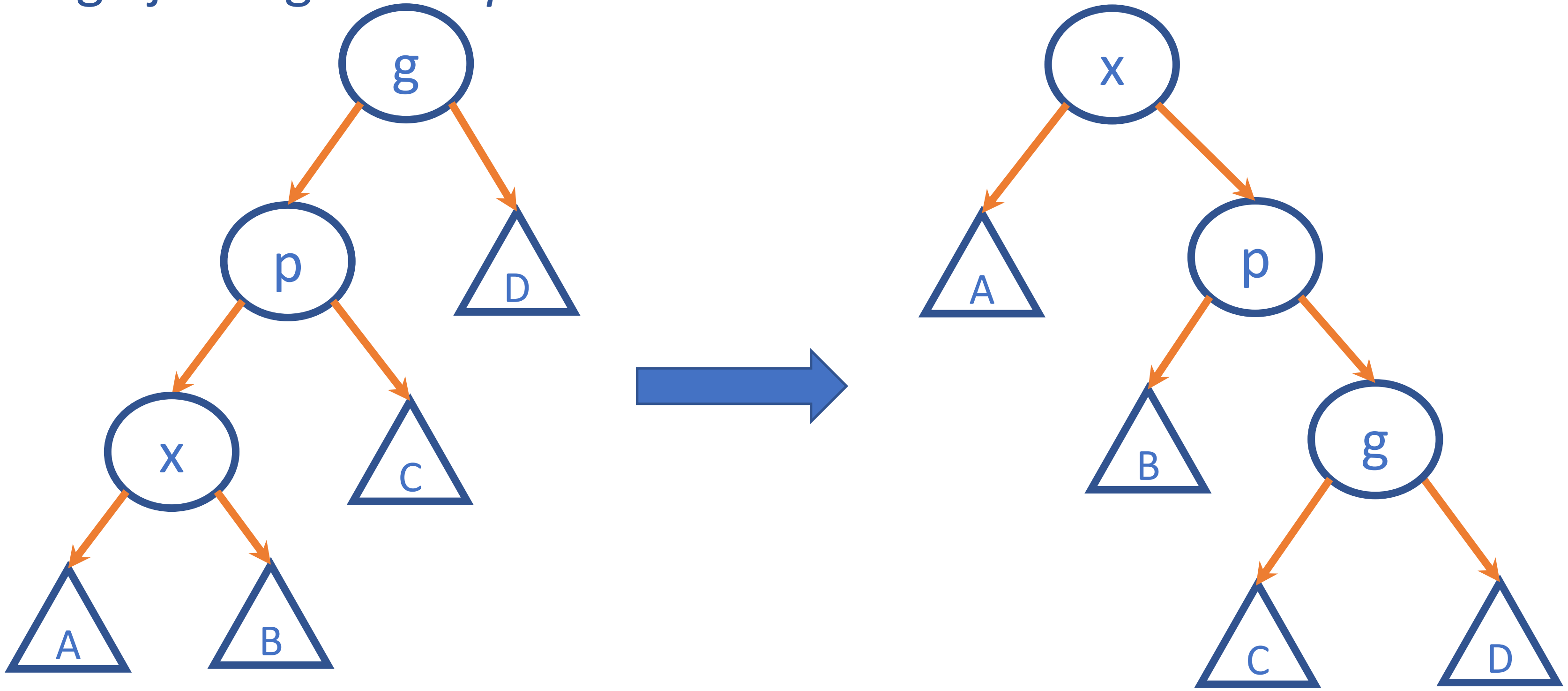
Splay Tree

Zig step : this step is done when *p* is the root. The tree is rotated on the edge between *x* and *p*.



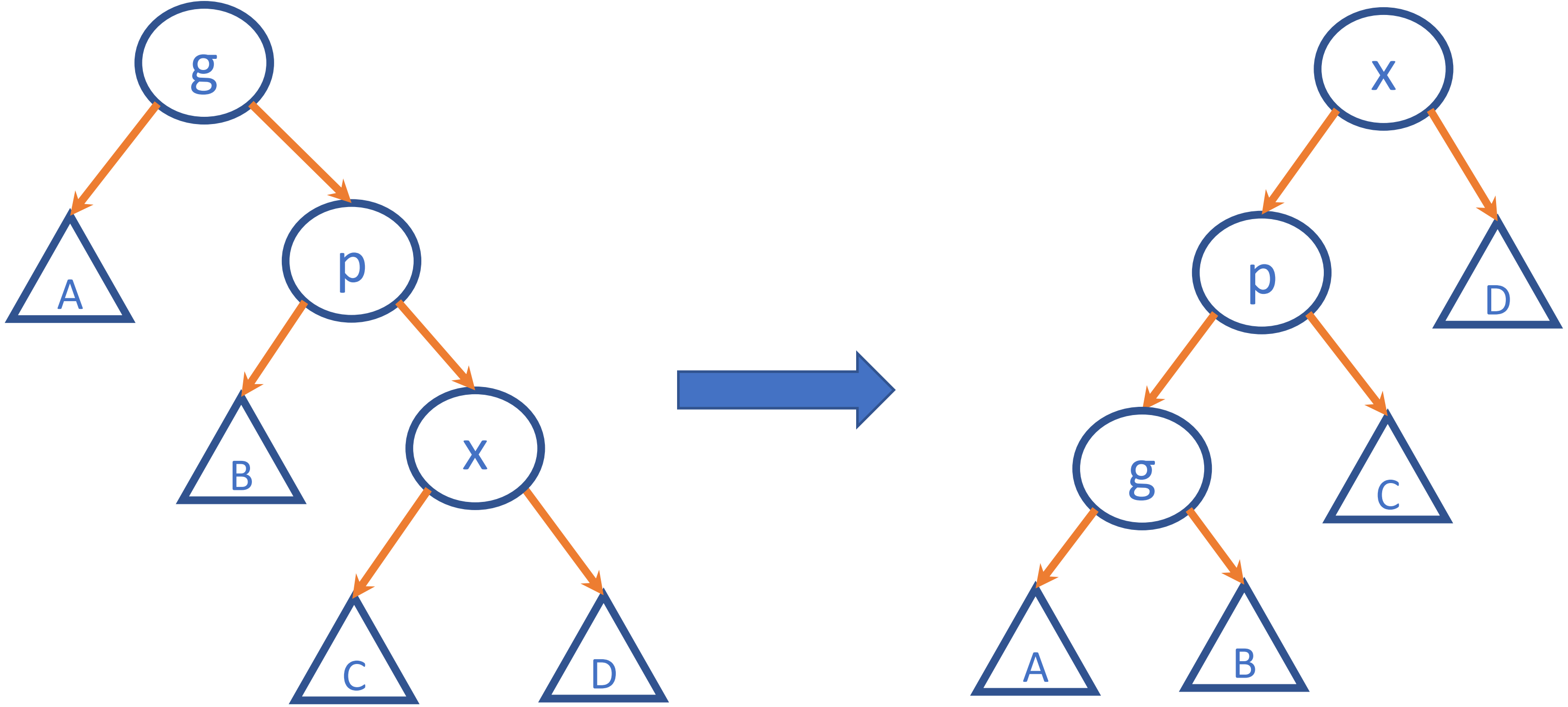
Splay Tree

Zig-Zig step : this step is done when *p* is not the root and *x* and *p* are either both right children or are both left children. The picture below shows the case where *x* and *p* are both **left** children. The tree is rotated on the edge joining *p* with *its* parent *g*, then rotated on the edge joining *x* with *p*.



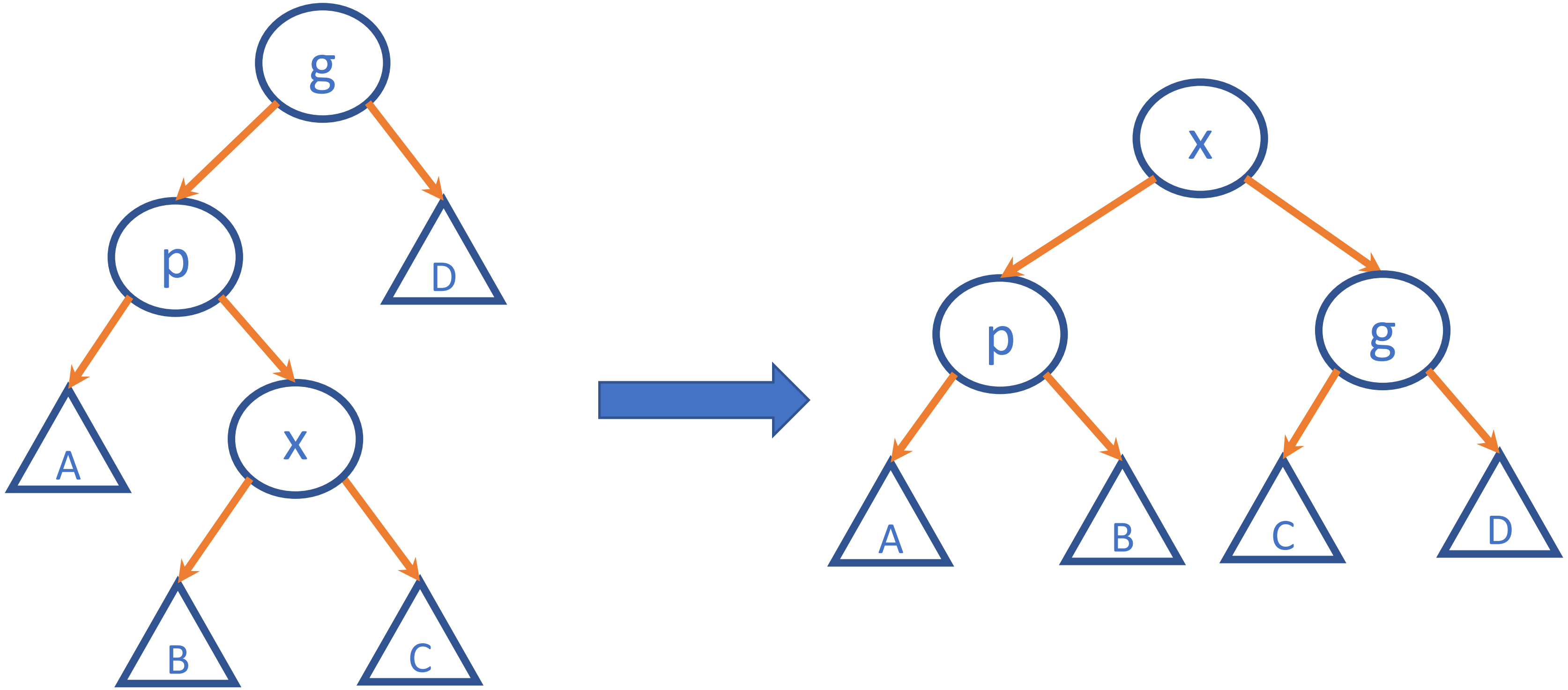
Splay Tree

Zig-Zig step : this step is done when *p* is not the root and *x* and *p* are either both right children or are both left children. The picture below shows the case where *x* and *p* are both **right** children. The tree is rotated on the edge joining *p* with *its* parent *g*, then rotated on the edge joining *x* with *p*.

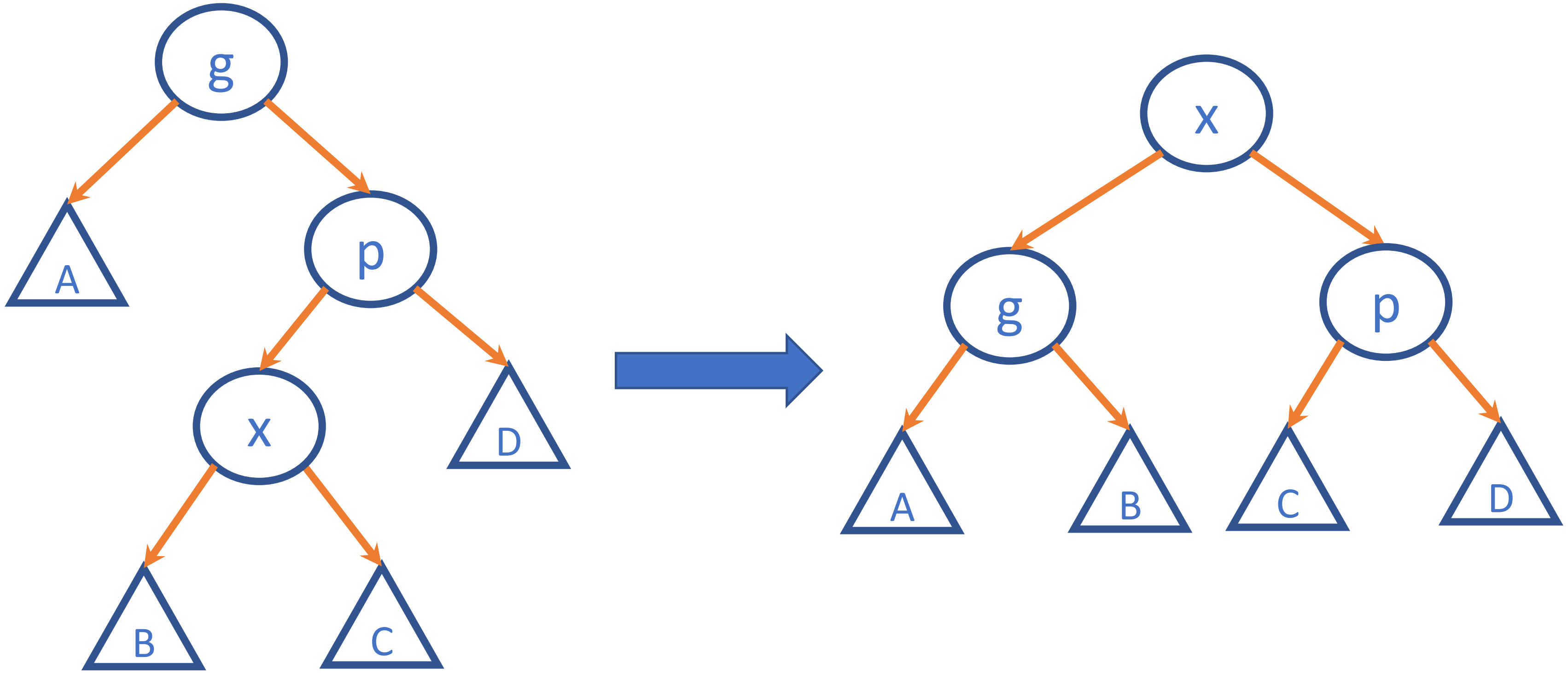


Splay Tree

Zig-Zag step : this step is done when *p* is not the root and *x* is a right child and *p* is a left child or vice versa (*x* is left, *p* is right). The tree is rotated on the edge between *p* and *x*, and then rotated on the resulting edge between *x* and *g*.



Zig-Zag step : this step is done when *p* is not the root and *x* is a left child and *p* is a right child or vice versa (*x* is right, *p* is left). The tree is rotated on the edge between *p* and *x*, and then rotated on the resulting edge between *x* and *g*.



Insertion

To insert a value x into a splay tree:

Insert x as with a normal binary search tree.

when an item is inserted, a splay is performed.

As a result, the newly inserted node x becomes the root of the tree.

Deletion

To delete a node *x*, use the same method as with a binary search tree:

If *x* has two children:

- Swap its value with that of either the rightmost node of its left sub tree (its in-order predecessor) or the leftmost node of its right subtree (its in-order successor).
- Remove that node instead.

In this way, deletion is reduced to the problem of removing a node with 0 or 1 children.

Unlike a binary search tree, in a splay tree after deletion, we splay the parent of the removed node to the top of the tree.

1. What are splay trees?

a) Self-adjusting binary search trees

b) Self-adjusting binary trees

c) Trees with strings

d) Trees with probability distributions

1. What are splay trees?

a) Self-adjusting binary search trees

b) Self-adjusting binary trees

c) Trees with strings

d) Trees with probability distributions

2. What is a splay operation in splay trees?

a) Move parent below child

b) Move a node to root

c) Move root to leaf

d) Remove leaf node

2. What is a splay operation in splay trees?

a) Move parent below child

b) Move a node to root

c) Move root to leaf

d) Remove leaf node

3. Why are splay trees preferred over other balanced trees in some scenarios?

a) Easier to program

b) Space efficiency

c) Easier to program and fast access to recently accessed items

d) Quick searching

3. Why are splay trees preferred over other balanced trees in some scenarios?

a) Easier to program

b) Space efficiency

c) Easier to program and fast access to recently accessed items

d) Quick searching

4. What happens during a Zig-Zag step in splay tree operation?

a) Node and its parent both rotate in the same direction

b) Node rotates with parent while parent rotates with grandparent, in opposite directions

c) Node moves directly to root

d) No rotation happens

4. What happens during a Zig-Zag step in splay tree operation?

a) Node and its parent both rotate in the same direction

b) Node rotates with parent while parent rotates with grandparent, in opposite directions

c) Node moves directly to root

d) No rotation happens

5. Which of the following is a typical application area of splay trees?

a) Cache implementation

b) Sorting algorithms

c) Database indexing

d) Network routing

5. Which of the following is a typical application area of splay trees?

a) Cache implementation

b) Sorting algorithms

c) Database indexing

d) Network routing

References

- https://en.wikipedia.org/wiki/Splay_tree
- "Data Structures and Program Design in C", Robert Kruse, Bruce Leung, C.L Tondo, Shashi Mogalla, Pearson, 2nd Edition, 2019.

PES UNIVERSITY

THANK YOU

Kusuma K V

Department of Computer Science & Engineering