جامعة بيرزيت
BIRZEIT UNIVERSITY

Splay Trees

Dr. Abdallah Karakra

Computer Science Department

COMP242

Thursday, June 1, 2023

Splay Tree: Characteristics

- **A splay tree** is **a type of binary search tree**.
- Structurally, it is **identical** to an ordinary binary search tree; **the only difference** is in the algorithms for finding, inserting, and deleting entries.
- All splay tree operations run in **O(log n) time** on average, where n is the number of items in the tree.

Splay Tree: Characteristics

- ❑ Any **single operation** can take O(n) time in the worst case, where n is the number of items in the tree.
- ❑ Splay trees are simpler and easier to program than BST.
- ❑ Fast access to entries accessed recently.

Splay Tree : Tree Rotations

splay trees are not kept perfectly balanced, but they tend to stay reasonably well-balanced most of the time, thereby averaging O(log n) time per operation in the worst case (and sometimes achieving O(1) average running time in special cases).

Splay Tree Operations

[1] Entry find(Object k);

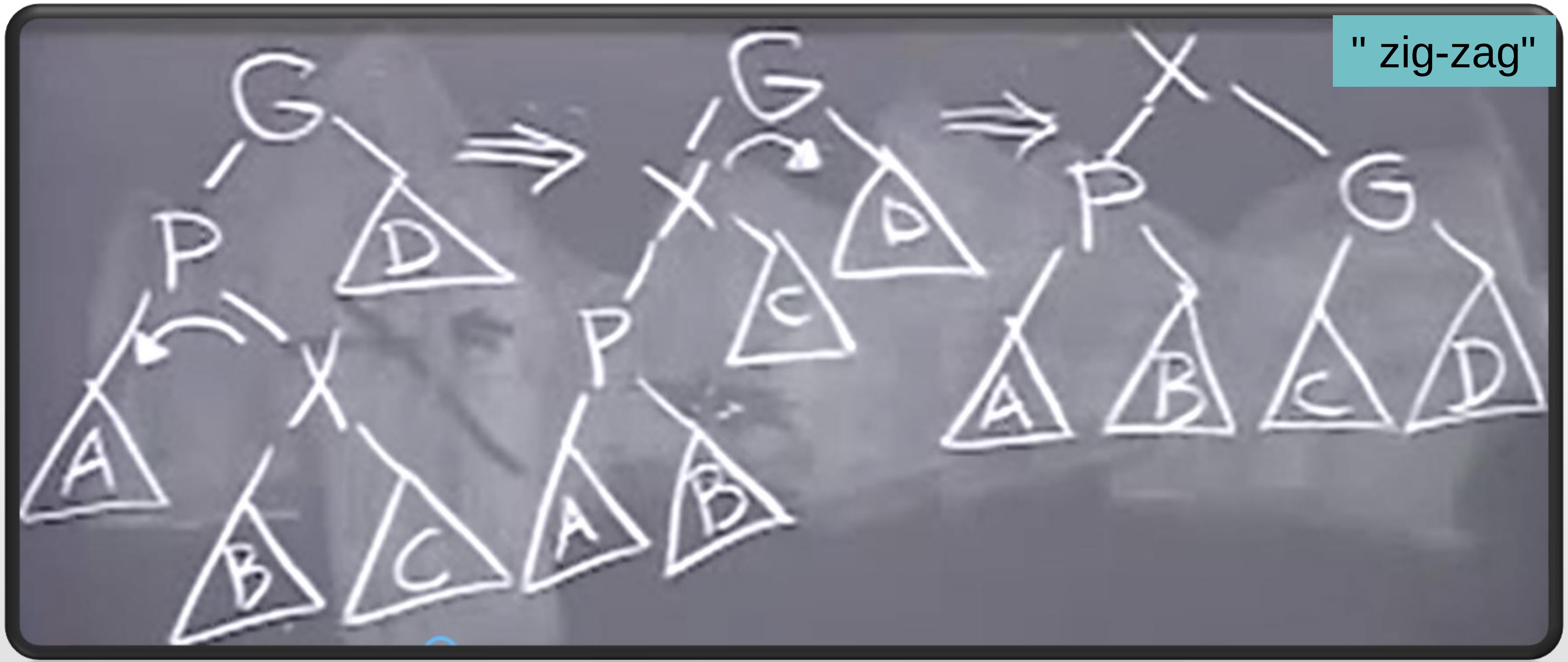
The find() operation in a splay tree begins just like the find() operation in an ordinary binary search tree: we walk down the tree until we find the entry with key k, or reach a dead end (a node from which the next logical step leads to a null pointer).

- Let X be the node where the search ended, whether it contains the key k or not. We splay X up the tree through a sequence of rotations, so that X becomes the root of the tree. **Why?**
  - **One reason is so that recently accessed entries are near the root of the tree, and if we access the same few entries repeatedly, accesses will be very fast.**
  - **Another reason is because if X lies deeply down an unbalanced branch of the tree, the splay operation will improve the balance along that branch.**

Splay Tree Operations

When we splay a node to the root of the tree, there are **three cases** that determine the rotations we use.

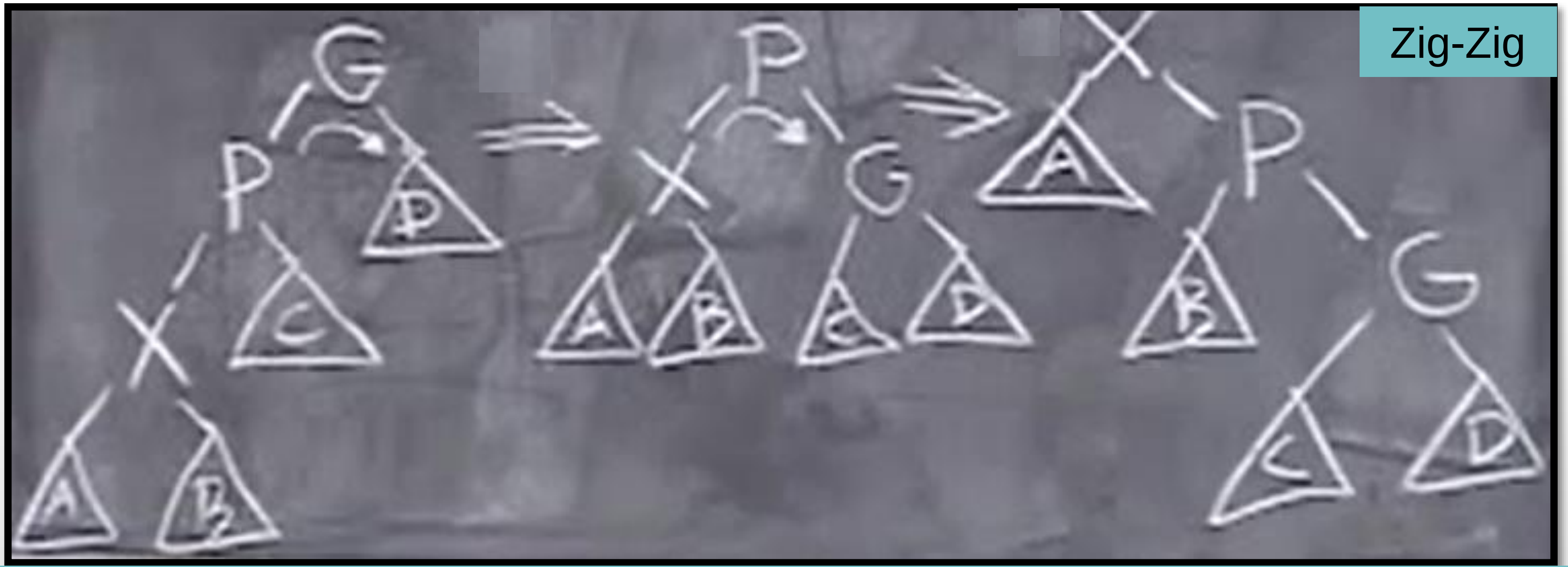
1. X is the right child of a left child (or the left child of a right child)



Splay Tree Operations

When we splay a node to the root of the tree, there are **three cases** that determine the rotations we use.

2. X is the left child of a left child (or the right child of a right child)

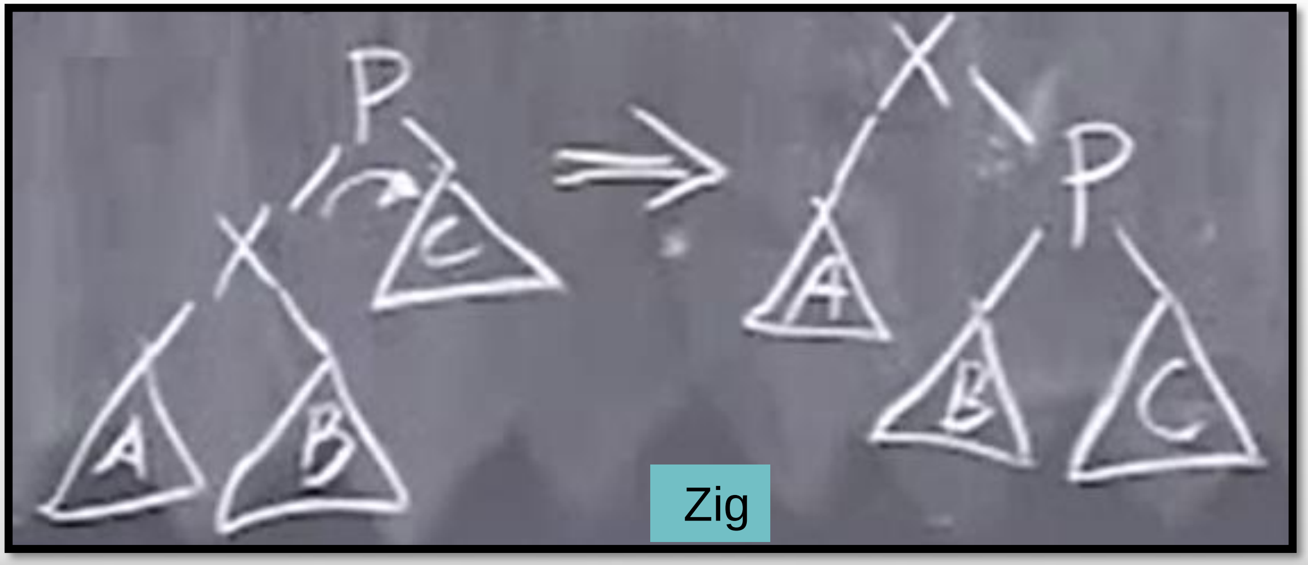


Note for Zig-Zig:
We start with the grandparent, and rotate G and P right Then, we rotate P and X right.

Splay Tree Operations

When we splay a node to the root of the tree, there are **three cases** that determine the rotations we use.

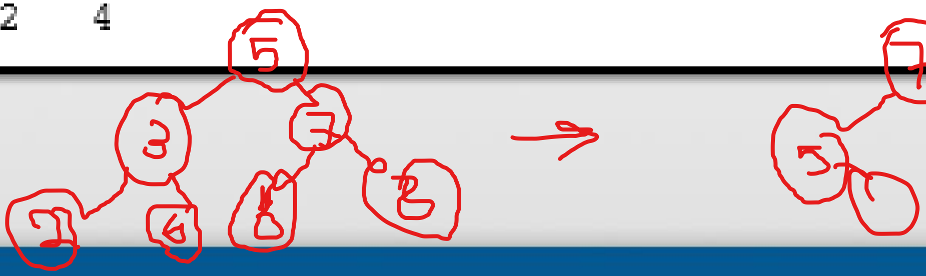
3. x is child of the root



Splay Tree Operations

Here’s an example of "find(7)". Note how the tree’s balance improves.

```
     11                    11                      11                  [7]
    /  \                   /  \                   /  \                /  \
   1    12                1   12                [7]   12             1   11
  / \                    / \                     / \                 /\   / \
 0   9                  0   9                   1   9                0 5   9 12
    / \                    / \                 / \ / \                / \ / \
   3   10  =zig-zig=>    [7] 10  =zig-zag=>   0  5 8 10   =zig=>     3  6 8 10
  / \                    / \                    / \                 / \
 2   5                  5   8                  3   6               2   4
    / \                / \                    / \
   4  [7]             3   6                  2   4
      / \            / \
     6   8          2   4
```



Splay Tree Operations

SELF STUDY

[2] Entry min();
Entry max();

These methods begin by finding the entry with minimum or maximum key, just like in an ordinary binary search tree. Then, they splay the node containing the minimum or maximum key to the root

[3] Entry insert(Object k, Object v);

insert() begins by inserting the new entry (k, v), just like in an ordinary binary search tree. Then, it splays the new node to the root

[4] Entry remove(Object k);
An entry having key k is removed from the tree, just as with ordinary binary search trees. Recall that the node containing k is removed if it has zero or one children.

جامعة بيرزيت BIRZEIT UNIVERSITY

Example

Example

Extra Exercises

Question?

“Success is the sum of small efforts, repeated day in and day out.”
Robert Collier

Reference:

Kathy Sierra and Bert Bates, *Head First Java*, O'Reilly, 2005. ISBN # 0-596-00920-8

Michael T. Goodrich and Roberto Tamassia, *Data Structures and Algorithms in Java*, John Wiley & Sons, 2010. ISBN # 0-470-38326-7.