

ACOL106: Data Structures and Algorithms

Ragesh Jaiswal, IIT Delhi

Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
- ② Depth property: All leaves have the **same depth**.

- Claim 1: The height of any (2, 4)-tree storing n entries is $O(\log n)$.

Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Claim 1: The height of any (2, 4)-tree storing n entries is $O(\log n)$.
 - Question: What is the running time of search operation on (2, 4)-Tree?

Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Claim 1: The height of any (2, 4)-tree storing n entries is $O(\log n)$.
 - Question: What is the running time of search operation on (2, 4)-Tree? $O(\log n)$

Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Claim 1: The height of any (2, 4)-tree storing n entries is $O(\log n)$.
 - Question: What is the running time of search operation on (2, 4)-Tree? $O(\log n)$
 - Question: How do we perform an insert operation?

Data Structures

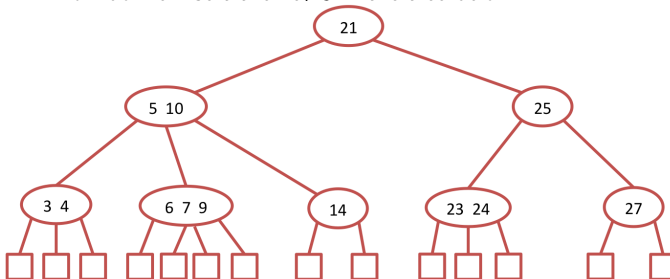
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform an insert operation?
 - How do we insert the key 8 in the tree below?



Data Structures

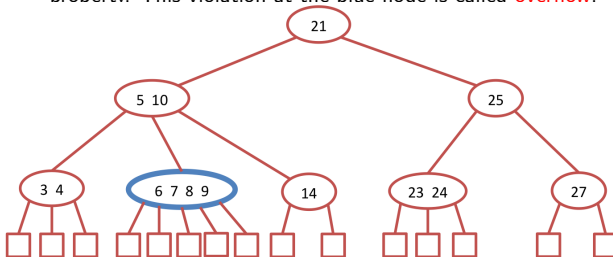
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform an insert operation?
 - How do we insert the key 8 in the tree below?
 - The tree below is not a (2, 4)-Tree since it violates the size property. This violation at the blue node is called **overflow**.



Data Structures

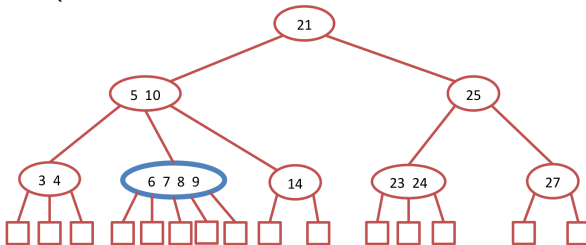
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform an insert operation?
 - How do we insert the key 8 in the tree below?
 - The tree below is not a (2, 4)-Tree since it violates the size property. This violation at the blue node is called **overflow**.
 - Question: How do we resolve an overflow?



Data Structures

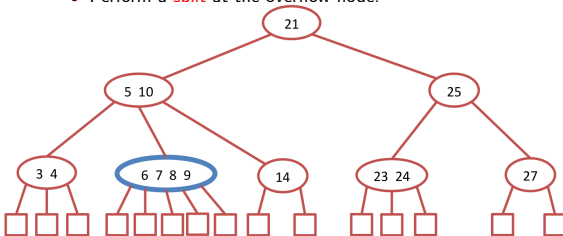
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform an insert operation?
 - How do we insert the key 8 in the tree below?
 - The tree below is not a (2, 4)-Tree since it violates the size property. This violation at the blue node is called **overflow**.
 - Question: How do we resolve an overflow?
 - Perform a **split** at the overflow node.



Data Structures

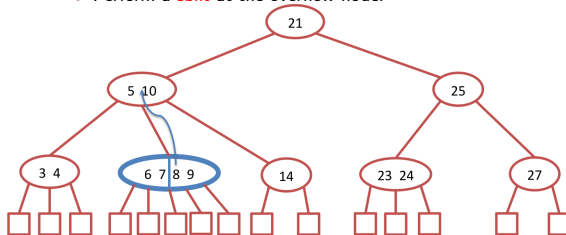
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
- ② Depth property: All leaves have the **same depth**.

- Question: How do we perform an insert operation?
 - How do we insert the key 8 in the tree below?
 - The tree below is not a (2, 4)-Tree since it violates the size property. This violation at the blue node is called **overflow**.
 - Question: How do we resolve an overflow?
 - Perform a **split** at the overflow node.



Data Structures

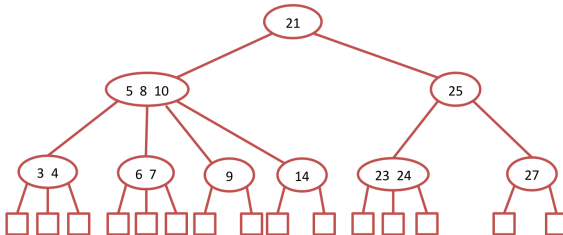
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
- ② Depth property: All leaves have the **same depth**.

- Question: How do we perform an insert operation?
 - How do we insert the key 8 in the tree below?
 - The tree below is not a (2, 4)-Tree since it violates the size property. This violation at the blue node is called **overflow**.
 - Question: How do we resolve an overflow?
 - Perform a **split** at the overflow node.



Data Structures

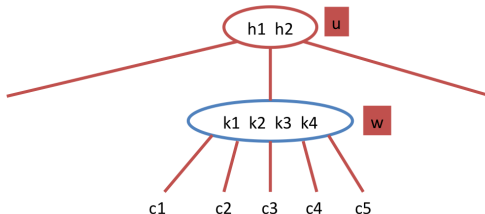
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Split operation on an overflow node w :



Data Structures

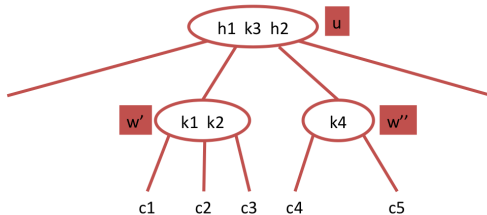
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Split operation on an overflow node.



Data Structures

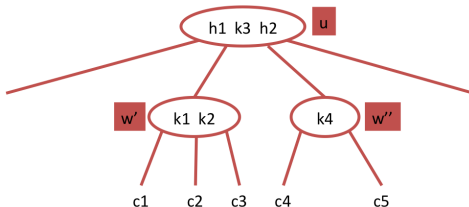
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Split operation on an overflow node.
- Complications:
 - What if the overflow node w is the root node?



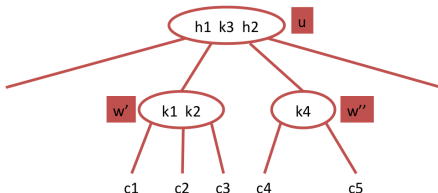
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2,4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Split operation on an overflow node.
 - Complications:
 - What if the overflow node w is the root node? **create a new root node**
 - What if after the split, the node u overflows?



Data Structures

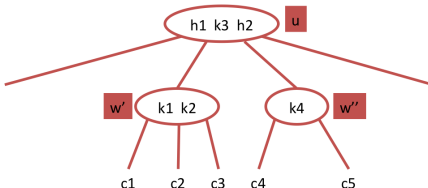
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2,4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
- ② Depth property: All leaves have the **same depth**.

- Split operation on an overflow node.
- Complications:
 - What if the overflow node w is the root node? **create a new root node**
 - What if after the split, the node u overflows? **continue performing split at u**



Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: What is the running time of insert in (2, 4)-Tree?

Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: What is the running time of insert in (2, 4)-Tree?
 $O(\log n)$

Data Structures

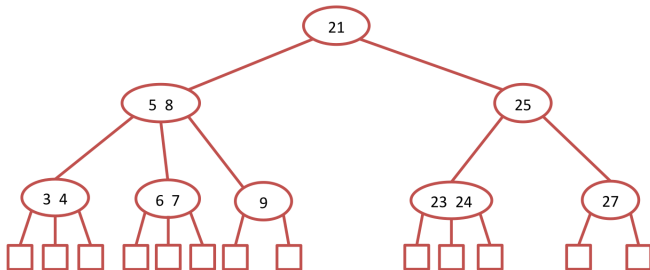
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
- ② Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - How do we delete the entry with key 8 from the tree below?



Data Structures

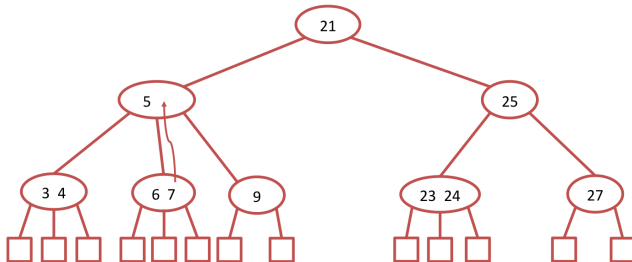
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2,4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - How do we delete the entry with key 8 from the tree below?
Similar to BST



Data Structures

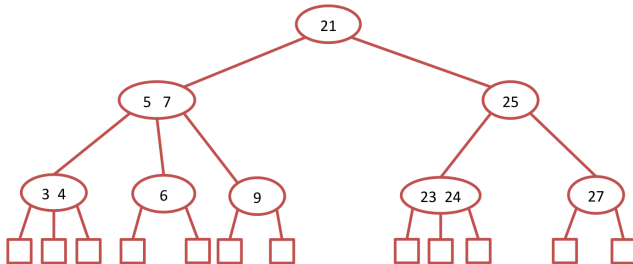
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2,4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - How do we delete the entry with key 8 from the tree below?
Similar to BST



Data Structures

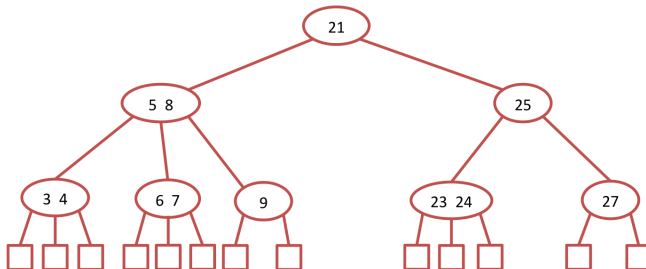
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
- ② Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - How do we delete the entry with key 27 from the tree below?



Data Structures

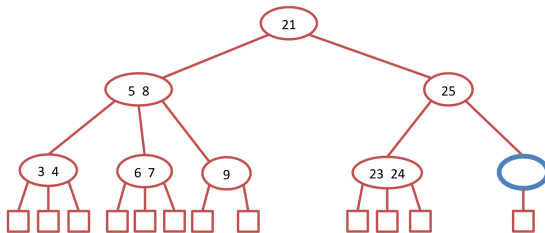
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
- ② Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - How do we delete the entry with key 27 from the tree below?
 - Since the children of node with key 27 are leaves, we can simply remove 27. This, however, creates a node with fewer than 2 children.
 - This condition is called **underflow**.



Data Structures

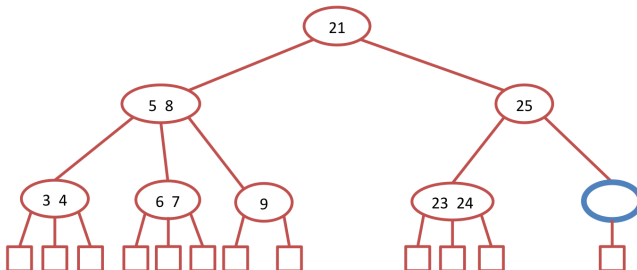
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node?



Data Structures

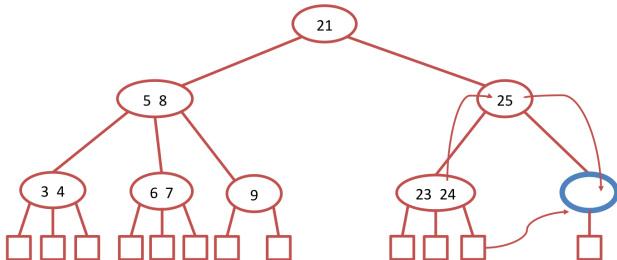
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2,4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node?
 - If possible, borrow an entry from a sibling.



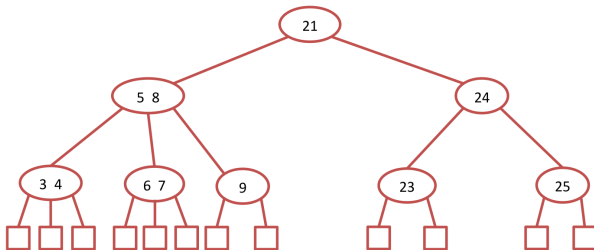
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible**
 - Now consider deleting 21.



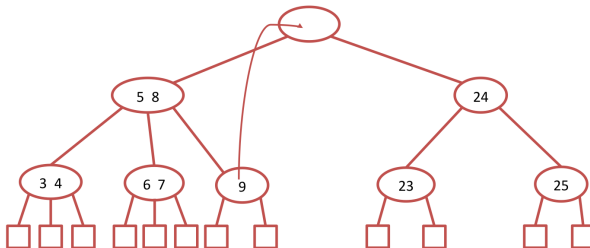
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible**
 - Now consider deleting 21.



Data Structures

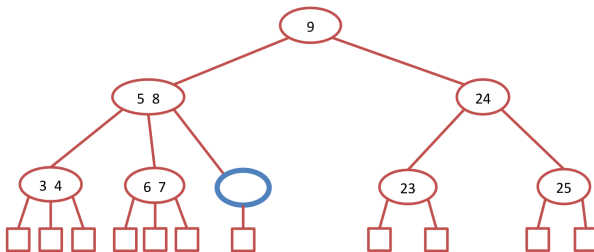
Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- 1 Size property: Every internal node has **at most 4 children**.
- 2 Depth property: All leaves have the **same depth**.

- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible**
 - Now consider deleting 21.



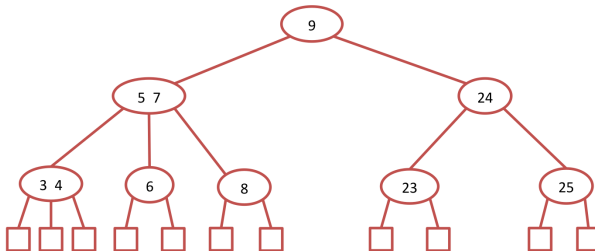
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible**
 - Now consider deleting 8.



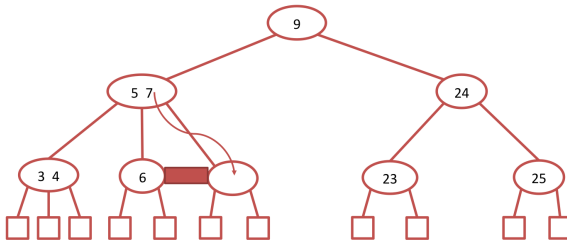
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2,4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible**
 - Now consider deleting 8.
 - Since borrowing is not possible, perform **fusion**.



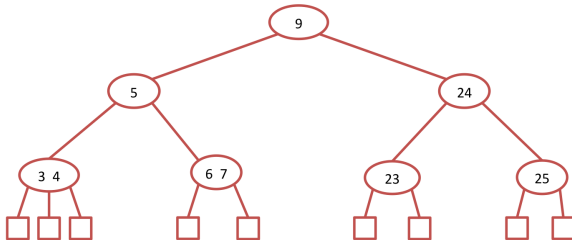
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible**
 - Now consider deleting 8.
 - Since borrowing is not possible, perform **fusion**.



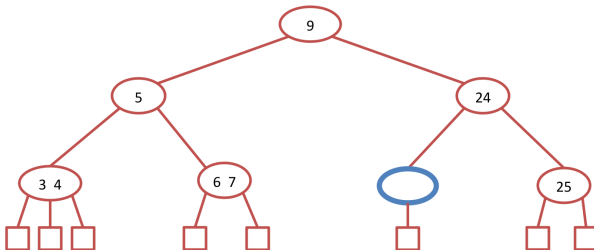
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible, else perform fusion**.
 - Now consider deleting 23.



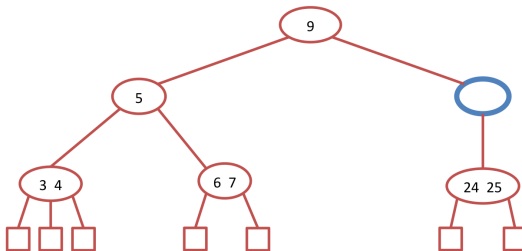
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible, else perform fusion**.
 - Now consider deleting 23.



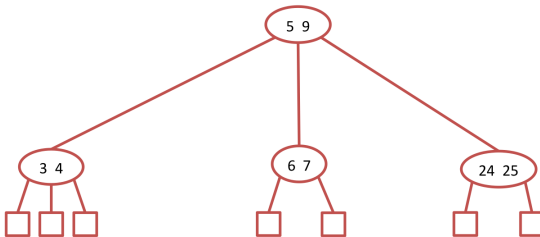
Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

Definition ((2-4)-Tree)

A (2, 4)-Tree is a multiway search tree with the following two additional properties:

- ① Size property: Every internal node has **at most 4 children**.
 - ② Depth property: All leaves have the **same depth**.
- Question: How do we perform deletion?
 - Question: How do we resolve **underflow** at a node? **borrow from sibling if possible, else perform fusion**.
 - Now consider deleting 23.



Data Structures

Multiway Search Trees $\rightarrow$ (2,4)-Trees

- We can easily generalize the techniques of (2, 4)-Tree to multiway search trees, where instead of every internal node having at least 2 and at most 4 children, to multiway search trees where every internal node has at least d and at most $2d$ children, where d is some constant.
- Such trees are known by the name **B-tree** and are used in modern filesystems and database implementations.

Data Structures

Other Balanced Search Trees

- AVL Tree and $(2, 4)$ -Tree are just two examples of balanced search trees.
- There are many more examples of such trees.
- The book gives two other examples: red-black tree and Splay tree.

Data Structures: B-Tree

Data Structures

B-Tree

- Binary Search Tree (BST), AVL Tree, and (2,4)-tree are implementations of the Abstract Data Type called *Map* where key-value pairs with all distinct keys are stored and the primary supported operations are: get (search), put (insert), and remove (delete).
- All the above data structures and in fact all the data structures that we have seen (and implemented) in this class until now are memory-based. Meaning, that they are stored and accessed from primary memory.
- Suppose the data is so large that it does not make sense to implement a memory-based data structure and we have to design a **disk-based** data structure.
- For this, we will first have to understand how the disk is accessed.

Data Structures

B-Tree

- Suppose the data is so large that it does not make sense to implement a memory-based data structure and we have to design a **disk-based** data structure.
- For this, we will first have to understand how the disk is accessed:
 - There are **slow** mechanical operations involved when accessing data in a disk storage.
 - There is **seek time** for positioning the head at the correct place and **transfer time** for reading (or writing) data.
 - Disk access is performed in data chunks called **blocks** (typically size 4KB)
 - Disk access is significantly slower than memory access.

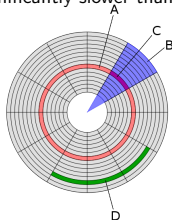


Figure: Tracks, Sectors, and Blocks on a disk.

Data Structures

B-Tree

- Suppose the data is so large that it does not make sense to implement a memory-based data structure and we have to design a **disk-based** data structure.
- For this, we will first have to understand how the disk is accessed:
 - There are **slow** mechanical operations involved when accessing data in a disk storage.
 - There is **seek time** for positioning the head at the correct place and **transfer time** for reading (or writing) data.
 - Disk access is performed in data chunks called **blocks** (typically size 4KB)
 - Disk access is significantly slower than memory access.
- Question: Are BST, AVL-Tree, (2, 4)-tree appropriate for disk-based implementation?

Data Structures

B-Tree

- Suppose the data is so large that it does not make sense to implement a memory-based data structure and we have to design a **disk-based** data structure.
- For this, we will first have to understand how the disk is accessed:
 - There are **slow** mechanical operations involved when accessing data in a disk storage.
 - There is **seek time** for positioning the head at the correct place and **transfer time** for reading (or writing) data.
 - Disk access is performed in data chunks called **blocks** (typically size 4KB)
 - Disk access is significantly slower than memory access.
- Question: Are BST, AVL-Tree, (2, 4)-tree appropriate for disk-based implementation?
- Goals of disk-based implementation:
 - Space usage should be linear in the size of the data.
 - The number of disk accesses should be as small as possible.

End