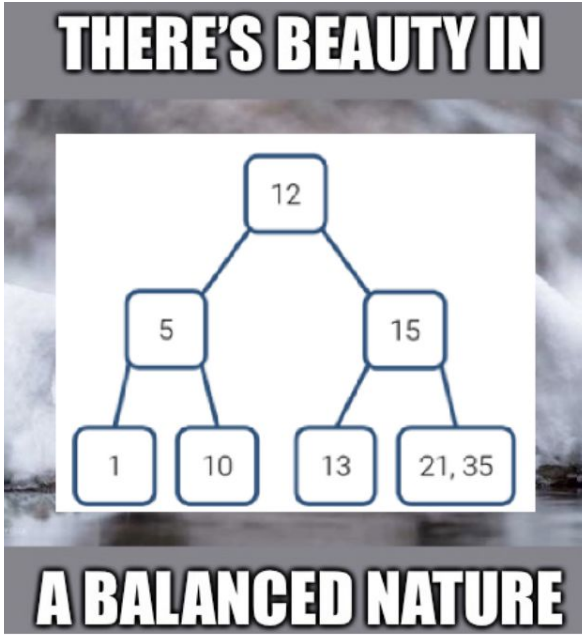
CS 1332R
WEEK 9

2-4 Trees

Exam 2 Review



ANNOUNCEMENTS

- ❑

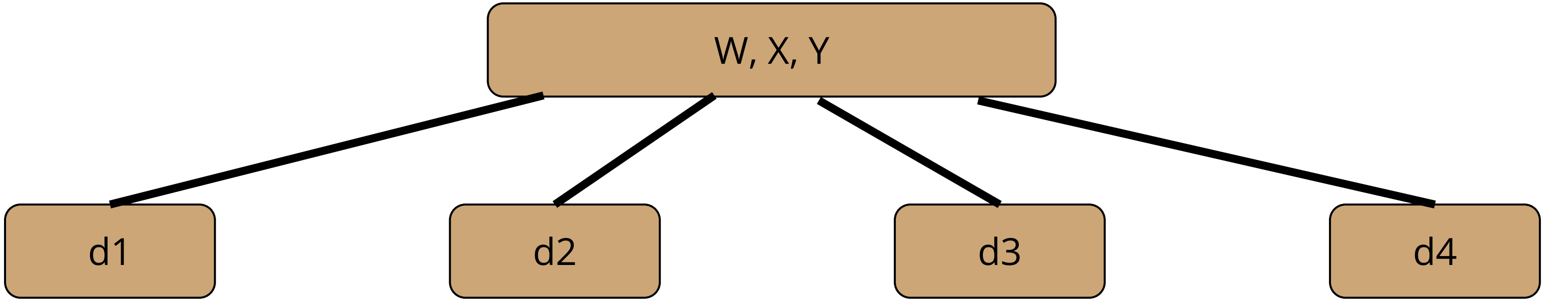
2-4 Tree

SHAPE Property

1. The tree must be **full and complete.**
2. A node must have 1 - 3 pieces of data.
3. A node with n pieces of data must have n + 1 children. Therefore, each node has ***2-4 children.***

ORDER Property

1. The pieces of data within a node are in ascending order.
2. All data in the left child node must be less than the parent data.
3. All data in the right child node must be greater than the parent data.



d1 < W < d2 < X < d3 < Y < d4

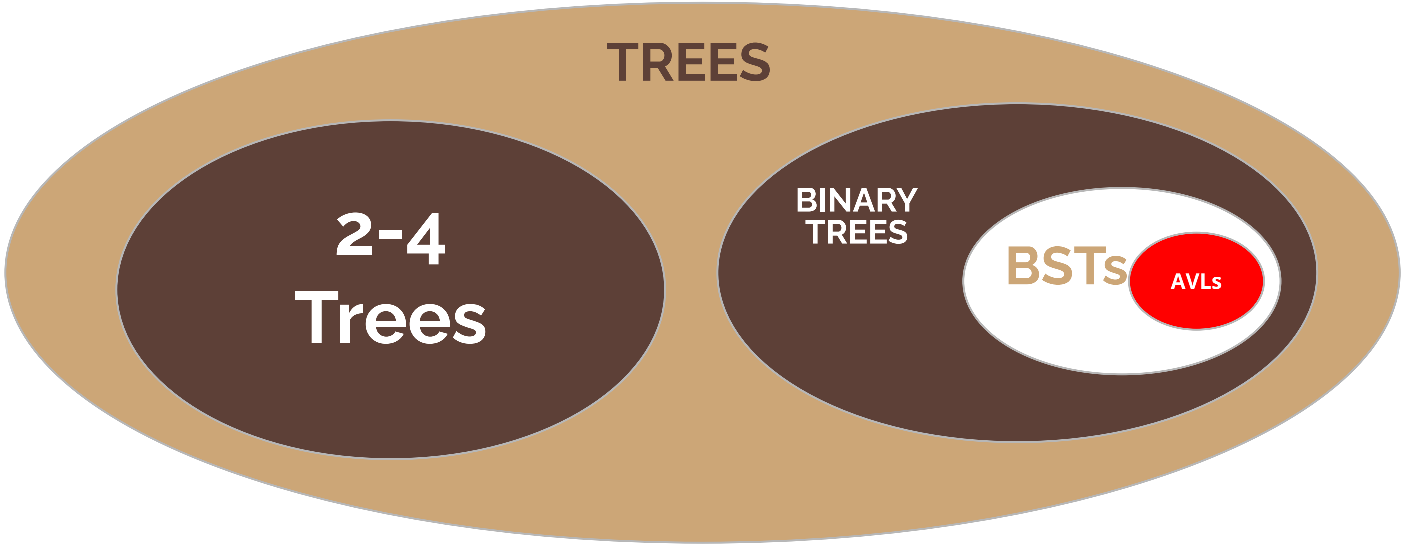
2-4 Tree

SHAPE Property

1. The tree must be **full and complete.**
2. A node must have 1 - 3 pieces of data.
3. A node with n pieces of data must have n + 1 children. Therefore, each node has ***2-4 children.***

ORDER Property

1. The pieces of data within a node are in ascending order.
2. All data in the left child node must be less than the parent data.
3. All data in the right child node must be greater than the parent data.



2-4 Tree: Add

PROCEDURE:

1. Search for the data (intuitive approach using the order property).
2. Add the data in the appropriate leaf node.
3. If there are **4 data elements** in a leaf node →

HANDLING OVERFLOW

Method: Promotion

1. Choose the second or third item in the node.
2. Split the node: *items less than the promoted data* vs. *items greater than the promoted data.*

What if the parent node now has 4 pieces of data?

repeat promotion steps

When do we terminate?

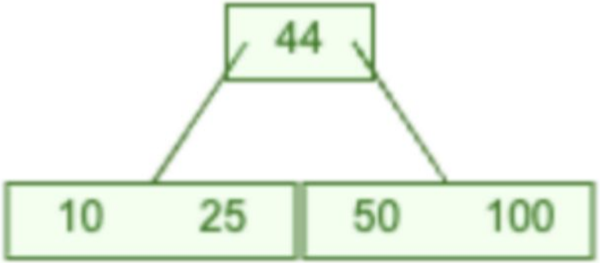
when a new root is created

2-4 Tree: Practice

Draw the tree following these operations:
add(75), add(15), add(60)

IMPLEMENTATION

Promote: ***second piece of data***

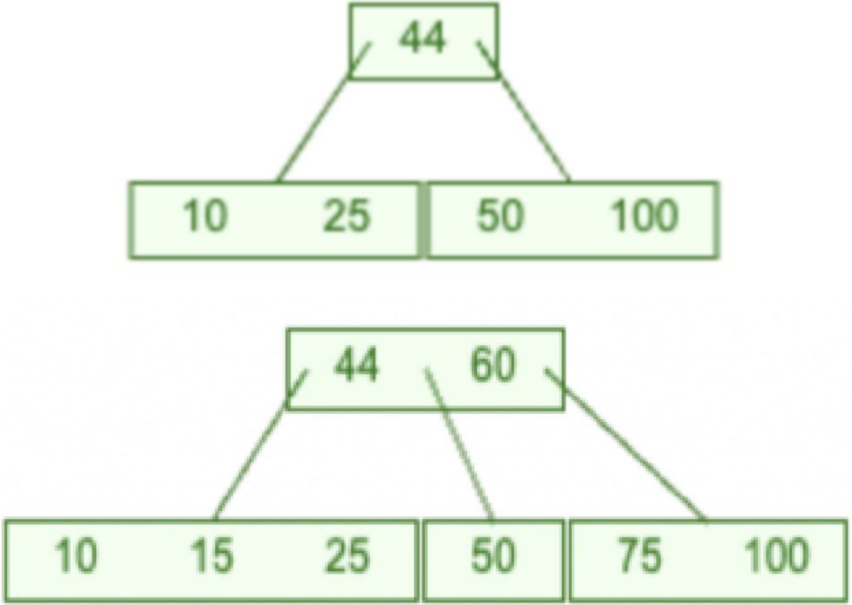


2-4 Tree: Practice

Draw the tree following these operations:
add(75), add(15), add(60)

IMPLEMENTATION

Promote: ***second piece of data***



2-4 Tree: Remove

PROCEDURE:

1. Search for the data
2. If the data is in an internal node, replace it with its predecessor/successor data.
3. If a **node becomes empty** →

HANDLING UNDERFLOW

Method 1: Transfer

1. If a sibling node has > 1 data, pull down the shared parent data and replace parent data with extra sibling data.

A transfer ends the removal process.

Method 2: Fusion

1. If no sibling node has > 1 data, pull down the parent data and fuse with the sibling node that shared the parent.

What if the parent node now has no data?

perform another transfer or fusion

2-4 Tree: Remove

PROCEDURE:

1. Search for the data
2. If the data is in an internal node, replace it with its predecessor/successor data.
3. If a **node becomes empty** →

HANDLING UNDERFLOW

Method 1: Transfer

1. If a sibling node has > 1 data, pull down the shared parent data and replace parent data with extra sibling data.

A transfer ends the removal process.

Method 2: Fusion

1. If no sibling node has > 1 data, pull down the parent data and fuse with the sibling node that shared the parent.

What if the root node now has no data?

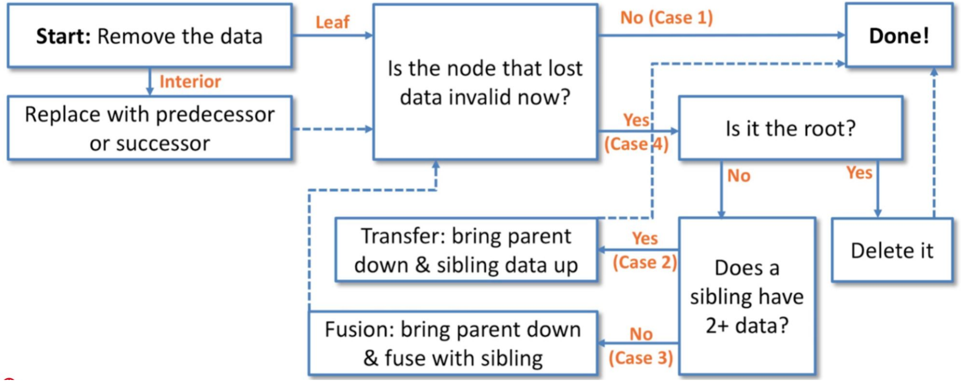
delete the root

2-4 Tree: Remove

PROCEDURE:

1. Search for the data
2. If the data is in an internal node, replace it with its predecessor/successor data.
3. If a **node becomes empty** →

HANDLING UNDERFLOW



Start: Remove the data

Leaf

Interior

Replace with predecessor or successor

Is the node that lost data invalid now?

No (Case 1)

Done!

Yes (Case 4)

Is it the root?

No

Yes

Delete it

Does a sibling have 2+ data?

Yes (Case 2)

Transfer: bring parent down & sibling data up

No (Case 3)

Fusion: bring parent down & fuse with sibling

2-4 Tree: Practice

Draw the tree following these operations:
remove(12), remove(95), remove(112), add(12), add(15).

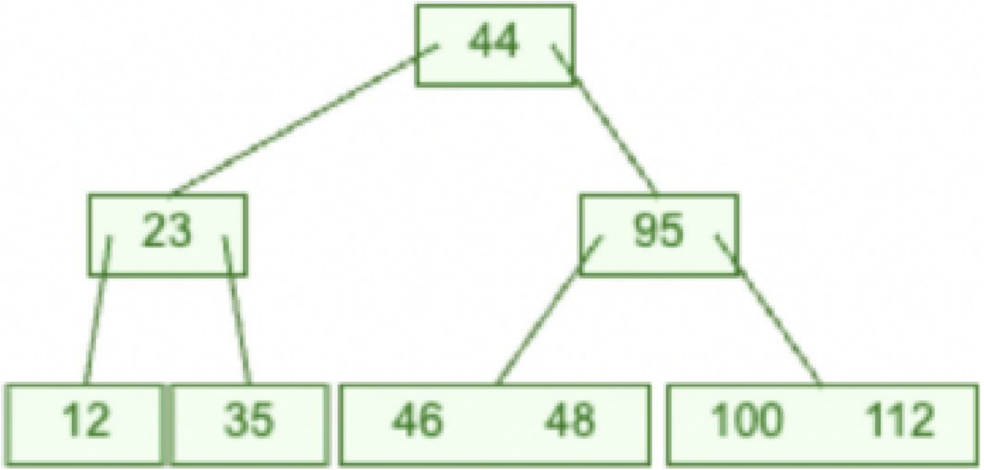
IMPLEMENTATION

Promote: ***second piece of data***

Pred/Succ: If removing from an internal node, replace with the ***predecessor***.

Transfer: If multiple siblings have extra data, take from the ***right sibling***.

Fusion: If a piece of data had two parents, pull down the ***leftmost parent*** data.



2-4 Tree: Practice

Draw the tree following these operations:
remove(12), remove(95), remove(112), add(12), add(15).

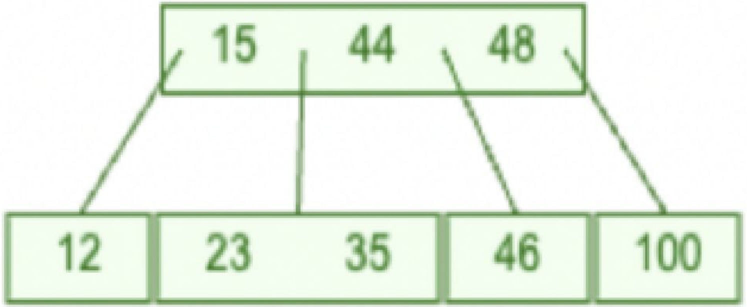
IMPLEMENTATION

Promote: ***second piece of data***

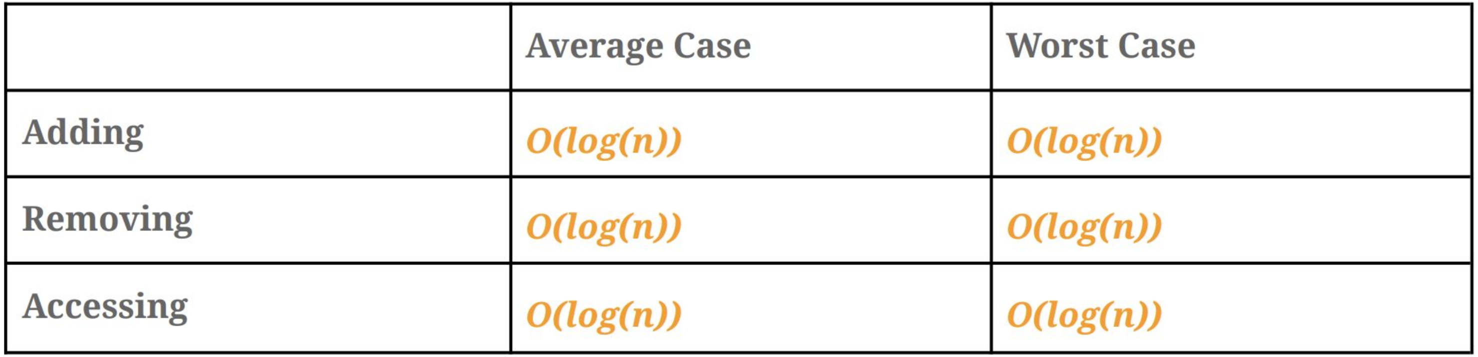
Pred/Succ: If removing from an internal node, replace with the ***predecessor***.

Transfer: If multiple siblings have extra data, take from the ***right sibling***.

Fusion: If a piece of data had two parents, pull down the ***leftmost parent*** data.



2-4 Tree: Efficiencies



	Average Case	Worst Case
Adding	*O(log(n))*	*O(log(n))*
Removing	*O(log(n))*	*O(log(n))*
Accessing	*O(log(n))*	*O(log(n))*

WHY?

A 2-4 tree is full and complete.

2-4 Tree: Practice

1. What is the worst case time complexity to remove data in a leaf of a 2-4 tree? **O(logn)**
2. What is the average case time complexity of adding to a 2-4 tree that triggers a promotion in the root node? **O(logn)**
3. What is the worst case time complexity to perform a fusion between two internal nodes? **O(logn)**
4. What is the average case time complexity to perform a promotion into the parent node? **O(1)**
5. What is the worst case time complexity of removing an element from a 2-4 tree which causes underflow to propagate all the way up to the root? **O(logn)**

EXAM 2 REVIEW

Socrative: CS1332

(There's also a practice exam in Canvas: Files -> resources -> recitation materials -> recitation practice exams)

Any questions?

Name
Office Hours
Contact

Name
Office Hours
Contact

Let us know if there is anything specific you want out of recitation!