Lecture 21: 2-3 Trees

2-3 Trees

2-3 Trees are a way to enforce a balance condition. They come at a cost: the tree is no longer binary, which makes everything uglier.

- In a 2-3 tree, every node has either 1 or 2 values.
- If a node has 1 value and is not a leaf, then it has 2 children, and we call it a 2-node.
- If a node has 2 values and is not a leaf, then it has 3 children, and we call it a 3-node.
- All leaves must be at the same level.
- For all 2-nodes, the binary search tree property holds.
- For all 3-nodes with keys $k_1 < k_2$, and subtrees $T_1, T_2, T_3$, the keys in $T_1$ are $\leq k_1$, the keys in $T_2$ are $\geq k_1$ and $\leq k_2$, and the keys in $T_3$ are $\geq k_2$.

We generalize the node class as follows:

```cpp
template<class T>
class Node {
    int numKeys; //1 or 2
    T key[2]; //space for 2 keys
    Node* subtree[3]; //space for 3 children
};
```

Insertion

To insert a value, search for it in the tree. If the leaf has only one value, insert the new value.

**Question 1** What does the tree look like after inserting 21?

If The leaf has two values, insert it anyway (it now invalidly has 3 values), and move the middle value up. If the parent has one value, you now have a valid 2-3 tree.
Question 2  Continuing with the same tree, what does it look like after inserting 22?

If the parent has two values, keep moving the middle node up. It may eventually become your new root node.

Question 3  Continuing with the same tree, what does it look like after inserting 12?

You can think of inserting a value as unzipping the tree. As the middle value propagates up the tree, you break it apart until the whole process stops.

Remove

We only remove from leaf nodes. Search for the value in the tree, then replace it with the next highest value, which will be a leaf value.

If we moved something from a 3-node, we’re done.

Question 4  Continuing with the same tree, what does it look like after removing 14?

If we delete a 2-node, then we have a hole in the tree. This hole can steal from an adjacent sibling.

Question 5  Continuing with the same tree, what does it look like after removing 4?

We can also borrow from the parent.

Question 6  Continuing with the same tree, what does it look like after removing 17?

Otherwise we have to bubble the hole up towards the top. We always merge the children of a “hole” into a single child.

Question 7  Continuing with the same tree, what does it look like after removing 12?

Exercise 1  Starting with an empty tree, perform the following operations in sequence.

Insert: 1, 3, 4, 6, 7, 9, 10, 8, 11, 12, 2, 13
Delete: 11, 4, 13, 3