Advanced Programming

Samir Datta[*]

March 15, 2018

Instructions

1. Modify the attached file assignment.3.py as *yourusername.3.py*. For example, I would have used aalok.3.py.
2. The file that you submit should be a modification of the attached assignment.3.py file, i.e, it should contain the classes and methods that have already been defined in this file. You may want to define more, but the given ones must be there.
3. The classes and methods provided will be treated as explained in the problem statements and the comments. You are free to change the names for the arguments, but the code will be tested by running the methods as specified in the comments and the code.
4. The *return* type of each function/method should be as specified in the assignment.3.py file.

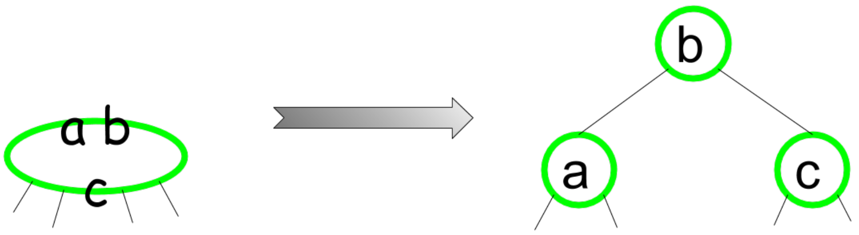
Problem 1

- A 2-3 tree is a kind of Balanced Search Tree. You may refer to this video for visual illustrations of this concept.
- A 2-3 tree has two kinds of nodes:
  - **2-Nodes:** A node which contains 1 key (say a), and has 2 children
    - All the keys in the left subtree of a 2-node are smaller than a
    - All the keys in the right subtree of a 2-node are larger than a
  - **3-Nodes:** A node which contains 2 keys (say $a < b$), and three children
    - All the keys in the left subtree of a 3-node are smaller than a
    - All the keys in the right subtree of a 3-node are larger than b

[*]Teaching Assistants - Kishlaya Jaiswal, Agnishom Chattopadhyay

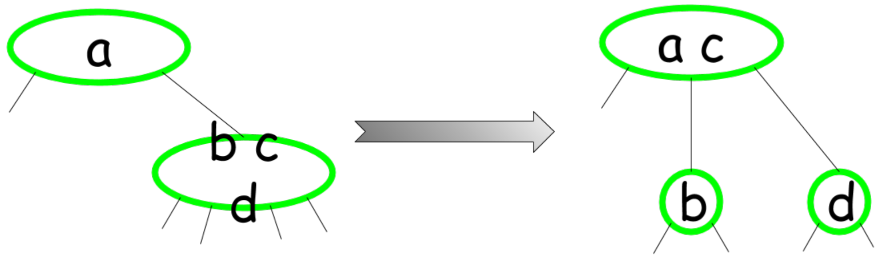
        - All the keys in the middle subtree of a 3-node are strictly between a and b

- A 2-3 tree often has a temporary node called a **4-node**.
    - The 4-node has analogous properties as that of the other nodes.
    - The 4-nodes will show up only during insertions, and we shall get rid of them
- **Search** ing if a value is present should be straightforward.
    - Go left, right, or to the middle depending on how the target value compares with the keys at the current node
- **Insert**ing a value is permissible only at a leaf
    - If the current leaf is a 2-node, just add the value, this would make this a 3-node.
    - If the current leaf is a 3-node, add the value. This would make it a temporary 4-node. We need to **split** this node.
- **Split**ing 4-nodes is the trickiest thing to implement.
    - What if the current 4-node is the root?



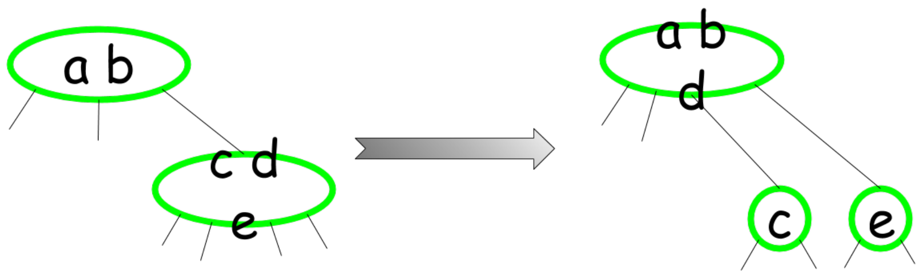
        *

    - What if the current 4-node is the child of a 2-node?



        *

    - What if the current 4-node is the child of a 3-node?



  - *
  - * Recursively split the parent node!

- 2-3 trees have a surprisingly elegant property. **Any** leaf is the same distance away from the root.
  - (Bonus; not graded) Prove that the insertion preserves this property
  - (Bonus; not graded) Thus, show that the tree is not too tall and search and insert works in logarithmic time.
- (Bonus; not graded) Think about how you could do deletion in the 2-3 tree.

Your Task

- In this assignment you shall implement a 2-3 Tree. To do this, you shall implement a class `Node`, with (at least) the following methods:
  - `.split()`
  - `.insert(data)`
  - `.search(data)`
- Read through the comments in the provided template to understand how this is to be done.
- A class `TwoThreeTree` has been implemented for you using the `Node` objects. Make sure you understand how the whole thing fits in.

Two is company,
three's a crowd,
but 2-3 is a tree