Lab 7 – More trees that you ever wanted 🌴🌲🌳

Instructors: Lorenzo De Carli & Maan Khedr

Slides by Lorenzo De Carli

What we are going to focus on today

- Practicing AVL trees

Exercise #1: the importance of Balance [1.5 pts]

- BST performance depends on whether the tree is balanced or not
- In this exercise we will introduce the balance metric

Exercise #1 /2

1. Implement a binary search tree with insertion and search operations as seen in class **[0.2 pts]**
    1. It should extend the template provided on D2L with an `insert()` and a `search()` method
2. Implement code to measure **balance** for each node in the tree **[0.4 pts]**
3. Generate a 1000 random search tasks **[0.3 pts]**
    1. Generate the list of the first 1000 integers
    2. Generate 1000 different tasks by shuffling the list 1000 times
4. For each task, measure average performance (i.e. across searching each integer in the tree) and largest absolute balance value **[0.3 pts]**
5. Generate a scatterplot with absolute balance on the X axis and search time on the Y axis **[0.3 pts]**

Exercise #1 - What to deliver

- Submit a file named ex1.py with your answer to questions 1 to 5

Exercise #2 – baby steps [1.5 pts]

- In this exercise, we'll start turning our BST into an AVL tree

Exercise #2 /2

- Extend the BST implementation from exercise 1 by extending the `insert()` method:

1. Implement code to identify the pivot node on node insertion **[0.3 pts]**
2. Implement code to identify case 1 (pivot does not exist) **[0.4 pts]**
   - In this case, the code should print “Case #1: Pivot not detected”
   - The code should also update node balances
3. Implement code to identify case 2 (pivot exist but node is being inserted into shorter subtree) **[0.6 pts]**
   - In this case, the code should print “Case #2: A pivot exists, and a node was added to the shorter subtree”
   - The code should also update node balances

Exercise #2 /3

4. Implement four test cases. Each test case consists into adding an appropriate sequence of nodes: **[0.2 pts]**
   - Adding a node results in case 1
   - Adding a node results in case 2
   - Adding a node results in case 3 (the code should print “Case 3 not supported”)

Exercise #2 - What to deliver

- Submit a file named ex2.py with your answer to questions 1 to 4

Exercise #3: more AVL trees [2 pts]

- Extend the baby AVL tree implemented in Exercise 3 to support case 3A by:

1. Implementing an internal `_left_rotate()` method to support left rotation of nodes as discussed in class **[0.5 pts]**
2. Implementing an internal `_right_rotate()` method to support right rotation of nodes as discussed in class **[0.5 pts]**
3. Extending the `insert()` method to support case 3a: **[0.8 pts]**
    1. The code should also print “Case #3a: adding a node to an outside subtree”
    2. Remember to update the weights!
4. Extend the test cases from exercise 2 with two new more cases: **[0.2 pts]**
    1. One resulting in case 3a
    2. One resulting in case 3b (the code should print “Case 3b not supported”)

Exercise #3 - What to deliver

- Submit a file named ex3.py with your answer to questions 1 to 4

Exercise #4: a fully grown AVL tree [2 pts]

- Turn the AVL tree of Exercise 3 into a fully grown AVL tree by supporting case 3b:

1. Create a `_lr_rotate()` method to perform LR rotation **[0.5 pts]**
2. Create a `_rl_rotate()` method to implement RL rotation **[0.5 pts]**
3. Extend `insert()` to support case 3b, using the methods above **[0.8 pts]**
4. Create two extra test cases, both testing case 3b (now the code should not return an error!) **[0.2 pts]**

Exercise #4 - What to deliver

- Submit a file named ex4.py with your answer to questions 1 to 4

How to submit

- Upload a zip file to the “Lab 7” dropbox on D2L, containing the required content for every exercise

Grading rubric

- You get **3 pts** for uploading a **partial solution** by **end of lab**
  - **Must not be an empty file or irrelevant material**
- Then, you'll have until **11:59PM of the day before the next lab** to upload the **complete solution**. That will be graded as follows:
  - Exercises 1, 2: 1.5 pts each
  - Exercises 3, 4: 2 pts each
  - **Can upload the complete solution to the same dropbox**

That’s all folks!