Student: PinHo Wang
Instructor: Prof. Robin Hillyard
NUID: 001443435

INFO 6205 Program Structure & Algorithms

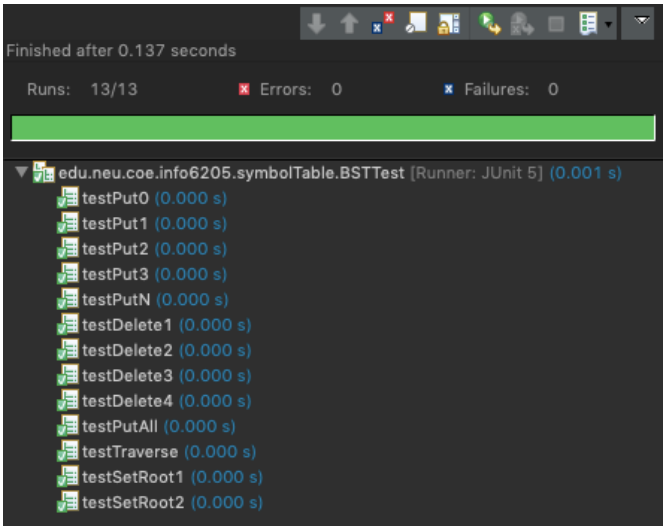
Summer Fall 2018

Assignment 5

Files Description:

+- PinHo_Wang_assignment5
    +- Report
    +- src
        +- main
        | +- edu.neu.coe.info6205.bqs
        | +- Element.java (default)
        | +- Queue_Elements.java (default)
        | +- Queue.java (default)
        | +- edu.neu.coe.info6205.symbolTable
        | +- BST.java (default)
        | +- BSTdetail.java (default)
        | +- BSTSimple.java: Implement getMaxHeight() method to get the
        | maximum of tree height
        | +- Driver.java: main function
        | +- Experiment.java: Implement the experiment method including
        | modify “magic” numbers
        | +- edu.neu.coe.info6205.util
        | +- PrivateMethodTester.java (default)
        | +- PrivateMethodTesterTest.java (default)
        +- test
        | +- edu.neu.coe.info6205.symbolTable
        | +- BSTTest.java (default)

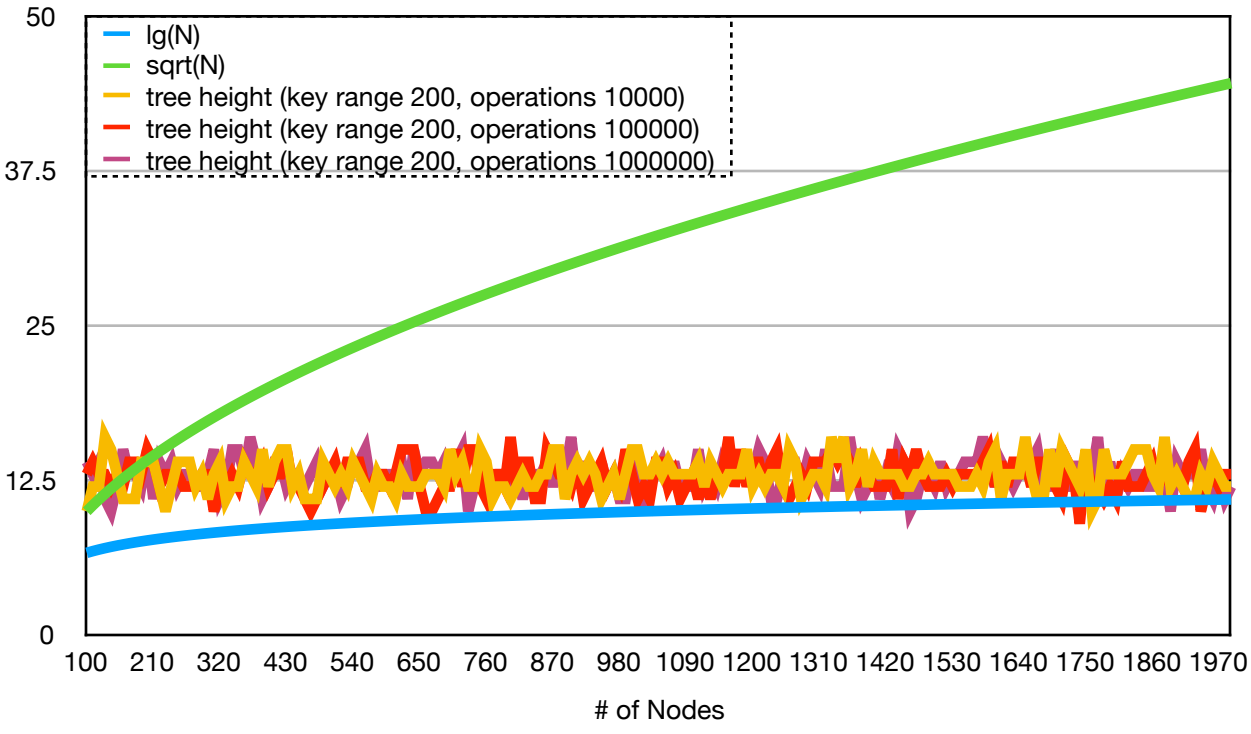
Test Cases:



13 test cases for testing BSTSimple.java

Experiment:

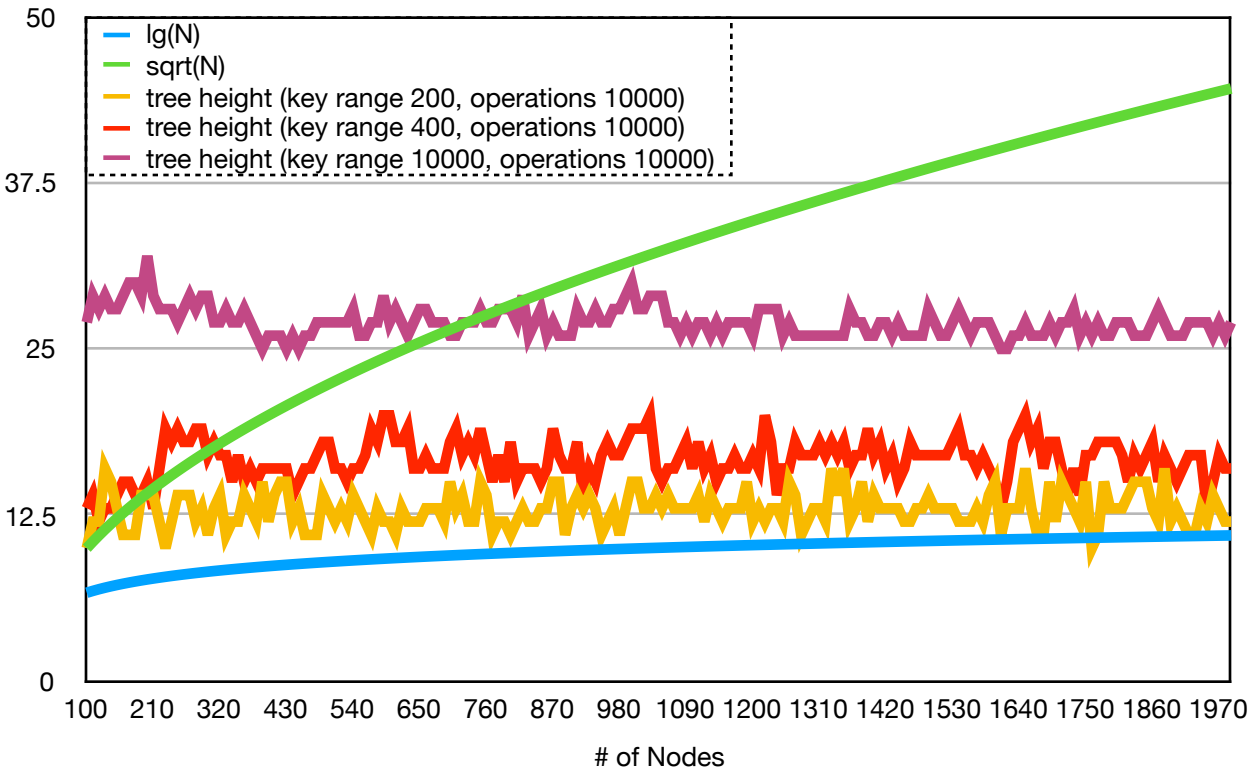
Fig. 1



First of all, I set key range with 200 and test the effect of insertions/deletions operations, increasing by multiplying 10, and the nodes number from 100 to 2000. The result shows in Fig. 1, including lg(N), sqrt(N) and the tree height, it shows that **the order corresponding to lg(N) with different operations**.

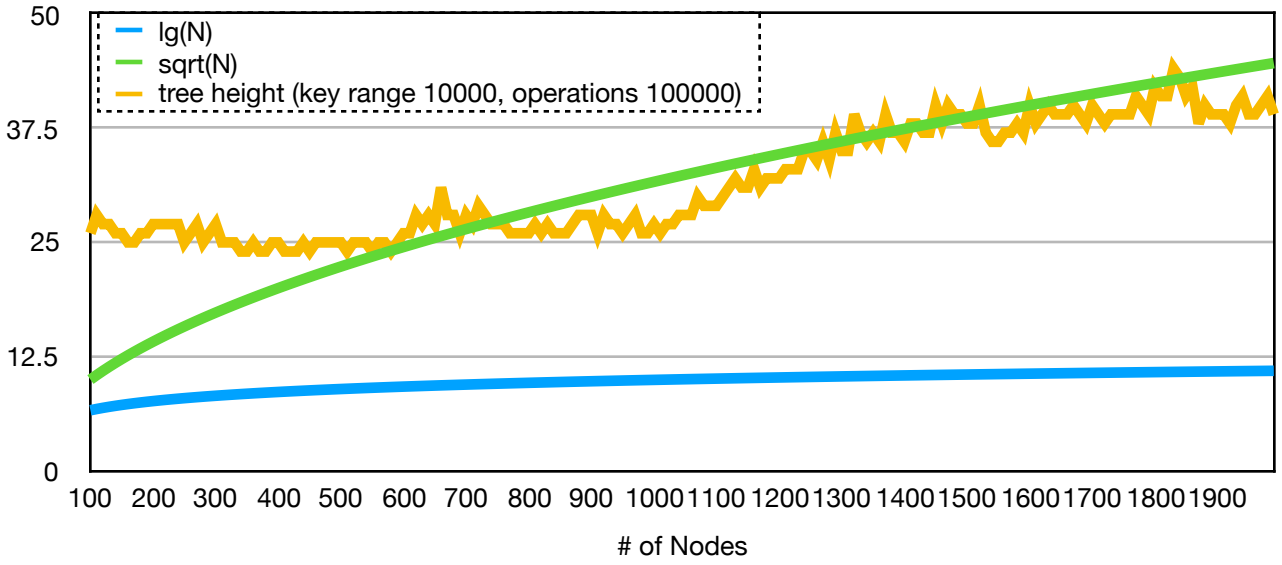
Then, in contrast, I set number of operations to 10000 and change key range from 200 to 1000. **The order seems still corresponding to lg(N), but while increasing operations, the tree height increasing**, as the result shows in Fig. 2.

Fig. 2



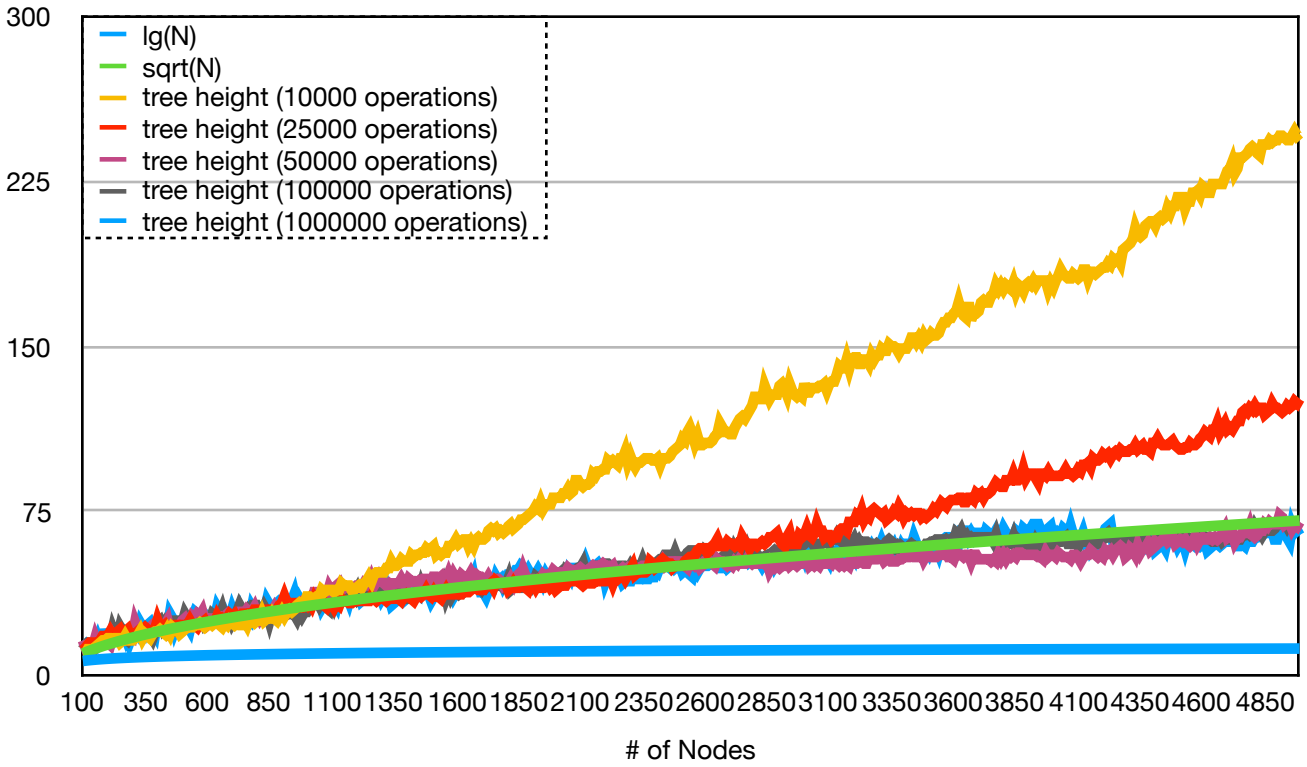
Then, I assume that if key range and operations are too small, the order of the result will correspond to lg(N). So, I increasing both key range and operations, where key range is 10000 and operations is 100000. Turn out that **the order seems corresponding to lg(N) while the node number is small, and the order change to correspond sqrt(N) while node number increasing**, , as the result shows in Fig. 3.

Fig. 3



Finally, I let the key range increasing with the node numbers, multiplying by 2, and set operations with 10000, 25000, 50000, 100000, and 1000000, as the result shows in Fig. 4. As you can see, when the operations is 10000, yellow line, the order corresponds sqrt(N) in small node number region, but it becomes worse while node number increasing. **In more operations cases, the order becomes to sqrt(N) in large node number**.

Fig. 4



Conclusion:

- While key range and times of insertions/deletions operations are small, the order corresponds to lg(N)
- The order changes from lg(N) to sqrt(N) while the time of operations is larger, if and only if, the key range can hold it.
- While the node number becomes larger, it needs more operations and key range to maintain sqrt(N) order, or it becomes worse time complexity (tree height).