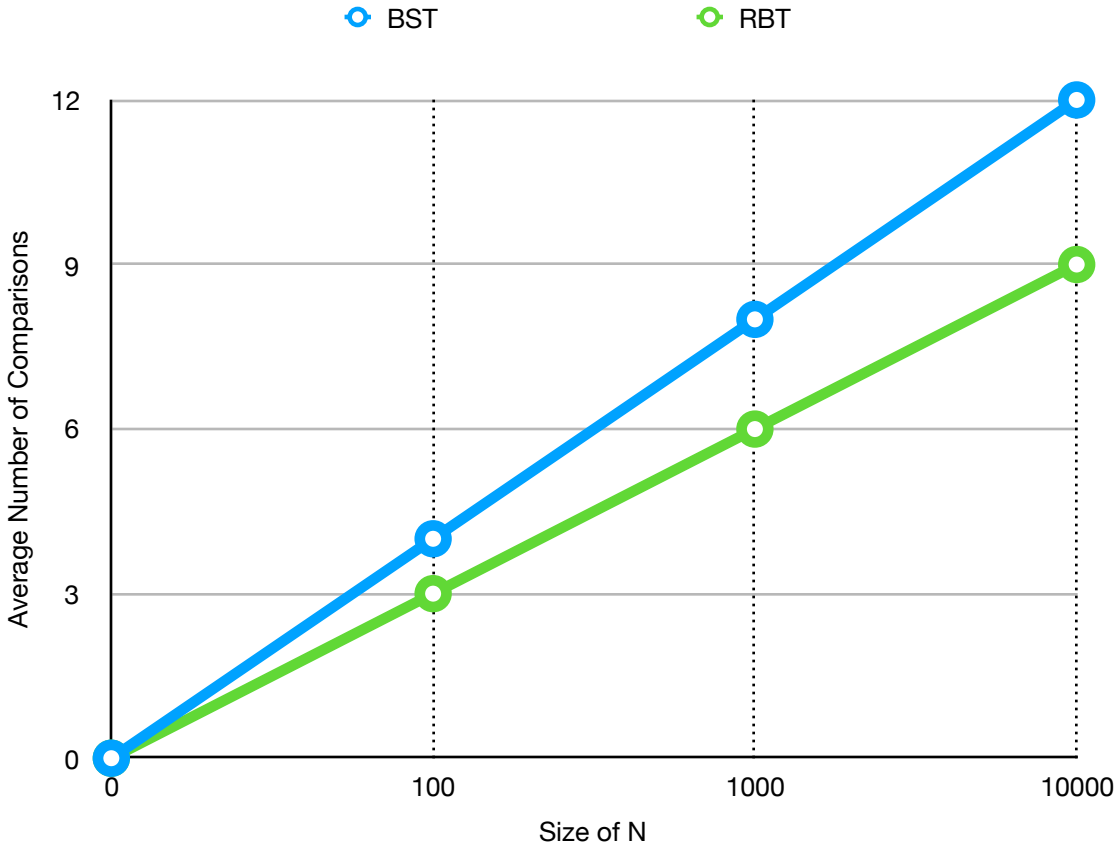
Parankush Bhardwaj

Algorithms Workshop 8

1. Code on separate .java files
2. Code on separate .java files
3. Graph below:
4. Graph also below (shared with BST, #3):



5. The rate of growth for RBT is clearly 3 * log(N) and for BST is 4 * log(N).

First, we know that log(100) = 1, log(1000) = 2, and log(10000) = 3.

Second, we know that, specifically for RBT, the average number of comparisons for a input size of N = 100 is 3.

Further, we know that 3 * log(100) = 3 (by the rule of Algebra). Note that 3 is also the average number of comparisons for 100 inputs. Thus, the average number of comparisons, denoted as Y(N), is equal to 3 * log(N). The same logic can be applied for all sizes of N.

Similarly, for RBT, log(1000) = 2, and 3 * log(1000) = 6, which is the average number of comparisons for a input size where N = 1000.

The same can be said for BST. As its input size increases by log(N) + 1, its output keeps increasing linearly at a rate of about 4 * log(N). In BST, 4 * log(100) = 4, which is the average number of comparisons for a input size of n = 100. Similarly, 4 *log(1000) = 8, which again is

the average number of comparisons for a input size of n = 1000. The same can be said for the case of N = 10000.

6.

1. On more random input, the results would likely of been slightly slower. We can see this from the lecture slides 41 and 42, which shows that with random inputs, trees end up relatively less balanced than if it were given a ordered list of inputs. Thus, if we dealt with inputs that were even more random than the ones tested, we can expect the tree to be slightly more unbalanced, leading to more comparisons, and thus higher search times.

2. The results find that the maximum depth of the binary search tree is larger than that of the Red black tree. We know this to be true because under the BST, long chains of nodes can sometimes form when dealing with insertions, leading to not only very deep chains but also a search that could take closer to linear time. With RBT, however, since they are self-balancing, it always guarantees a logarithmic search time, which means its depth will be usually always be relatively shorter than that of BST's.