Caitlin Coggins

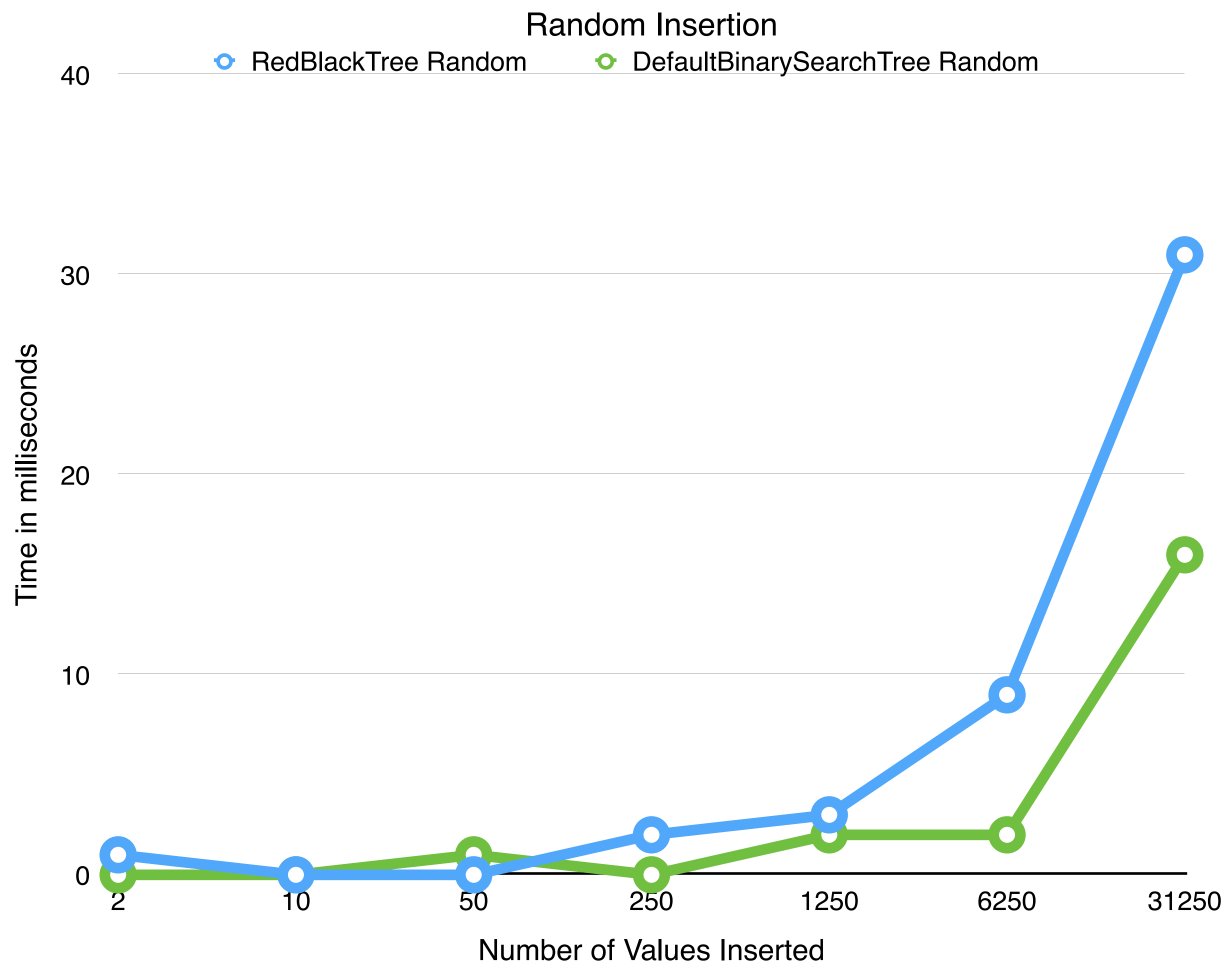
Random Values

Number of Values Inserted	RedBlackTree Insertion Time (milliseconds)	DefaultBinarySearchTree Insertion Time (milliseconds)	RedBlackTree Search Time (milliseconds)	DefaultBinarySearchTree Search Time (milliseconds)
2	1	0	0	0
10	0	0	0	0
50	0	1	0	0
250	2	0	0	0
1250	3	2	0	0
6250	9	2	0	0
31250	31	16	0	0

Sorted Values

Number of Values Inserted	RedBlackTree Insertion Time (milliseconds)	DefaultBinarySearchTree Insertion Time (milliseconds)	RedBlackTree Search Time (milliseconds)	DefaultBinarySearchTree Search Time (milliseconds)
2	0	0	0	0
10	0	0	0	0
50	0	0	0	0
250	0	0	0	0
1250	1	3	0	1
6250	1	133	0	1
31250	12	stalled so long it broke (took at least 10 seconds)	n/a	n/a

Caitlin Coggins



A RedBlackTree may be slower when it comes to insertion with random data sets, but it has many benefits in the long run over BinarySearchTrees. When I tested the trees with random data sets, RedBlackTrees were somewhat slower. Though the bigO notation of the insertion itself is equivalent, O(log n), the RedBlackTree also had to re-sort itself to follow its rules. This means it ends up taking slightly longer to insert. However, when inserting an already-sorted data set, the RedBlackTree was much faster. Since the RedBlackTree balances itself, there cannot be a case where every node is on one side of the tree. This proved detrimental to the BinarySearchTree. While the RedBlackTree took 13 milliseconds to insert 31250 integers, the BinarySearchTree took so long and had to insert so far down one side of the tree that it StackOverflowed. In this case, the bigO notation of the RedBlackTree stayed O(log n) plus a little for restructuring, and the BinarySearchTree was O(n). This takes much longer to do.

In terns of searching, both the RedBlackTree and the BinarySearchTree were extremely fast. With a random data set, they both take O(log n), so they both took the same amount of time. With a sorted set, the BinarySearchTree began to take a little longer to find the value, but it would be hardly noticeable. However, with larger and larger amounts of data, it will take longer and longer to find the value. This is another benefit of the RedBlackTree: searching cannot much take longer than O(log n) because the tree cannot become horribly unbalanced.

Overall, even though the RedBlackTree may take longer to insert random data sets, there are enough benefits to them when inserting larger amounts of data to use them over BinarySearchTrees.