Matthew Moore

Dr. Bowers

CPSC 223

12/15/20

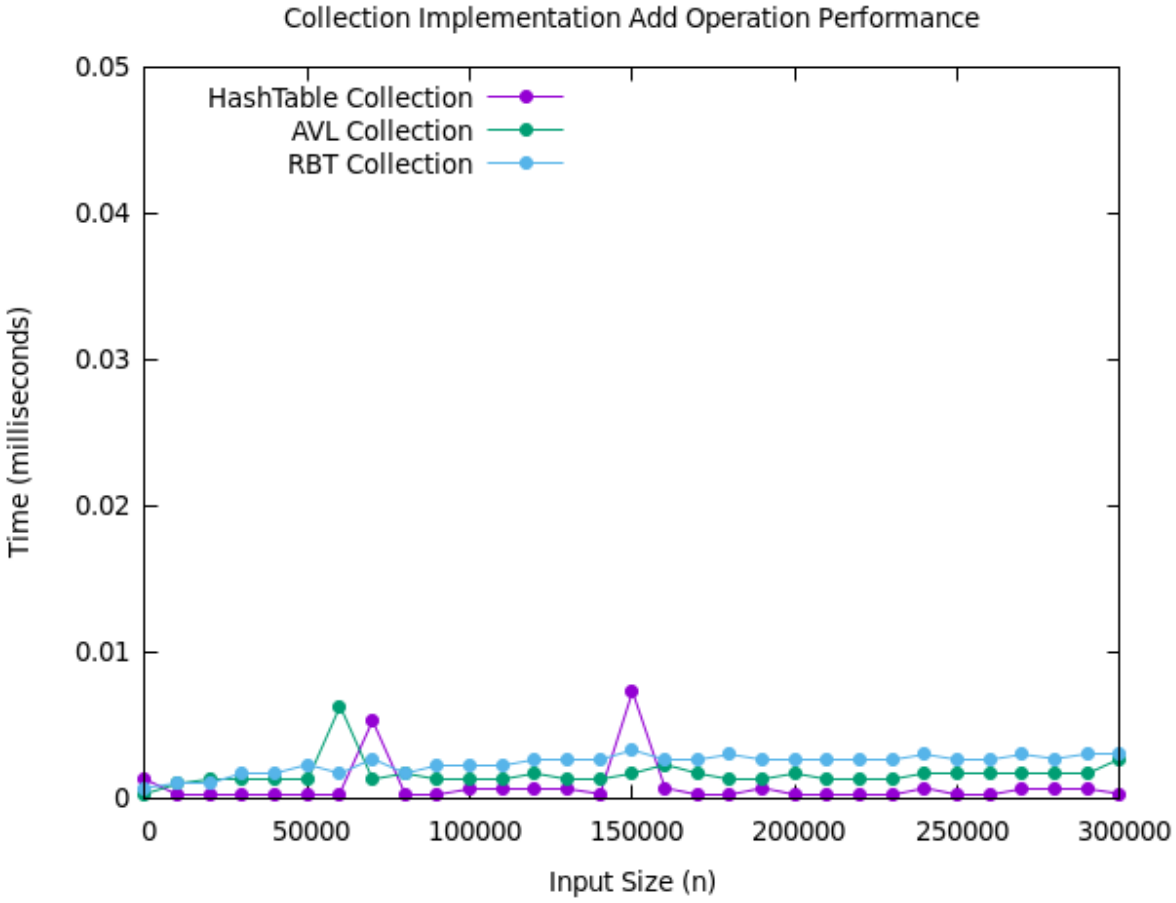
Hw9-graphanalysis

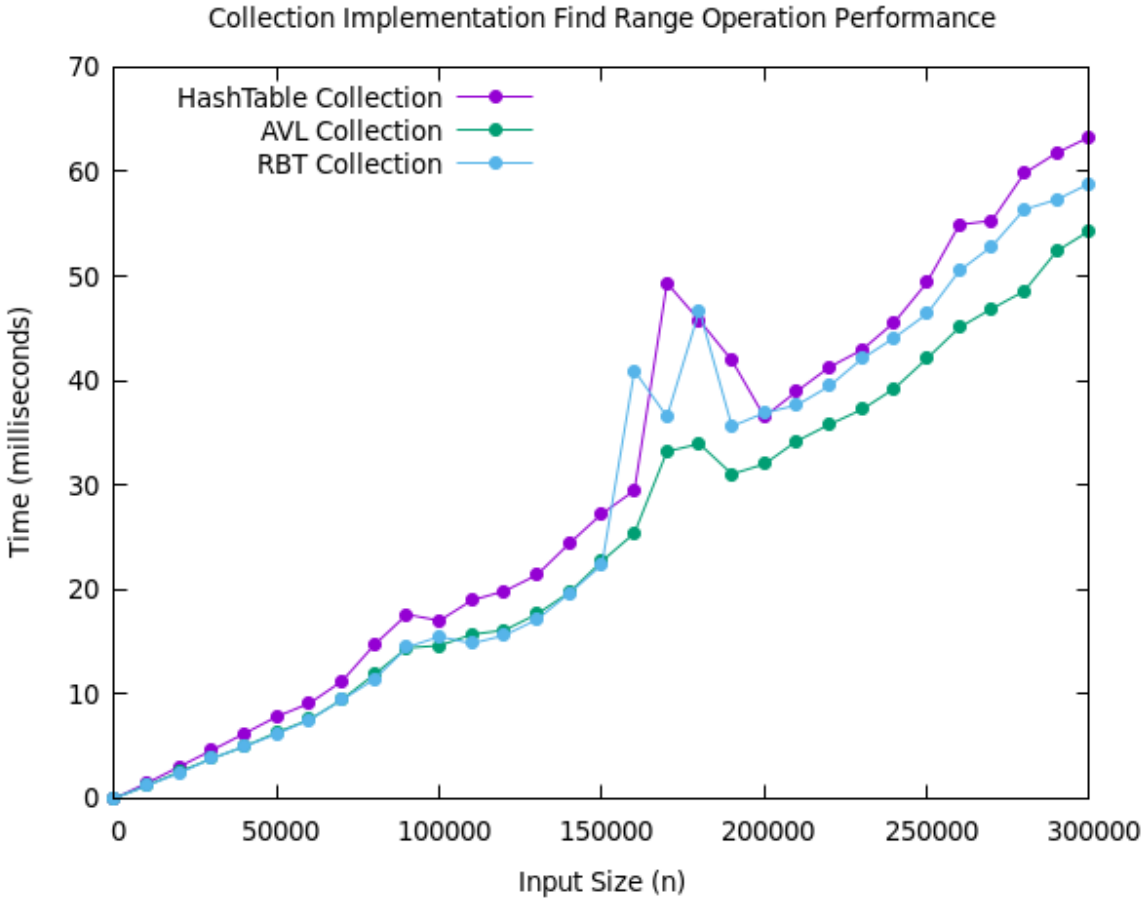
Overall Analysis:

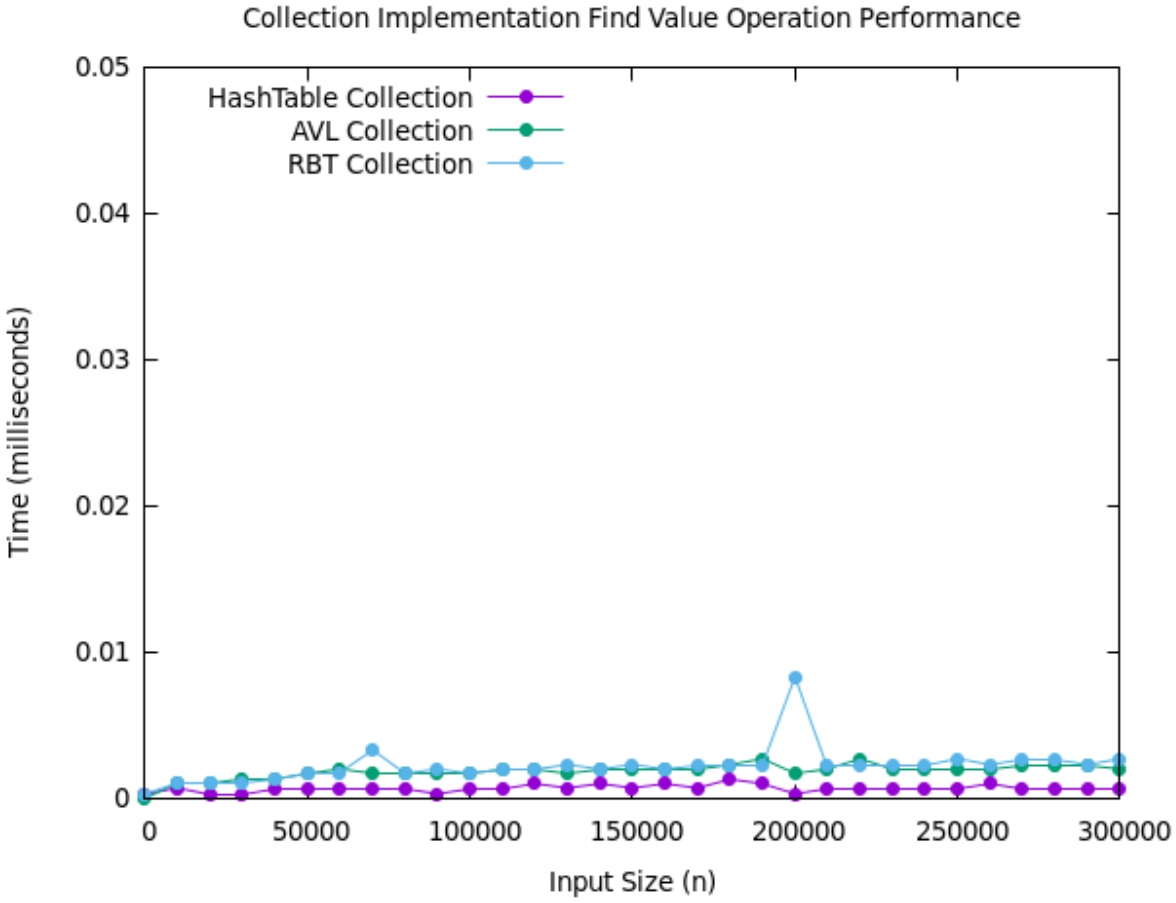
Overall, we can see that the AVL tree shows to be the most efficient collection data structure out of the three options displayed in the chart. This makes sense because as we discussed in class, the RBT Collection should be the most efficient option, yet designing the implementation that would be more efficient than an AVL structure would be quiet a difficult task to accomplish. So, in our case the AVL structure is going to win in just about every operation. Meanwhile, the HashTable Collection makes a good run for the efficiency with being the most efficient for remove, add, and find value. We know that it will always be lagging when it comes to the sorting operation.

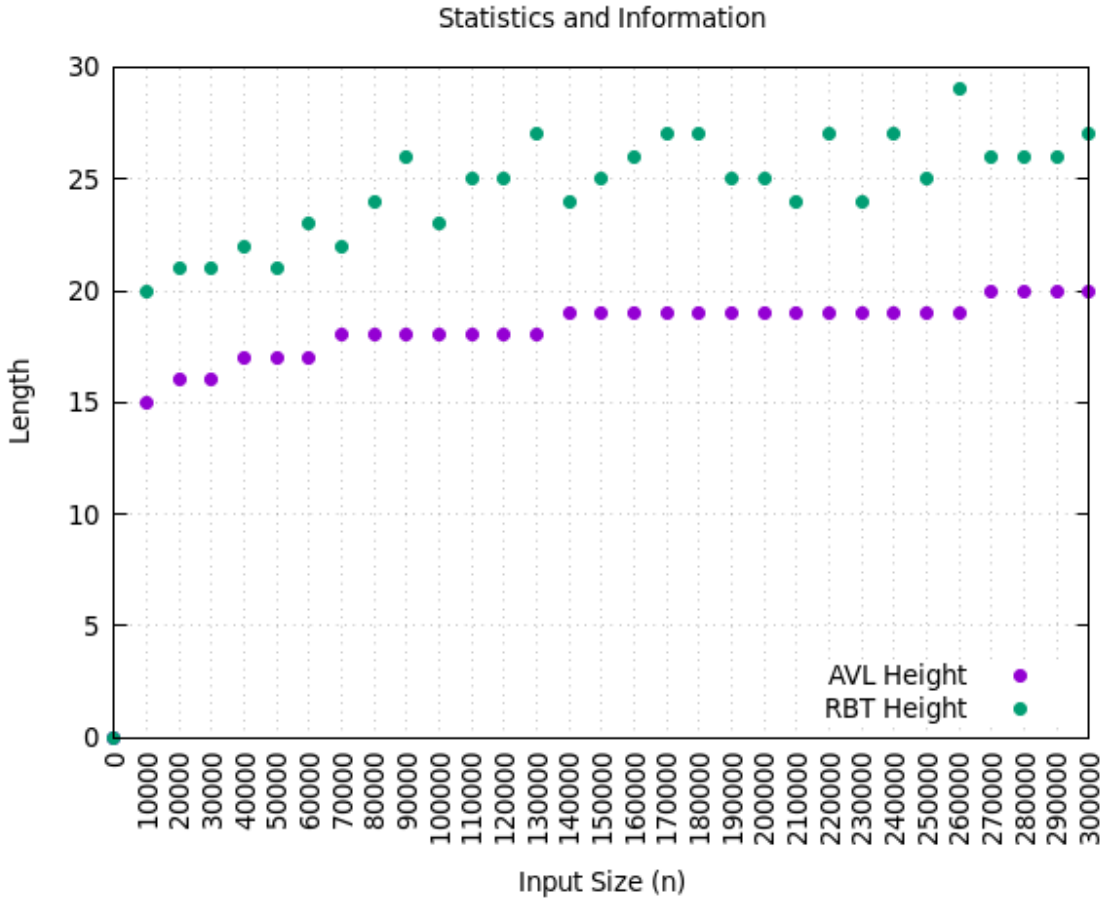
In terms of the complexity of all of these we know that the RBT is pretty much identical to the AVL Collection with log n defining much of its operations as we will see in the table depicted.

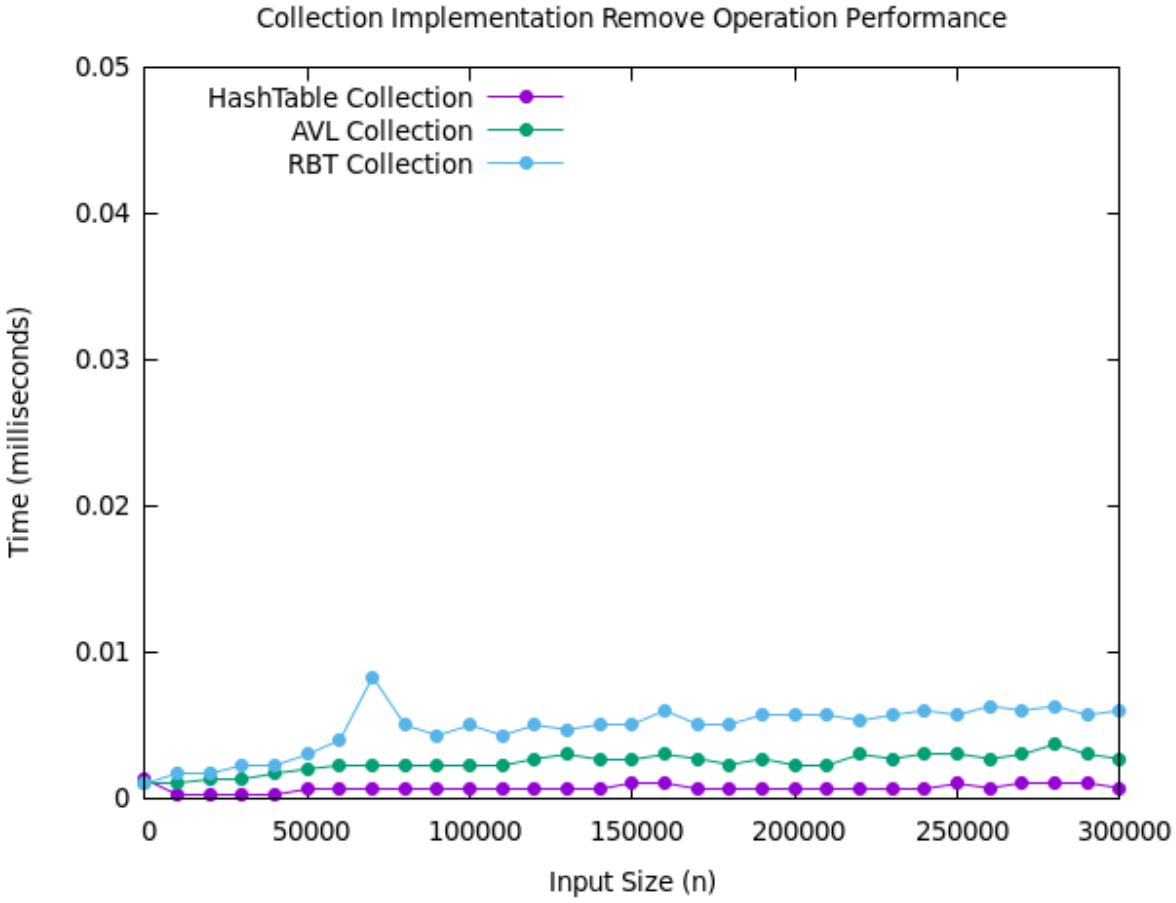
Functions	**Array List**	**LinkedList**	**Sorted Array**	**Hash Table**	**AVL**	**BST**	**RBT**
Add	O(1)	O(1)	O(n)	O(1)	O(log n)	O(n)	O(log n)
Remove	O(n)	O(n^2)	O(n)	O(1)	O(log n)	O(n)	O(log n)
Find Val	O(n)	O(n^2)	O(log n)	O(1)	O(log n)	O(n)	O(log n)
Find Range	O(n)	O(n^2)	O(n)	O(n)	O(n)	O(n)	O(n)
Sort	O(n^2)	O(n^2)	O(n)	O(n^2)	O(n)	O(n)	O(n)

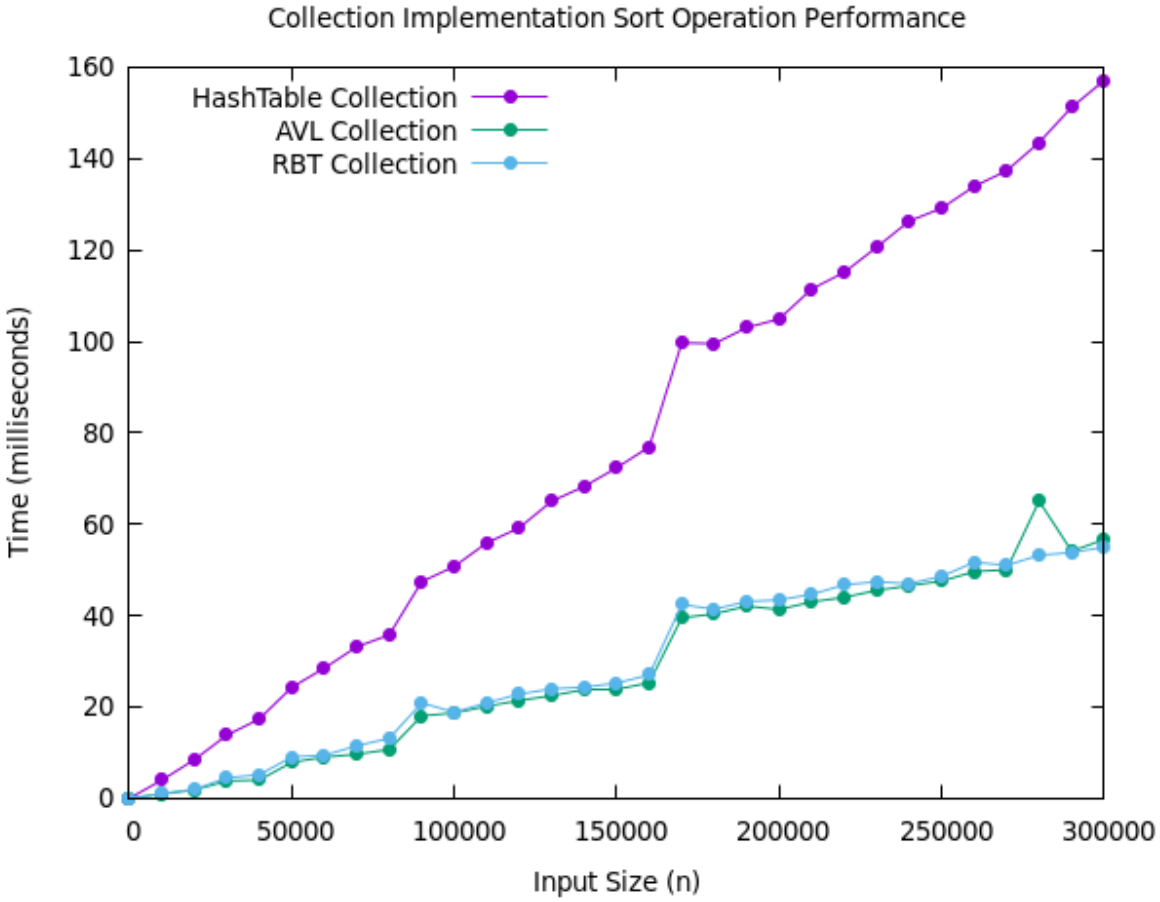

Collection Implementation Add Operation Performance
HashTable Collection
AVL Collection
RBT Collection
Time (milliseconds)
Input Size (n)
0.05
0.04
0.03
0.02
0.01
0
0
50000
100000
150000
200000
250000
300000


Collection Implementation Find Range Operation Performance
HashTable Collection
AVL Collection
RBT Collection
Time (milliseconds)
Input Size (n)
0
10
20
30
40
50
60
70
0
50000
100000
150000
200000
250000
300000


Collection Implementation Find Value Operation Performance
HashTable Collection
AVL Collection
RBT Collection
Time (milliseconds)
Input Size (n)
0.05
0.04
0.03
0.02
0.01
0
0
50000
100000
150000
200000
250000
300000


Statistics and Information
Length
Input Size (n)
AVL Height
RBT Height
0
5
10
15
20
25
30
0
10000
20000
30000
40000
50000
60000
70000
80000
90000
100000
110000
120000
130000
140000
150000
160000
170000
180000
190000
200000
210000
220000
230000
240000
250000
260000
270000
280000
290000
300000


Collection Implementation Remove Operation Performance
HashTable Collection
AVL Collection
RBT Collection
Time (milliseconds)
Input Size (n)
0.05
0.04
0.03
0.02
0.01
0
0
50000
100000
150000
200000
250000
300000


Collection Implementation Sort Operation Performance
HashTable Collection
AVL Collection
RBT Collection
Time (milliseconds)
Input Size (n)
0
20
40
60
80
100
120
140
160
0
50000
100000
150000
200000
250000
300000