Osama Kergaye
CS 2420
9/14/2016
Dr. Meyer

Assignment 03 Analysis Document

1. Who is your programming partner? Which of you submitted the source code of your program?

My programming partner was Sam Bridge and I (Osama Kergaye) submitted the code.

2. How often did you and your programming partner switch roles? Would you have preferred to switch less/more often? Why or why not?

We didn't switch too often. I don't think it was crazy relevant who had whatever role. Reason being, whatever needed to be written in the code could just be relayed by the partner. So for most of the assignment I was behind the keyboard and unless I if I wasn't able to understand what Sam was telling me to code, we would simply switch.

3. Evaluate your programming partner. Do you plan to work with this person again?

Sam was an excellent programming partner, and I absolutely plan to work with him again.

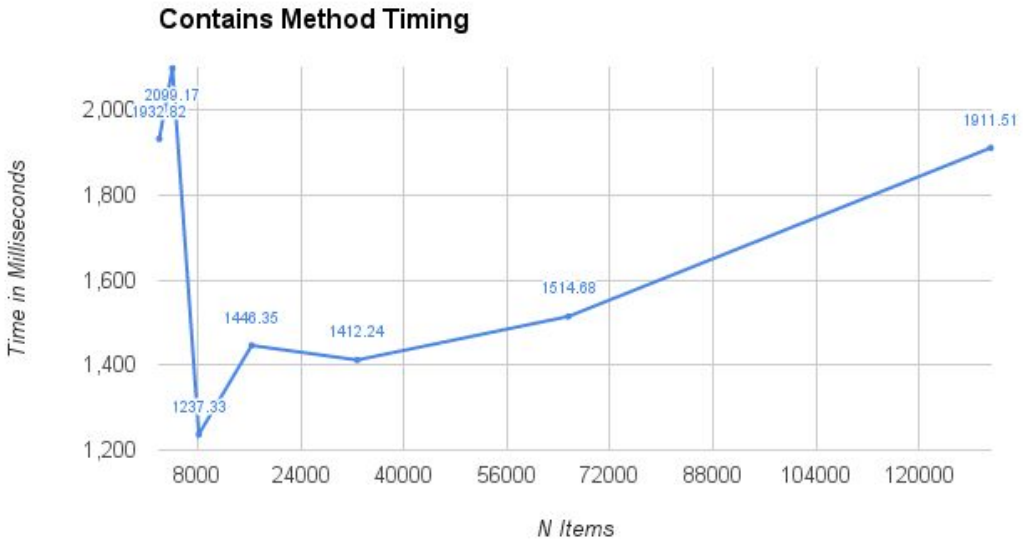
4. If you had backed the sorted set with a Java List instead of a basic array, summarize the main points in which your implementation would have differed. Do you expect that using a Java List would have more or less efficient and why? (Consider efficiency both in running time and in program development time.)

Well, a bunch of my methods would be simpler to write because I could just use Java List's methods instead of making my own. I could have called Java List's methods for each of the methods we were told to implement, including its binary sort. In the case of running time, it would be a bit longer, because I would have 2 layers of methods to go through before my code executes, for instance, if I called BinarySearchSets contains method, it would then call Java List's contains method. To me, this seems very inefficient. As for programming development time, it would have been much quicker, because I wouldn't have had to use my brain at all. All that would be necessary is to keep using Javas List's methods for everything.

5. What do you expect the Big-O behavior of BinarySearchSet's contains the method to be and why?

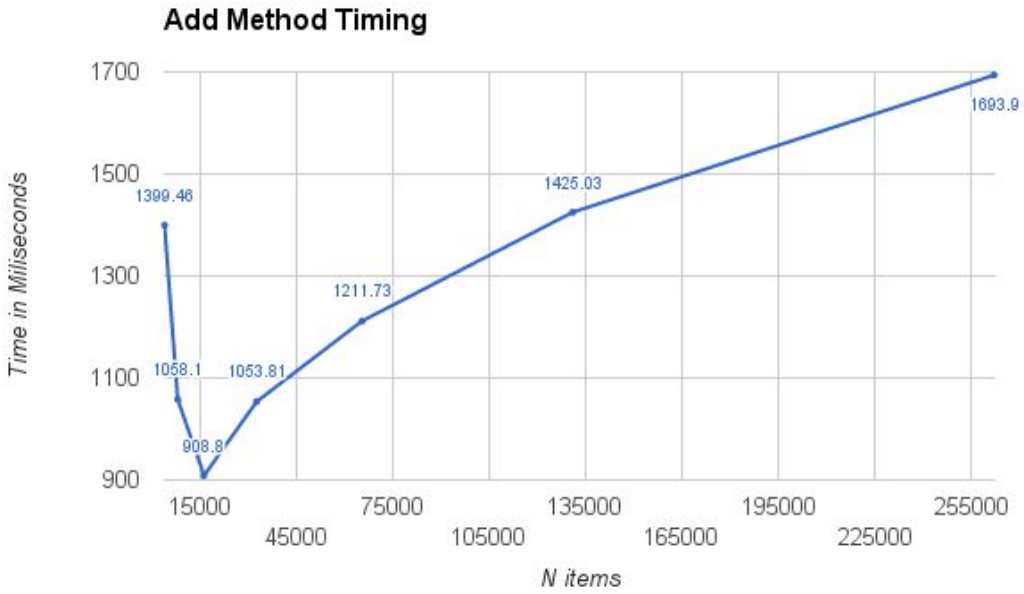
I expect a Big-O behavior of O(logN) because the contains method implements a binary search which has a complexity time of O(logN). Another reason is because we are always halving the set size, which leads to a logarithmic pattern mathematically.

6. Plot the running time of BinarySearchSet's contains method, using the timing techniques demonstrated in Lab 2. Be sure to use a decent iteration count to get a reasonable average of running times. Include your plot in your analysis document. Does the growth rate of these running times match the Big-oh behavior you predicted in question 5?



The growth rate in the run times for the contains method match my prediction for an O(logN) growth behavior. Also, the first item in my contains method timing was omitted to create a clearer chart.

7. Consider your add method. For an element not already contained in the set, how long does it take to locate the correct position at which to insert the element? Create a plot of running times. Pay close attention to the problem size for which you are collecting running times. Beware that if you simply add N items, the size of the sorted set is always changing. A good strategy is to fill a sorted set with N items and time how long it takes to add one additional item. To do this repeatedly (i.e., iteration count), remove the item and add it again, being careful not to include the time required to call remove() in your total. In the worst-case, how much time does it take to locate the position to add an element (give your answer using Big-oh)?



For an element not in the set, it takes about O(logN) amount of time to locate the position it should be inserted at. In the worst case, it still would take O(logN) time. Also, the first two timed tests in my add timing experiment were removed to create a clearer chart.

8. How many hours did you spend on this assignment?

It took about 30-40 hours to finish the assignment.