Programming Assignment #4

TEAM PROGRAMMING PROJECT IMPLEMENTING

SORTING AND FUNCTION POINTERS

By

David Garcia

&

Michael Fort

Regis University - CS372

1/15/2015

1 PROGRAM BREAK DOWN

This program demonstrates 4 different sorting algorithms:

- Bubble sort
- Insertion sort
- Merge sort
- Quick sort

The program starts off by getting input from the user on which two types of sorts to run and how many times to run them. An array of 100,000 integers is then created which will then be used by the sorting functions the user has selected. After each sort function is run, the program displays the amount of time it took to run each sort type followed by the average time. The program will continue to go through this process until the user chooses to exit the program.

2 ANALYSIS

The testing of this program was implemented by running each sorting function ten times each. As you can see in table 2.1 below, bubble sort was by far the slowest of the four sorting algorithms. Bubble sort on average was 6 times slower than insertion, about 1700 times slower than merge and about 2300 times slower than quick sort. Insertion sort came in on average as the 3rd slowest being 274 times slower than merge and 382 times slower than quick sort. Merge and quick sort were the fastest of the four algorithms but merge sort fell behind quick sort about 28% on average. However, when looking at each test individually merge sort sorted just

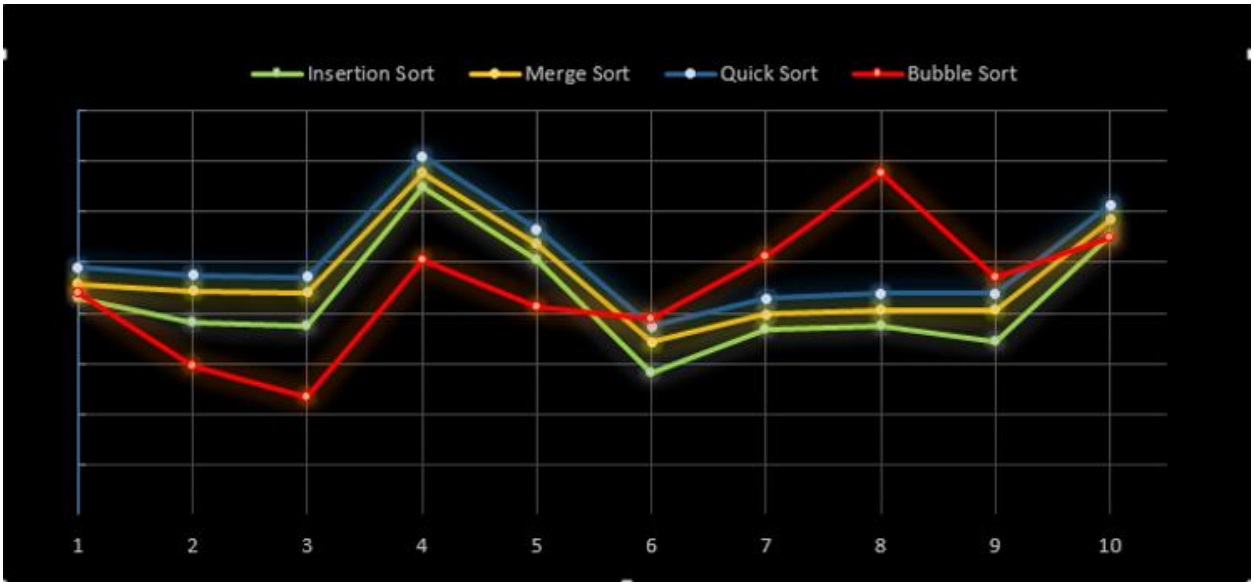
as fast or ±1 clock tick than quick sort 6 out of 10 times. If you take a look at table 2.2 you can see how each sort performed on each set of random numbers per test. From 2.2's stacked line graph you can see that all the tests handle the random numbers the same except for test 6 – 9. Bubble sorts line peaked indicating it had to take more time to sort through the random numbers that were generated in tests 6 – 9. Again, bubble sort on average was the least efficient algorithm.

2.1 PC Clock Tick Sort Times

Test #	Bubble Sort	Insertion Sort	Merge Sort	Quick Sort
1	36482	5964	15	16
2	36192	5941	31	15
3	36065	5938	32	16
4	36612	6074	15	16
5	36427	6002	16	15
6	36378	5890	31	16
7	36626	5934	15	15
8	36952	5937	16	16
9	36542	5922	31	16
10	36698	6026	16	15

Program was tested on a Dell XPS 8700 running Windows 8.1 with an Intel i7-4790 CPU & 12 GB RAM

2.2 SORT PERFORMANCE PER TEST



Program was tested on a Dell XPS 8700 running Windows 8.1 with an Intel i7-4790 CPU & 12 GB RAM

3 CONCLUSION

From the data gathered we can conclude that bubble and insertion sort (bubble sort being the very least) are the least efficient of the four algorithms tested. Both bubble and insertion sort have an average efficiency of $O(n^2)$ but as n grows, insertion becomes the better of the two. Merge and quick sort were the fastest of the sorts due the fact they use a “divide and conquer” approach which splits the problem at hand into manageable sort sizes. Both merge and quick sort carry a very good average efficiency of O(n lon n) which shows in this test to be correct.