Name: Yunfei Li

Machine: 2013 Late MacBook Pro

Processor: 2.3 GHz Intel Core i7

RAM: 16GB 1600 MHz DDR3

Term Project Bubble Sort vs. Selection Sort

A lot of programmers that make some of the coolest and most useful software today, such as many of the stuff we see on the Internet or use daily, don’t have a theoretical computer science background. They’re still pretty awesome and creative programmers and we thank them for what they did.

However, theoretical computer science has its uses and applications and can turn out to be quite practice. In this report, targeting at programmers who know their art but who don’t have any theoretical computer science background, I will present one of the most pragmatic tolls of computer science: Big O notation and algorithm complexity analysis. As someone who has worked both in computer science academic setting and in building low-level software in the industry, this is the tool I found to be one of the truly useful ones in practice, so I hope reading this report you can apply it in your own code to make it better. After reading this report, you’ll have a solid understanding of two major sorting techniques in practice. Bubble sort and selection sort might not be the best sorting algorithms but they are very essential in understanding how things are sorted.

In this report I have tested both bubble sort and selection sort and their sorting times in which is attached in the dot java file. As the numbers may vary due to the random number generating for each time the method is called. However, after multiple testing, it has been seen that selection sort will always be better than bubble sort in all cases (result have been recorded down in the excel file). In the graph provided in the excel file, we can see that as number of n (number of elements in the array) increases, selection sort have a slower processing time than bubble sort even at the slightest n. Another graph shows that as num\_i(number of iterations) increases, selection sort still out performs bubble sort in call cases.

Even thought both selection sort and bubble sort have a Big O notation of n^2, there’s something called the “worst-case”, which means if bubble sort has a reverse sorting elements in the array, it’ll take much longer than selection sort. One of the big advantage that bubble sort has over selection sort is that in the “best-case”, which means the array is sorted or almost sorted, bubble sort has a build in method to know when the list is almost sorted making it more efficient than selection sort. In the “average-case”, which is taking the best case and worst case and divide it by 2, selection sort will always out perform bubble sort. In the excel file attached, I notice that no matter how small the array is, selection sort is out performing bubble sort.

Even though bubble sort and selection sort are not the best way to sort items, it is still key to understand the fundamentals of Big O notation and the time it takes for each algorithm to sort. In this report, we have testing values using n = 50, 250, 500 and num\_i = 100, 1000, 10000, and in all cases selection sort out performs bubble sort. What I have learned through out this project is that there are comparisons of worst-case, best-case, and average-case scenario but taking the leading coefficient is the Big O notation, and how to calculate worst-case, best-case, and average-case scenarios.