Grant McGovern

Dr. Pauca

CSC 222

March 2, 2015

Lab 2 Write-Up

Perhaps some of the most historical sorting algorithms in the history of computer science, Quick Sort, Merge Sort, and Selection Sort have served as benchmark algorithms for understanding run time complexities within the backdrop of large data sets. While basic sorting algorithms like Bubble Sort do exist, their complexity is inferior to the above sorting algorithms that leverage powerful techniques such as divide and conquer steps. In this lab, I will implement and examine the above sorting algorithms and test them against different data sets. The first algorithm is Selection Sort whose worst case time complexity is O(n2). The second algorithm I will implement and examine is Merge Sort whose worst case time complexity is O(nlogn). Lastly, the final sorting algorithm I will implement and examine is Quick Sort, whose worst-case time complexity is also O(n2). For the purpose of this lab, all of the above algorithms will be implemented in the C++ programming language. The dataset I will be using is a 235,886 thousand-line file containing random words. Each entry in the file is delimited with a new line character. Each sorting algorithm is constructed via class and a public *sort* method which can be called from the main program. All of the files are linked against a *makefile*, which upon execution, runs the program and outputs the results to a text file. This text file contains information regarding the size of the input data, as well as the time taken by each algorithm to sort it. In this lab, we will do analysis on each algorithm by examining these results and ultimately plotting them side-by-side on a graph to visually see their complexities.