Report on Sorting Algorithms

Rachel Ross

12.14.18

In this program, 4 sorting algorithms were tested: being insertion sort, quick sort, merge sort, and bubble sort. The time was measured using the ¡time.h¿ header file, and measured in clock ticks per second. The file contains about 100,000 doubles, which is a data set size less favorable for 2 of the algorithms. I was not surprised at the contrast in runtimes for each algorithm, but quick sort exceeded my expectations in terms of efficiency. Since C++ as a programming language is not as high level in comparison to other common ones, it required more re-compiling when coding.

The results showed that quick sort was the most efficient in this specific case, taking 15,115 clock ticks per second to sort the list of doubles. Merge sort was the next fastest (2,679,131 clock ticks per second), followed by insertion sort (12,446,274 clock ticks per second) then bubble sort (32,370,426 clock ticks per second). Both quick sort and merge sort are algorithms that use recursion, each with an average Big-Oh runtime of O(n logn). Bubble sort, the least efficient of the sorting algorithms, has a quadratic Big-Oh runtime on average. Having previous knowledge about Big-Oh runtimes of these sorting algorithms, these algorithms confirmed what my expectations were.

The shortcomings of performing this specific empirical analysis is time efficiency, and also that limited information can be gathered in case one wanted to look for a more detailed analysis. Measuring the runtimes only takes into account a limited number of factors. Creating a program to test these algorithms was also more time consuming than analyzing these algorithms mathematically, and that would likely be inconvenient on a larger scale.