Chase Mitchell

05/13/2018

CPSC-350-01

Assignment 6 – Sorting Algorithms – Report

For the given assignment, 4 sorting algorithms were implemented, executed, and compared on an array of double values of variable length and value retrieved from a chosen formatted file input. The 4 algorithms chosen were Quick Sort, Insertion Sort, Bubble Sort, and Pancake Sort. The arrays passed through each algorithm were identical, and timing for each was conducted using the Chronos class with units of microseconds. The comparison was executed with three sample files of 500, 2500, and 5000 length respectfully and executed 10 times for comprehensiveness and variance, the results are shown below:

Sorting Algorithm	**500**	**2500**	**5000**
Quick	184-251 μs	1072-1202 μs	2166-2304 μs
Insert	664-755 μs	16194-18340 μs	57454-61872 μs
Bubble	3047-4584 μs	73812-78723 μs	264759-286848 μs
Pancake	2331-2915 μs	48112-53085 μs	180006-196258 μs

As expected, quick sort performed the best over the other three algorithms, even more so in the higher length arrays; this further reinforces the effectiveness of divide-and-conquer approaches to sorting problems. Insertion sort also performed quit well for a rather rudimentary sorter, showing that even simpler algorithms can still be effective, especially when wanting to avoid the pitfalls of some more complex counterparts, such as the high CPU intensity of quick sort. Pancake sort was an interesting algorithm to implement; characteristically, it is similar to selection sort in that it rearranges the maximum elements to the end of the array, but the focus is to perform the fewest array reversals rather than fewest array comparisons. With a selection of few variables to sort, its performance was similar to bubble, but as the values increased, it

showed notable improvement. Naturally, all results found through this process are heavily reliant on the system they are run on, as the capabilities and limitations of each are vastly different; runtimes will likely vary drastically between different systems – a rather annoying limitation of empirical analysis in this regard.