Data Structures Assignment Six Report

Nate Everett

November 30, 2018

Assignment six explored a few of the many different types of sorting algorithms in the C++ language. The algorithms analyzed in this program are:

Algorithm	Worst Case	Average Case	Best Case	Space Complexity
Bubble Sort	$\Omega(n \log n)$	$O(n^2)$	$\Omega(n)$	$O(1)$
Quick Sort	$O(n^2)$	$\Theta(n \log n)$	$\Omega(n \log n)$	$O(n \log n)$
Insertion Sort	$O(n^2)$	$\Theta(n^2)$	$\Omega(n)$	$O(1)$
Selection Sort	$O(n^2)$	$\Theta(n^2)$	$\Omega(n^2)$	$O(1)$

The algorithms were timed on efficiency (in ms) with varying sized data samples. Beginning with a set $n = 10$, the differences did not show to be significant. At at set of $n = 100$, the Bubble Sort and Selection Sort proved to take relatively longer compared to the remaining two. The biggest significance, as expected, was when $n = 500$ where Quick Sort and Insertion Sort proved to be the quickest algorithms by factors of hundreds of ms. Given the name "Quick Sort", it was not shocking to find this to consistently be the quickest algorithm.

The trade offs for the different algorithms comes in the space complexity. Although as n increased the performance of Quick Sort proved best in run time, it takes a significant more amount of space. In addition, the complexity of the code for the algorithms varied respectively. Although not tested with other programming languages, it can be assumed that C++ will run equivalent to other languages according to theoretical/mathematical analysis.

Overall, this empirical analysis faced some shortcomings. There was

no equivalent environment to test the equivalent code with, therefore there could be hidden machine bias. In addition, the data was randomly generated and did not represent any real-life situation. However, it gave an insight into the run-times of these algorithms aside from the mathematical representation