Kevin Shannon

Dr. Syrotiuk

CSE 310

1 October 2018

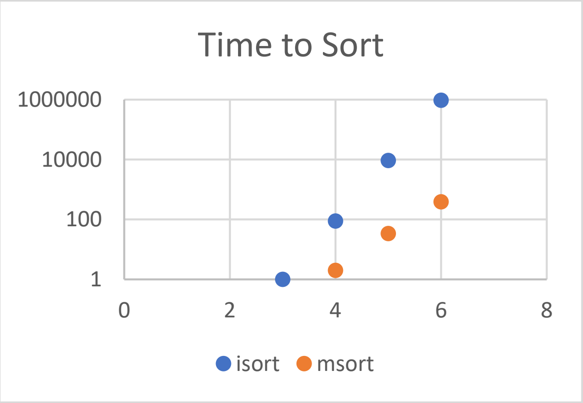
Experiment Report

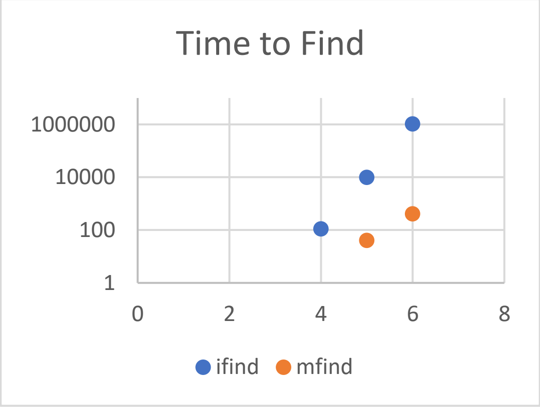
Abstract

The experiment to be tested is whether insertion-sort or merge-sort is faster. Given that insertion-sort has big O complexity of $n^2$and merge-sort $n \log n$, I expected merge-sort to be faster for large arrays, but possible insertion sort could be faster for small arrays.

Implementation

Files filled with random numbers where created with the provided randSeq program and a new program was created to sort and find the largest values based on the implementation used in part 1 of this project. Each algorithm would sort a copy of the array created from the file so that comparison in time is fair. I compiled and ran the code through general.asu servers.





The x-axis represents the order of magnitude of the array and y-axis is a logarithmic scale of time in milliseconds.

Results

As seen from the graphs isort was considerably slower in every case. For small arrays the program was able to sort faster than the clock function could measure hence why the data points are missing in the beginning. It is possible that with more precise time measurement methods that isort could be faster than msort, however the difference would be miniscule. Also, from the graph it can be seen that isort has a greater slope which means it gets exponentially worse than msort as the size of the array grows. The difference between sort and find times is nearly identical which can be accounted for by it needing to output the max value to the screen.