Shilp Shah (RUID: 170009205)

Mathew Varghese (RUID 170009651)

Assignment 2 Analysis

We conducted various tests to see how the speed of multithreading compared to the speed of multiprocessing for sorting files. We ran each test 5 times and took the average to account for any scheduling differences by the computer.

Tests

- In a single directory, we tested having 1, 2, 4, 8, 32 and 64 files. A few of these files were large so we did not increase the number of files above 64.
- For subdirectories, we tested both programs with 1, 2, 4, 8, and 16 directories/subdirectories, each with the same files.

Results

- Our results showed some differences between multithreading and multiprocessing. For a lot of the tests, the multithreading program was faster than the multiprocessing program. The biggest difference we saw was the time it took per file as we increased the number of files. It is amazing how multithreading and multiprocessing can we so efficient when the number of files grows very large.
- We did compare the times for both program through these test cases but the programs actually works slightly different from each other to have a straightforward comparison on time. The multithreading program creates 1 output file, meaning it has to sort a file that is a combined data of all the files in the directory and subdirectories. The multiprocessing program creates 1 output file for each CSV file that is sorted. Because of this slight difference, the programs run times are not fully comparable to each other.
- Mergesort is the right sorting function to use if we were only doing multiprocessing. Since the multithreaded program, sorts all the merged data, mergesort is not the best function to use