Serial and Multi-Threaded Architecture

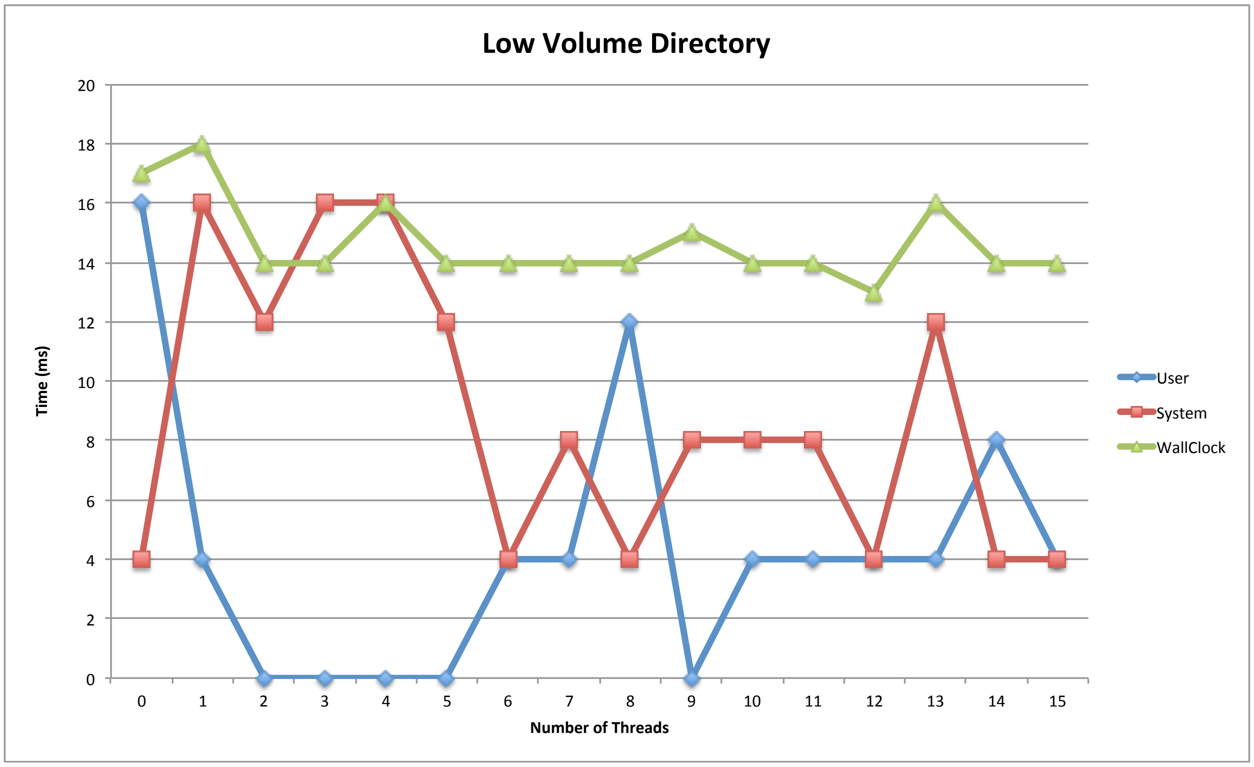
Charles Lovering - cjlovering@wpi.edu

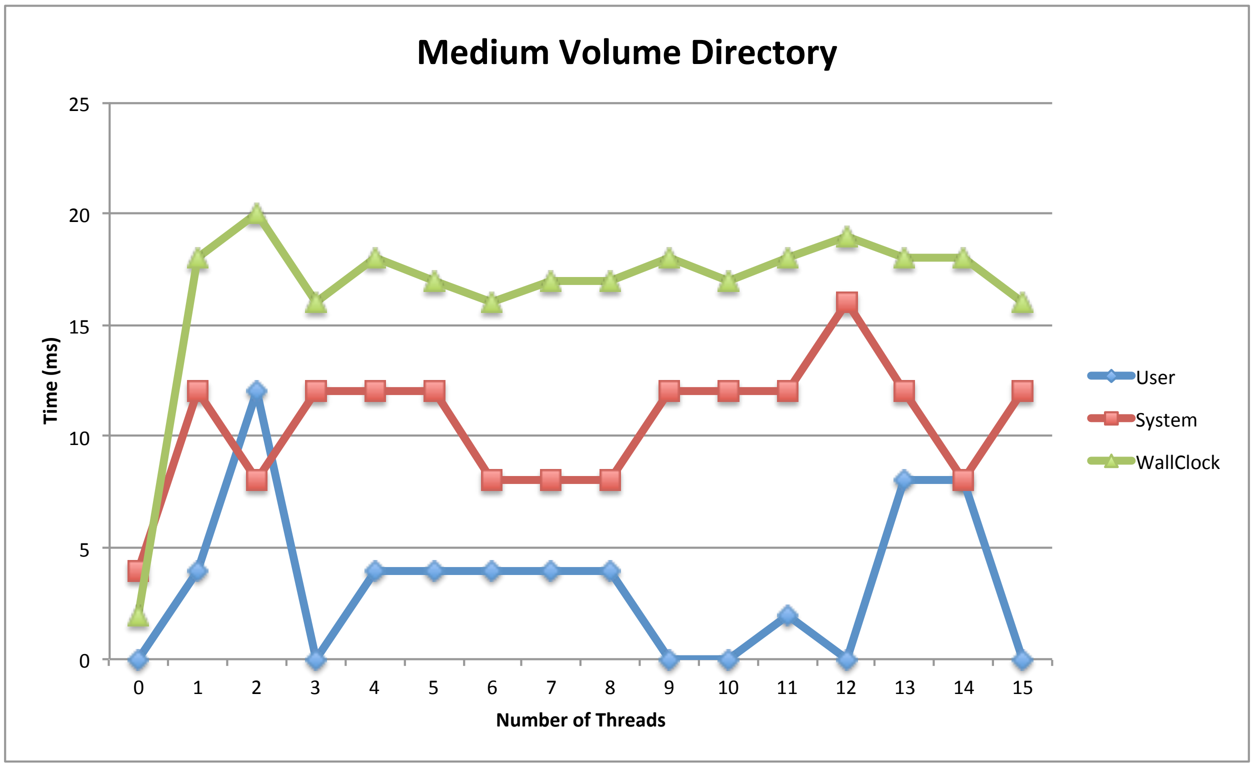
Introduction:

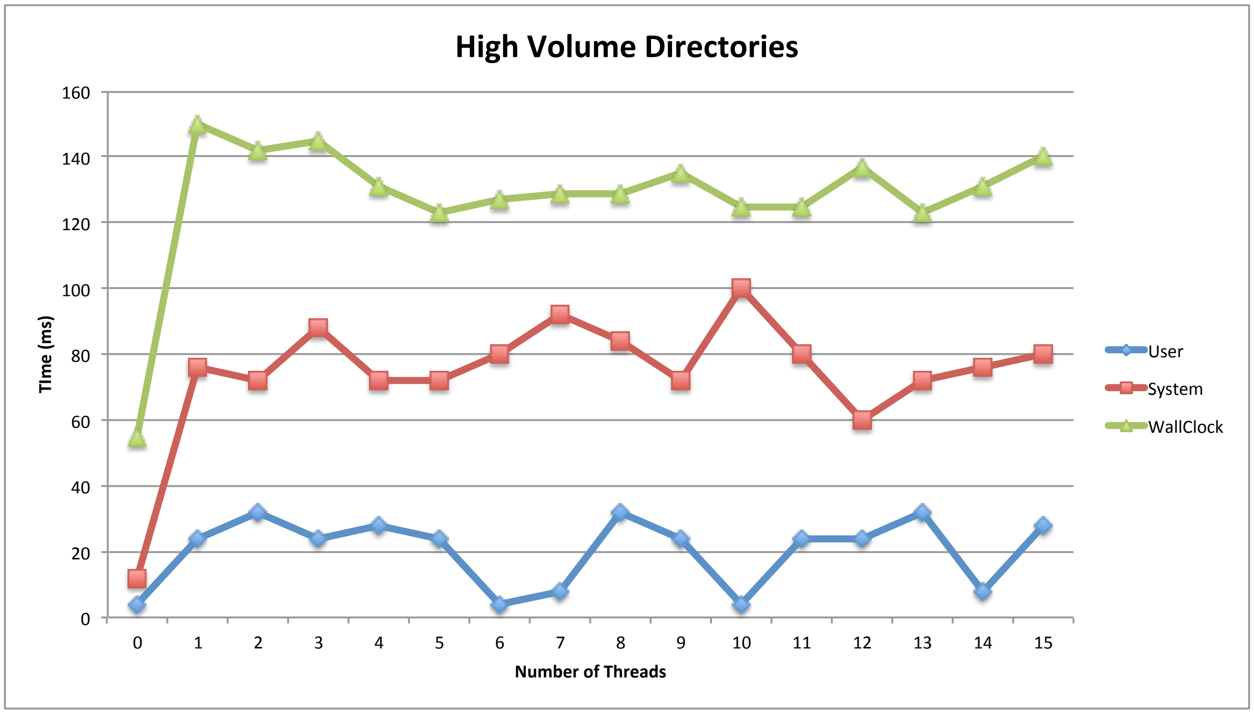
This is an inspection of the tradeoffs and effects of a serial versus multi-threaded architecture design. It is based off the results of a c++ function that reads from the stdin and returns statistics on the input regards to its validity as files.

Data:

Graphs:







Analysis:
The general results of this set of experiments point strongly to the notion that thread overhead is a significant *problem*. This overhead seems to increases dramatically (potentially exponentially) with the number of files that have to be read. This largely makes follows expected logic, as the program itself was not built specifically to recycle the threads, and as the computation per thread is relatively low the required switching and creation of threads is relatively significant compared to the increase in efficiency. The number of threads that are created is one to one with the number of files to be read, so while on a small scale experiment (experiment 1 with a directory of 6 files), there was some measure of immediate improvement when using threads. This follows, as the parallelism of the threads allows the process to complete more quickly. It is important to note that while there is improvement when a low number of threads are created (exactly 6 - the number of files to be processed), as that number increases there is no increase in performance, as the maximum number of threads to be run are already being run.

When looking at inputs with a medium or large number of files, the trend that there is a huge and immediate decrease in throughput is clearly evident. Though this could point directly to flaws in the code itself, it does speak loudly about the negative potentials of multi-threaded architecture.

Conclusions:
The sample size of these experiments, while large enough to draw preliminary ideas, is too small to make firm conclusion. Increasing the range of inputs, and number of trials would highly increase the validity of these trials.

That being noted, it would be an informed decision to inspect the concept of recycling threads to decrease the overhead of this process. Similarly, this versus a function that utilized children processes could be used to illuminate more clearly the advantages of a thread versus a processes. This could be useful for a situation where parallelism was essential.