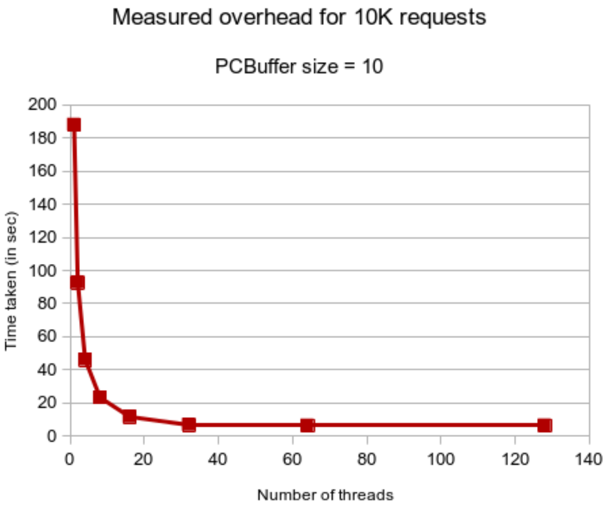
Daniel Frias

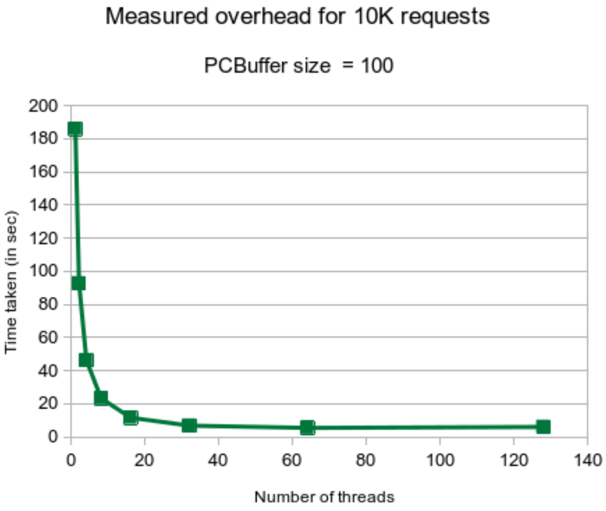
CSCE 313

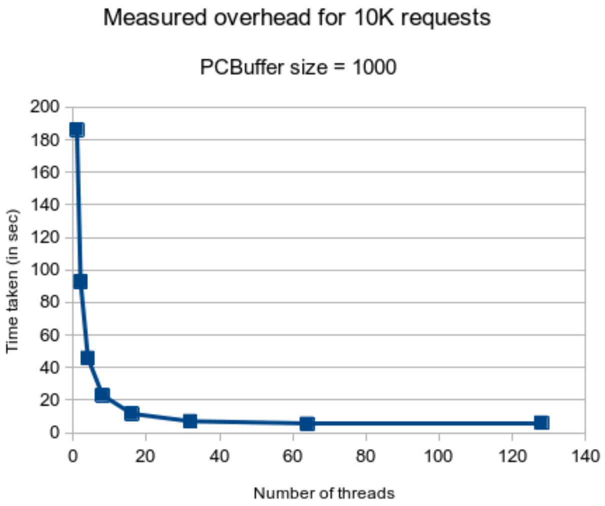
Section 516

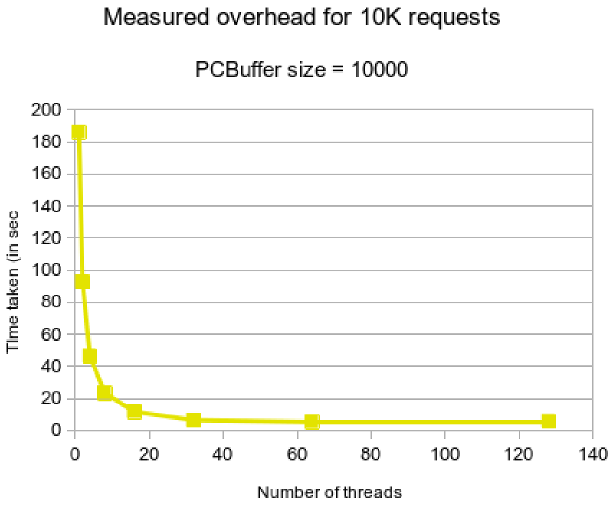
Machine Problem 3 Analysis

In this analysis, we look at the performance of the client/server program with a multithreaded implementation. The idea of adding threads to the client/server program is to allow a large amount of requests in parallel to be processed instead of just having two processes going through, say, one million queries which would take a long time. After implementing the client program, the time that the program took to process ten-thousand queries was measured, this was done by using the gettimeofday() function. Trials were done by starting with a buffer size of 10 and one worker thread, then doubling the amount of worker threads up to 128. Then, the buffer size was multiplied by a factor of 10 up to 10000, still doubling the amount of worker threads up to 128 as well. These are the graphs that were produced by the data collected from the program:









Looking at the graphs, there does not seem to be much difference when increasing the PCBuffer, even at the 10000 size. However, the number of threads executing in the program **does** have a big impact on performance, up until after 32 threads, where the time it takes to go through 10000 queries flattens out to about 6 seconds. Before that threshold, the amount of overhead is halved at every power of two number of threads. In conclusion, the size of the PCBuffer seems to have little impact on the performance of the client/server program, and the number of threads has significant impact on performance (up until about 32 threads).