I tested the performance of various read methods by reading three different files, a small file, a medium file, and a large file. The small file was 19KB, the medium file was 5.6MB and the large file was 750MB. I performed the testing on a 64 bit Ubuntu vm with 3 processor cores and 2GB of RAM. There were no other programs running while I did the testing.

First I tested the number of page faults that occurred when using the different read methods. As can be seen in the graph below, I only encountered one major page fault while testing. This result can be explained by the fact that there were very few processes running on the system, so there was always enough memory to load the files into. It can also be assumed that getrusage does not report the cold start page faults. This implies it either does not report them, or the system optimized access by loading the pages before they were requested.

My testing of the wall clock time required to read the files seems to indicate that the optimal running condition is 4 threads with a memory mapped file. This makes sense since more threads will allow the process to complete faster. The reason that the optimal number of threads is 4 versus the maximum of 16 is most likely due to the fact that the machine I was testing on had 3 processor cores. Since it could only run three threads at a time, adding more threads only slowed down execution since the threads could not perform simultaneously, and the system had to deal with the extra overhead from creating more threads. In general the results also indicated that memory mapping even with one thread was faster than using calls to read. This becomes more apparent the larger the files being read become. This can be attributed to the fact that the program did not have to make as many systems calls, so it avoided the extra overhead involved with invoking those system calls.