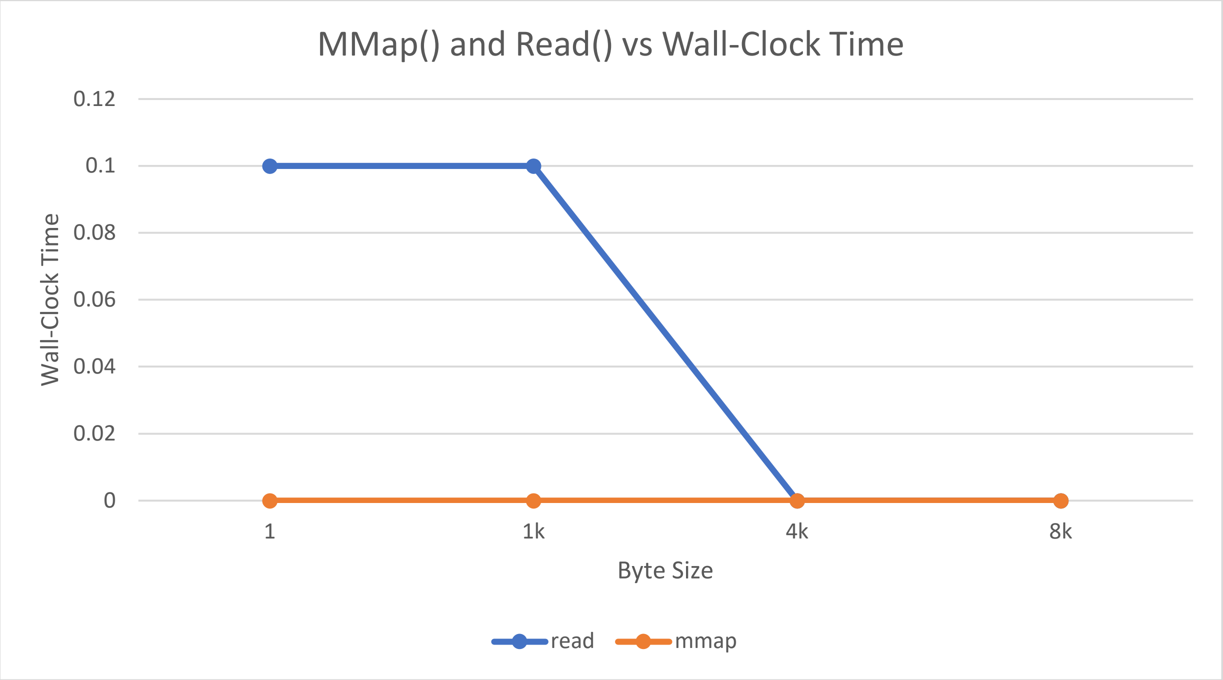
Performance Analysis of Proj2.cpp

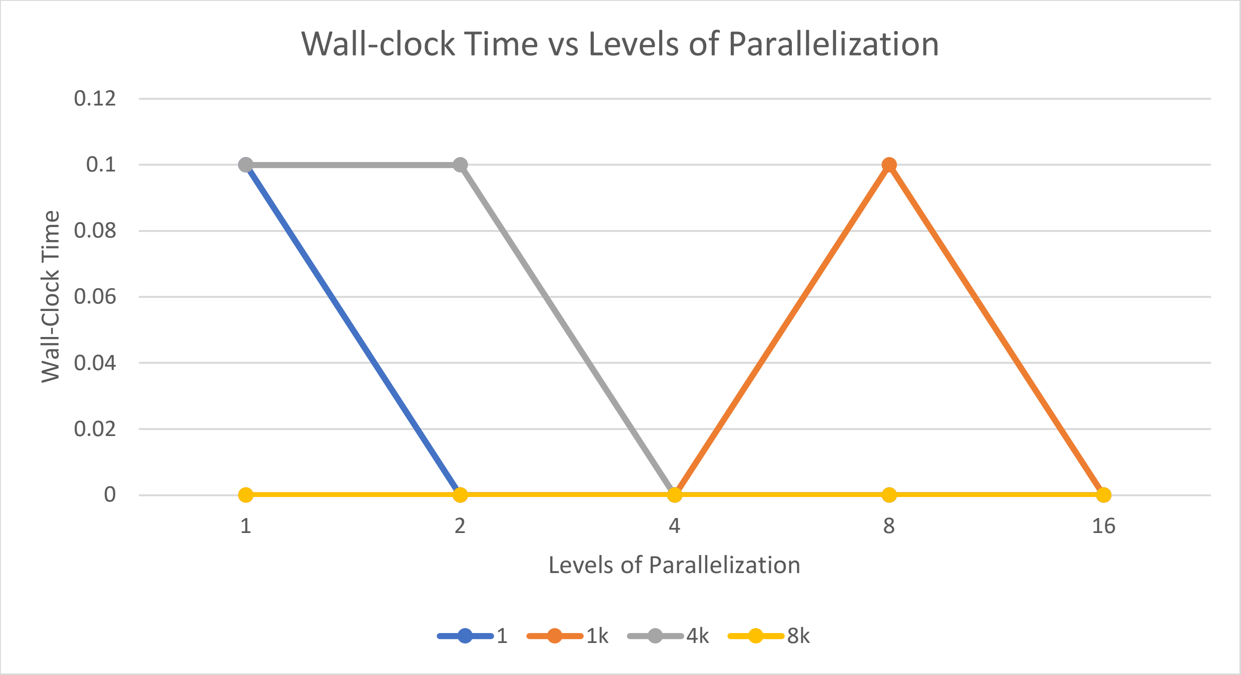
Professor Wills

9/17/2021

By Tate Donnelly



From testing my proj2.cpp file in Atom, I found that the mmap function was significantly more efficient than the read() function for both files with 1 byte and files with 1,000 bytes. This is because the mmap function took about 0 milliseconds to map the contents of both files while the read function took around 0.1 milliseconds. It is unclear whether or not mmap has a similar effect on files with 4,000 bytes and 8,000 bytes as both the mmap function and the read function took approximately 0 milliseconds of wall-clock time. Given our data, we can assume that the mmap function is a better technique for files with 1 to 1,000 bytes, however, we can't assume the same for larger files.



Additionally, from testing parallelization, I've found that parallelization's benefits depend on the size of the file and the number of processes. For instance, as shown in the graph, when files with 1 byte are mapped into 1 process they take longer than when they have higher levels of parallelization. However, for files with 8,000 bytes any parallelization seems to have roughly the same effect as it takes around same time (0 milliseconds for all of them). All file sizes seem to be most efficient at parallelization levels 4 and 16, which indicates that using these parallelization levels is the best technique for files with between 1 and 8000 bytes.