As you can see with small file sizes there is no difference with number of page faults for mmap and file I/O. They both result in no major page faults. However, as you get to bigger and bigger sizes that are not seen in this graph there are more page faults for reading a file then mapping a file.

Below, is a graph for the wall-clock time of both file I/O and mapping and as you can see they are the same. However, with larger page sizes you would see that the wall clock time of read would be larger than the wall clock time of mapping.

However, we must keep in mind that memory mapping causes slightly more minor page faults then reading to a file, as seen by the graph below.

This graph below is for the wall-clock time of a 8K file with different thread sizes. With the small size of the file, no major page faults are found. However, there is a discrepancy in the wall-clock time with different thread sizes. Overall, the more threads you use the longer wall-clock time. This is probably due to the amount of time it takes for the main thread to send and receive messages from the worker threads.

Overall, given a small file size it does not matter it does not make a big difference if you read or do memory mapping. However, with bigger size files it will be better to use memory mapping to reduce the number of major page faults that may occur. Additionally, with larger file sizes it is best to use threads to distribute the work amoung worker threads because it reduces the amount of wall clock-time as seen by

./doit proj ovs-agent-3.0-1.el6.x86\_64.rpm mmap

Shell command is ./proj

Usage: ./proj <ovs-agent-3.0-1.el6.x86\_64.rpm>

File size:733820

Strings greater than 4: 11473

Maximum string length is: 408

Time User CPU: 013 ms

CPU System time: 000 ms

Number of Page Faults: 0

Number of Page Faults that can be satisfied by reclaiming memory: 412

 Number of Processes Preempted involuntarily: 9

Number of Time Processes Give Up: 1

 Wall-clock time: 014 ms

$

$

$ ./doit proj ovs-agent-3.0-1.el6.x86\_64.rpm mmap p2

Shell command is ./proj

Time User CPU: 000 ms

CPU System time: 000 ms

Number of Page Faults: 0

Number of Page Faults that can be satisfied by reclaiming memory: 201

 Number of Processes Preempted involuntarily: 1

Number of Time Processes Give Up: 1

 Wall-clock time: 001 ms

Getting a wall clock time of 1ms vs 14 ms. In situations where the file size is small though, multiple threads increase wall-clock time. In conclusion, if a file is small it will not not make a big difference if you choose mapping or reading and it is best practice to avoid using multiple threads. However as the file size increases to large sizes it is best to use mapping and multiple threads to reduce the wall-clock time and minor page faults.