MP3 Report
yuguang2 Yuguang WEI

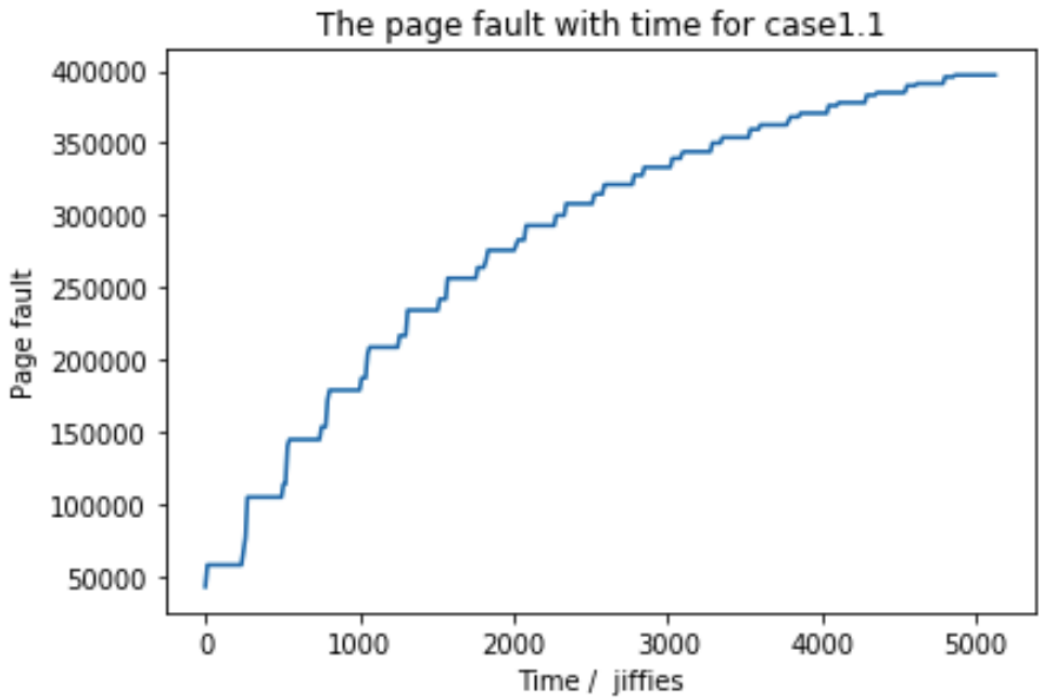
Design and implementation:

Workqueue: workqueue is implemented as the main service of periodically record the minor, major and cpu computation time for each process. Then summing up all the page faults and cpu time together, passing into kernel buffer.

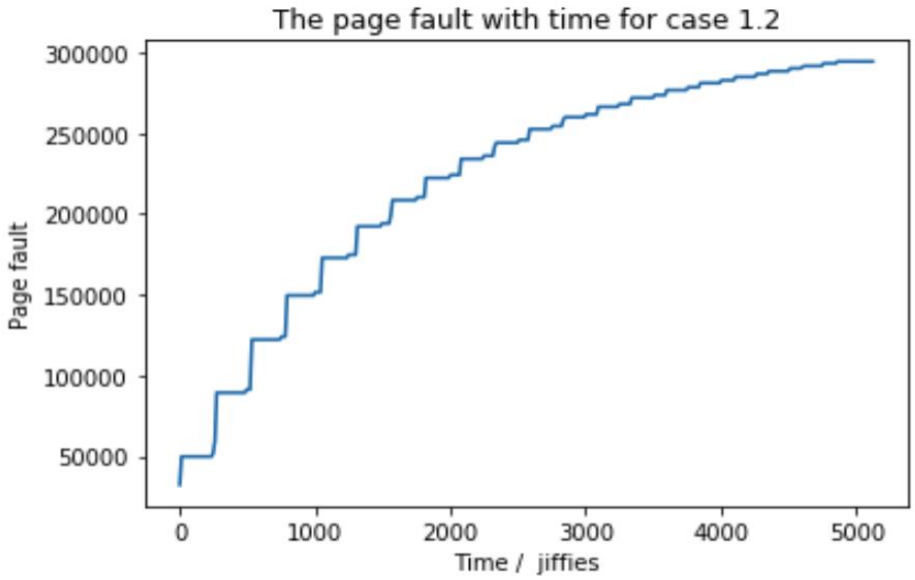
Character device: The core of the character device is mmap callback function. The trick of mmap is mapping continuous physical memory from kernel virtual memory to the user virtual memory. Each page frame is mapped one by one to make sure the continuity of physical memory.

Case1:

Work process 1: 1024 MB Random 50,000 per process
Work process 2: 1024 MB Random 10,000 per process



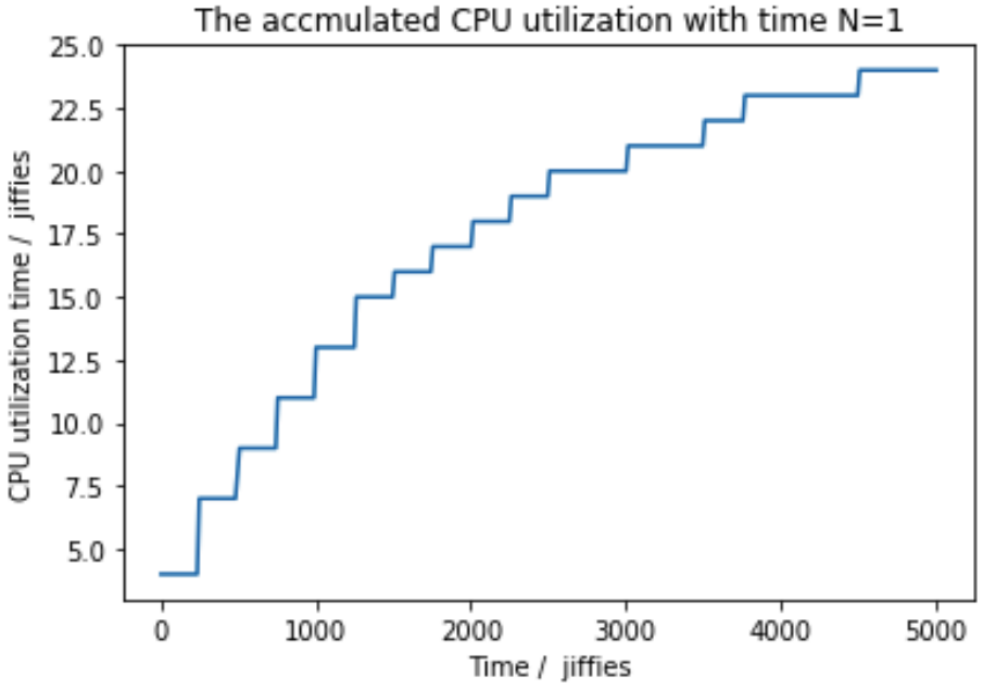
Work process 3: 1024 MB Random 50,000 per process
Work process 4: 1024 MB Locality 10,000 per process

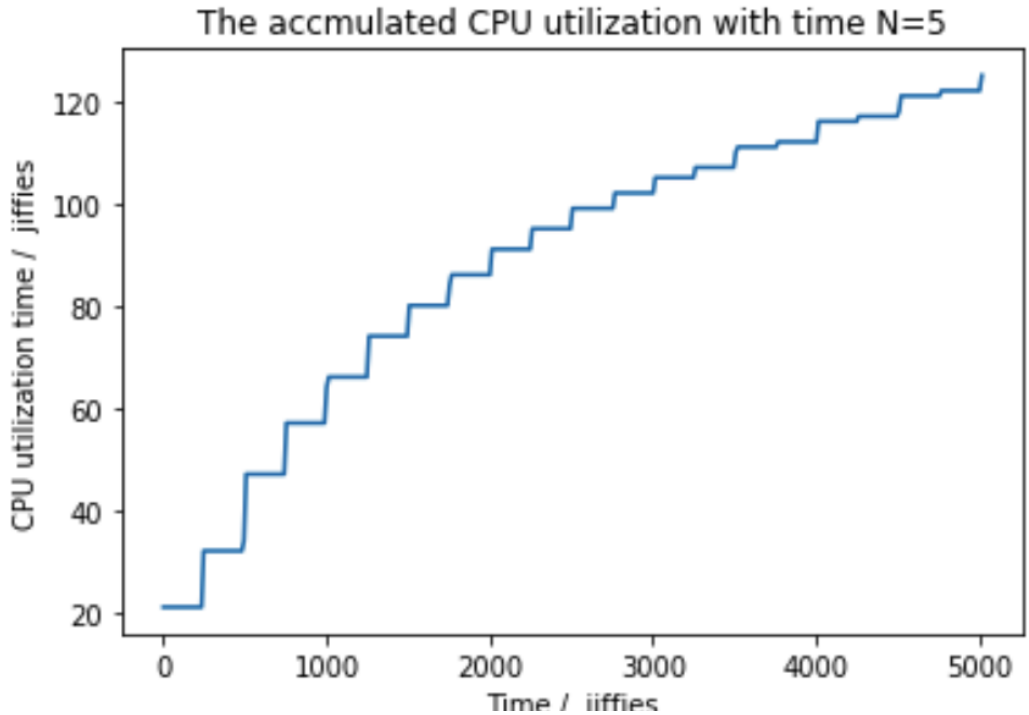


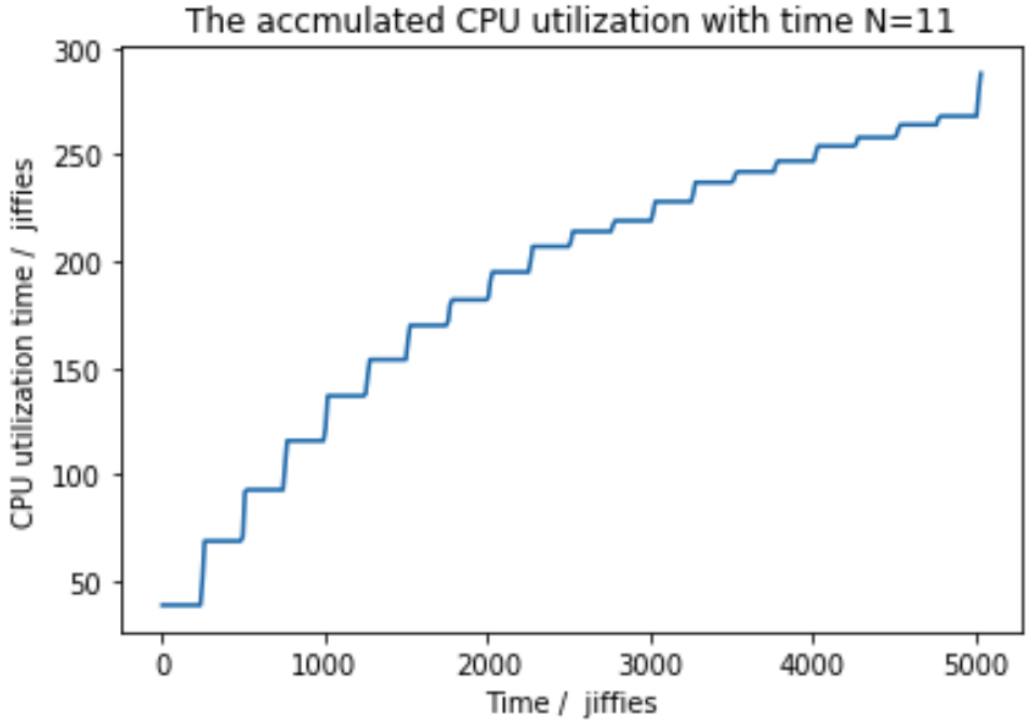
From these two instances, the allocated memory and access times for each process in each instance are same. The only variable is the access pattern and the result is consistent with our expectation. The random access would have more page faults since the page fault replacement policy would have less effect on reducing the page fault with respect to random access.

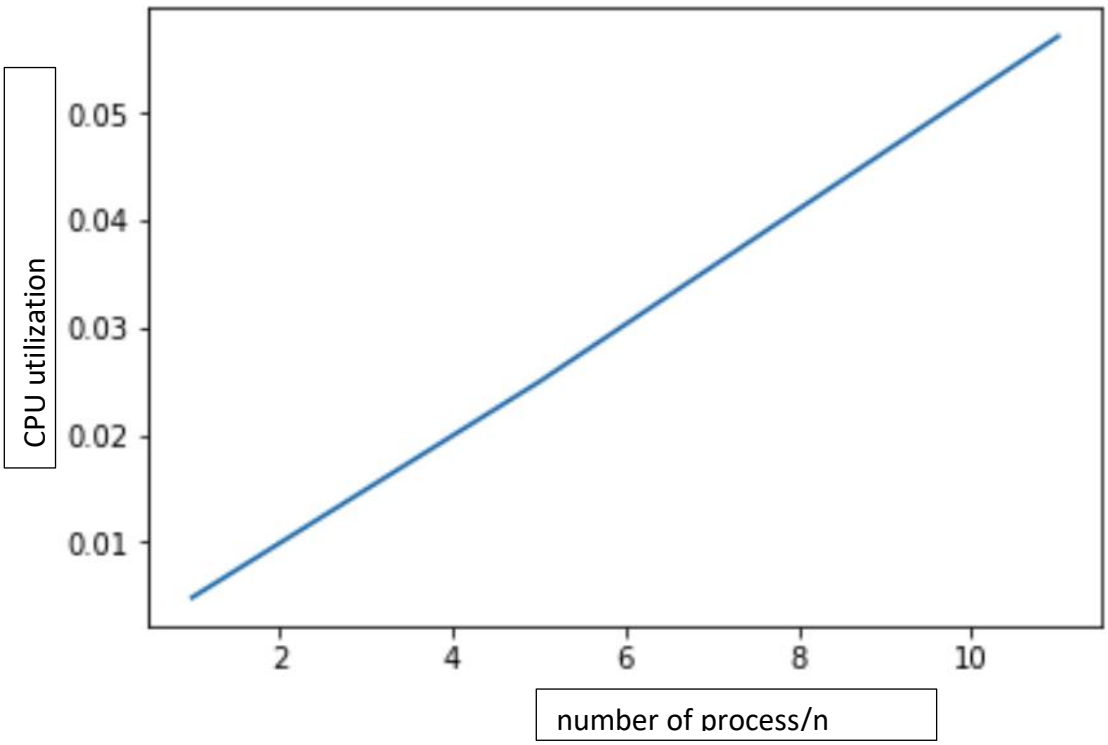
The completion of time these two cases are almost the same 5000 jiffies, which is dominant by fixed number of iterations in work.c.
Case2:

Work Process: 200MB Random Locality 10,000

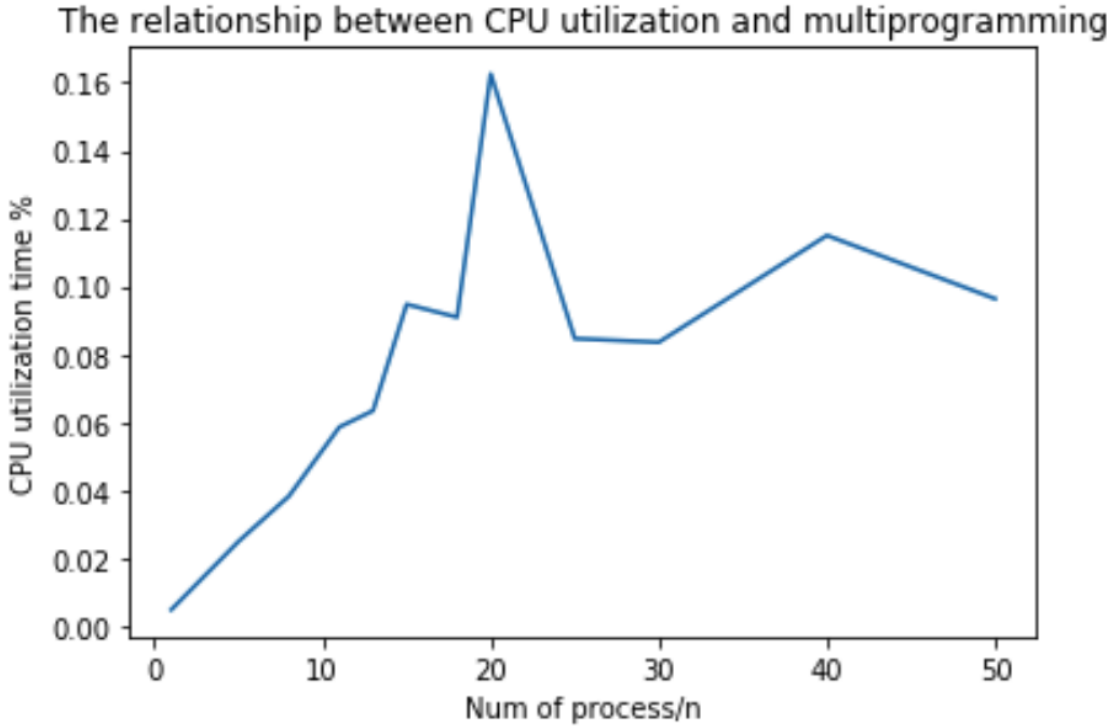



The accmulated CPU utilization with time N=5
CPU utilization time / jiffies
120
100
80
60
40
20
0
1000
2000
3000
4000
5000
Time / jiffies


The accmulated CPU utilization with time N=11
CPU utilization time / jiffies
300
250
200
150
100
50
0
1000
2000
3000
4000
5000
Time / jiffies



It can be observed that the utilization of the CPU almost linearly increases as the number of processes increases, which is also consistent with my hypothesis that multiprogramming can improve the utilization of the CPU when N is relatively small. There is no constrains for increasing CPU utilization when page faults occur in low frequency.
However, if we increase the amount of concurrent processes, the following graph is obtained:



It can be seen that a similar characteristic line to thrashing is shown as above graph. The result indicates that thrashing occurs when N is relatively large. The CPU is Idle during the page fault handler is trapped in continuous replacement of pages. As a result, the utilization drop down later.

In the linear region, the completion time is almost the same as 5000 jiffies. However, that of each instance increases a lot ,especially, when the CPU utilization drops down.