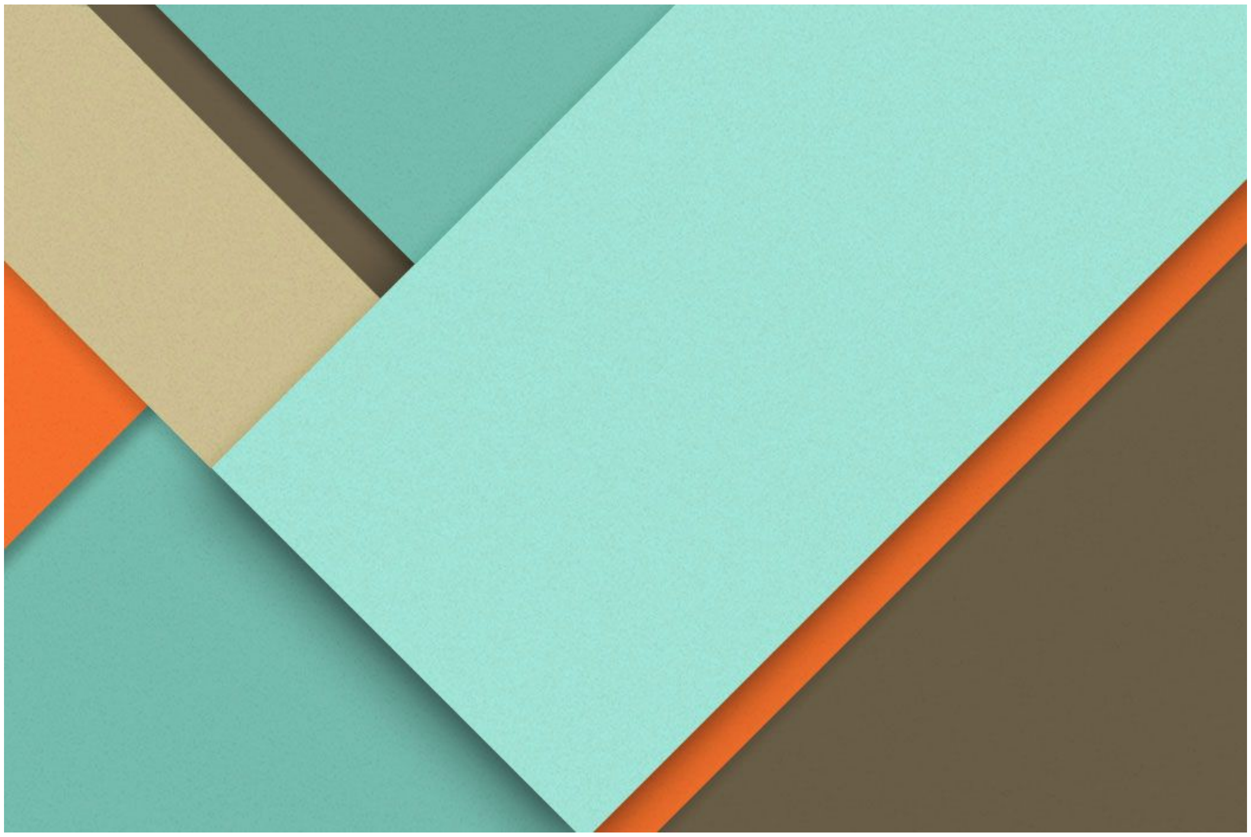


Report for Virtual Memory

30.03.2018

Arun Bhusri
5680607

Overview

We were assigned to build a simple but fully functional demand paged virtual memory. This was implemented at a user level which is exactly the technique used by modern virtual machines. At the start of this project, we were provided with a starter pack which included the implementation of a page table and virtual disk. We were asked to implement three kinds of replacement algorithms which will trap page faults and report what should be done to solve the problem.

The three types of replacement algorithm are:

- Random page fault handler
- FIFO page fault handler (first in first out)
- LRU page fault handler (least recently used)

Purpose of the experiment

Purpose of this experiment is to compare the three page fault handlers in terms of how their performance by looking at the read counts, writes counts, and fault counts that we print in the end of the experiment. We will use read counts, writes counts and faults counts data to plot the graph.

Experimental Setup

The implementation of these replacement algorithms was done on a MAC machine. Due to not reading the instruction properly, I tried to compile to code on MAC machine and ran into a compilation error. Therefore; I decided to “scp” my code to Ubuntu VM which was provided in the File Processing class. I was able to compile and perform test on a LINUX machine without any problem.

Here are some instructions.

The program must be invoked as follows:

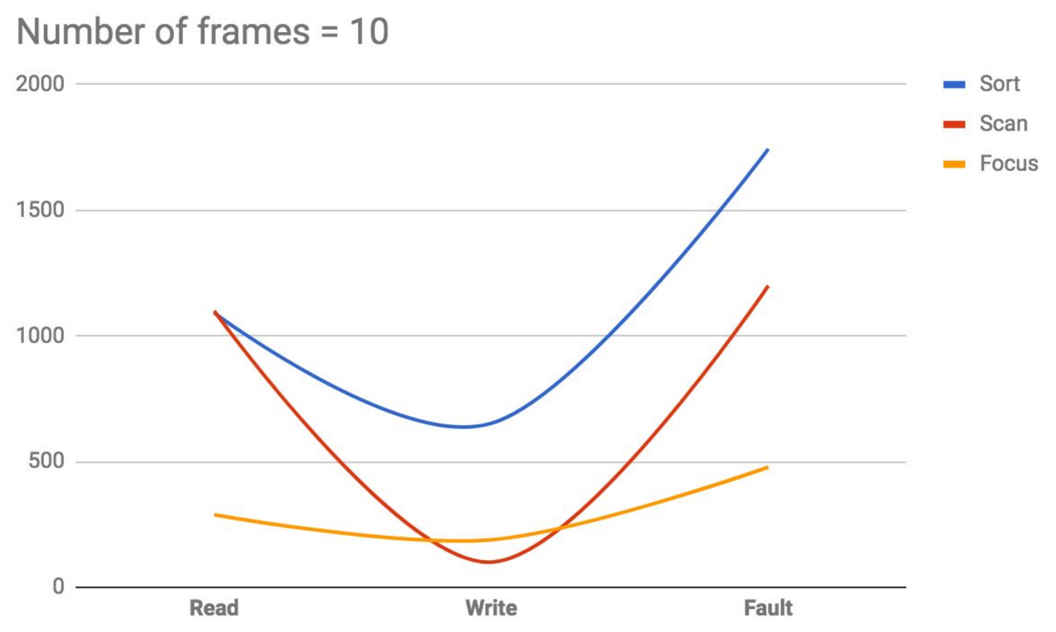
```
./virtmem npages nframes [rand|fifo|lru] [scan|sort|focus]
```

Below are the commands for testing different types of replacement algorithms:

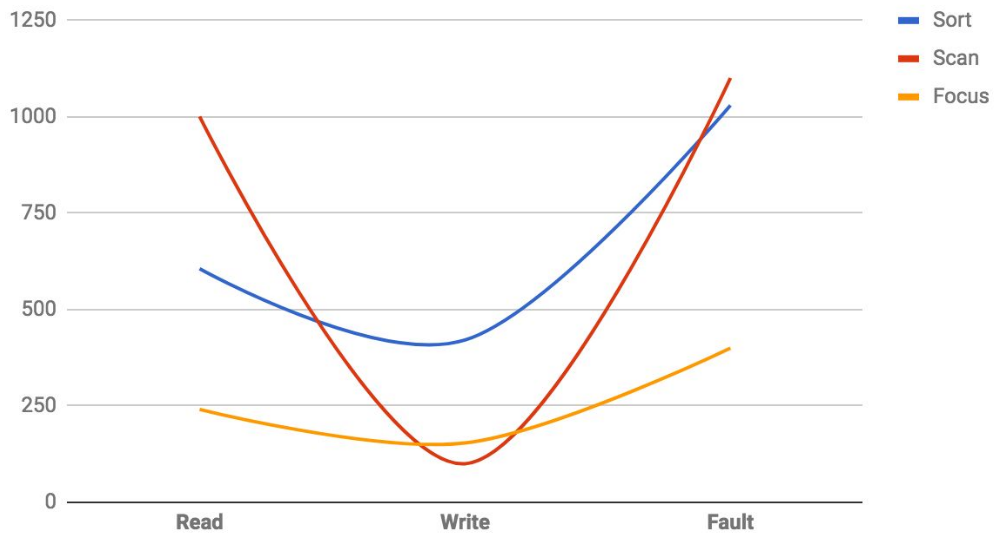
1. Random page fault handler
   a. ./virtmem <npages> <nframes> rand <scan|sort|focus>
2. First in first out page fault handler
   a. ./virtmem <npages> <nframes> fifo <scan|sort|focus>
3. Least recently used page fault handler
   a. ./virtmem <npages> <nframes> lru <scan|sort|focus>

Example: ./virtmem 100 10 rand sort

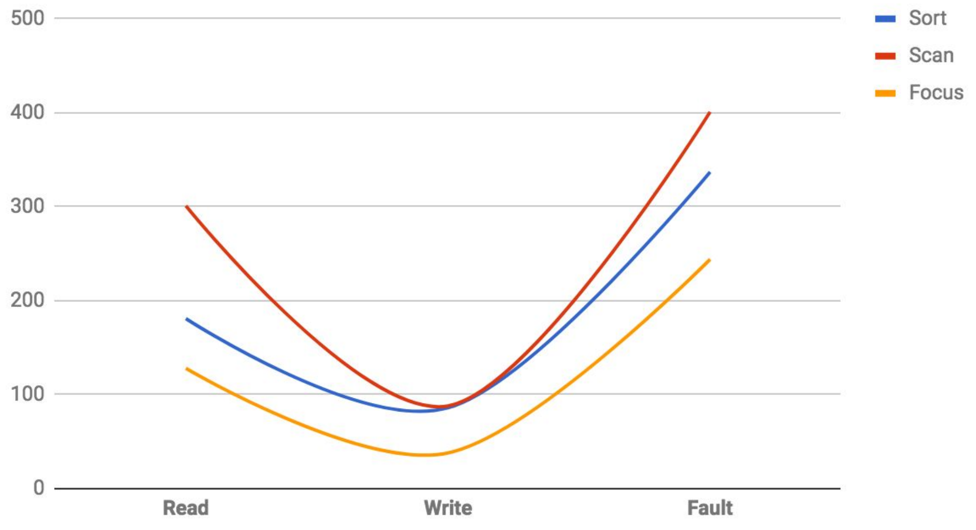
Results for Random page fault handler



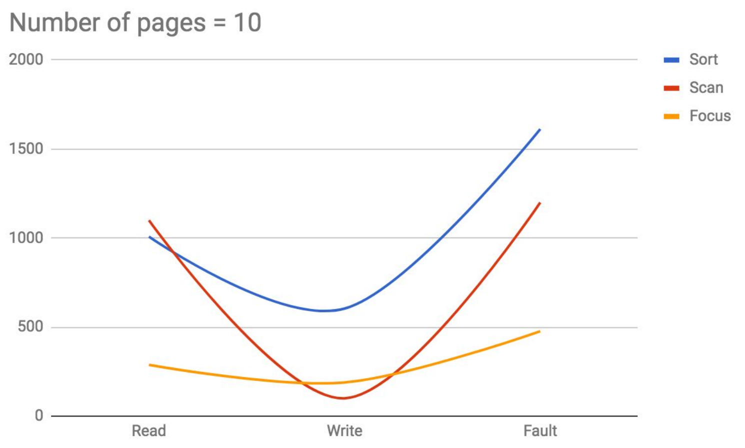
Number of frames = 40

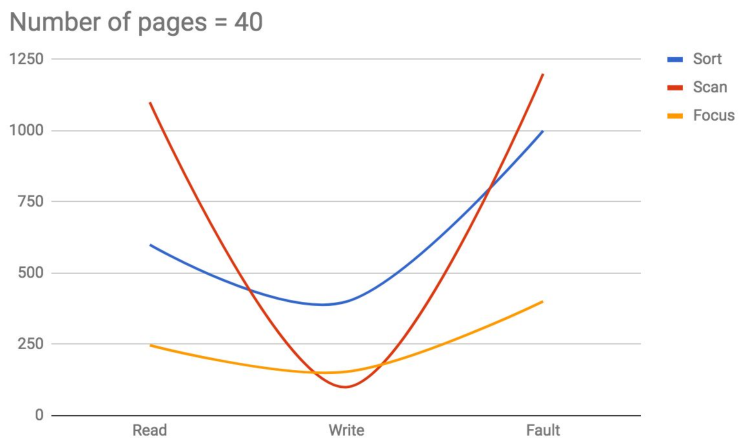


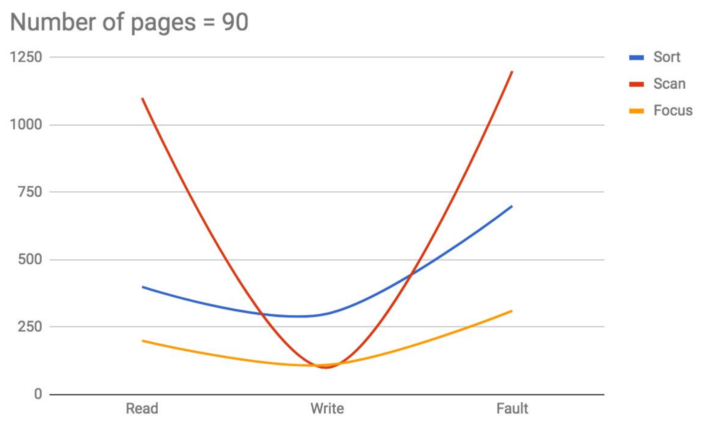
Number of frames = 90



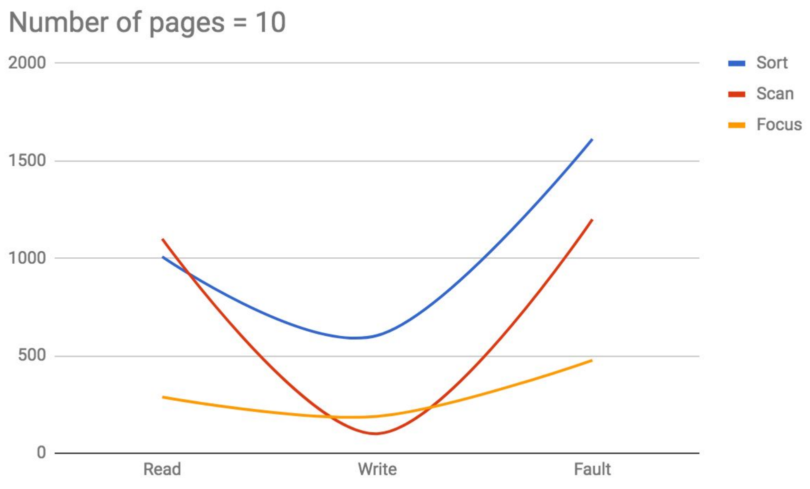
Results for FIFO page fault handler

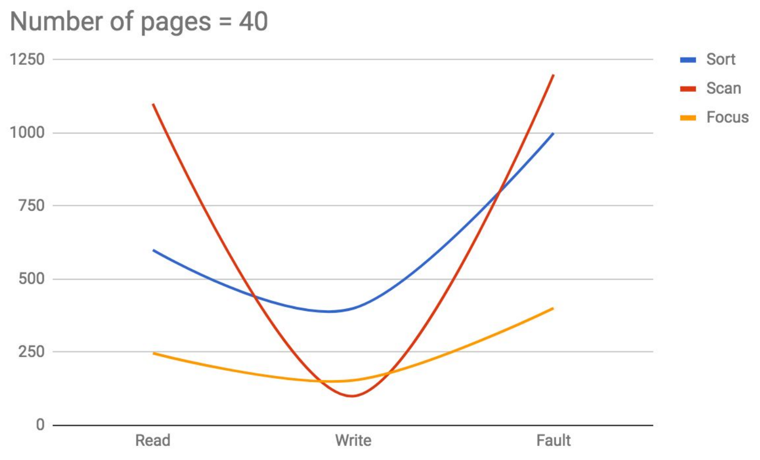


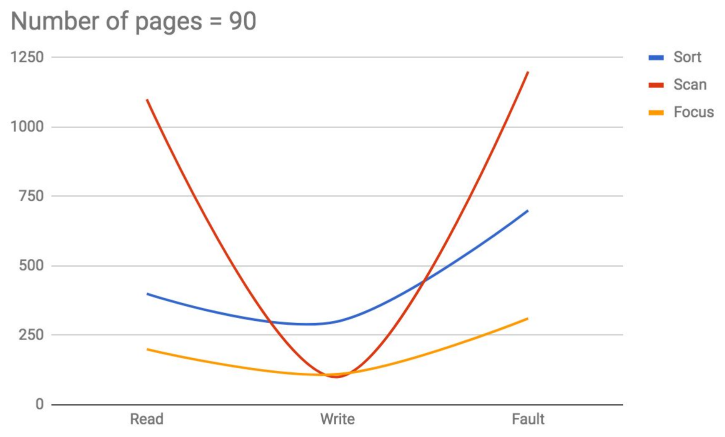




Results for LRU page fault handler







Conclusion

After running the test we can see that the result are similar for almost every types of page fault handler. If we look closely at the number, we can see that random page fault handler performs better. The difference between fifo and random page fault handler is very small. I chose random page fault handler as the best algorithm because of the data that I've collected after running the test. For random page fault handler, we can see that as the number of frames increases, the number of fault decreases; which is a good sign. Overall, I would rate random page fault handler as the best page replacement algorithm.