CS-314 OPERATING SYSTEMS LAB-8 REPORT

(M. Suresh – 210010030, K.S.N Manikanta – 210010050)

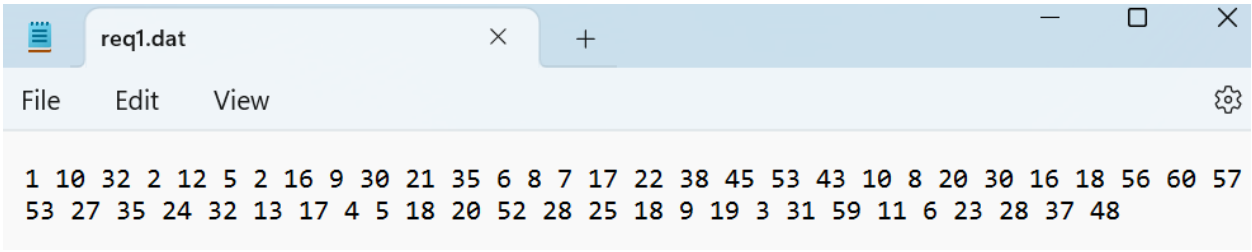
1. Introduction

In this assignment, we have used FIFO, LRU and random page replacement policies for various request files. In FIFO, the page that is accessed first should be evicted when a page fault occurs. It is simple and fair because all pages have equal chances of getting replaced. However, it may replace a page that is accessed frequently, leading to more page faults. In LRU, the page that is least recently used should be evicted when a page fault occurs. It is near to optimal and gives good performance in 80-20 workload, which is closer to reality. However, it is a little difficult to implement this and does not help in the case of sequential accesses. In Random policy, when a page fault occurs, a random page is evicted. It is simple to implement and has no overhead. However, its performance is very uncertain. Here, we assume that the pages start from 1 and not zero, i.e., **virtual page 1 ⇔ VPN 0 or virtual page {i} ⇔ VPN {i-1}.**

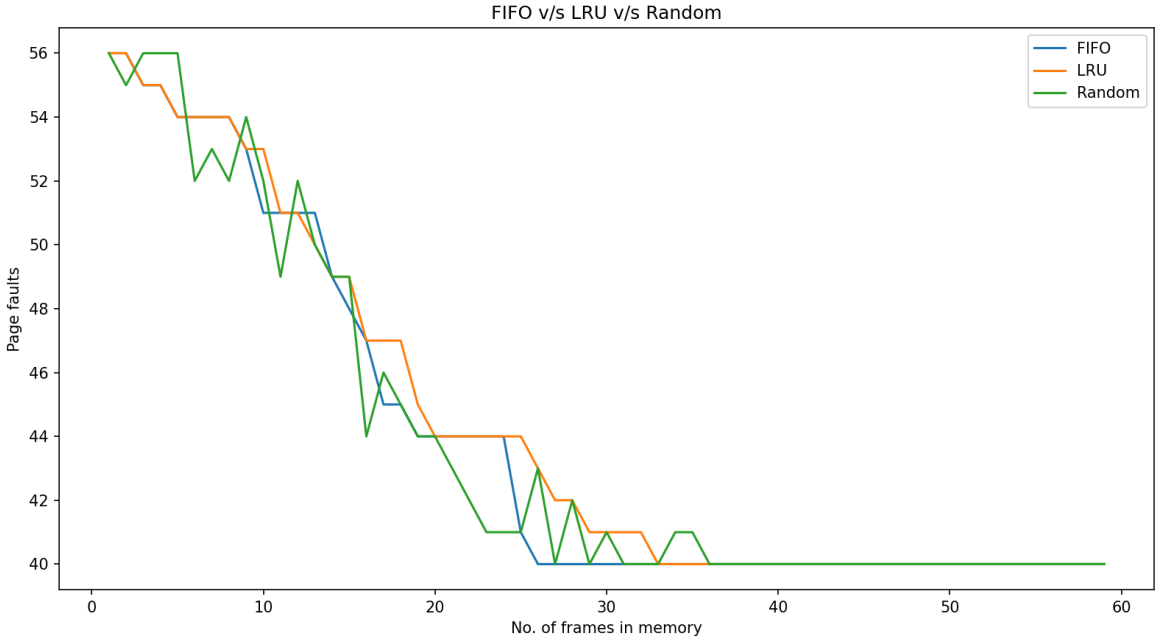
2. Analysis

2.1. Testcase: req1.dat

```
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./random 60 20 60 req1.dat
No. of Page Hits: 10
No. of Page Faults: 46
Percentage of page fault: 0.821429
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./random 60 20 60 req1.dat
No. of Page Hits: 11
No. of Page Faults: 45
Percentage of page fault: 0.803571
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./fifo 60 20 60 req1.dat
No. of Page Hits: 12
No. of Page Faults: 44
Percentage of page faults: 0.785714
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./lru 60 20 60 req1.dat
No. of Page Hits: 12
No. of Page Faults: 44
Percentage of page fault: 0.785714
```

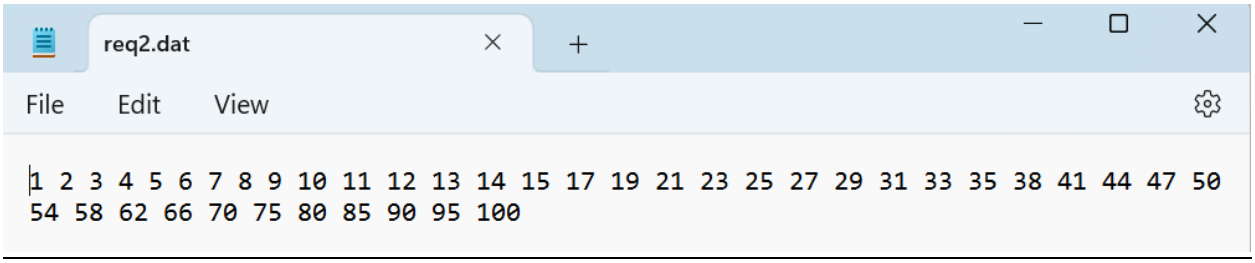


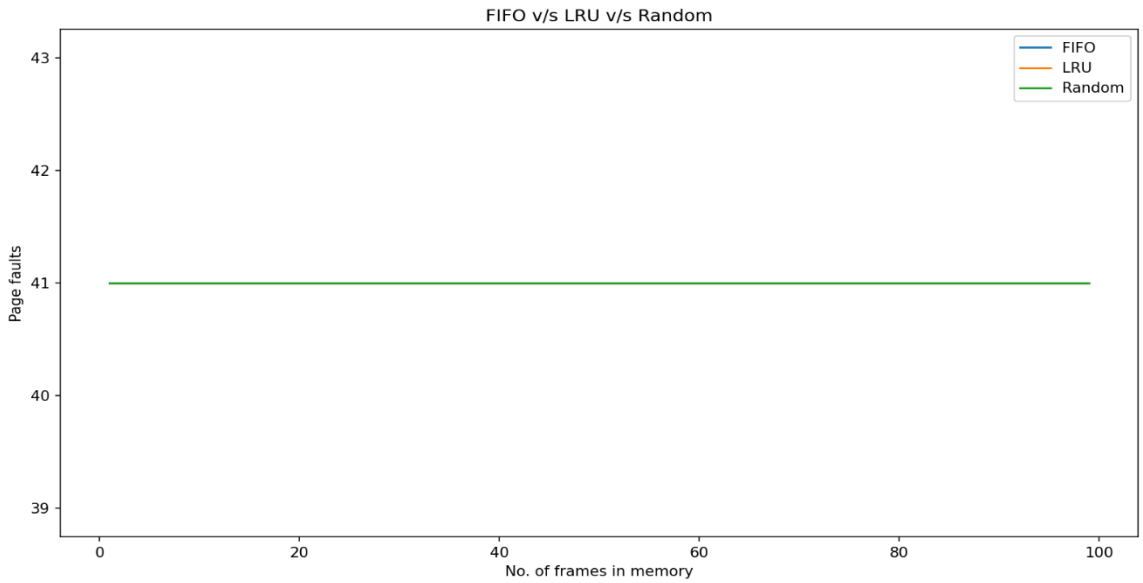
From the graph below, we observe that FIFO performs better than LRU. Random, as we know, works randomly, i.e., its performance is very uncertain. We observe that as the number of frames in memory increases, the number of page faults decreases. After sometime, we see the number of page faults remains constant for all 3 algorithms as there are enough pages in memory that no swapping occurs. That constant value is 40 which are compulsory faults that occur at the beginning when the memory is empty.



2.2. Testcase – req2.dat

```
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./lru 100 50 100 req2.dat
No. of Page Hits: 0
No. of Page Faults: 41
Percentage of page fault: 1
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./fifo 100 50 100 req2.dat
No. of Page Hits: 0
No. of Page Faults: 41
Percentage of page faults: 1
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./random 100 50 100 req2.dat
No. of Page Hits: 0
No. of Page Faults: 41
Percentage of page fault: 1
```

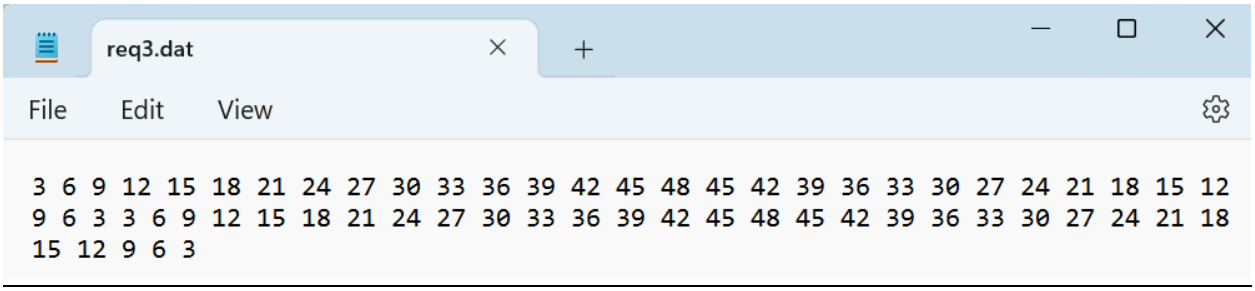


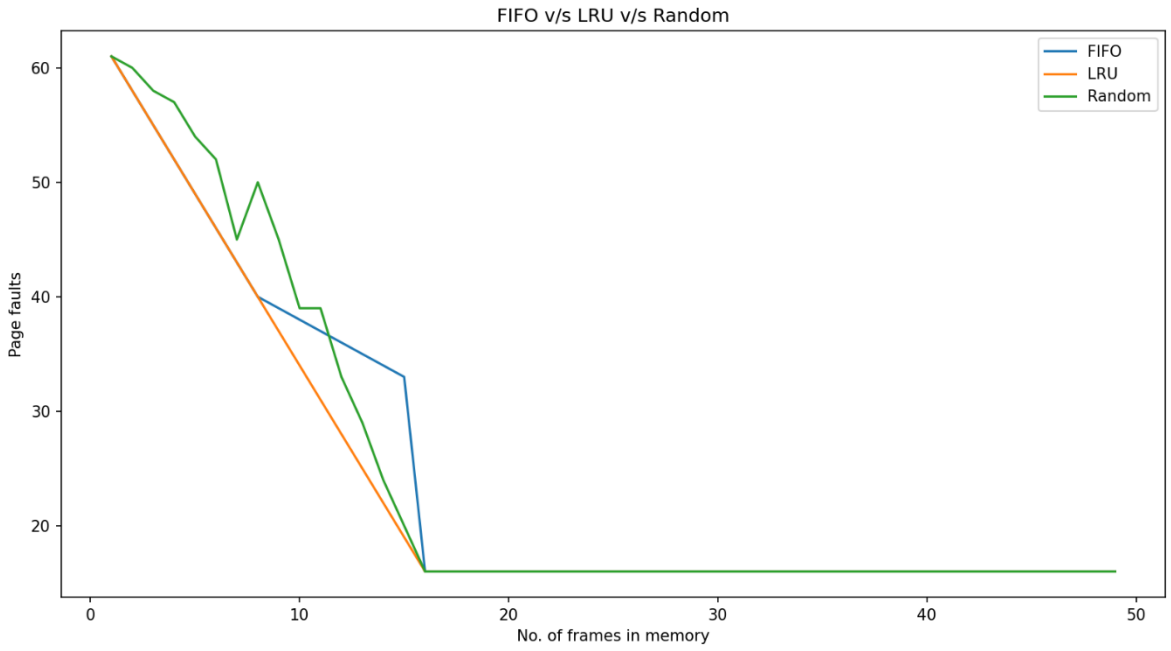


From the above graph, we observe that all accesses lead to page faults since the pages are accessed sequentially and hence number of page faults are same for all values of no. of frames.
Since no page is accessed again in future after it has been accessed in the past, all three algorithms give the same result. The no. of page faults is equal to the no. of accesses in the input file, for all three algorithms.

2.3. Testcase - req3.dat

```
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./lru 50 13 50 req3.dat
No. of Page Hits: 37
No. of Page Faults: 25
Percentage of page fault: 0.403226
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./fifo 50 13 50 req3.dat
No. of Page Hits: 27
No. of Page Faults: 35
Percentage of page faults: 0.564516
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8> ./random 50 13 50 req3.dat
No. of Page Hits: 32
No. of Page Faults: 30
Percentage of page fault: 0.483871
```



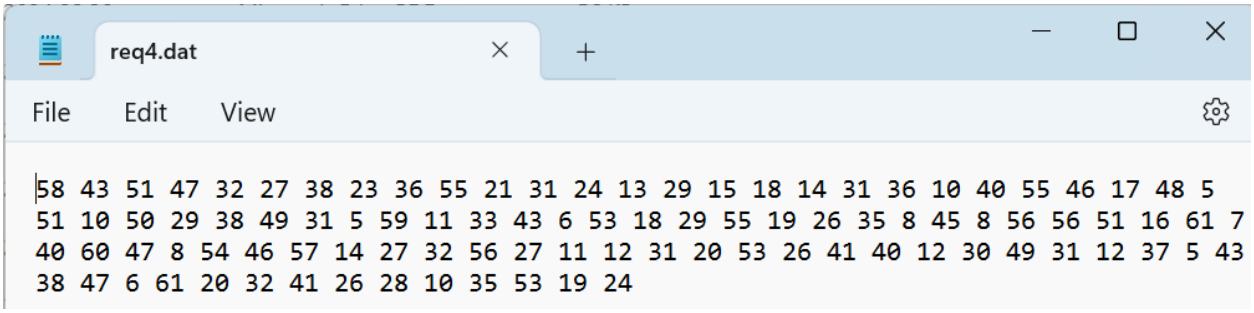


From the above graph, we observe that LRU works better than both FIFO and random since the accesses are such that those that

are accessed recently are accessed again soon. LRU avoids swapping the frames, which are accessed quite often. This leads to fewer page faults. Regardless of the algorithm, the no. of page faults decreases with an increase in the number of frames in memory. The page faults settle at a constant value of around 8 or 9, which are the compulsory faults that occur at the beginning when the memory is empty. After a point, increasing the number of frames doesn't reduce the number of page faults, as there is enough space in memory such that nearly no swapping occurs.

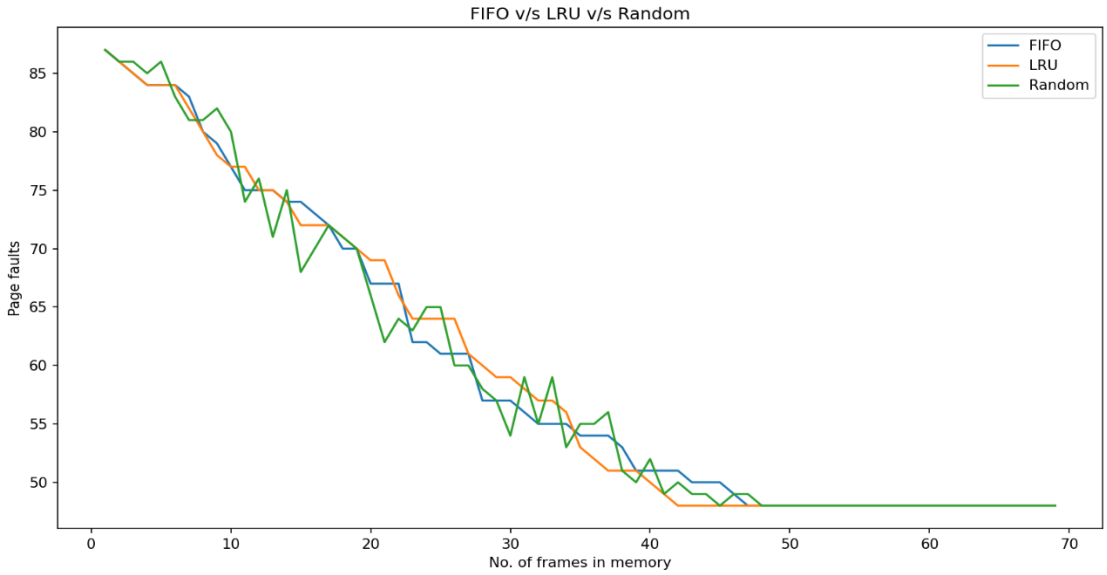
2.4. Testcase – req4.dat

```
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8>
 ./fifo 70 30 70 req4.dat
No. of Page Hits: 31
No. of Page Faults: 57
Percentage of page faults: 0.647727
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8>
 ./lru 70 30 70 req4.dat
No. of Page Hits: 29
No. of Page Faults: 59
Percentage of page fault: 0.670455
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8>
 ./random 70 30 70 req4.dat
No. of Page Hits: 30
No. of Page Faults: 58
Percentage of page fault: 0.659091
```



From the below graph, we can see that FIFO performs better than LRU till the no. of frames reaches around 35 and then LRU performs better than FIFO till the no. of frames reaches around 48 where the no. of page faults reaches a constant value of around 47

or 48 which are the compulsory faults that occur at the beginning when the memory is empty. After a point, increasing the number of frames doesn't reduce the number of page faults, as there is enough space in memory that no swapping occurs. Due to the random nature of the requests, all the replacement policies perform similarly.



2.5. Testcase – req5.dat

```
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8>
 ./random 120 20 120 req5.dat
No. of Page Hits: 32
No. of Page Faults: 52
Percentage of page fault: 0.619048
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8>
 ./lru 120 20 120 req5.dat
No. of Page Hits: 33
No. of Page Faults: 51
Percentage of page fault: 0.607143
PS C:\AlphaParadise\1.CSE\SEM6\OPERATING SYSTEMS\CS-314_OS-Lab_Minix\LAB8>
 ./fifo 120 20 120 req5.dat
No. of Page Hits: 31
No. of Page Faults: 53
Percentage of page faults: 0.630952
```

req5.dat

File Edit View

```
1 2 40 60 90 45 67 83 69 1 4 10 12 56 96 78 53 41 89 52 45 52 89 41 75 65 32 41
85 97 1 6 8 4 5 78 96 45 7 45 96 85 58 69 47 24 26 25 35 21 26 27 28 29 26 21 25
23 24 28 29 24 25 26 21 24 69 63 63 68 57 57 57 4 1 23 32 63 43 75 73 1 6 9
```

From the below graph, we can see that the random policy is quite good, but its performance is highly uncertain. Due to the random nature of the requests, all the replacement policies perform similarly. LRU performs well in this request file as LRU avoids swapping the frames, which are accessed quite often. The page faults settle at a constant value of around 45, which are the compulsory faults that occur at the beginning when the memory is empty. After a point, increasing the number of frames doesn't reduce the number of page faults, as there is enough space in memory such that nearly no swapping occurs.