CS-342

Operating Systems

Section: 1

PROJECT 3

Mehmet Hasat Serinkan

21901649

Mehmet Eren Balasar

22001954

04.12.2023

Frame count = 4, Input.txt lines=740

Algorithm ⇩	Page Fault Count
FIFO	616
LRU	616
CLOCK	618
ECLOCK	623

Frame count = 32, Input.txt lines=740

Algorithm ⇩	Page Fault Count
FIFO	570
LRU	571
CLOCK	572
ECLOCK	585

Frame count = 128, Input.txt lines=740

Algorithm ⇩	Page Fault Count
FIFO	494
LRU	484
CLOCK	480
ECLOCK	503

Frame count = 4, Input.txt lines=100000

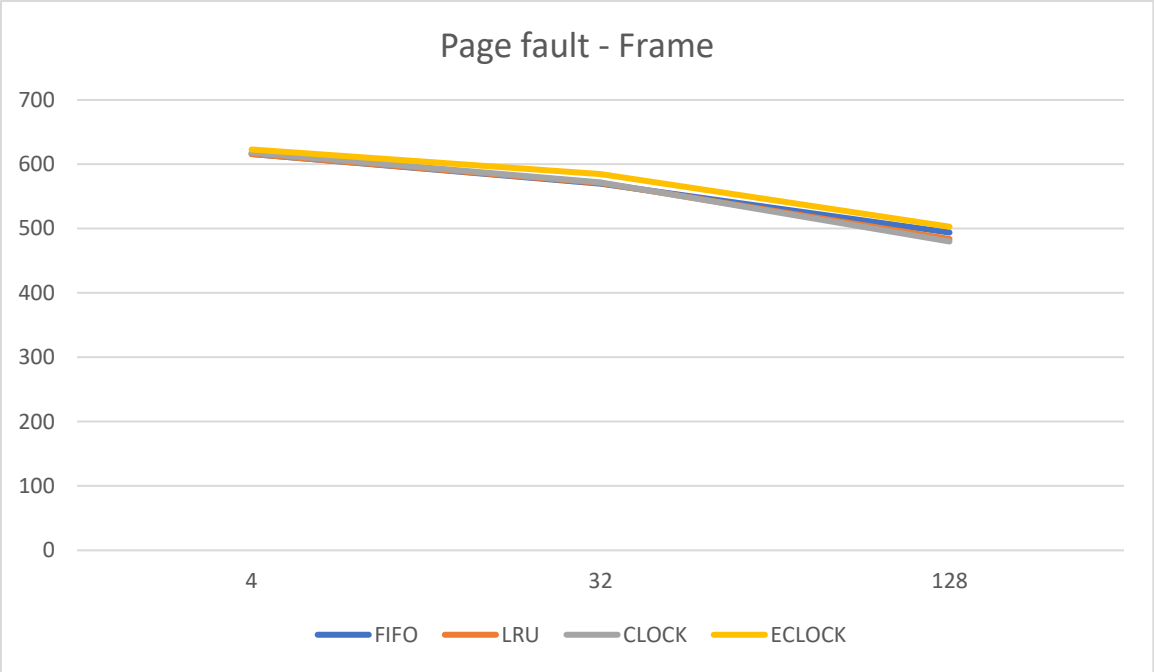
Algorithm ⇩	Page Fault Count
FIFO	99632
LRU	99633
CLOCK	99634
ECLOCK	99638

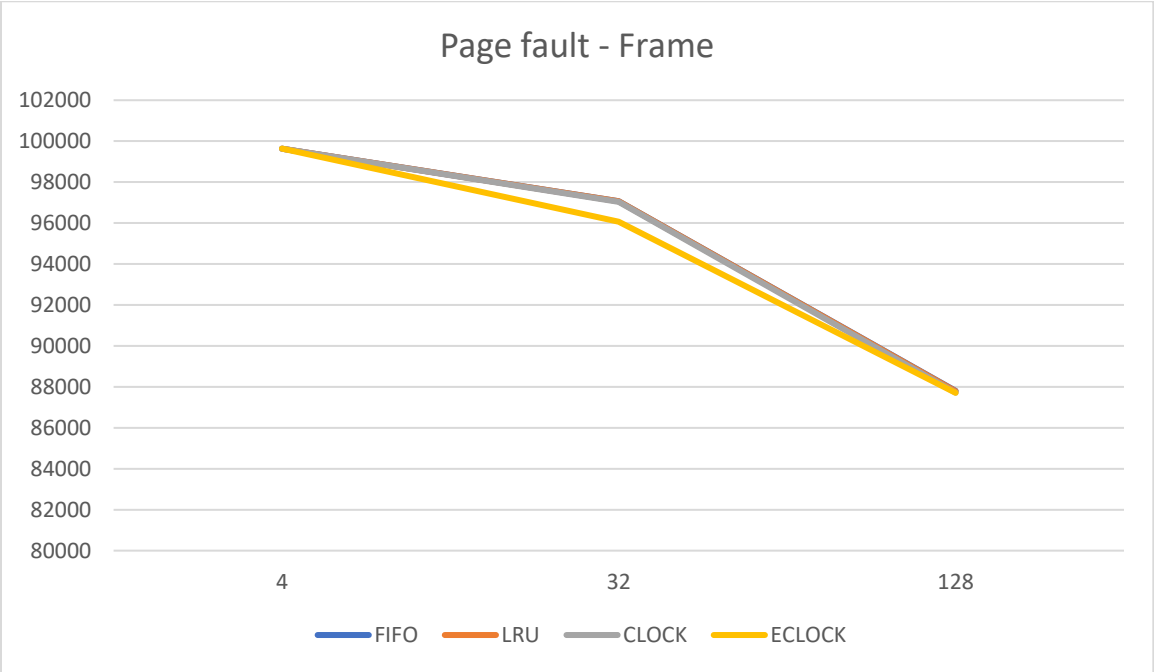
Frame count = 32, Input.txt lines=100000

Algorithm ⇩	Page Fault Count
FIFO	97067
LRU	97062
CLOCK	97042
ECLOCK	96982

Frame count = 128, Input.txt lines=100000

Algorithm ⇩	Page Fault Count
FIFO	87805
LRU	87788
CLOCK	87717
ECLOCK	87707





We have used two different address files. One of them was completely randomly generated by a Python program and had 100k lines. Other one was created by hand. While creating this file, we tried to simulate a real program's memory references, i.e. addresses are somewhat grouped and these groups generally contain relatively close addresses.

For 4 frames, all algorithms have a relatively high number of page faults due to the limited number of frames available.

As the number of frames increases to 32, the number of page faults decreases across all algorithms, which is expected because more physical memory is available to accommodate more pages.

At 128 frames, the number of page faults further decreases, showing that having more physical memory generally reduces the need for page replacements.

The Enhanced CLOCK (ECLOCK) algorithm seems to have a slightly higher number of page faults compared to CLOCK in all cases, which may be due to the overhead of considering both the reference and modified bits.

Unfortunately, we did not see much of a performance difference between algorithms, even though one might expect a better performance with a more complicated algorithm. The reason for this, most probably, might be the nature of our input files. We believe we could not generate input files that greatly resemble a real program's memory references. Nevertheless, we saw that LRU and CLOCK algorithms generally perform better, but one can argue that this slight performance increase is not worth the longer implementation-time, more logical complexity and the overhead of maintaining additional data structures.