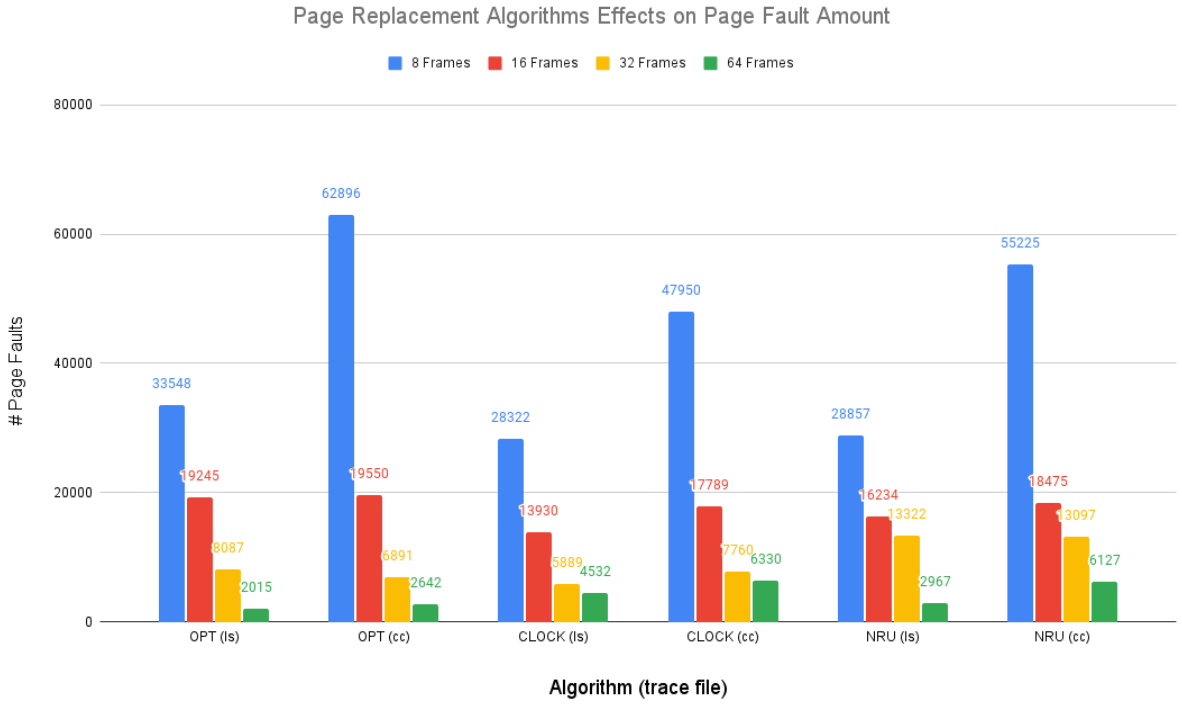
Christian Lester

22 April 2024

Collected Data

Algorithm	Trace file	8 frames	16 frames	32 frames	64 frames
OPT	cc	62896	19550	6891	2642
OPT	ls	33548	19245	8087	2015
CLOCK	cc	47950	17789	7760	6330
CLOCK	ls	28322	13930	5889	4532
NRU*	cc	55225	18475	13097	6127
NRU*	ls	28857	16234	13322	2967

*Refresh rate set to 128



Analysis

My expectations when gathering this data would be that OPT would outperform the other tests and algorithms by a noticeable margin. Since the implementation has perfect knowledge of what instructions are ahead and what to evict, I was surprised when my tests came back showing that at lower frame numbers, it was not as effective as I hoped. Eventually, the number of page faults

will lower to the point where the more frame numbers led to the lowest page faults among the algorithms.

For NRU, I decided to use a refresh rate of 128, which was in a good middle spot where it was not so low that there would be an incredibly high amount of page faults but not high enough where we could also run into the same issue. Although the data for clock and NUR are about the same, there are noticeably less page faults when running the clock algorithm, which could be due to the refresh rate I choose, but also for other reasons. The use of the second chance algorithm in clock allows for a consistent rate at which we evict, allowing for an optimal strategy towards marking our pages as invalid or dereferenced at a predictable rate.

For use in an operating system, the clock algorithm would be the best algorithm to use. While we can also get perhaps more desirable results with a certain refresh rate in NRU, clock is generally guaranteed to perform at its most optimal under most circumstances. Its use of a circular linked list allows for it to be easily updated and will be able to evict pages in a non-demanding and intuitive manner. It will allow for pages to have enough time to remain valid and could help prevent getting rid of certain pages too soon.