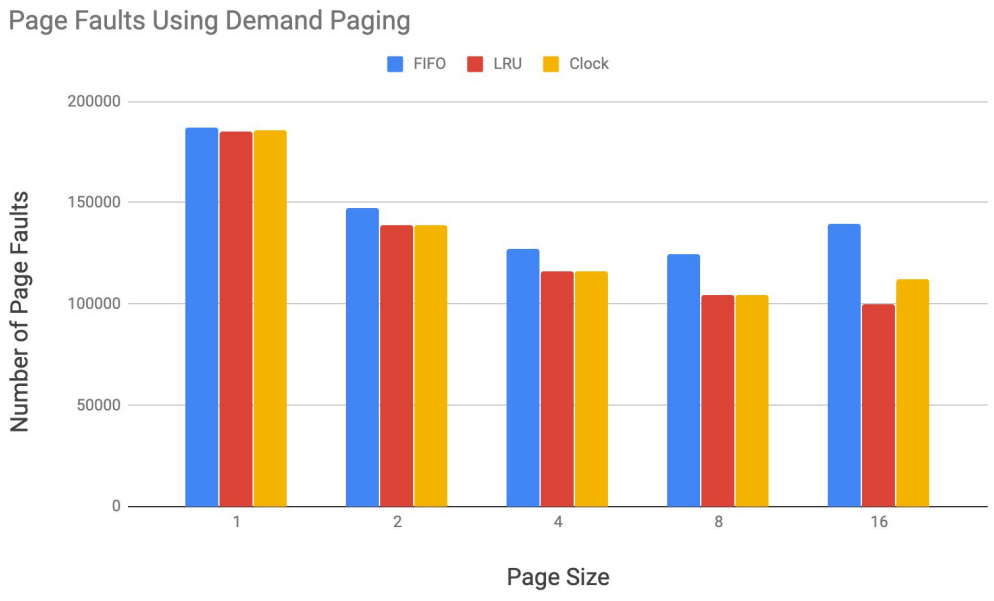
Team Morpheus: Daniel Kramer & Michael Curry
CS 492
Spring 2019

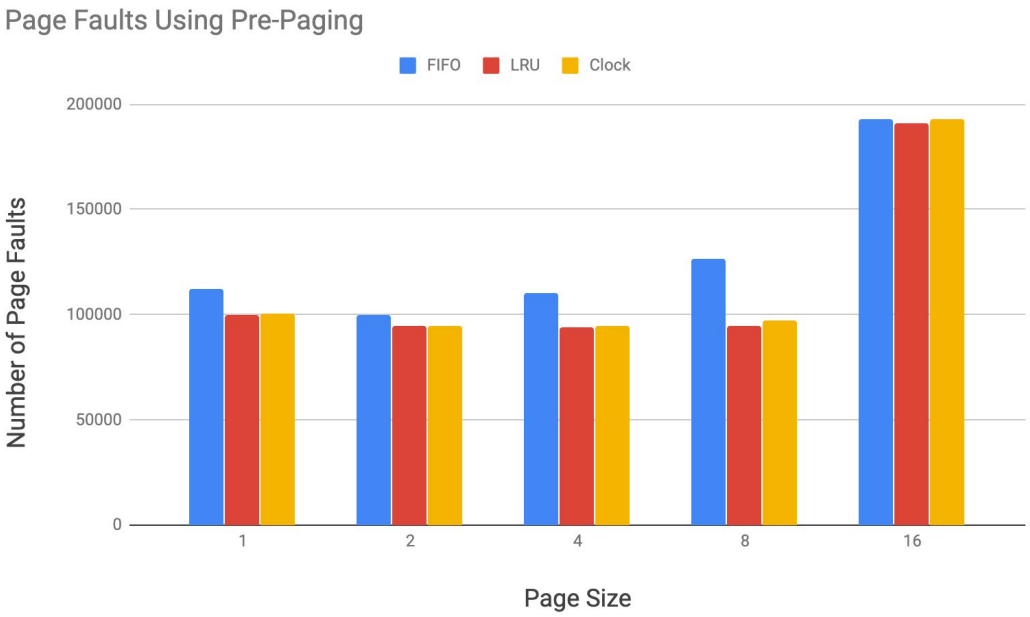
Homework 3 - Analysis of Results

Virtual Memory System Result Data:

Page Faults With Demand Paging	1	2	4	8	16
FIFO	186913	147253	127322	124787	139807
LRU	185335	138786	116020	104467	99676
Clock	185493	138913	116019	104567	111977
Page Faults With Pre-Paging	1	2	4	8	16
FIFO	111977	99858	110221	126365	192798
LRU	99983	94725	94206	94481	191286
Clock	100263	94935	94444	97077	192889

Result Graphs:





Explanation of Results:

The results we obtained from the output of our program closely matched our expectations for the overall performance of the three algorithms. In particular, we noticed that the Clock algorithm performed the same or better than the FIFO implementation across all of our test cases. We also observed that there was a dramatic increase in the number of page faults when using pre-paging with a page size of 16, likely caused from the constant swapping of the larger pages.

Implementing the algorithms was generally straightforward for the FIFO and LRU implementations because of the of the data structures we created to represent the page table. The clock algorithm proved to be more difficult due to the fact that we needed an additional state for the valid bit in order to prevent unnecessarily performing swaps. One significant difference in performance complexity is that LRU needs to scan the entire memory array to find the least recent access time across all the pages. This results in a generally lower amount of page faults as a trade off for introducing extra algorithmic complexity.

If using a completely random memory access trace, we would expect the results to contain a higher number of page faults, especially using a system with pre-paging enabled. This is because the loss of locality of reference will make it so that more pages need to be continuously swapped in and out of the main memory.