Parker Hague

Professor Aakur

CS 5323

18 April 2022

Assignment 4 Analysis

**Data Structure**

For my implementation, I went with a simple data structure. I decided to use an array as the primary data structure to store my frames in the working set. I chose an array because it’s simple to use and easy to implement. Though, I know that this isn’t the most optimal data structure in terms of performance. The array has O(n) lookups whereas a data structure like a hash table has O(1) lookups. I think it’s okay to take the performance hit in this scenario because our working set will never get bigger than a size of 5-10 realistically. The O(n) lookup would be a much bigger problem in a scenario where our n could be really big or if it’s unknown and unpredictable.

**Implementation**

I implemented the FIFO replacement algorithm by using a pointer that keeps track of the location of the victim frame. The pointer starts at zero and increments to the next array location after each page miss in a circular manner.

LRU was very straightforward as well. To find my victim frame, I used an array to keep track of the index that each memory location in the working set was read at. I then looped over these indices and looked for the minimum value. The memory address with the minimum index would indicate that it was used the longest ago or least recently. This was the page that I removed.

**Analysis**

This program was interesting to test. As expected, the LRU algorithm greatly outperformed the FIFO algorithm. For example, when using four frames in the working set, the bzip.txt file had 27.9% fewer page faults when going from FIFO to LRU. The same test on the gcc.txt file saw a 19.5% decrease in page faults. These differences in page fault occurrences are quite large and really demonstrate that LRU is the superior algorithm.

**Belady’s Anomaly**

I ran 100 tests across the files gcc.txt and bzip.txt. I tested each file for Belady’s anomaly by running the program with 1-50 frames in the working set. Every time the number of frames in the working set was increased, the number of page faults decreased. After n=50 for each file, I was not able to observe Belady’s anomaly occurring. The figures below show the number of page faults and how they decreased as the number of frames increased. They also give us an idea as to the ideal number of page frames in the working set. For example, for both files, we can tell that the number of page faults is dramatically decreasing until around the n=9-11 range. After that, the number of page faults begins to converge and plateau. This indicates that it might not be that efficient to have more than 10 frames or so in the working set. Of course, this may apply to only these trace files as each one is very unique.

**Table

Description automatically generated**