Operating Systems
HW-4
Selin Dinç
150150229

Q1)

Today, there are so many operating systems and many page replacement algorithms. However, only a some of the page replacement algorithms are used by today's operating systems.

Ubuntu uses Least Recently Used (LRU) algorithm for page replacement. LRU algorithm works on the idea that the pages that have been mostly used in the past instructions are most likely to be used heavily in the next instructions. That is why when a page needs to be replaced that is going to be the page that has not been referenced for the longest time. LRU algorithm performs almost optimally in theory. However, it is expensive to implement in practice.

Windows uses page replacement algorithms based on the type of the processor. On uniprocessors, Windows uses a variation of the clock (second chance) algorithm. On multiprocessor systems, Windows uses a variation of First in First Out (FIFO) algorithm.

Clock (second chance) page replacement algorithm works by keeping a circular list of pages in the memory. There is a pointer, set to the last examined page frame in the list and a reference bit for each frame. Reference bit is set to 1 when a page is first loaded into the frame or the corresponding page is referenced. If it's time to replace a page, reference bit started to inspect at the pointer's location. The first frame encountered with the reference bit set to 0 is replaced. While searching for replacement, each reference bit set to 1 is changed to 0.

FIFO page replacement algorithm requires keeping track of all the pages in memory in a queue. In the queue, the most recent arrival is at the back and the oldest arrival is at the front. If a page needs to be replaced, oldest page (the one at the front) is selected.

Android also uses Least Recently Used (LRU) algorithm for page replacing. LRU picks the least recently used page for replacement as I mentioned above.

Mac-OS also uses a FIFO like page replacement algorithm. There is a threshold value for comparing the number of the pages at the free list. This aims to balance the number of pages in the free list and the inactive list. The page replacement occurs by moving page from the inactive list to free list when the number of pages in the free list is below the threshold value. However, there is always a small number of pages at the inactive list. Until this part it's results like FIFO algorithm. The difference is if a page in the inactive list is faulted on then it's reactivated. This allows the inactive list to give the page a second chance to be replaced. if the free list size surpasses the threshold value, then the pager rests.

IOS also uses First in First Out like algorithm. However, in IOS kernel doesn't write pages out to backing store. If the amount of free memory drops below the threshold value, kernel removes pages that are inactive and unmodified and may also remove the running application to free up the memory.

Q2)

First in First Out algorithm is one of the simplest one between page replacement algorithms. Windows, MacOS and IOS use this algorithm for page replacement. This algorithm keeps track of all pages in the memory in a queue. If a page needs to replace, then the oldest one (the one in the front of the queue) selected for removal. That is why no future prediction is required. This algorithm doesn't have much overhead, because it doesn't need to keep track of the pages. Although this algorithm seems cheap and intuitive in theory, it's performs poorly in practical application. In real, when FIFO algorithm is selected as page replacement algorithm, it's generally is a modified form. This algorithm suffers from Belady's anomaly. Belady's anomaly happens when the number of page frames increase and that results in an increase in the number of page fault. Until this anomaly, it was generally believed that increasing the number of page frames would result in the same number of fewer page faults. However, FIFO algorithm disproves that idea. As a result, despite easy to understand and execute, this algorithm is not that effective because used pages can still be removed, needs to keep track of each page and can perform abnormally. Also, replacing the pages badly may increase the page fault rate and this results slower execution.

Least Recently Used (LRU) is a page replacement algorithm, which is used by operating systems like Ubuntu and Android. LRU algorithm stands that, the most heavily used pages in the past few instructions probably are going to be used heavily in the next few instructions. That is why when a page needs to be replaced that is going to be the page that has not been referenced for the longest time. Although this algorithm works nearly optimal in theory, it's very expensive to implement in practice. LRU algorithm needs to keep track of when the pages were used. This makes this algorithm more overhead than the FIFO. While implementing this algorithm, several methods are using. These methods generally aim to decreasing the cost but keeping the performance high as much as possible. However, some methods like linked list, increase the cost to highest. In this method, linked list contains all the pages in the memory. Linked list keeps the least recently used page at the back and most recently used page at the front. This method costs the most expensive because, pages in the list will have to move with every memory reference that will be made. That makes this method time consuming. Number of page faults in LRU algorithm are less than FIFO algorithm. That means LRU algorithm has more hits than FIFO algorithm because, LRU keeps the most recently used pages in the memory. Also, LRU algorithm doesn't suffer from Belady's anomaly like FIFO does. That's means, in the LRU, when the number of page frames increased, page fault decreases. As a result, there is no perfect page replacement algorithm in general. The perfect page replacement algorithm requires perfect knowledge of the future, which is impossible with today's technology.

References

Page replacement algorithm. (2018, November 26). Retrieved from https://en.wikipedia.org/wiki/Page_replacement_algorithm#First-in,_first-out

Memory Usage Performance Guidelines. (2013, April 23). Retrieved from https://developer.apple.com/library/archive/documentation/Performance/Conceptual/ManagingMemory/Articles/AboutMemory.html

Strickland, J. (2011, October 17). What's Ubuntu, and how is it different from Linux? Retrieved from https://computer.howstuffworks.com/ubuntu2.htm