Operating Systems

Homework 7 — Virtual Memory

Abraham Murciano

February 27, 2020

1. (a) Given n distinct page numbers, there must be at least n page faults, since each page must be initially mapped the first time it is referenced.

   (b) If there are p references, then there will be up to p page faults, since each reference may cause a page fault if there is too little memory and each page is swapped out before it is referenced each time.

2. The question is unclear. The question references a 'given computer system', but none it given.

3. The benefits of virtual memory are:

   - Each process has the illusion that it has control over the entire address space.
   - No process can affect the memory of another process because the operating system will map each virtual address to only memory that the process is allowed to access, otherwise it can kill the process.
   - It is possible to have a larger virtual address space than the amount of physical memory using techniques such as swapping pages.

   The costs of using virtual memory are:

   - Additional hardware must be used to map virtual addresses to physical ones in a speedy fashion.
   - It takes additional time to reference a location in memory since each logical address must be translated to a physical address.
   - The operating system must hold data structures (such as a page table if using the paging technique) to store information about the virtual addresses. This consumes additional memory.

   To ensure that the costs of using virtual memory don't exceed the benefits, it is important that the address translation from the virtual to physical address space is as fast as possible, which will often require the operation to be performed by specialized hardware. If memory is being swapped in and out, the operating system must try and minimize the number of page faults by choosing which pages to swap out very carefully. Furthermore, it should ensure that the data structures required to map addresses be as memory efficient as possible.

4. (a) There will be 5000 page faults in the first piece of code, because there will only be two consecutive memory accesses in the same page before it is swapped out for another page.

(b) There will be 50 page faults since for each page that is swapped in, there will be 200 consecutive references to an address on that page, after which that page can be swapped out never to be swapped in again.

5. If the hardware supports a validity bit, then this can be used to function as a reference bit. The first time a reference is made, a software interrupt can be sent to the operating system. The operating system can then set the validity bit to 1, and will have to also keep a software bit of its own which it will also set to 1.

   The cost of doing this would be the time taken for the trap to be handled.

6. The number of page faults that would occur given the page reference string

   $$1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6$$

   depending upon the algorithm used and the number of frames available would be:

Frames	1	2	3	4	5	6	7
LRU	20	18	15	10	8	7	7
FIFO	20	18	16	14	10	10	7
Optimal	20	15	11	8	7	7	7

7. Given that the degree of multiprogramming is fixed at four:

   (a) if CPU utilization is at 13 percent and disk utilization 97 is at percent, then the system is thrashing. We cannot increase the degree of multiprogramming and the paging is not helping; it is slowing the system down.

   (b) if CPU utilization is at 87 percent and disk utilization is at 3 percent, the CPU utilization is good, and the paging algorithm is good because there are few page faults. The degree of multiprogramming can be increased.

   (c) if CPU utilization is at 13 percent and disk utilization is at 3 percent, the CPU utilization is too low, so the degree of multiprogramming should be increased. The paging algorithm is good because there are few page faults.

8. A TLB miss without a page fault can occur when the page has been swapped into main memory but the page has been removed from the TLB.

   A TLB miss with a page fault can occur when a page that has been swapped out and has been removed from the TLB is referenced.

   A TLB hit without a page fault can occur when the page has been recently referenced, so it is in main memory and in the TLB.

   A TLB hit with a page fault cannot occur, since the TLB keeps a subset of the pages which are in main memory. Therefore a page will never be in the TLB if it isn't in main memory.

9. There must be a mistake in the question. Surely the computer cannot have only 222 bytes of physical memory, especially if the page size is 4096 bytes.

10. If a computer can hold three pages in memory, and the pages referenced are

    $$1, 1, 2, 3, 4, 1,$$

then by the time page 4 is referenced, page 1 has a frequency of 2 and all the other pages have a frequency of 1. Therefore page 2 will most likely be swapped out by the LFU algorithm. Then when page 1 is again referenced, it would not cause a page fault since page 1 is still in main memory. However, the LRU algorithm would swap out page 1 when page 4 is swapped in, so when page one is referenced immediately after, it would cause a page fault.

However if the order of page references is

$$1, 1, 2, 3, 4, 2,$$

then LFU would swap page 2 out, and then when page 2 is referenced immediately afterwards a page fault would occur. However LRU would swap out page 1, so there would not be a page fault when page 2 is referenced.

11. Consider a demand-paging system with the following time-measured utilizations:

    - CPU utilization 20
    - Paging disk 97.7
    - Other I/O devices 5

    (a) Installing a faster CPU would not help, since the current CPU is sitting idle most of the time anyway.

    (b) Installing a bigger paging disk is unlikely to help, since the problem is the main memory, not the swap space.

    (c) Increasing the degree of multiprogramming will not help, since the system is currently thrashing.

    (d) Decreasing the degree of multiprogramming is the solution to reduce thrashing.

    (e) Installing more main memory certainly would help, since currently there are too many page faults. More main memory will reduce that.

    (f) Installing more or faster hard disks would make the system faster if they are used for swapping, since they will make the page faults be able to be resolved faster.

    (g) Adding pre-paging is likely to help since it will reduce the number of page faults

    (h) Increasing the page size may help, since each time a context switch occurs and the new process's pages begin to be referenced, they will have to be swapped in. Since each process now uses less pages, less page faults would occur, but each page fault would take longer to handle.