Student: Yu Cheng (Jade)

ICS 412

Homework #6

November 25. 2009

Homework #6

Exercise 9.15: A simplified view of thread states is *Ready*, *Running*, and *Blocked*, where a thread is either ready and waiting to be scheduled, is running on the processor, or is blocked (i.e. is waiting for I/O). This is illustrated in the following figure. Assuming a thread is in the Running state, answer the following questions (Be sure to explain your answer).

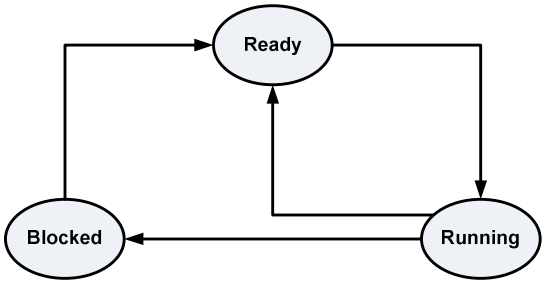


Figure. Thread state diagram for Exercise 9.15

a. Will the thread change state if it incurs a page fault? If so, to what new state?

Answer: Yes, a tread changes from the *Running* state to the *Blocked* state when a page fault occurs.

When a page fault occurs, the process starts to wait for I/O operation to finish. The OS does several things while the process is waiting. It checks whether the page is really invalid or just on disk, finds a free memory frame, schedules a disk access to load the page into t the frame, updates the page table with the new logical-physical mapping, updates the valid bit of that entry, and eventually restarts the process by change its state from *Blocked* to *Ready*.

b. Will the thread change state if it generates a TLB miss that is resolved in the page table? If so, to what new state?

Answer: Not necessarily. If a page table entry is not found in the TLB (TLB miss), the page number is used to index and process the page table. If the page is already in main memory, then TLB is updated to include the new page entry, while the process execution continues since there is no I/O operation needed. If the page is not in the main memory, a page fault is generated. In this

case, the process needs to change to the *Blocked* state and wait for I/O to access the disk. This is the same procedure as in the first question.

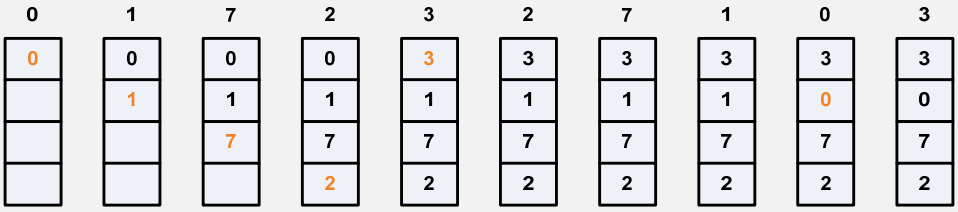
c. Will the thread change state if an address reference is resolved in the page table? If so, to what new state?

Answer: No, because no I/O operation is needed is the address reference is resolved in the page table, which indicates the page needed is loaded in the main memory already.

Exercise 2: Consider a system with four pages frames and a program that uses eight pages. Consider the reference string 0172327103 and assume that all four page frames are initially empty. How many page faults occur assuming:

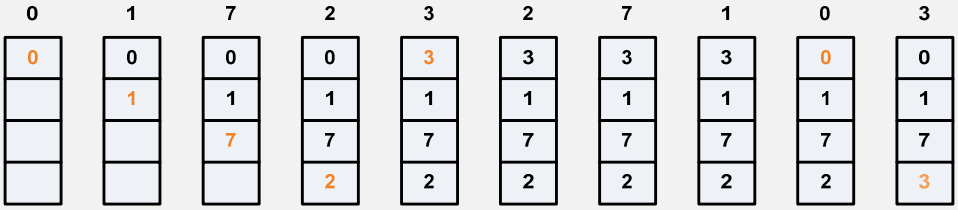
a. FIFO page replacement.

Answer: Six page faults will be generated. The orange colored accessing incurred page faults.



b. LRU page replacement.

Answer: Seven page faults would be generated. The orange colored accessing incurred page faults.



Exercise 3: Consider a demand-paging system with the following time-measured utilizations:

- CPU: 20%
- Paging disk (accesses): 97.7%
- Other I/O devices (accesses): 5%

For each of the following say whether it is likely to improve CPU utilization or not, or whether it is not clear. Explain your answer:

a. Install a faster CPU

Answer: No. The given system's CPU is underutilized, and it spends a lot of time accessing disk. This tells us that the system is thrashing. CPU is not the where the problem is, so a faster CPU will not help.

b. Install a bigger paging disk

Answer: Not necessarily, thrashing is more likely caused by the main memory capacity and disk accessing speed. Page disk size might not be where the problem is.

c. Increase the degree of multiprogramming

Answer: No. Increasing the degree of multiprogramming means even fewer frame per process. It worsen the thrashing. All processes might not be able to get their pages in a reasonable speed, and nobody gets to run smoothly.

d. Decrease the degree of multiprogramming

Answer: Yes. Decreasing the degree of multiprogramming means more frames per process. It reduces the thrashing. The remaining processes might be able to get their pages in a reasonably fast speed, and get to run smoothly.

e. Install more main memory

Answer: Yes. Installing more main memory means more frames per process. It reduces the thrashing. The processes might be able to get their pages in a reasonably fast speed, and get to run smoothly.

f. Install a faster hard disk

Answer: No. The problem is not the size of hard drive. Trashing is caused by the main memory capacity and the accessing speed.

g. Increase the page size

Answer: No likely. Increasing page sizes doesn't change the fact that the physical memory for each process's pages is not big enough. Plus, increasing pages sizes would cause a longer accessing time to the disk. The extra bits loaded from the disk increases, which is a waste.

Exercise 9.21: Assume we have a demand-paged memory. The page table is held in registers. It takes 8 milliseconds to service a page fault if an empty page is available or the replaced page is not modified and 20 milliseconds if the replaced page is modified. Memory access time is 100 nanoseconds. Assume that the page to be replaced is modified 70 percent of the time. What is the maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?

Answer: Let's used variable x for the page-fault rate. The average time consumption for pages that are available on the main memory is,

$$(1 - x) \times 100 \cdot 10^{-9}$$

The average time consumption for page faults if empty pages are available or the replaced pages are not modified is,

$$x \times (1 - 70\%) \times 8 \cdot 10^{-3}$$

The average time consumption for page faults if the replaced pages are modified is,

$$x \times 70\% \times 20 \cdot 10^{-3}$$

Therefore the total average access time is no more than 200 nanoseconds indicates,

$$(1 - x) \times 100 \cdot 10^{-9} + x \times (1 - 70\%) \times 8 \cdot 10^{-3} + x \times 70\% \times 20 \cdot 10^{-3} \leq 200 \times 10^{-9}$$

$$\Rightarrow -x + 140000x + 24000x \leq 1$$

$$\Rightarrow 163999x \leq 1$$

$$\Rightarrow x \leq 6.1 \times 10^{-6}\,.$$

The maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds is approximately 6.1×10^{-6}.