Laboratory Exercise 5 Solutions

cpe 453 Spring 2010

Due by (or before) 11:00am, Friday, May 28th
The Written Exercises (problems) are to be done individually.

Problems

1. (T&W 4-1) A computer system has enough room to hold four programs in its main memory. These programs are idle waiting for I/O half the time. Assuming that all I/O operations are distributed randomly with even probability—that is, any given process is blocked waiting for I/O with probability 1/2 at any given time in its execution—what proportion of the time will you expect the CPU to be idle?

   Solution:

   The probability that the CPU will be idle is the same as the probability that all four processes will be blocked simultaneously. Since at each moment each processess had a 1/2 *probability of being blocked, the probability of all four being blocked simultaneously is:*

   $$\frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{16}$$

   Thus, you would expect the CPU to be idle 1/16th of the time.

2. (T&W 4-4) Using the page table of Fig. 4-8 in T&W, give the physical address corresponding to each of the following virtual addresses:

   (a) 20

   (b) 4100

   (c) 8300

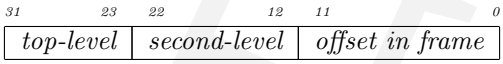
   Solution:

   The page size is 4096 (or 4k), so

Virtual				*Physical*		
Address	*Page*	*Offset*		*Frame*	*Offset*	*Address*
20	*0*	*20*	$\mapsto$	*2*	*20*	*8212*
4100	*1*	*4*	$\mapsto$	*1*	*4*	*4100*
8300	*2*	*108*	$\mapsto$	*6*	*108*	*24684*

3. (T&W 4-8) A computer with a 32-bit address uses a two-level page table. Virtual addresses are split into a 9-bit top-level page table field, an 11-bit second-level page table field, and an offset. How large are the pages and how many are there in the address space?

   Solution:
   This page table structure will look like:



   There are 12 bits of offset in each page, so pages are of size 2^{12}*, or 4096 bytes (4kB). Because there are 20 bits of combined page number, there are* 2^{20} *pages required to cover the entire address space.*

4. (T&W 4-14) A computer has four page frames. The time of loading, time of last access, and the R and M bits for each page are as shown below (the times are in clock ticks):

Page	Loaded	Last Ref.	R	M
0	126	279	0	0
1	230	260	1	0
2	120	272	1	1
3	160	280	1	1

   (a) Which page will NRU replace?

   Solution:
   NRU will replace 0 because it has been neither read nor modified in the last clock period.

   (b) Which page will FIFO replace?

   Solution:
   FIFO will replace page 2 (the oldest)

   (c) Which page will LRU replace?

   Solution:
   LRU will replace page 1 (last referenced at 260)

   (d) Which page will second chance replace?

   Solution:
   Second chance will replace page 0 because all of the others have been referenced in the last clock period.

5. (Based on T&W 4-15) If FIFO page replacement is used with four page frames and eight pages, how many page faults will occur with the reference string 0713272013 if the four frames are initially empty. What will be the final contents of memory?

Solution:

The final contents of memory are pages 0, 1, 2, and 3. There will have been 6 page faults. The evolution of memory is illustrated in Figure 1.

Time	Reference	Action	Frame Contents 0	1	2	3	FIFO Queue ←Older			
0	—	—	—	—	—	—	—	—	—	—
1	0	Miss. Rep. page 0	0	—	—	—	—	—	—	0
2	7	Miss. Rep. page 1	0	7	—	—	—	—	0	7
3	1	Miss. Rep. page 2	0	7	1	—	—	0	7	1
4	3	Miss. Rep. page 3	0	7	1	3	0	7	1	3
5	2	Miss. Rep. page 0	2	7	1	3	7	1	3	2
6	7	Hit.	2	7	1	3	7	1	3	2
7	2	Hit.	2	7	1	3	7	1	3	2
8	0	Miss. Rep. page 1	2	0	1	3	1	3	2	0
9	1	Hit.	2	0	1	3	1	3	2	0
10	3	Hit.	2	0	1	3	1	3	2	0

Figure 1: FIFO Page replacement for problem 5.

6. Repeat Problem 5 for LRU replacement.

Solution:

The final contents of memory are pages 0, 1, 2, and 3. There will have been 8 faults. The evolution of memory is illustrated in Figure 2.

Time	Reference	Action	Frame Contents 0	1	2	3	LRU Stack ←Older			
0	—	—	—	—	—	—	—	—	—	—
1	0	Miss. Rep. page 0	0	—	—	—	—	—	—	0
2	7	Miss. Rep. page 1	0	7	—	—	—	—	0	7
3	1	Miss. Rep. page 2	0	7	1	—	—	0	7	1
4	3	Miss. Rep. page 3	0	7	1	3	0	7	1	3
5	2	Miss. Rep. page 0	2	7	1	3	7	1	3	2
6	7	Hit.	2	7	1	3	7	1	3	2
7	2	Hit.	2	7	1	3	7	1	3	2
8	0	Miss. Rep. page 2	2	7	0	3	7	3	2	0
9	1	Miss. Rep. page 3	2	7	0	1	7	2	0	1
10	3	Miss. Rep. page 1	2	3	0	1	2	0	1	3

Figure 2: LRU Page replacement for problem 6.

7. (T&W 4-20) It has been observed that the number of instructions executed between page faults is directly proportional to the number of page frames allocated to a program. If the available memory is doubled, the mean interval between page faults is also doubled. Suppose that a normal instruction takes 1 microsecond, but if a page fault occurs, it takes 2001 microseconds (i.e. 2 msec to handle the fault). If a program takes 60 seconds to run, during which time it gets 15,000 page faults, how long would it take to run if twice as much memory were available?

   Solution:

   Dissecting the problem, we see that the program took $15\,000$ *page faults at a cost of* $2000\mu sec$ *each for a total of 30 seconds:*

   $$15\,000 \times 2\,000\mu sec = 30\,000 msec = 30s$$

   The remaining 30s is time that is required to simply execute the instructions. Thus, there are $30\,000\,000$ *total instructions to be executed at* $1\mu sec$ *per instruction and a mean interval between page faults of 2000 instructions.*

   Given the above claim, doubling the available memory should double the mean interval between to page faults to be 4000 instructions. This gives us 30 000 000 / 4000 or 7500 expected page faults. The total cost of execution should then be given by:

   $$\begin{aligned} Total\ Time &= 30\,000\,000 \times 1\mu sec + 7500 \times 2msec \\ &= 30\,000 \times 1msec + 15000 \times 1msec \\ &= 45\,000 \times 1msec \\ &= 45\,000 msec \\ &= 45s \end{aligned}$$

What to turn in

For the Written Problems: individually written solutions to the problems according to the guidelines set forth in the syllabus.