Homework 2 - System Programming

Instructor for CS 241: Klara Nahrstedt
Deadline, May 3, midnight
Department of Computer Science
University of Illinois, Urbana Champaign

The homework 2 is an **individual effort activity!! (no pairs)**. Each student submits his/her own solution in the electronic form using the PDF format. The format should use 11pt or 12pt font. Each student submits his/her work to Illinois Compass in the same manner as he/she submitted machine problems:

1. Log into Illinois Compass (`http://compass.uiuc.edu`) and select cs241 from the *Course List*.
2. Select HW2 Handin from the CS 241 Home Page
3. Click the *Add Attachments* button in the *Submissions* section.
4. In the *Choose Files* window, click *My Computer*.
5. Select your PDF document in the file selection dialog.
6. Click *Submit*, then check *OK* in the confirmation alert.
7. On the *Confirmation* page, click *Continue*.

1 Problems on Input/Output (25 Points)

1. (9 Points) Although DMA does not use the CPU, the maximum transfer rate is still limited. Consider reading a block from the disk. Name three factors that might ultimately limit the rate transfer.

   Solution: The first factor could be the limiting speed of the I/O device - transfer rate between the device and the main computer, second factor could be the speed of bus, the third factor could be no internal buffering on the disk controller or too small internal buffering space, and forth factor could be erronoues disk or transfer of block (i.e., if the checksum is incorrect, an error is signalled and no transfer of the block happens, new block must be found, or block must be retransmitted.

2. (10 Points) An alternative to interrupts is polling. What are the cicumstances in which polling is better than interrupts? Give at least two circumstances.

   Solution: Polling could be a better choice in very small dedicated systems that are not running multiple processes. Polling is beneficial in situations if the anticipation is that only very short waiting is expected. This can happen in device drivers, other very low level software where synchronization is very common and executed in very short time intervals.

3. (6 Points) Many disks contain an ECC at the end of each sector. If the ECC is wrong, what actions might be taken and by which piece of hardware or software?

   Solution: The ECC means Error-Correcting Code which represents the checksum of the information in the sector. The ECC checksum is used to verify the block of bytes coming from the disk in the disk controller. If the checksum is incorrect, the controller signals to DMA (or CPU depending who sent the request for disk block) that an error occured and no transfer to memory is done.

2 Problem on Basic Memory Management (30 Points)

1. (15 Points) In this problem you are to compare the storage needed to keep track of free memory using a bitmap versus using linked list. The 128 MB memory is allocated in units of n bytes. For the linked list, assume that memory consists of an alternating sequence of segments and holes, each 64 KB. Also assume that each node in the linked list needs 32-bit memory address, a 16-bit length, and a 16-bit next-node field. How many bytes of storage are required for each method? Which one is better?

   Solution: The bitmap needs 1 bit per allocation unit. (128MB = 2^{27}, 64 KB = 2^{16}). With $2^{27}/n$ allocation units (bits), this is $(2^{27}/2^3)/n = 2^{24}/n$ bytes. The linked list has $2^{27}/2^{16}$ or 2^{11} nodes, each of 8 bytes (4 bytes for memory address, 2 bytes length and 2 bytes next-node filed) for a total of $2^{11} * 2^3 = 2^{14}$ bytes. For small n, the linked list is better. For large n, the bitmap is better. The cross-over point can be calculated by solving equation: $2^{24}/n = 2^{14}$, $n = 2^{10}$bytes, i.e., 1KB. So if $n < 1KB$, linked list is better, for $n > 1KB$, bit map is better.

   Of course the assumption about segments and holes alternating every 64KB is very unrealistic. Also, we need $n \leq 64KB$ if the segments and hole are 64KB.

2. (12 Points) Consider the swapping system in which memory consists of the following hole sizes in memory order: 10KB, 4KB, 20KB, 18KB, 7KB, 9KB, 12KB, and 15 KB. Which hole is taken for successive segment requests of (a) 18KB, (b) 2KB, (c) 13 KB for the following policies: first fit, best fit, and worst fit? Specify clearly what is the list of holes after the allocation of the segment requests (a), (b) and (c) for each policy.

   Solution: Note that the three requests need to be considered together one after another for each policy.

   First fit: for the sequence of requests 18KB, 2KB and 13KB, the policy will choose: 20KB hole for 18KB request (we would have after this allocation holes 10, 4, 2, 18, 7,9,12, 15), 10KB hole for 2 KB request (we would have then holes 8,4,2,18,7,9,12, 15), and 18KB for 13 KB request (we would then have the holes 8,4,2,5,7,9,12, 15).

   Best fit: 18KB hole (10,4,20,0,7,9,12,15); 4KB hole (10,2,20,7,9,12,15), 15KB hole (10,2,20,7,9,12,2)

   Worst fit: 20KB (10,4,2,18,7,9,12,15), 18KB (10,4,2,16,7,9,12,15), 16KB (10,4,2,3,7,9,12,15)

3. (3 Points) What is the difference between a physical address and a virtual address?

   Solution: Real memory uses physical addresses. These are the numbers that the memory chips react to on the bus. Virtual addresses are the logical addresses that refer to a process' address space. Thus a machine with a 16-bit word can generate virtual addresses up to 64K, regardless of whether the machines has more or less memory than 64KB.

3 Problem on Virtual Memory Management (20 Points)

1. (5 Points) Let us assume processes with 1024 pages in their address spaces. Furthermore, we assume that each process keeps its page tables in memory. The overhead required for reading a word from the page table is 5 nsec. To reduce the overhead of accessing page table, the computer has a TLB, which holds 32 (virtual page, physical page frame) pairs, and can do a look up in 1 nsec. What is the minimum hit rate needed to reduce the mean overhead (of accessing the page table) to at most 2 nsec?

   Solution: The efficient access time is $T = (b)h + (a + b)(1 - h)$, where a is access to the page table in main memory, b is the access to page table entires in TLB, and T is the effective access time to the page table. In our case $a = 5nsec$, $b = 1nsec$. The goal is to find h, if $T = 2nsec$, $2 = 1h + 6(1 - h), h = 3/4 * 100\% = 75\%$. So the minimum hit rate is 75%

2. (5 Points) A machine has 48-bit virtual addresses and 32-bit physical addresses. Pages are 8KB. How many entries are needed for the page table? Provide the exact calculation how you determine the number of entries.

   Solution: With 8KB pages and a 48 bit virtual address space, the number of virtual pages is $2^{48}/2^{13}$, which is 2^{35} (which is about 34 billion).

3. (10 Points) A computer has four page frames. The time of loading, time of last access, and the R (reference bit) and M (modified bit which can be dirty bit = 1 or clean bit = 0) are as shown below (the times are in clock ticks):

   (a) Which page will FIFO replace?

Page	**Loaded**	**Last reference**	R	M
0	126	280	1	1
1	230	256	0	01
2	140	270	0	0
3	110	285	1	1

Table 1: *Load Information (Note: M=01 means that the page was not written to in this cycle, but it was written to in the previous cycle)*

(b) Which page will LRU replace?

(c) Which page will second chance replace?

(d) Which page will be replaced using 'page classes' replacement?

For each algorithm specify the evicted page and explain clearly why this page was evicted. (Note: If just the victim page will be specified, and no explanation will be given, points will be taken off.)

Solution:

(a) FIFO will replace the Page 3 because page 3 was loaded first. FIFO replaces the first come first evict page. So the Loaded column decided which page to evict.

(b) LRU will replace the Page 1 because page 1 was last referenced out of the 4 pages. LRU keeps the most recently used and evicts the least recently used. So the last reference column decided which page to evict.

(c) Second chance evicts the Page 2. The reason is that the last reference column determines the order, where page 1 was first referenced and page 3 last referenced (order according to last reference time, i.e., page 3, page 0, page 2, page 1). In the FIFO order we go: page 3 has R bit set one so it gets second chance (R=0), but it will not be evicted. page 0 has R=1, so it gets second chance, R=0, but it will not be evicted. The next page 2 has R=0, so it will be evicted - no second chance.

(d) Page classes algorithm determines that the first victim will be a page that has R=0 and M=0 should be evicted. In this case it is page 2.

4 Problems on File Systems (25 Points)

1. (9 Points) Consider the i-node structure with 10 direct addresses and one indirect address. If the i-node contains 10 direct addresses of 4 bytes each and all disk blocks are 1024 Bytes, what is the largest possible file?

   Solution: With the 10 direct addresses we get file size of 10K. With the single indirect address we can point to one block of addresses which means (1024/4=256) 256 addresses to blocks. This also means that we can address 256KB with the indirect addressing. All together we can have the largest possible file of 256+10 = 266KB.

2. (16 Points) The beginning of a free space bitmap looks like this after the disk partition is first formated: 1000 0000 0000 0000 (the first block is used by the root directory). The system always searches for free blocks starting at the lowest numbered block, so after writing file A, which uses 6 blocks, the bitmap looks like this: 1111 1110 0000 0000. Show the bitmap after each of the following additional actions:

   (a) File B is written, using 5 blocks;
   (b) File A is deleted;
   (c) File C is written, using 8 blocks;
   (d) File B is deleted.

   Solution:

   (a) 1111 1111 1111 0000
   (b) 1000 0001 1111 0000
   (c) 1111 1111 1111 1100
   (d) 1111 1110 0000 1100