HW6

**DUE DATE: Friday April 17 2020 11 PM.** Submit MS Word or PDF file using Canvas. **You must show how the answer was obtained by giving the main steps in the procedure.**

**Problem 1** (17 points) Recall the various deadlock detection and prevention algorithms we’ve discussed in this course. The following is the state of a system with these processes: P1, P2, P3, P4, P5 and these resources: R1, R2, R3, R4.

There are no current outstanding queued unsatisfied requests.

Currently Available Resources

|  |  |  |  |
| --- | --- | --- | --- |
| **R1** | **R2** | **R3** | **R4** |
| 2 | 1 | 2 | 0 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Current Allocation** | | | | **Max Need** | | | | **Still Needs** | | | |
| **Process** | **R1** | **R2** | **R3** | **R4** | **R1** | **R2** | **R3** | **R4** | **R1** | **R2** | **R3** | **R4** |
| **P1** | 0 | 0 | 3 | 4 | 6 | 6 | 5 | 6 | 6 | 6 | 2 | 2 |
| **P2** | 2 | 0 | 0 | 0 | 2 | 7 | 5 | 0 | 0 | 7 | 5 | 0 |
| **P3** | 0 | 0 | 1 | 2 | 0 | 0 | 3 | 2 | 0 | 0 | 2 | 0 |
| **P4** | 2 | 3 | 5 | 4 | 4 | 3 | 5 | 6 | 2 | 0 | 0 | 2 |
| **P5** | 0 | 3 | 3 | 2 | 0 | 6 | 5 | 2 | 0 | 3 | 2 | 0 |

1. Is this system currently deadlocked, or can any process become deadlocked? Why or why not? If not deadlocked, give an execution order.

No, the system is not deadlocked because P3 is able to immediately use free resources and release its current allocation.

Execution order: P3 🡪 P4 🡪 P5 🡪 P2 🡪 P1

1. If a request from a process P3 arrives for (0, 4, 2, 0), should the request be immediately granted? Explain why or why not? If yes, show an execution sequence.

No, because the request exceeds the need AND there is not enough resources available.

iii) If a request from a process P2 arrives for (0, 1, 2, 0), should the request be immediately granted? Explain why or why not? If yes, show an execution sequence.

No, because granting the request would result in an unsafe state.

**Problem 2** (10 points) A system with virtual memory has these page accesses: 1 4 3 2 4 3 2 1 4 1 1 3 1 2 and that there are three frames in the system. Show how the **FIFO** replacement algorithm will work, and what will be the final contents of the three frames following the execution of the given reference string? Use the table below to the successive contents of the frames. When there is no page faults, leave the column blank.

**Answer**: Final contents: F0=3, F1=2, F2=4

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 4 | 3 | 2 | 4 | 3 | 2 | 1 | 4 | 1 | 1 | 3 | 1 | 2 |  |
| f0 | 1 | 1 | 1\* | 2 |  |  |  | 2 | 2\* |  |  | 3 |  | 3 |  |
| F1 |  | 4 | 4 | 4\* |  |  |  | 1 | 1 |  |  | 1\* |  | 2 |  |
| F2 |  |  | 3 | 3 |  |  |  | 3\* | 4 |  |  | 4 |  | 4 |  |

**Problem 3** (12 points) Given these sequential page accesses: 1 2 3 4 2 3 4 1 2 1 1 3 1 4 and a system with three page frames, what will be the final contents of the three frames after the LRU algorithm is applied? Use the table below to the successive contents of the frames. When there is no page faults, leave the column blank.

**Answer:** Final contents: F0=3, F1=1 F2=4

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 2 | 3 | 4 | 1 | 2 | 1 | 1 | 3 | 1 | 4 |  |
| F0 | 1 | 1 | 1 | 4 |  |  |  | 4 | 4 |  |  | 3 |  | 3 |  |
| F1 |  | 2 | 2 | 2 |  |  |  | 1 | 1 |  |  | 1 |  | 1 |  |
| F2 |  |  | 3 | 3 |  |  |  | 3 | 2 |  |  | 2 |  | 4 |  |

**Problem 4** (11 points) Assume we have a demand-paged memory. Assume that the time needed to access the page table is negligible. It requires 8 milliseconds to handle a page fault if an empty page is available or the replaced page is not dirty, and 20 milliseconds if the replaced page is dirty. Memory access time is 100 nanoseconds. Assume that the page to be replaced is dirty 70 percent of the time. What is the maximum acceptable page-fault rate for an effective access time of no more than 200 nanoseconds?

Show your work here:

200ns = (1 – P)(100ns) + (0.3P)(8ms) + (0.7P)(20ms)

0.2µs = (1 – P)(0.1µs) + (0.3P)(8000µs) + (0.7P)(20000µs)

0.2 = 0.1 – 0.1P + 2400P + 14000P

P = 0.1 / (2400 + 14000 – 0.1)

Answer: P = 0.0000061

**Problem 5** (5 points)

a. A system uses 32-bit logical addresses, a 16K byte (214) page size, and 36-bit physical addresses (64 GB memory). What is the size of the page table?:

|  |  |
| --- | --- |
| **□** 222 entries (236-14). | **□** 24 entries (236-32). |
| **□** 218 entries (232-14). | **□** 214 entries |

b. We had seen an example of determining the Working Sets using a given value of Δ. For that example, obtain the value of WS(t1) and WS(t2) if Δ = 5.

Answer: WS(t1) = {1,5,7}

WS(t2) = {3,4}

**Problem 6** (13 points) The page table below is for a system with 16-bit virtual as well as physical addresses and with 4,096-byte pages. The reference bit is set to 1 when the page has been referenced. Periodically, a thread zeroes out all values of the reference bit. A dash for a page frame indicates the page is not in memory. The LRU page-replacement algorithm is used. The numbers are given in decimal.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Page | | Frame | | ReferenceBit |
| 0 | 9 | | 0 | |
| 1 | 1 | | 0 | |
| 2 | 14 | | 0 | |
| 3 | 10 | | 0 | |
| 4 | – | | 0 | |
| 5 | 13 | | 1 | |
| 6 | 8 | | 0 | |
| 7 | 15 | | 1 | |
| 8 | – | | 0 | |
| 9 | 0 | | 0 | |
| 10 | 5 | | 1 | |
| 11 | 4 | | 0 | |
| 12 | – | | 0 | |
| 13 | – | | 0 | |
| 14 | 3 | | 0 | |
| 15 | 2 | | 1 | |

A. Convert the following virtual addresses (in hexadecimal) to the equivalent physical addresses(hex).

|  |  |
| --- | --- |
| Virtual Address | Physical Address |
| 0xF12C | 0x212C |
| 0x5A9D | 0xDA9D |
| 0xA9D9 | 0x59D9 |
| 0x7001 | 0xF001 |
| 0xACA1 | 0x5CA1 |

B. Will these logical addresses (in hexadecimal) results in a page fault: 0xD123? 0x5000?

Answer: 0xD123 will result in a page fault. 0x5000 will not result in a page fault.

C. Give the complete list of frames which may be chosen by the LRU page-replacement algorithm in resolving a page fault?

Answer: {9, 1, 14, 10, 8, 0, 4, 3}

**Problem 7** (8 points) A system that uses demand-paging with a disk that has an average page access and transfer time of 20 milliseconds. Addresses are mapped using a page table in main memory, with an access time of 1 microsecond per access. To improve this time, the system designers have added an associative memory that reduces access time to one memory reference, when the page-table entry is in the associative memory. If 90 percent of the accesses result in an associative memory hit, and of those remaining, 10 percent (or 1 percent of the total) cause page faults, what is the effective memory access time?

Answer:

Effective Access Time = (0.9)(1µs) + (0.09)(2µs) + (0.01)(20002µs)

EAT = 0.9 + 0.18 + 200.2 = 201.28µs

Answer: 0.2 ms (milliseconds) [ must have correct units]

**Problem 8**  (18) Consider the following page reference string:  
6, 2, 3, 1, 2, 5, 3, 4, 7, 6, 6, 1, 0, 5, 4, 7, 2, 3, 0 , 1.  
Assuming demand paging with three frames, what page fault rate would be encountered for the following replacement algorithms?  
• LRU replacement  
• FIFO replacement  
• Optimal replacement

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 6 | 2 | 3 | 1 | 2 | 5 | 3 | 4 | 7 | 6 | 6 | 1 | 0 | 5 | 4 | 7 | 2 | 3 | 0 | 1 |  |
| F0 | 6 | 6 | 6 | 1 | 1 | 1 | 3 | 3 | 3 | 6 | 6 | 6 | 6 | 5 | 5 | 5 | 2 | 2 | 2 | 1 |  |
| F1 |  | 2 | 2 | 2 | 2 | 2 | 2 | 4 | 4 | 4 | 4 | 1 | 1 | 1 | 4 | 4 | 4 | 3 | 3 | 3 |  |
| F2 |  |  | 3 | 3 | 3 | 5 | 5 | 5 | 7 | 7 | 7 | 7 | 0 | 0 | 0 | 7 | 7 | 7 | 0 | 0 |  |

LRU: Page fault rate: \_\_\_18/20\_\_\_ = \_\_0.9\_\_

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 6 | 2 | 3 | 1 | 2 | 5 | 3 | 4 | 7 | 6 | 6 | 1 | 0 | 5 | 4 | 7 | 2 | 3 | 0 | 1 |  |
| F0 | 6 | 6 | 6\* | 1 | 1 | 1 | 1 | 1\* | 7 | 7 | 7 | 7\* | 0 | 0 | 0\* | 7 | 7 | 7\* | 0 | 0 |  |
| F1 |  | 2 | 2 | 2\* | 2 | 5 | 5 | 5 | 5\* | 6 | 6 | 6 | 6\* | 5 | 5 | 5\* | 2 | 2 | 2\* | 1 |  |
| F2 |  |  | 3 | 3 | 3 | 3\* | 3 | 4 | 4 | 4\* | 4 | 1 | 1 | 1\* | 4 | 4 | 4\* | 3 | 3 | 3 |  |

FIFO: Page fault rate: \_\_\_17/20\_\_\_ = \_\_0.85\_\_

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 6 | 2 | 3 | 1 | 2 | 5 | 3 | 4 | 7 | 6 | 6 | 1 | 0 | 5 | 4 | 7 | 2 | 3 | 0 | 1 |  |
| F0 | 6 | 6 | 6 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |  |
| F1 |  | 2 | 2 | 2 | 2 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 4 | 7 | 2 | 3 | 3 | 3 |  |
| F2 |  |  | 3 | 3 | 3 | 3 | 3 | 4 | 7 | 6 | 6 | 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |  |

Optimal: Page fault rate: \_\_\_13/20\_\_\_ = \_\_0.65\_\_

**Problem 9** (6 points)

* 1. Give a block diagram of a TLB. Mention where the input comes from and where does the output go. Identify where the page numbers are stored and where the corresponding frame numbers are stored.

See diagram below

* 1. A system has 32 bit logical and physical addresses. The frames are 4096 bytes each. What is number of bits in the TLB input and output.

Answer: 232-12 = 220 = 20 bits in TLB input/output

9a:  
The input to the TLB comes from the CPU via the logical address page number and the output goes to the physical address in order to access the physical memory. The TLB stores the page numbers and corresponding frame numbers.

