**EECE 5129/6029: Operating Systems**

**FALL 2019**

**HOMEWORK IV**

**Given: Oct. 27, 2019**

**Due: Nov. 6, 2019 (Wednesday), 2019 (NO LATER THAN 11:59PM)**

**SUBMISSION INSTRUCTIONS:**

**SUBMIT ONLY ON-LINE FILES ON BLACKBOARD BEFORE MIDNIGHT.**

**NO HARD COPY WILL BE ACCEPTED.**

Dr. Rui (April) Dai

Assistant Professor

Department of Electrical Engineering & Computer Science

University of Cincinnati

Office: ERC 540 Tel: (513)556-0134 Email: rui.dai@uc.edu

1. Consider the following snapshot of a system:



Answer the following questions using the banker’s algorithm:

a) Illustrate that the system is in a safe state by demonstrating an order in which the processes may complete.

b) If a request from process P1 arrives for (1, 1, 0, 0), can the request be granted immediately?

c) If a request from process P4 arrives for (0, 0, 2, 0), can the request be granted immediately?

2. Compare the memory organization schemes of contiguous memory allocation, pure segmentation, and pure paging with respect to the following issues:

a) External fragmentation

b) Internal fragmentation

c) Ability to share code across processes

3. Consider a logical address space of 256 pages with a 4-KB page size, mapped onto a physical memory of 64 frames.

a) How many bits are required in the logical address?

b) How many bits are required in the physical address?

4. Consider a paging system with the page table stored in memory.

a) If a memory reference takes 50 nanoseconds, how long does a paged memory reference take?

b) If we add TLBs, and 75 percent of all page-table references are found in the TLBs, what is the effective memory reference time? (Assume that finding a page-table entry in the TLBs takes 2 nanoseconds, if the entry is present.)

5. 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 (for example, waiting for I/O). This is illustrated in Fig. 1. Assuming a thread is in the Running state, answer the following questions, and explain your answer:

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

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?

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



Figure 1. Thread state diagram for Question 5

6. Programming problem:

Design a programming solution to the bounded-buffer problem using the producer and consumer processes shown in Chapter 5 of textbook. Use standard counting semaphores for empty and full and a mutex lock to represent mutex. The producer and consumer—running as separate threads—will move items to and from a buffer that is synchronized with the empty, full, and mutex structures. Please check the attached code, where the main() function and the producer and consumer functions are given. You task is to implement the insert\_item() and remove\_item() functions which manipulate the buffer:

int insert item(buffer item item) {

/\* insert item into buffer

return 0 if successful, otherwise

return -1 indicating an error condition \*/

}

int remove item(buffer item \*item) {

/\* remove an object from buffer

placing it in item

return 0 if successful, otherwise

return -1 indicating an error condition \*/

}

The attached code uses Pthreads API. You can also solve this problem using Windows API or Java.

**Please submit your source code for this question.** **To receive full credits for this problem, please include screen shots of your test results.**