STAMFORD UNIVERSITY BANGLADESH

Department of Computer Science and Engineering

Semester Final Examination, Summer-2022

CSI 313: Operating System

CT: Mohammad Zainal Abedin

Date and Time: 05/12/2022 & 7:00 PM-9:30 PM

Batch: MCSE **Campus:** Siddeswari

Duration: 2.30 hours **Full Marks: 40**

(There are **FOUR** questions. Answer all of them. Figures in the right margin indicate marks

1. a) Consider the following resource-allocation policy. Requests for and releases of resources are allowed at any time. If a request for resources cannot be satisfied because the resources are not available, then we check any threads that are blocked waiting for resources. If a blocked thread has the desired resources, then these resources are taken away from it and are given to the requesting thread. The vector of resources for which the blocked thread is waiting is increased to include the resources that were taken away. [04]

For example, a system has three resource types, and the vector Available is initialized to (4,2,2). If thread T0 asks for (2,2,1), it gets them. If T1 asks for (1,0,1), it gets them. Then, if T0 asks for (0,0,1), it is blocked (resource not available). If T2 now asks for (2,0,0), it gets the available one (1,0,0), as well as one that was allocated to T0 (since T0 is blocked). T0's Allocation vector goes down to (1,2,1), and its Need vector goes up to (1,0,1).

1. Can deadlock occur? If you answer "yes," give an example. If you answer "no," specify which necessary condition cannot occur.
2. Can indefinite blocking occur? Explain your answer.

b) Consider the following snapshot of a system: [06]

Table 2 Resource Allocation State

Threads	Allocation A B C D	Max A B C D	Available A B C D
TO	3 1 4 1	6 4 7 3	2 2 2 4
T1	2 1 0 2	4 2 3 2	
T2	2 4 1 3	2 5 3 3	
T3	4 1 1 0	6 3 3 2	
T4	2 2 2 1	5 6 7 5	

Answer the following questions using the banker's algorithm:

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

ii. If a request from thread T4 arrives for (2, 2, 2, 4), can the request be granted immediately?

2. a) There are two resource-allocation graphs shown in Fig. 1 and 2. Determine is there any deadlock in those graphs . Justify your answer. [02]

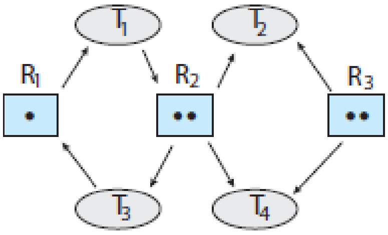

Fig. 1

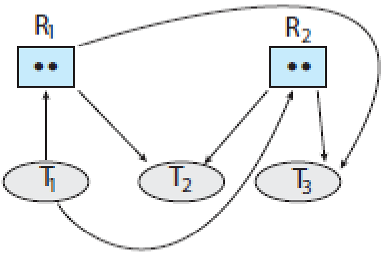

Fig.2

b) Compare the memory organization schemes of contiguous memory allocation and paging with respect to the following issues: [03]

1. External fragmentation
2. Internal fragmentation

c) Given six memory partitions of 100 MB, 170 MB, 40 MB, 205 MB, 300 MB, and 185 MB (in order), how would the first-fit, best-fit, and worst-fit algorithms place processes of size 200 MB, 15 MB, 185 MB, 75 MB, 175 MB, and 80 MB (in order)? Indicate which—if any—requests cannot be satisfied. Comment on how efficiently each of the algorithms manages memory. [05]

3. a) Consider a logical address space of 2,048 pages with a 4-KB page size, mapped onto a physical memory of 512 frames. [05]

1. How many bits are required in the logical address?
2. How many bits are required in the physical address?

b) Consider a paging system with the page table stored in memory. [03]

1. If a memory reference takes 50 nanoseconds, how long does a paged memory reference take?
2. If we add TLBs, and if 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.)

c) How protection of memory is implemented in paging memory allocation ? [02]

4 a) A simplified view of process states is ready, running, and blocked, where a process is either ready and waiting to be scheduled, is running on the processor, or is blocked (for example, waiting for I/O). [03]

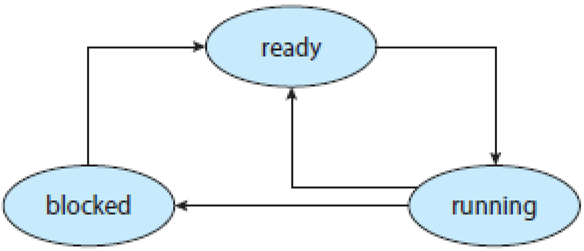

Fig. 3

Assuming a process is in the running state, answer the following questions, and explain your answers:

1. Will the process change state if it incurs a page fault? If so, to what state will it change?
2. Will the process change state if it generates a TLB miss that is resolved in the page table? If so, to what state will it change?

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

b) Consider the following page reference string: [07]
7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4, 6, 2, 3, 0 , 1.
Assuming demand paging with three frames, how many page faults would occur for the following replacement algorithms? Compare this two algorithms with the benchmark optimal page replacement algorithm.

1. LRU replacement
2. FIFO replacement

/// *the end*//