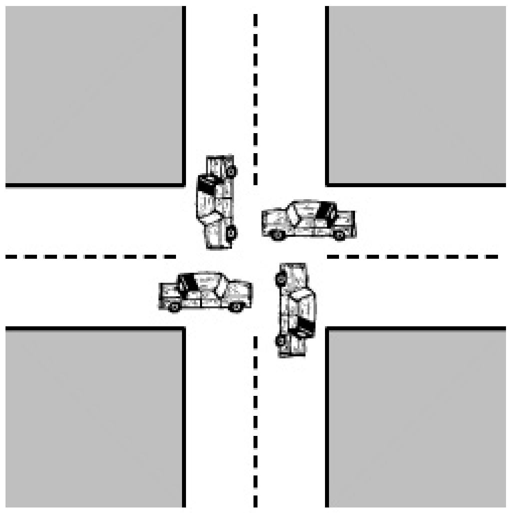
Department of Mathematics and Computer Science,
SUNY at Fredonia, Fredonia, New York.

CSIT 431 Introduction to Operating Systems Spring 2002 <u>Second Mid-Term Test Solution</u>

NAME: *********************SOLUTION*********************************

Q1.

(a) Look at the diagram shown below that depicts a four-way intersection. There are four cars present in the intersection that are heading North, South, East and West. List all four conditions that must be present for a deadlock to occur and discuss if these conditions are actually present in this scenario or not.



(0.5 marks for listing conditions; 0.5 marks to show their presence here)

FOUR CONDITIONS

1) *Mutual Exclusion: PRESENT because road segments are non-sharable. Cars cannot stand over one another*
2) *Hold & Wait: PRESENT because each car is holding onto a segment of road and waiting for a segment to open up*
3) *No Preemption: PRESENT because you cannot forcibly remove the road from a car*

4) *Circular Wait: PRESENT because car heading N is waiting for W-bound car to go. W-bound car is waiting for S-bound car to go. S-bound car is waiting for E-bound car to go and E-bound car is waiting for N-bound car to leave.*

(b) Suppose the processes in a system are designed like below:

Process Pi:

```
Start processing data
Request resource 1
Request resource 2
Process data with resource 1
Process data with resource 2
Release resource 1
Release resource 2
```

Are these processes prone to deadlock? If yes, Can the design be changed to avoid deadlock?

2 Marks

(0.5 marks for YES, 0.5 marks for redesigning)
YES, they are prone to deadlock. Circular wait can result in the way the resources are requested)
Design can be changed to request a resource, process data with it and then release it before requesting another resource

Q2.

(a) What is the difference between deadlock prevention and deadlock avoidance? Which technique would result in a more efficient use of resources? Why?

(0.5 marks for correct definition of prevention and avoidance. 0.5 marks for arguing which one is more efficient)
Deadlock prevention tries to break one of the four necessary conditions for a deadlock. Deadlock avoidance allows the 3 necessary conditions to exist but

it considers every request to see if it would lead to a deadlock. If yes, it will deny that request.

Deadlock avoidance is more efficient in use of resources because we do not limit or restrict the allocation of resources in the normal run until an offending request is made.

(b) Define banker's algorithm by explaining:

(0.5 marks for each part)

(I) Claim Matrix: *One row for each process and one column for each resource. Lists total requirements for each resource from all processes*

(II) Allocation Matrix: *Shows currently allocated resources for all processes*

(III) Resource Vector: *Gives total amount of all resources present in the system*

(IV) Safe State and Unsafe state: *Safe state is one in which all processes can run to completion in at least one sequence. Unsafe state is one in which such a sequence does not exist*

3 Marks

Q3.

(a) How is a deadlock detected? What steps can be taken to break the deadlock?

(0.5 marks for detection; 0.5 marks for breaking the deadlock)
O.S. runs a deadlock detection algorithm to detect the circular wait condition present for a set of processes. It uses a marking method to mark all processes that are not currently deadlocked.
Deadlock can be broken by one of the following methods:

(a) Abort all deadlocked processes
(b) Backup to a deadlock free checkpoint and start over
(c) Abort one by one the deadlocked processes
(d) Pre-empt resources

(b) Consider a system with a total of 150 instances of a resource available for allocation. The following table shows currently held and maximum needs of three active processes

Process	**Max Need**	**Current Allocation**	**Still Needs**
P1	70	45	
P2	60	40	
P3	60	15	

Complete the last column and then consider a new process P4 that will need a maximum of 60 units. Use Banker's algorithm to determine if the system will be in a safe or unsafe state if P4 is allocated its current request. Do it for both scenarios I and II and show full sequence of steps:

(i) P4 is currently requesting 25 units
(ii) P4 currently needs 35 units

2 Marks

(0.5 marks for part (i) and 0.5 marks for part (ii)
PART i

(a) P4 requests 25 units, it is allocated 25 units; Remainder: 25 units
(b) P1 is given 25 units; it is finished; Remainder:70 units
(c) P2 is given 20 units; P2 is finished; Remainder:110 units
(d) P3 is given 45 units; P3 is finished; Remainder:125 units
(e) P4 is given 35 units; P4 is finished; Remainder: 150 units

ALL FINISHED SUCCESSFULLY; IT IS A SAFE STATE

PART ii

(a) P4 requests 35 units, it is allocated 35 units; Remainder: 15 units
(b) No process can run to completion if they keep their current allocation and request their balance. IT IS AN UNSAFE STATE

Q4.

(a) In the dining philosophers problem, five processes represent the philosophers. Each process is supposed to think for a random amount of time and then try to eat by acquiring forks on left and right. Using semaphores, list the steps taken by each process for completing the thinking and eating cycle. Since there are only 5 forks available, the system may get into a deadlock. State a possible solution that can avoid deadlock completely.

(0.5 marks for listing steps as below; 0.5 marks for deadlock free solution)
-think(random)
-wait(left fork)
-wait(right fork)
-eat()
-release the forks (signal(left fork) signal(right fork)

Thinking random amount of time may reduce the probability of deadlock but to completely eliminate deadlock, the only solution is to either increase the number of forks to 6 or reduce the number of diners to 4

(b) Discuss how semaphores are implemented in UNIX. Mention their components, system calls and specific values of interest.

2 Marks

(Too general statement: only 0.5 marks. UNIX specific discussion: 1 mark)
Components of UNIX semaphores:

(1) Current value
(2) PID of last process to operate on this semaphore
(3) Number of processes waiting for increment
(4) Number of processes waiting for zero
(5) Queues of the waiting processes

System calls: semctl (to set values) and semop (to operate on a semaphore)
Specific values of interest: 0;and negative values

Q5.

Suppose that a total of 64MB RAM is available in a system. This memory space is partitioned into 8 fixed size slots of 8MB each. Assume 8 processes are currently requesting memory usage with sizes indicated as below:
[2M, 4M, 3M, 7M, 9M, 6M, 1M, 8M]
Calculate the size of memory wasted due to external and internal fragmentation. Derive the memory utilization ratio by dividing the total allocated memory by total requested memory.

2 Marks

(correct internal fragmentation result: 1 mark; external fragmentation:0.5 mark; correct utilization ratio: 0.5 mark)

Assuming exactly one slot can be given to each process:
INTERNAL FRAG: 6+4+5+1+2+7+0=25M
EXTERNAL FRAG: 8M (One slot unused because the process won't fit)
UTIL RATIO: 56/40 or 56/64(if sizes of processes are ignored)

Assuming more than one slot can be given to each process:
INTERNAL FRAG:6+4+5+1+7+2+7=32M
EXTERNAL FRAG: Zero (but one process is left out)
UTIL RATIO: 64/40 or 64/64 (if sizes of processes are ignored)

Q6.

(a) How will each of the listed schemes schedule a set of processes? Which one is not a preemptive scheme?

(0.5 for first two parts, 0.5 for third part and stating which one is non-preemptive)

(i) RR (Round robin): *allows a time slot to each process; then preempts and brings the next one*

(ii) SPN (Shortest Process Next): *It selects shortest process from current waiting list and runs it to completion. SPN IS NON_PREEMPTIVE*

(iii) Feedback : *On any job arrival/completion of time slot, current job is preempted and moved to lower priority queue. Punishes longer running jobs and favors short or fresh jobs*

(b) Perform scheduling on this group of processes using SPN and Feedback Schemes.

Process Name	**Arrival Time**	**Processing Time**
A	0	1
B	1	9
C	2	1
D	3	9

2 Marks

(0.5 for SPN; 0.5 for Feedback)

SPN:
0-1: A
1-10:B
10-11:C
11-20:D

Feedback: (When time slice continues to increase as power of 2)
0-1:A
1-2:B
2-3:C
3-4:D
4-6:B
6-8:D
8-12:B
12-16:D
16-18:B
18-20:D

Q7.
(a) How many scheduling classes are defined in Linux and what are their priorities?
(0.5 for mentioning real-time; 0.5 for specific classes)
SCHED_FIFO (Real-Time Highest Priority)
SCHED_RR (Real-Time Middle Priority)
SCHED_OTHERS (Non-RT other jobs Lowest)

(b) Why does "gang scheduling" result in improved throughput on a multiprocessor system?

2 Marks

(1 mark)
"Gang Scheduling" allows a set of related threads to run on a set of processors. It improves throughput because all threads of a job run concurrently thus eliminating waste of time in synchronization attempts

Q8.
(0.5 marks for each part)

(a) What is the difference between periodic and aperiodic tasks?: *Periodic*

tasks arrive regularly after set time periods. Aperiodic tasks can

arrive any time.

(b) Define RMS (Rate Monotonic Scheduling): *RMS attaches highest*

priority to the periodic tasks with shortest period

(c) What is meant by a “soft deadline”?: *A deadline that can be missed by a small margin*

(d) When we talk about coarse, medium and fine-grained synchronization, what are we referring to?: *Synch. Frequency is the number of instructions executed before the threads or processes synchronize with each other.*

2 Marks

Q9.

Assume three periodic tasks A, B and C arrive at the time t=0 to a real-time system. Process A will continue to arrive every 20ms, it has execution time of 10ms and a deadline of 20ms on arrival. Process B arrives every 50ms, has execution time of 10ms and a deadline of 50ms on arrival. Process C will arrive every 50ms, has execution time of 15ms and a deadline of 50ms on arrival. Using EDF (Earliest Deadline First), show that all processes meet their deadlines from t=0 to t=60ms by drawing the schedule of the system.

3 Marks

(1 mark for each of A, B and C being scheduled correctly)

0-10:A1
10-20:B1
20-30:A2
30-45:C1
45-55:A3
55-60:B2….

Another solution

0-10:A1
10-25:C1
25-35:A2
35-45:B1
45-55:A3
55-60:C2….