Name-Muhammed Raihan P.S.
Enroll No - 130
Course - MCA 1C
OS Assignment - 3

2016

Q-1) What is Deadlock Avoidance? Explain the recovery process from Deadlock?
A Deadlock avoidance algorithm dynamically examines the resource allocation state to ensure that a circular wait condition can never exist. The resource allocation state is defined by the number of available & allocated resources & the maximum demands of the processes.

-Safe State - A state is safe if the system can allocate resources to each process in some order & still avoid a deadlock. A system is in safe state only if there exists a ~~strigle~~ safe sequence.

⇒ Recovery from Deadlock

- Process Termination - To eliminate deadlock by aborting a process, we use one of two methods.
  - Abort one process at a time - This method incurs considerable overhead, since after each process is aborted, a deadlock detection algo must be invoked to determine whether any processes are still deadlocked.
  - Abort all deadlocked process - This method will clearly break cycle, but at great expence, the deadlocked processes have computed for a long time, & the results of these partial computations must be discarded & probably will have to be recomputed later. CPU time they have used till now will get wasted.

⇒ Resource Preemption - To eliminate deadlock using resource preemption, we can successfully preempt some resources from processes & give these resources to another processes until deadlock cycle is broken.

- Selecting a victim - Which resource & which process are to be preempted. As in process termination, we must determine the order of preemption to minimize cost. Cost factors may include such parameters as the no. of resources a deadlocked process is holding & the amount of time the process has consumed during its execution.
- **Roll back** - If we preempt a resource from a process, what should be done with that process, clearly it cannot continue with its normal execution, it is missing some needed resources. we must roll-back the process to some safe-state.
- Starvation - In a system where victim selection is based primarily on cost factors, it may happen that the same process is always picked as a victim. As a result, this process never ~~continues~~ completes its task.

Q-2) What is Deadlock characterization ? Suggest two methods for Deadlock Handling :

What is Deadlock characterization - Refer Q-1 2017

Suggest two methods for deadlock Handling

Deadlock Prevention
Deadlock Avoidance

2017

Q-1) Mention the characteristics of a deadlock system. Explain various deadlock recovery techniques.

There are four necessary conditions hold simultaneously in a system

-Mutual Exclusion- Only one process at a time can use the resource. If a process requests that resource, the requesting process must be delayed until the resource has been released.

- Hold & Wait - A process must be holding at least one resource & waiting to aquire additional resources that are currently being held by other process.

- No Preemption - Resources connat be preempted, that is a resource can be released by the process holding it, after that process has completed its task.

-Circular Wait - One process is waiting for a resource that is held by other process, other process is waiting for a resource that is held by next process & so on.

Explain various deadlock recovery techniques

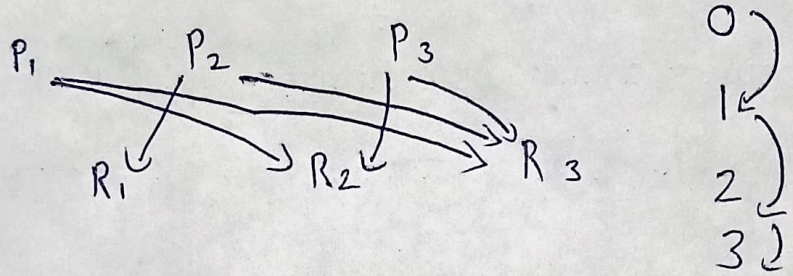
Refer Q-No 1 of 2016.

2018

Q-1(a) What do you mean by circular wait condition in deadlock? Design a protocal which violates this condition & justify its worthiness.

Circular Wait - One process is waiting for a resource that is held by other process, other process is waiting for a resource that is held by next process & so on.

Protocal - Each process can request resources only in an increasing order.



A process can initially request any no. of instances of a resource type - R_i After that, the process can request instances of resource type R_j if & only if $F(R_j) > F(R_i)$.

Q-1 b Lost Page

2019

Q-1)

	Allocation			Max			Available			Need		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	0	1	0	7	5	3	3	3	2	7	4	3
P_1	2	0	0	3	2	2				1	2	2
P_2	3	0	2	9	0	2				6	0	0
P_3	2	1	1	2	2	2				0	1	1
P_4	0	0	2	4	3	3				4	3	1

(i) Need Matrix

	Need		
	A	B	C
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

Available - 3 3 2

P_1 2 0 0

Available - 5 3 2

P_3 2 1 1

Available - 7 4 3

P_4 0 0 2

Available - 7 4 5

P_2 3 0 2

Available - 10 4 7

P_0 - 0 1 0

Available - 10 5 7

We can satisfy there needs in the following sequence

$P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_2 \rightarrow P_0$

therefore system is in safe state

(iii) If the request from process P1 arrives for (102), can the request be granted immediately ?

	Allocation			Max			Available			Need		
	A	B	C	A	B	C	A	B	C	A	B	C
P_0	0	1	0	7	5	3	2	3	0	7	4	3
P_1	3	0	2	3	2	2				0	2	0
P_2	3	0	2	9	0	2				6	0	0
P_3	2	1	1	2	2	2				0	1	1
P_4	0	0	2	4	3	3				4	3	1

Available - 2 3 0

P_1 - 3 0 2

Available - 5 3 2

P_3 - 2 1 1

Available - 7 4 3

P_4 - 0 0 2

Available - 7 4 5

P_0 - 0 1 0

Available - 7 5 5

P_2 - 3 0 2

Available 10 5 7

Yes P_1's request will be granted as we are able to fulfill the needs of processes in the following sequence

$$P_1 \rightarrow P_3 \rightarrow P_4 \rightarrow P_0 \rightarrow P_2$$

2022

Q-1) Write Bankers Algo for deadlock avoidance

1) Initialization

Set finish [i] = false for all i ranging from 1 to n

2) Compute avail & need

set need [i,j] = max[i,j] - alloc [i,j]

for all process i & for all resources

3) Find a process i such that

finish [i] = false & need [i,j] ≤ avail [i,j] for all resources

if no such i exists, goto step 5

4) Schedule process i for execution so that it can utilize resources, then free up its resources by doing following

set avail [i] = Avail[i] + [alloc[i,j]

set need [i,0] = 0 for all

set finish [i] = true

Goto step 3

5) If Finish [i] = true for all i return [safe] else return [unsafe]

Q-2)

	Allocation				Max				Available				Need			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0	0	0	0	0
P_1	1	0	0	0	1	7	5	0					0	7	5	0
P_2	1	3	5	4	2	3	5	6					1	0	0	2
P_3	0	6	3	2	0	6	5	2					0	0	2	0
P_4	0	0	1	4	0	6	5	6					0	6	4	2

(a) Need Matrix

	Need			
	A	B	C	D
P_0	0	0	0	0
P_1	0	7	5	0
P_2	1	0	0	2
P_3	0	0	2	0
P_4	0	6	4	2

(b) Is the system in safe state

Available	1	5	2	0
P_3	0	6	3	2
Available	1	11	5	2
P_4	0	0	1	4
Available	1	11	6	6
P_1	1	0	0	0
Available	2	11	6	6
P_2	1	3	5	4
Available	3	14	11	10

System is in safe state we can satisfy the needs of these processes in the following sequence

$P_3 \rightarrow P_4 \rightarrow P_1 \rightarrow P_2$

(c) If P1 request (0,4,2,0) will it be granted

	Allocation				Max				Available				Need			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	1			0	0	0	0
P_1	1	4	2	0	1	7	5	0	1	1	0	0	0	3	3	0
P_2	1	3	5	4	2	3	5	6					1	0	0	2
P_3	0	6	3	2	0	6	5	2					0	0	2	0
P_4	0	0	1	4	0	6	5	6					0	6	4	2

With the available 1100 we cannot satisfy the needs of any processes.

Request of P1 will not be granted

2018

Q-4 (b) Write an algo for deadlock detection for multiple instances of resources ?

1) If alloc [i,j] ≠ 0 then
   set finish [i] = false;
   else
   set finish [i] = true;
   for all process i = 1 to n
   for all process j = 1 to n

2) Find a process i such that the condition
   finish [i] = false & Request [i,j] ≤ avail [j]
   If no such i exists go to step 4

3) Set avail [j] = avail [j] + alloc [i,j]
   set finish [i] = true;
   goto step 2

4) If finish [i] = false for some process
   then system is in deadlock state.