Tribhuvan University
Institute of Science and Technology
2076
✡

Bachelor Level / Second Year/ Forth Semester/ Science
Computer Science and Information Technology (CSC 259)
(Operating Systems)

Full Marks: 60
Pass Marks: 24
Time: 3 hours.

Candidates are required to give their answers in their own words as for as practicable.
All figures in the margin indicate full marks.

Attempt all the questions.

Section A

Long Answer Questions
Attempt any Two questions (2x 10=20)

1 Defined interactive system goals? List various interactive scheduling algorithms. Consider following process data and compute average waiting time and average turnaround time for RR (quantum 10) and priority scheduling algorithms.

PID	Burst Time	Arrival Time	Priority
A	16	0	1
B	37	12	2
C	25	7	3

2 How Second Chance page replacement algorithm differs from FIFO page replacement policy? Discuss the concept of Belady's anomaly with suitable example.

3 What is the main objective of disk scheduling algorithms? Why SSTF is not practically feasible? Assume that we have disk with 100 tracks and currently head is at track number 35. What will be the seek time for the algorithms SCAN and LOOK for processing IO requests queue: 52, 67, 27, 11, 43, 85, 18, 75, 92, 8?

Group 'B'

Short Answer Questions
Attempt any Eight questions. (8x5=40)

4 What are two modes of OS? Discuss different OS structures briefly.

5 When threads are better than processes? Explain the concept of user level threads in detail.

6 Differentiate between multi programming and Monoprogramming. What will be the CPU utilization with 6 processes with 60% IO waiting time are in memory?

7 How can you manage free disk space? Explain the linked list approach of managing free disk space with example.

8 When programmed IO is suitable than other IO handling techniques? Explain the process of IO handling using DMA.

9 Differentiate between deadlock and starvation? Discuss the process of detecting deadlocks when there are multiple resources of each type.

10 What is problem associated with semaphores? Explain the concept of monitors in brief.

11 Why program relocation and protection is important? Explain the technique of achieving program relocation and protection.

12 Write short notes on

- Linux File System
- Resource Allocation Graph

Operating System - 2076

- Ankit Pangeni

Section A.

Q.1. Define interactive system goals? List various interactive scheduling algorithms. Consider following process data and compute average waiting time and average turnaround time for RR (quantum 10) and priority scheduling algorithms.

P.ID	Burst time	Arrival time	Priority
A	16	0	1
B	37	12	2
C	35	7	3

⇒ In interactive system, processes are scheduled according to the priority. Some of the interactive system goals are:-

Response time :- minimize the response time for user
Waiting time :- minimize total time spend waiting in queue
Throughput :- Maximize jobs per given time period

Various interactive scheduling algorithms are:-

i) Round-Robin scheduling
ii) Priority scheduling
iii) Multiple queues

1st Round Robin

Ready queue: A | C | A | B | C | B | C | B | C | B

Gantt Chart: A | C | A | B | C | B | C | B | C | B

0 10 20 26 36 46 56 66 76 84 88

P.ID	Arrival time	Burst Time	Completion Time	Turn around	waiting time
A	0	16	26	26	10
B	12	37	78	66	29
C	7	25	61	54	29

∴ Average turn around time: $\frac{26+76+74}{3} = 58.66$

Average waiting time: $\frac{10+39+29}{3} = 32.66$

2nd Priority schedule.

Gantt Chart: A | A | A | B | C

0 7 12 16 53 78

P.I.D	Arrival time	Burst Time	Completion Time	TAT	WT
A	0	16	16	16	0
B	12	37	53	41	4
C	7	25	78	71	46

∴ Average turn around time: $\frac{16+41+71}{3} = 42.66$

Average waiting time: $\frac{0+4+46}{3} = 16.66$

DATE

Q.2. How Second Chance page replacement algorithm differs from FIFO page replacement policy? Discuss the concept of Beldy's anomaly with suitable example.

⇒ Second Chance page replacement algorithm is just the modified version of First In First out (FIFO) page replacement policy. In FIFO, when a page needs to be replaced, the oldest ie. first in page in the front of the queue is replaced.

The 2nd Chance algorithm is based on the FIFO algorithm and it even degenerates to FIFO in its worst-case scenario. The main difference between them is that, in 2nd chance algorithm, a FIFO replacement is implemented along with a reference bit. If reference bit is 0, then we proceed to replace the page, but if the reference bit is 1, then we give page the 2nd Chance -, then its reference bit is cleared.

Belady's anamoly is the phenomena in which increasing the number of page frames (ie. memory size) ~~for~~ results in an increase in the number of page faults for certain memory access patterns. This phenomena is commonly experienced when using the FIFO page replacement algorithm.

Let's take an example of Belady's anamoly in page replacement.

classmate

PAGE

DATE

Let's take a reference string
1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Initially lets take Memory size = 3
Let's find the total page faults when using FIFO page replacement algorithm

f_3			3	3	3	2	2	2	2	2	4	4
f_2		2	2	2	1	1	1	1	1	3	3	3
f_1	1	1	1	4	4	4	5	5	5	5	5	5
	*	*	*	*	*	*	*	H	H	*	*	H

Total page faults = 9

Now, let's increase the memory size to 4 and check on the same reference string
1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

f_4				4	4	4	4	4	4	3	3	3
f_3			3	3	3	3	3	3	2	2	2	2
f_2		2	2	2	2	2	2	1	1	1	1	5
f_1	1	1	1	1	1	1	5	5	5	5	4	4
	*	*	*	*	H	H	*	*	*	*	*	*

Total page faults = 10

Here, we can see, on increasing memory size to 4 from 3, the total page faults increase to 10 from 9. Which is called the Belady's anamoy.

PAGE

DATE

Q.3 What is the main objective of disk scheduling algorithms? Why SSTF is not practically feasible? Assume that we have disk with 100 tracks and currently head is at track number 35. What will be the seek time for the algorithms SCAN and LOOK for processing. IO requests queue:
52, 67, 27, 11, 43, 85, 18, 75, 92, 8

=> Disk scheduling is done by the operating system to schedule I/O requests arriving for the disk. It is also known as I/O scheduling.

There are two main objective of disk scheduling algorithms:

- Maximize the throughput :- It is the average number of requests satisfied per time unit
- Minimize the seek time :- It is time taken to reach upto the desired track in the disk

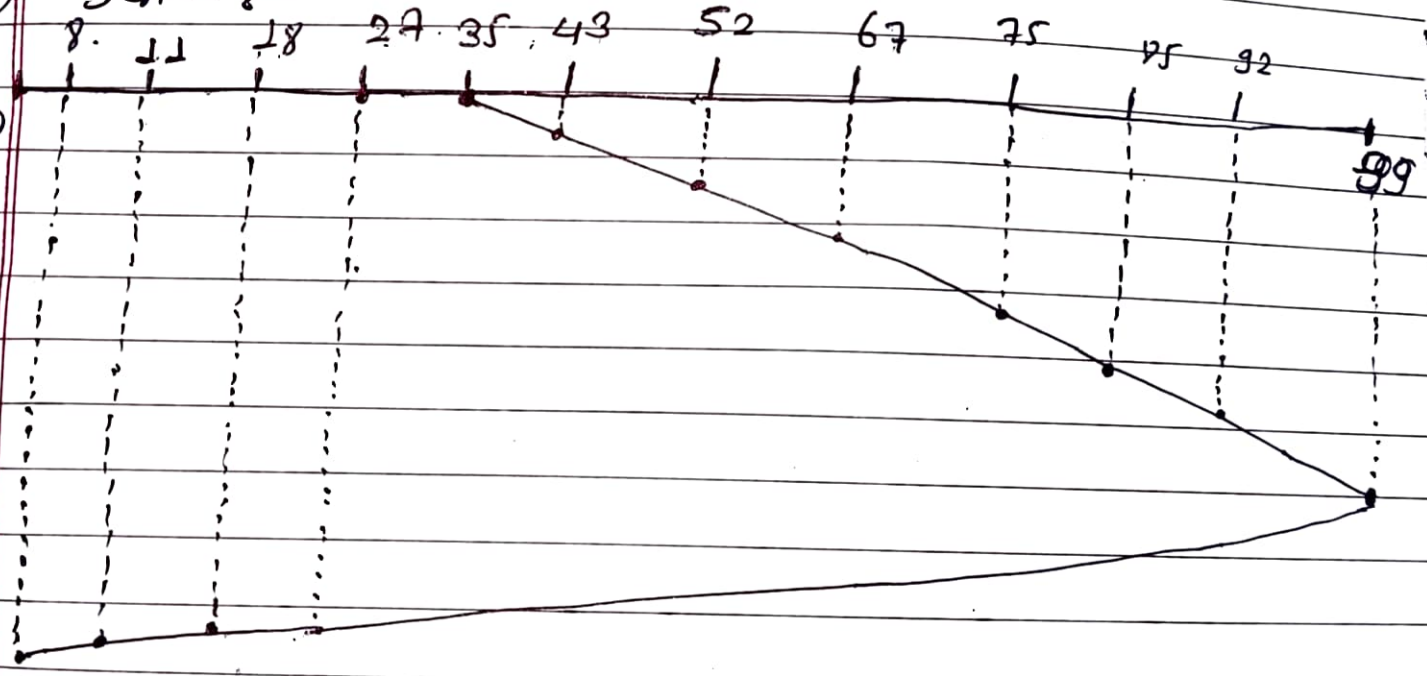
In Shortest-seek Time first (SSTF), requests having shortest seek time are executed first. So the seek time of every request is calculated in advance in queue & they are scheduled according to their calculated seek time. As a result, the request near the disk arm will get executed first.

But, it is not practically feasible because starvation is possible for some requests as it favours easy to reach request and ignores the far away processes and there is lack of predictability because of high variance of response time. And theoretically it is possible to compare seek time of every request, but practically what request will come in future isn't known at all.

Given, To requests, queue

52, 67, 27, 11, 43, 85, 18, 75, 92, 8

1) SCAN :-



∴ Total seek count by R/W head.

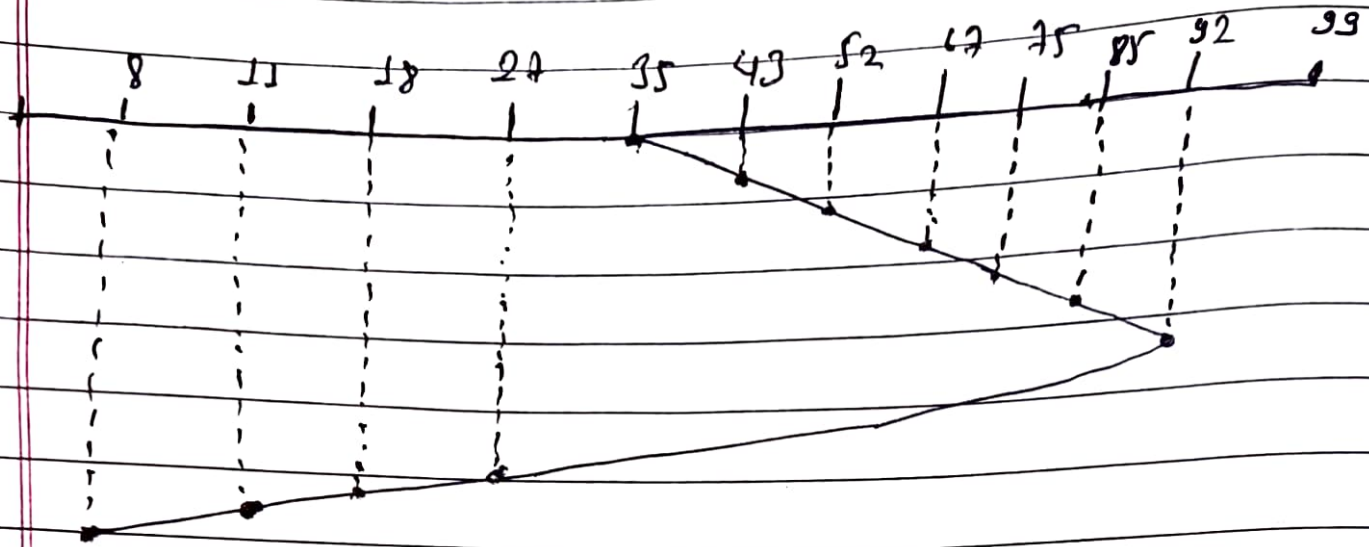
$= (99-35) + (99-8)$

$= 155$

Hence, Seek time is 155ms.

DATE

2. Look:



∴ Total seek count by R/W head = (92-35) + (92-8)
= 141

Hence, seek time = 141 ns

Section 'B'

DATE

Qu. What are the two modes of os? Discuss different os structures briefly.

=> The operating system has two modes of operation to ensure it works correctly

User mode: When the computer runs user applications like file creation or any other application program, in the user mode, this mode doesn't have direct access to the computers hardware. For performing any hard-ware related tasks, it must switch to kernel mode. To access, RAM or other hardware resources it has to make system calls. The bit mode is 1.

Kernel mode: The system starts in kernel mode when it boots up. It has direct access to all the underlying hardware resources. It also executes privileged functions / instructions which are not provided with user access. The bit mode is 0.

The Os has 3 structure

1) Simple Structure:

These type of os have simple system & rapidly expanded much further than their scope. Ex: MS-DOS. It was designed simply for a small groups of people and company. Here, application program can access ROM / BIOS, so if any application program crashes, whole system crashes.

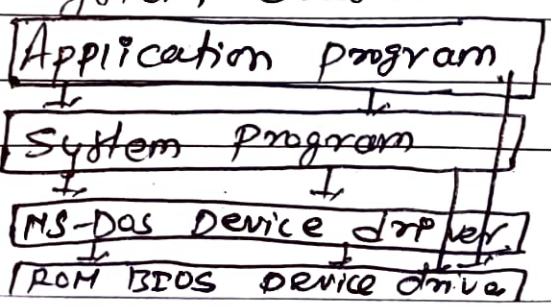


fig: MS DOS structure.

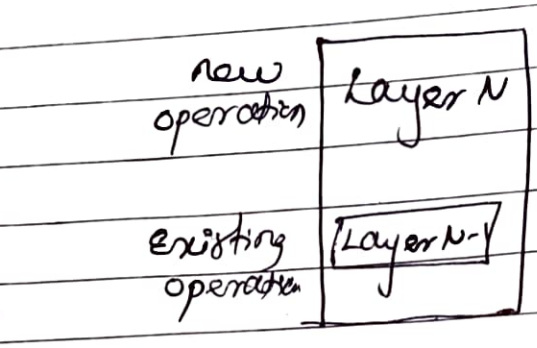
PAGE

DATE

b) Layered structure

Here, the OS is organized as a hierarchy of layers, each one constructed upon one below it. One way to achieve modularity is layered approach Bottom layer is hardware & topmost layer is UI

Layers function:
5- Operator
4- User program
3 - I/O Management
2- Operator process
1 - Memory Management
0 - processor allocation & multiprogramming



c) Monolithic system

These systems are written as a collection of procedures, each of which can call any of the other ones whenever it needs to. This organization suggests a basic structure for the

→ A main program that invokes the requested service procedure
→ A set of service procedures that carry out system calls
→ A set of utility procedures that help in service procedures

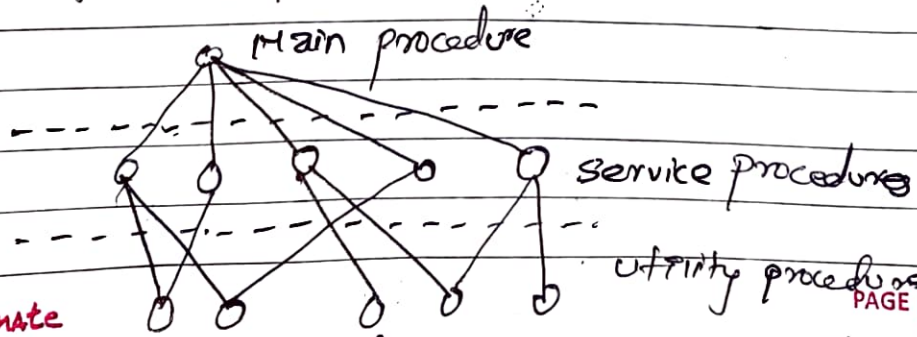


fig: Monolithic system structure

classmate

PAGE

DATE

Q.: When threads are better than process? Explain the concept of user-level threads in detail.

⇒ A thread is a single sequential flow of execution of tasks of a process. It is often referred as light-weight process. The process can be split down into so many threads. Threads are better than process as:-

→ It takes far less time to create a new thread in an existing process than to create new process.
→ Threads can share common data, they don't need to use inter-process communication
→ Context switching is faster when working with thread.

User-level thread

The OS doesn't recognize the user-level thread & these are easily implemented by the users. If a user performs a user level thread blocking operation, the whole process is blocked. The kernel level thread knows nothing about user-level thread.

Advantages:-
- Can be easily implemented & is more faster than Kernel level.
- Context switching is faster
- It is simple to create, switch & synchronize threads without interfering the process.

Disadvantage:- They lack coordination between the thread & Kernel.

PAGE

classmate

DATE

Q. Differentiate between multiprogramming and Monoprogramming. What will be the CPU utilization with 6 processes with 60% I/O waiting time are in memory?

A) Multiprogramming

It is the ability of an OS to execute more than one program on a single processor machine. More than one task/program/process can reside into the main memory at a point of time. Only one program at a time is able to get the cpu for executing its instructions while all the others are waiting for their turn. During this time, CPU switches back & forth from process to process. It happens in few millisecond.

Monoprogramming

It is the ability of OS to execute only one program at a time for the memory management scheme, sharing the memory between the program & the OS. Only one program can reside into the main memory at a point of time. It prevents or protects OS from getting into deadlock state.

DATE

We know, when there are n processes with p% of the time waiting for I/O. Then,

CPU utilization is given by $= 1 - p^n$

There are 6 processes with 60% I/O waiting time. $\therefore p = 60\% = 0.6$, $n = 6$

So, CPU utilization $= 1 - (0.6)^6$

$= 0.9533$

$= 95.33\%$

Hence, CPU is 95.33% utilized

DATE

Q.7. How can you manage free disk space? Explain the linked list approach of managing free disk space with example

⇒ As the disk space is limited, we need to reuse the space from deleted files for new files, if possible. To keep track of free space, the system maintains a free space list. The free space list records all free disk blocks those not allocated to some file or directory. There are mainly two approaches by which the free blocks in the disk are managed:

- Bitmap free space management
- Linked List free space management

Linked list free space management

It is one approach for free space management. This approach suggests linking together all the free blocks and keeping a pointer in cache which points to the first free block. So, all the free blocks on the disk will be linked together with a pointer.

When a block gets allocated, its previous free block will be linked together to its next free block. We would keep a pointer to block 2, as the first free block. Block 2 would contain a pointer to block 3, which would point to block 4

DATE

which would point to block 5 & so on. However, this scheme is not efficient; to traverse the list, we must read each block, which requires a lot of time

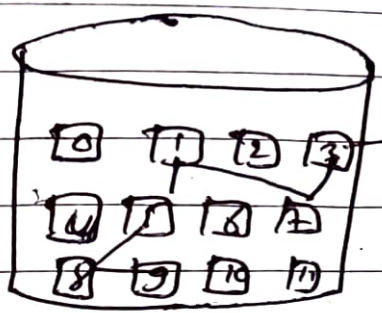


fig: Linked list free space in the disk.

Q8 When programmed IO is suitable than other IO handling techniques? Explain the process of IO handling with DMA.

=) Programmed IO is suitable than other IO handling technique when the cpu is not busy in other higher priority tasks and it can check each IO device in sequence & in effect "ask" each one of them if it needs communication with the processor. It is suitable when no interrupt hardware support is required.

DMA (Direct memory access) is the data transfer technique in which the peripherals (IO/devices) manage the memory buses for direct interaction with main memory without involving the cpu.

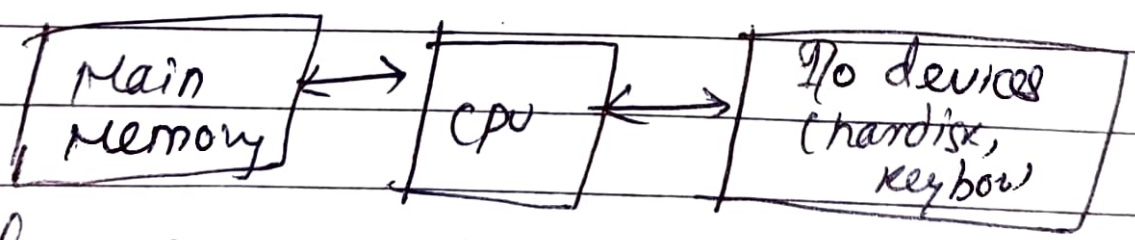


fig: Communication without DMA.

DMA uses an additional piece of hardware - DMA controller.
The DMA controller can take over the system bus and transfer data between an I/O module & main memory without the intervention of the CPU. Whereas CPU wants to transfer data, it tells the DMA controller the direction of the transfer, the I/O module involved, the location of data in memory, & size of block of data to be transferred. It can then ~~continue~~ continue with other instructions & the DMA controller will interrupt it when the transfer is complete.

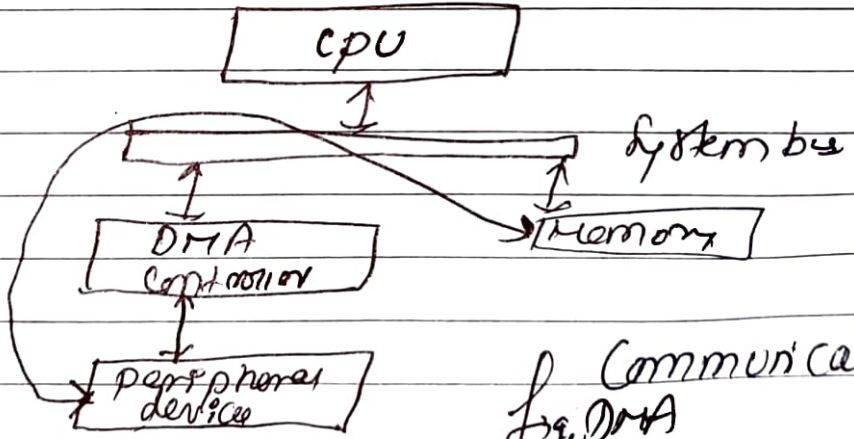


Fig. Communication using DMA

DATE

Q.9. Differentiate between deadlock and Starvation? Discuss the process of detecting deadlocks when there are multiple resources of each type.

=>

Deadlock	Starvation
i) It is also known as circular waiting.	i) It is also known as lived lock.
ii) In deadlock state, requested resources are blocked by the other process.	ii) In starvation, the requested resources are continuously used by high priority process.
iii) Requires external intervention to solve deadlock	iii) May or may not require external intervention to solve

The process of detecting deadlocks when there are multiple resource of each type is explained as

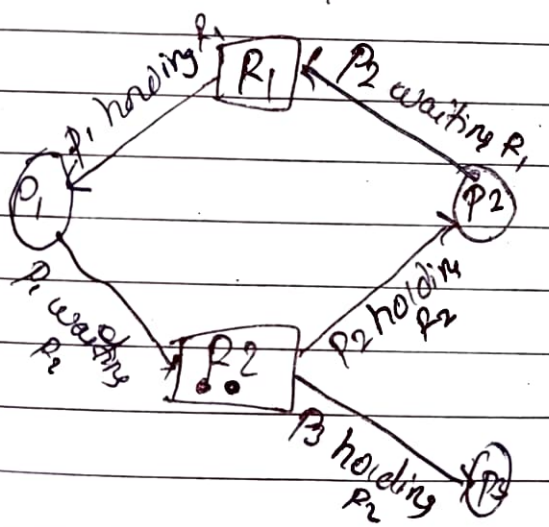
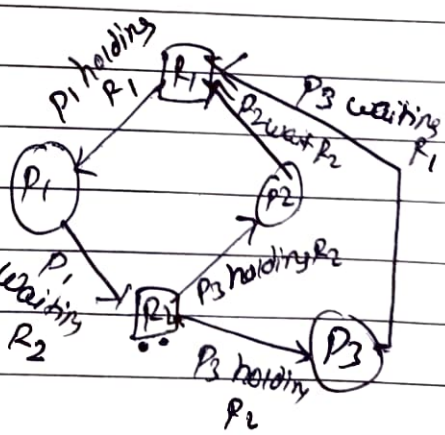


fig: Multiple instance resource without deadlock



fig(2) Multi instance resource with deadlock

classmate DATE

DATE

matrix representation of

Here, the Resource allocation graph for figure (2) a

Let 1 → True, 0 → False.

Process	Allocation resources		Request resource	
	R_1	R_2	R_1	R_2
P_1	1	0	0	1
P_2	0	1	1	0
P_3	0	1	0	0

fig (1) matrix representation

To see if there is deadlock, 1st we have to see the available resource. The available resource is [00]. as P_3 doesn't require any resource to execute. So, it releases R_2 after execution. Since, there are two instances of R_2, one is hold by P_2 & another is waiting by P_1. Then, P_1 executes & releases R_1. Then, P_2 executes & releases R_2. In this way there is no deadlock in above RAG. So, we can say, in multiinstance resource cycle is not sufficient condition for deadlock

For, fig 2.

Process	Resource Allocation		Resource request	
	R_1	R_2	R_1	R_2
P_1	1	0	0	1
P_2	0	1	1	0
P_3	0	1	1	0

Since, there are no available resource. as the resource R_2 is hold by P_3 and P_3 is waiting for R_1 which is hold by P_1. So, the dead lock occurs. Hence, if there are no available resource in matrix, then deadlock occurs

classmate PAGE