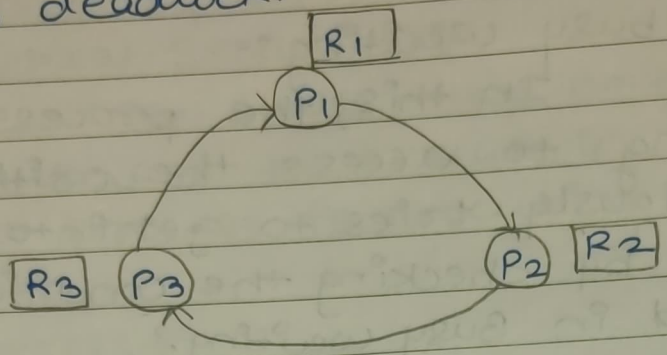
Date 28/11/22

Assignment no. 5

Q.1) What is deadlock?

→ • If a process is waiting for particular resource for a infinite time. It is called as deadlock.



• Deadlock can arise if four conditions hold simultaneously :-

a) Mutual Exclusion :-

Only one process at a time can use a resource.

b) Hold and wait :-

A process holding atleast one resource is waiting to acquire additional ~~requirements~~ resources held by other processes.

c) No preemption :-

A resource can be released only voluntarily by the process holding it, after that ~~it~~ process has completed its task.

d) Circular wait :-

There exists a set $\{P_0, P_1, \ldots, P_n\}$ of waiting processes such that P_0 is waiting for a resource that is held by P_1, P_1 is waiting for a resource that is held by $P_2, \ldots, P_{n-1}$ is waiting for

resource that is held by P_n, and P_n is waiting for a resource that is held by P_0.

Q.2) What are the conditions that must be satisfied to occur in details? deadlock?

→ · If a process is waiting for particular resource for a infinite time. It is called as deadlock.

· Deadlock can arise if four conditions hold simultaneously:-

a) Mutual Exclusion:-

Only one process at a time can use a resource.

b) Hold and Wait:-

A process holding atleast one resource is waiting to acquire additional resources held by other processes.

c) No preemption:-

A resource can be released only voluntarily by the process holding it, after that process has completed its task.

d) Circular Wait:-

There exists a set $\{P_0, P_1, \cdots, P_n\}$ of waiting processes such that P_0 is waiting for a resource that is held by P_1, P_1 is waiting for a resource that is held by $P_2, \cdots, P_{n-1}$ is waiting for resource ~~to~~ that is held by P_n, and P_n is waiting for a resource that is held by P_0.

Q.3) Explain Resource Allocation Graph?

→ • It is the set of vertices (V) and set of edges (E).

• Vertices (V) is partitioned into two types:

a) $P = \{P_1, P_2, \ldots, P_n\}$, the set containing of all the ~~resources~~ processes in the system.

b) $R = \{R_1, R_2, \ldots, R_n\}$, the set ~~of~~ consisting of all the resources (R) in the system.

• There are two types of edges:-

1) request edge:- Process to Resource.

$P_i \rightarrow R_j$.

2) Assignment edge:- Resource to Process.

$R_j \rightarrow P_i$.

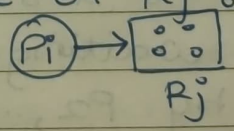
• If graph contains no cycle then there is no deadlock.

• If graph contains a cycle then if only one instance per resource type, then deadlock occurs. If several instances per resource type, then there is possibility of deadlock.

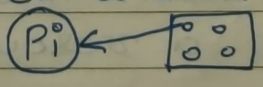
• Process is denoted by circle 'O'.

• Resource Type with 4 instances:-

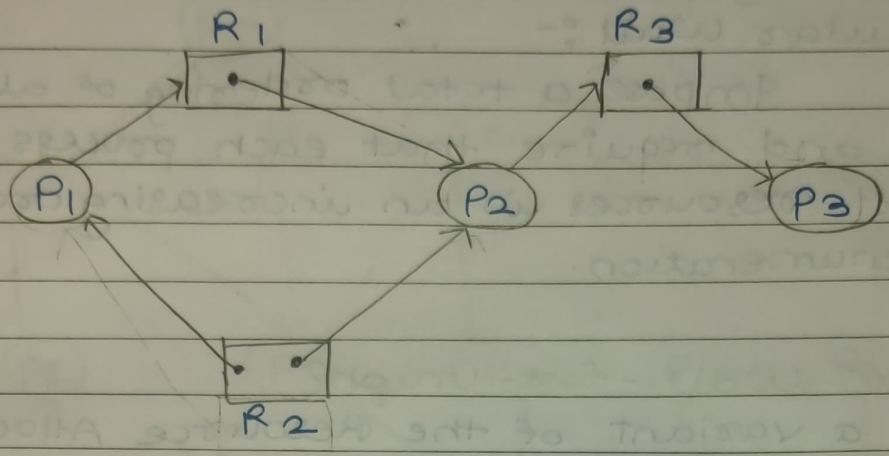
• Pi requests instance of Rj:-



• Pi is holding an instance of Rj:-



• Example of Resource Allocation Graph:-



Q.4) How can we prevent deadlock?

→ There are 4 methods by which we can prevent deadlock. They are as follows:-

a) Mutual Exclusion:-

It allows a shared resources among the process.

b) Hold and Wait:-

It must guarantee that whenever a process request a resource, it does not hold any other resources. Requires process to request and be allocated all its resources before it begin execution, or allow process to request resources only when the process has none allocated to it.

c) No preemption:-

If a process that is holding resources request another resource that cannot be immediately allocated to it, then all resources currently being held are released. Preempted resources are added

to the list of resources for which the process is waiting.

d) Circular Wait :-

Impose a total ordering of all types and require that each process request resources in an increasing order of enumeration.

Q.5) Explain Wait-for-Graph?

→ • It is a variant of the Resource Allocation graph. ~~In th~~

• In this algorithm, we only have processes as vertices in the graph.

• If the Wait-For-Graph contains a cycle then we can say the system is in Deadlock state.

• We need to remove Resources while converting from Resource Allocation Graph to Wait-For-Graph.

• Resource Allocation :- Contains processes and Resources.

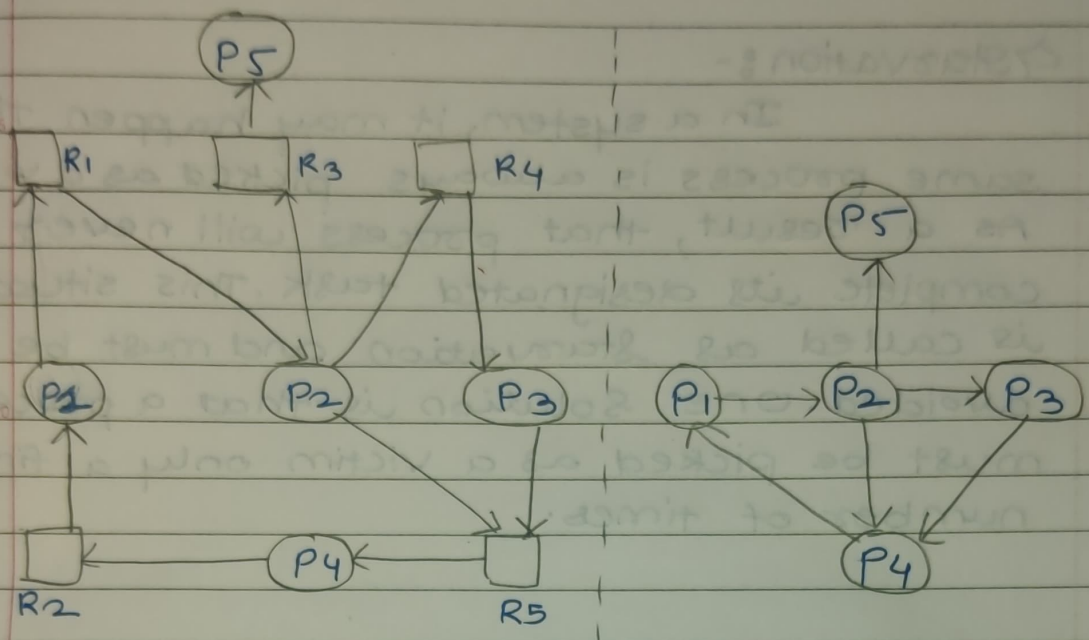
• Wait-For-Graph :- Contains only processes after removing the resources while conversion from Resource Allocation Graph.

~~Eg:-~~

~~P5~~

~~R1~~ ~~R3~~ ~~R2~~

Ex.



Resource Allocation Graph

Wait-For-Graph

Q6) How can we recover from the deadlock?

→ To eliminate deadlocks using resource preemption, we preempt some resources from processes and give those resources to other process.

a) Selecting a victim:-

We must determine which resource and which processes are to be preempted and also the order to minimize the cost.

b) Rollback:-

We must determine what should be done with the process from which resources are preempted. One simple idea is total rollback. That means abort

the process and restart it.

c) Starvation :-

In a system, it may happen that same process is always picked as a victim. As a result, that process will never complete its designated task. This situation is called as starvation and must be avoided. One solution is that a process must be picked as a victim only a finite number of times.