OPERATING SYSTEM

Deadlock

What Is Deadlock?

In a multiprogramming environment, several processes may compete for a finite number of resources. A process requests resources, if the resources are not available at that time, the process enters a wait state. It may happen that waiting processes will never again change state, because the resources they have requested are held by other waiting processes. This situation is called a deadlock.

Resource Concept

As a resource manager, the OS is responsible for the allocation of a vast array of resources of various types. It is the part of what makes the OS design.

When a particular process reaches a point at which it cannot effectively use a processor (such as during long wait for I/O completion), the OS dispatches another process to that processor. A user program currently occupying a particular range of locations in main memory may be removed or preempted by another program.

When resources are non-preemptible, they cannot be removed from the processes to which the are assigned until the processes voluntarily release them. For example tape driver and optical scanners are normally assigned to a particular process for period or minutes or hours.

Types of Resources

- **Preemptive Resources** : Resource that may be removed form process such as process or memory. These resources cannot be involved in dead lock. Ex – Memory, Processor.

- **Non-preemptive Resources** : Resource that cannot be forcefully removed from processes. Ex – Scanner, tape driver.

- **Shared Resources** : Resource that can be used by only process at a time. Ex – Disk drive.

- **Dedicated Resource** : Resource that may be used by only one process at a time.

Resources may be involved in Deadlock

- Resource that may be removed from process such as processor or memory. It can't be involved in Deadlock.
- Resource that cannot be forcibly removed from a process. Ex – Tape driver, scanner. It can be involved in deadlock.
- Resource that can be accessed by more than one process may be involved in deadlock.
- Resource that may be used by only one process at a time. It is also known as serially reusable resource.

Resource Deadlock

- Resource sharing is primary goal in multiprogramming computing system.

- When resources are shared among a set of processes, each process maintaining exclusive control over particular resources allocated to it, dead lock can develop in which some processes will never be able to complete execution.

- The result can be loss of work and reduced system throughput and system failure.

Four major conditions for dead lock to exist

- Mutual exclusion : A resource may be acquired exclusively by only one process at a time.
- Hold and wait : There must exist a process that is holding at least one resource and is waiting to acquire additional resources that are currently being held by other processes.
- No Preemption : A resource can be released only voluntarily by the process holding it, after that process has completed its task.
- Circular wait : Two or more processes are locked in a "circular chain" in which each process is waiting for one or more resources that the next process in the chain is holding.

Resource Deadlock

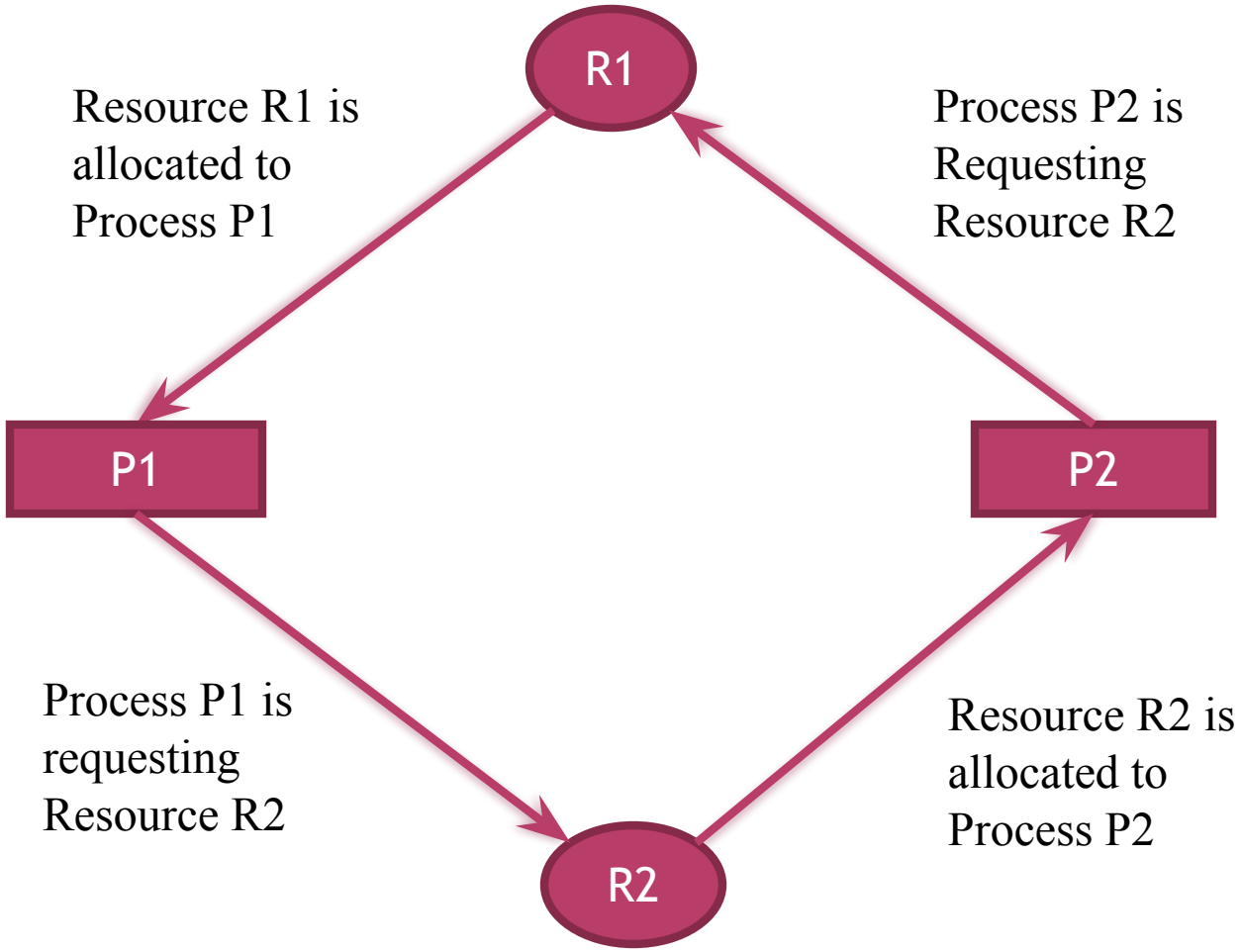


Fig : Resource Dead Lock

Deadlock Solutions

- Dead Lock Prevention
- Dead Lock Avoidance
- Dead Lock Detection
- Dead Lock Recovery

Deadlock Prevention

Process of disallowing deadlock by eliminating one of the four necessary conditions for deadlock.

1. In deadlock prevention our concern is to condition a system to remove any possibility of deadlocks occurring.
2. Prevention is a clean solution as far as deadlock itself is concerned.
3. Prevention methods can often result in poor resource utilization.
4. Deadlock prevention consider various methods and examine the effect on both users and system, especially from the stand point of performance.

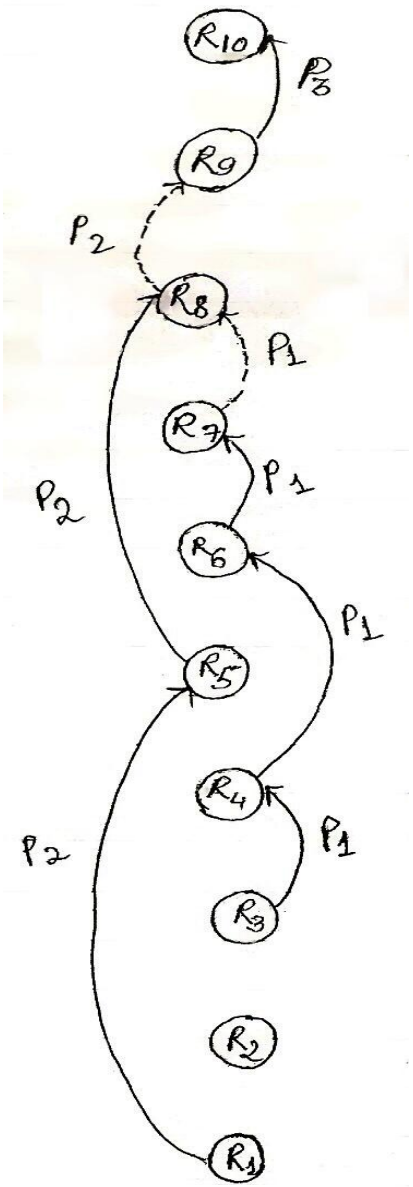
Deadlock Prevention Strategies

- Each process must request all its required resources at once and cannot proceed until all have been granted.

- If a process holding certain resources is denied a further request, it must release its original resources and if necessary, request them again together with the additional resources.

- A linear ordering of resources must be improved on all processes i.e. if a process has been allocated certain resources, it may subsequently request only those resources later in the ordering.

Havender's linear ordering or resources for preventing Deadlock



Process P1 has obtained resources R3, R4, R6, R7 and is requesting resource R8 (indicated by dotted line). No circular wait can be developed because all arrows point upwards.

Deadlock Avoidance

Avoidance methods do not precondition the system to remove all possibility o deadlock, instead they allow the possibility to loom, but whenever a deadlock is approached it is carefully sidestepped.

Strategy that eliminates deadlock by allowing a system to approach deadlock but ensuring that deadlock never occurs.

Deadlock Detection

Deadlock detection is the process of determining that a deadlock exists and identifying the processes involved in the deadlock.

"Process of determining whether or not a system is deadlocked. Once detected, a deadlock can be removed from a system, typically resulting in loss of work."

Resource allocation graph

1.

P1 is requesting a resource of type R1 of which these are two identical resources.

Process P1 is requesting a resource of type R1. The arrow from P1 touches only the extremity of the large circle, indicating that the resource request is under consideration

Resource allocation graph

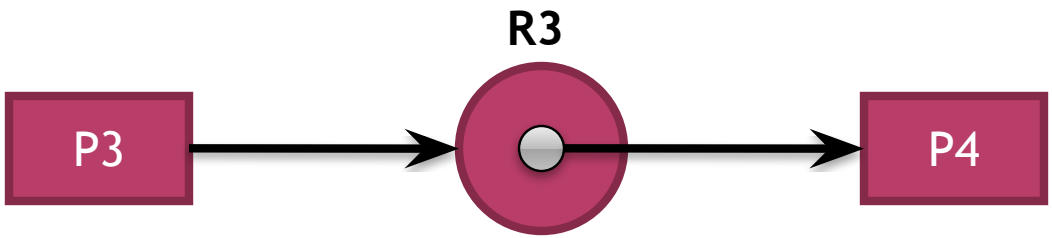
2.

One of the two identical resources of type R2 has been allocated to process P2

The arrow is drawn from the small circle within the large circle, R2 to the square P2 to indicate that the system has allocated a specific resource of that type to the process.

Resource allocation graph

3.

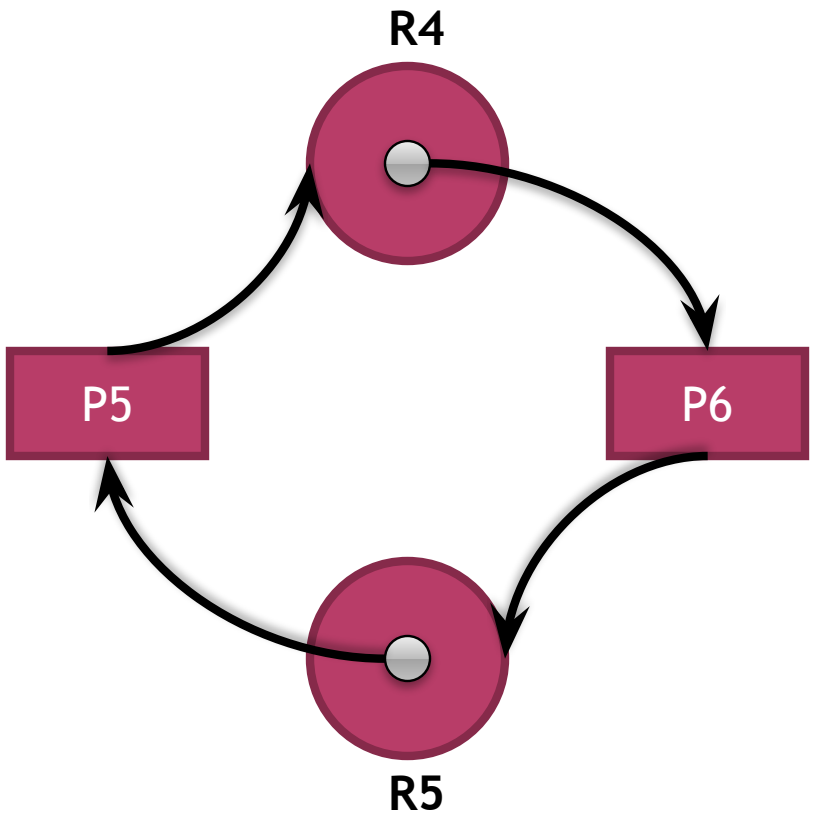


Process P3 is requesting resource R3, which has been allocated to process P4.

It indicates situation somewhat closer to a potential deadlock. Process P3 is requesting a resource of type R3 but the system has allocated the only R3 resources to the process P4

Resource allocation graph

4.



Process P5 has been allocated resource R5 that is being requested by process P6 that has been allocated resource R4 that is being requested by process P5 (the classic “circular wait”.

Deadlock Recovery

Deadlock recovery methods are used to clear deadlocks from the system so that it may operate free of deadlocks, and so that the deadlocked process may complete their execution and free their resources.

Recovery from deadlock is complicated by several factors:

1. It may not be clear that the system has become deadlocked. Ex – most system contain processes that wake periodically, perform certain task then go back to sleep because such processes do not terminate until the system is shutdown, and because they rarely enter the active state, it is difficult to determine if they are deadlocked.

2. Most systems do not provide the means to suspend a process indefinitely remove it from the system and resume it.
3. Recovery from deadlock is complicated because the deadlock could involve many processes (tens or even hundreds)
4. Recovery somehow seems like an appropriate term because some processes are in fact “killed” for the benefit of he others.
5. The suspend/resume mechanism allows the system to put a temporary hold on process and when it is safe to do so, to resume the held process without loss of work.

ASSIGNMENT QUESTIONS ON UNIT-2

- Q.1. Explain Processor scheduling? Explain types of scheduler. 6
- Q.2. Explain Scheduling Criteria? 4
- Q.3. Differentiate Preemptive and Non-preemptive 5
- Q.4. Explain Scheduling Algorithm & explain 1.FIFO 2. ROUND-ROBIN. 4
- Q.5. Explain Process Synchronization. 4
- Q.6. Explain Mutual Exclusion with example. 4
- Q.7. Explain Semaphore with proper example. 4
- Q.8. Explain Monitors with proper example. 4
- Q.9. What is deadlock ? explain and characterization of deadlock 4
- Q.10. What is Deadlock prevention and avoidance? Explain in detail. 6
- Q.11. What is Deadlock detection and recovery? Explain in detail. 6