Unit 3:

CPU Scheduling

CPU scheduling is a process which allows one process to use the CPU while the execution of another process is on hold due to unavailability of any resources, thereby making full use of CPU. The aim of CPU scheduling is to make the system efficient, fast and fair.

Whenever the CPU becomes idle, the operating system must select one of the processes in the ready queue to be executed.

Performance Criteria for CPU scheduling:

1. CPU utilization

CPU utilization is the average function of time, during which the processor is busy.

2. Throughput

It refers to the amount of work completed in a unit of time. The number of processes the system can execute in a period of time. The higher the number, the more work is done by the system.

3. Waiting time

The average period of time, a process spends in waiting.

4. Turnaround time

It is the time interval from the time of submission of a process to the time of the completion of the process.

5. Response Time

It is the difference between first execution time and Arrival time.

6. Priority

Give preferential treatment to processes with higher priorties.

7. Balanced Utilization

Utilization of memory, I/O devices and other system resources are also considered.

8. Fairness

Avoid the process from starvation. All the processes must be given equal opportunity to execute.

Objective of CPU Scheduling:

1) Efficiency must be very high.
2) Maximize throughput
3) Minimize response time
4) Minimize overhead
5) Maximize resource use
6) Avoid Indefinite postponement
7) Enforce priorities

Types of Scheduling

1) Pre-emptive Scheduling
2) Non Pre-emptive Scheduling

Pre-emptive Scheduling:

Processor can be pre-empted to execute a different process in the middle of execution of any current process.

Non pre-emptive Scheduling:

Once processor start to execute a process it must finish it before executing the other. It cannot be paused in middle.

Page No. _____
Date

Scheduler

Scheduler are special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run.

Schedulers are of three types –

- Long-term scheduler
- short-term scheduler
- Medium-term scheduler

Long-term scheduler

It is also called a job scheduler. A long-term scheduler determines which programs are admitted to the system for processing.

It selects processes from the queue and loads them into memory for execution. Process loads into memory for CPU scheduling.

Short-term Scheduler

It is also called as CPU scheduler. Its main objective is to increase system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the processes. CPU scheduler selects a process among the processes that are ready to execute and allocates CPU to one of them.

It is also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers.

Page No.
Date

Medium Term Scheduler

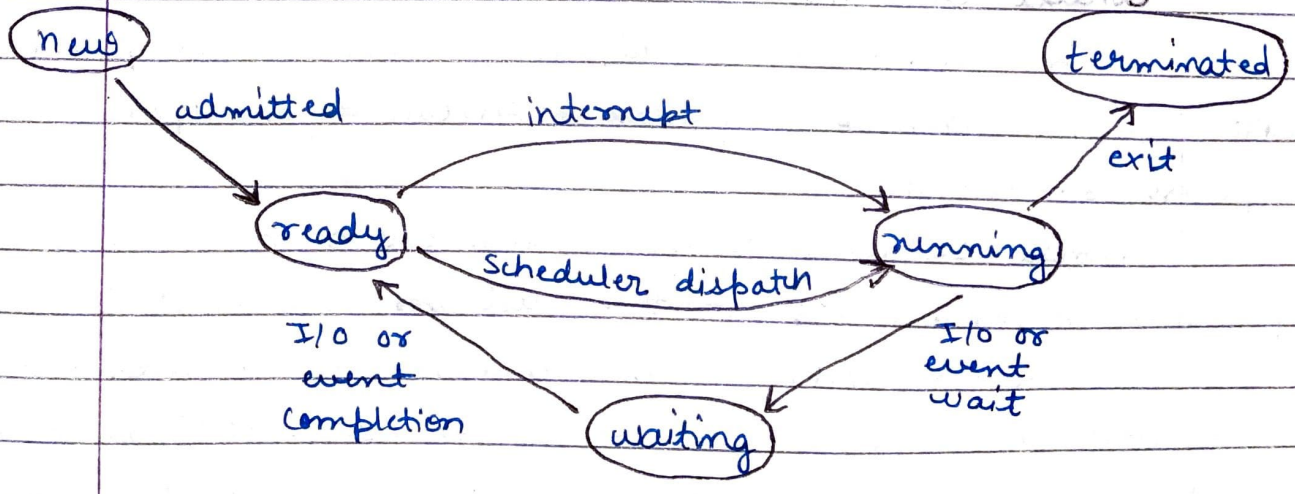
It is a part of scheduling. It removes the processes from the memory. It reduces the degree of multiprogramming.

A running process may become suspended if it makes an I/O request. A suspended processes cannot make any progress towards completion.

In this condition, to remove the process from memory and make space for other processes, the suspended process is moved to the secondary storage. This process is called swapping and the process is said to be swapped out or rolled out.

Different Process States

- New- The process is being created.
- Ready- The process is waiting to be assigned to a processor.
- Running - Instructions are being executed.
- waiting- The process is waiting for some event to occur.
- Terminated- The process has finished execution.



Page No.
Date

Process Control Block

There is a Process Control Block for each process enclosing all the information about the process. It is a data structure, which contains the following:

1) Naming the process
2) State of the process
3) Resources allocated to the process
4) Memory allocated to the process
5) Scheduling information
6) I/O devices associated with process

Components of Process Control Block

1) Process ID
2) Process State
3) Program Counter
4) Register information
5) Scheduling information
6) Memory related information
7) Accounting information
8) Status information related to I/O.

Context Switching:

A context switch occurs when a computer's CPU switches from one process or thread to a different process or thread.

Context switching allows for one CPU to handle numerous processes or threads without the need for additional processors.

A context switch is the mechanism to store and restore the state or context of a CPU in process control block so that a process execution can be resumed from the same point at a later time.

Date

Process Address space

An address space is a range of valid addresses in memory that are available for a program or process. That is, it is the memory that a program or process can access. The memory can be either physical or virtual and is used for executing instructions and storing data.

Address space can be of two types:

a) Physical address space:

Physical address space is created in RAM.

b) Virtual address space:

Virtual address space is an address space that is created outside the main memory inside the virtual memory and it is created in hard disk.

Thread

A thread is a flow of execution through the process code, with its own program counter, system registers and stack. A thread is a path of execution within a process. A process can contain multiple threads.

A thread is known as light weight process. The idea is to achieve parallelism by dividing a process into multiple threads.

Advantages of Thread:

1) Thread minimize context switching time.
2) Use of thread provides concurrency within a process
3) Efficient communication.
4) It is more economical to create and context switch threads.

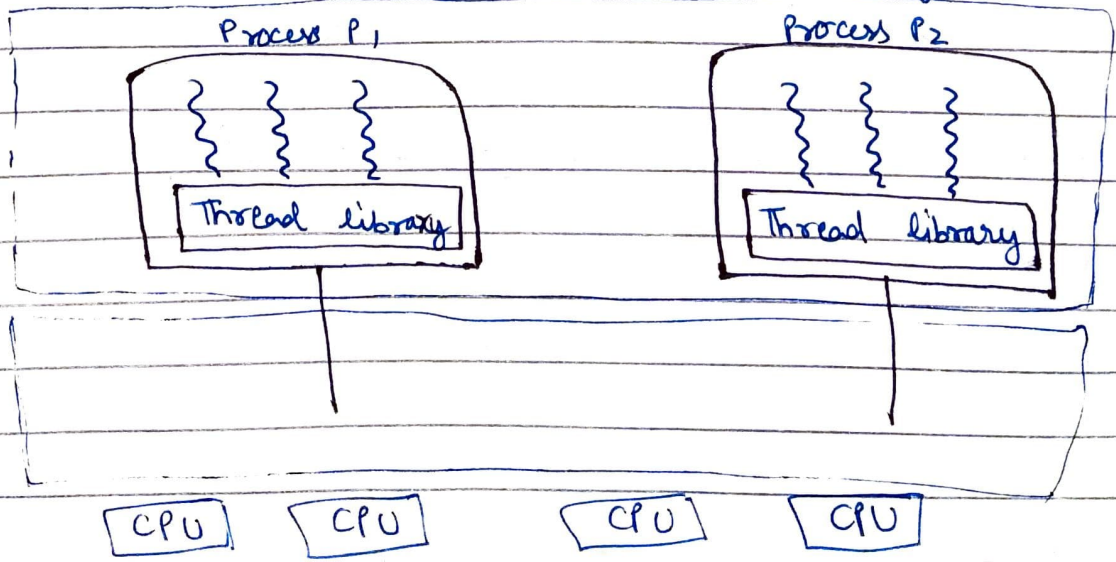
Process	Thread
1) Process is heavy weight.	1) Thread is light weight.
2) Process switching needs interaction with operating system	2) Thread switching does not need to interact with OS.
3) If one process is ~~execute~~ blocked, then no other process can execute.	3) while one thread is block, then second thread in the same task can run.
4) In multiple process implementation each process execute the same code but has its own memory and file resources	4) All threads can share same set of open files, child processes.
5) Each process operates independently of the other.	5) one thread can read, write or even completely wipe out another threads stack.

Types of Thread

1) User level threads
2) Kernel level threads

1) User level Threads

The thread library contains code for creating and destroying threads, for passing message and data between threads, for scheduling thread execution and for saving and restoring thread contexts. The application starts with a single thread.



Page No.
Date

Advantages:

1) Thread switching does not require Kernel mode privileges
2) User level thread can run on any OS.
3) User level thread are fast to create and manage.
4) Scheduling can be application specific.

Disadvantages:

1) In some OS, most system calls are blocking.
2) Multi threaded application cannot take advantage of multiprocessing.

2) Kernel Level Threads

Thread management is done by Kernel. Kernel threads are supported directly by the OS. Any application can be programmed to be multithreaded. All of the threads within an application are supported within a single process.

The kernel maintains context information for the process as a whole and for individuals threads within the process. Scheduling by the kernel is done on a thread basis. The Kernel performs thread creation, scheduling and management in kernel space. Kernel threads are generally slower to create and manage than the user threads.

Advantages:

1) If one thread in a process is blocked, the kernel can schedule another thread of the same process.
2) Kernel can simultaneously schedule multiple threads from the same process on multiple processes.
3) Kernel routines themselves can be multithreaded.

Disadvantages:

1) Transfer of control from one thread to another within the same process requires a mode switch to the Kernel.

2) Kernel threads are generally slower to create and manage than the user threads.

Thread Cancellation:

It is the task of terminating a thread before it has completed. For example, if multiple threads are concurrently searching through a database and one thread returns the result, the remaining threads might be cancelled.

First Come First Serve Algorithm

- Jobs are executed on first come, first serve basis.
- Easy to understand and implement
- Its implementation based on FIFO queue.
- Poor in performance as average wait time is high.

Advantages: 1) Better for long processes
2) No starvation and simple method

Disadvantages: 1) Even very small process should wait for its turn to come to utilize the CPU.
2) Throughput is not emphasized.

Completion time: Time taken for execution to complete, starting from arrival time.

Turn around time: Time taken to complete after arrival.
= Completion time - Arrival time

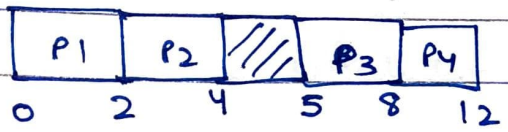
Waiting time: It is the difference between the Turn around time and burst time of the process.
= Turnaround time - Burst time

Page No.
Date

Q.

Process	Arrival time	Burst time	Completion time	Turn around	waiting time	Respone time
P_1	0	2	2	2	0	0
P_2	1	2	4	3	1	1
P_3	5	3	8	3	0	0
P_4	6	4	12	6	2	2

Gantt chart



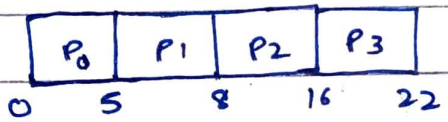
Average turn around time = $\frac{2+3+3+6}{4} = 3.5$

Average waiting time = $\frac{0+1+0+2}{4} = 0.75$

CPU utilization = $\frac{11}{12} \times 100 = 91.66\%$.

Q.

Process	Arrival time	Execution time	Completion time	Turn around	Waiting time
P_0	0	5	5	5	0
P_1	1	3	8	7	4
P_2	2	8	16	14	6
P_3	3	6	22	19	13



Gantt chart

Average turn around time = $\frac{5+7+14+19}{4} = 11.25$

Average waiting time = $\frac{0+4+6+13}{4} = 5.75$

CPU Utilization = $\frac{22}{22} \times 100 = 100\%$.

Page No. _____
Date

Q.

Process	Arrival time	Burst time	Turnaround time	Waiting time	Completion time
P_1	0	2	2	0	2
P_2	1	3	4	1	5
P_3	2	5	8	3	10
P_4	3	4	11	7	14
P_5	4	6	16	10	20

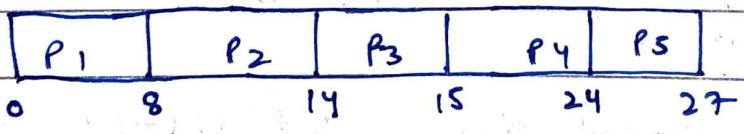
Average turn around time = $\frac{2+4+8+11+16}{5} = 8.2$

Average waiting time = $\frac{0+1+3+7+10}{5} = 4.2$

CPU utilization = $\frac{20}{20} \times 100 = 100\%$

Q.

	Burst	Priority
P_1	8	4
P_2	6	1
P_3	1	2
P_4	9	2
P_5	3	3



Average turn around time = $\frac{8+14+15+24+27}{5} = 17.6$

Average waiting time = $\frac{0+8+14+15+24}{5} = 12.2$

CPU utilization = 100%.

Shortest Job first (SJF) Scheduling Algorithm

Shortest job first or shortest job next (SJN) is a scheduling policy that selects the waiting process with the smallest execution time to execute next. SJN is a non-preemptive algorithm.

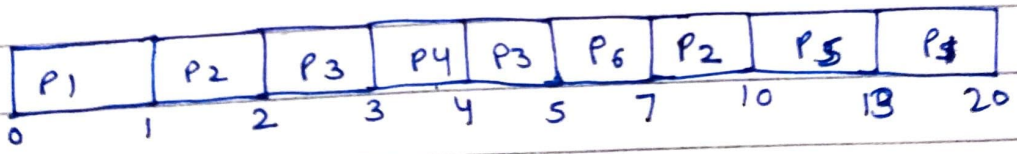
- SJF has the advantage of having minimum average waiting time among all scheduling algorithms.
- It is a Greedy Algorithm
- It may cause starvation if shorter processes keep coming.
- It is practically infeasible as OS may not know burst time and therefore may not sort them.

Advantages: Throughput is high.

Disadvantages: Starvation may be possible for longer processes.

Q.

Process	Arrival time	Burst time	Completion time	Turn Around time	Waiting time
P1	0	~~8~~ 7	20	20	12
P2	1	~~4~~ 3	10	9	5
P3	2	~~2~~ 1	5	3	1
P4	3	1	4	1	0
P5	4	3	13	9	6
P6	5	2	7	2	0



Average turn around time = $\frac{20+9+3+1+9+2}{6} = 7.333$

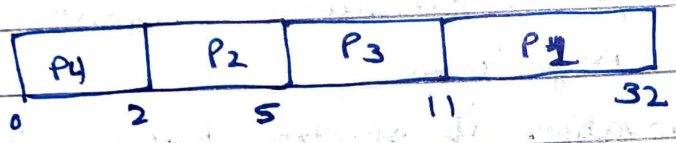
Average waiting time = $\frac{12+5+1+0+6+0}{6} = 4$

Date

Non-preemptive Shortest Job first

Q.

Process	Burst time
P_1	21
P_2	3
P_3	6
P_4	2

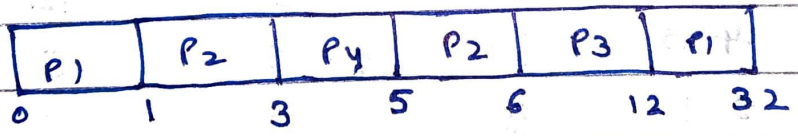


$$\text{Average waiting time} = \frac{0+2+5+11}{4} = 4.5$$

Pre-emptive Shortest Job first

Q

Process	Burst time	Arrival time	completion time	Turn Around time	waiting time
P_1	21 20	0	32	32	11
P_2	3 1	1	6	5	2
P_3	6	2	12	10	4
P_4	2	3	5	2	0



$$\text{Average turn around time} = \frac{32+5+10+2}{4} = 12.25.$$

$$\text{Average waiting time} = \frac{11+2+4+0}{4} = 4.25$$

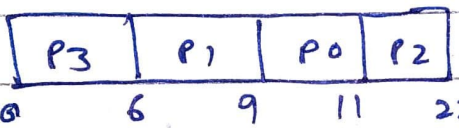
Date

Priority Scheduling Algorithm

- Priority scheduling is a non-preemptive algorithm
- Each process is assigned a priority. Process with highest priority is to executed first and so on.
- Processes with same priority are executed on first come first served basis.

Q.

Process	Arrival time	Execution time	Priority	Completion time	Turnaround time	waiting time
P_0	0	5	1	11	11	0
P_1	1	3	2	9	8	6
P_2	2	8	1	22	20	9
P_3	3	6	3	6	3	11



Average waiting time = $\frac{0+6+9+11}{4} = 6.5$

Average turn around time = $\frac{11+8+20+3}{4} = 10.5$

Advantages: 1) The priority of a process can be selected based on memory requirement, time requirement or user preference.

2) Priority scheduling provides a good mechanism where the relative importance of each process may be precisely defined.

Disadvantages:

If high priority processes use up a lot of CPU time, lower priority processes may starve and be postponed indefinitely.

Round Robin Scheduling Algorithm

Round Robin is a CPU scheduling algorithm where each process is assigned a fixed time slot in a cyclic way. It is simple, easy to implement and starvation free as all processes get fair share of CPU.

One of the most commonly used technique in CPU scheduling as a core. It is preemptive as processes are assigned CPU only for a fixed slice of time at most.

A fixed time is allocated for each process called quantum for execution. Once a process is executed for given time period that process is preempted and other process executes for given time period.

Advantages of RR scheduling:

1) fair treatment for all the processes.
2) Good response time.

Disadvantages of RR scheduling:

1) Care must be taken in choosing quantum value.
2) Throughput is low if time quantum is too small.

Q.

Process	Arrival time	Execution time	completion time	Turnaround time	Waiting time
A	0	4 2	6	6	2
B	2	7 5 3 1	22	20	13
C	3	3 1	15	12	9
D	3.5	3 1	16	12.5	9.5
E	4	5 3 1	21	17	12

Ready Queue

A	B	A	C	D	E	B	C	D	E	B	E	B

Running Queue

A	B	A	C	D	E	B	C	D	E	B	E	B

0 2 4 6 8 10 12 14 15 16 18 20 21 22

Page No. _____
Date

Average turnaround time = $\frac{6 + 20 + 12 + 12.5 + 17}{5} = 13.5$

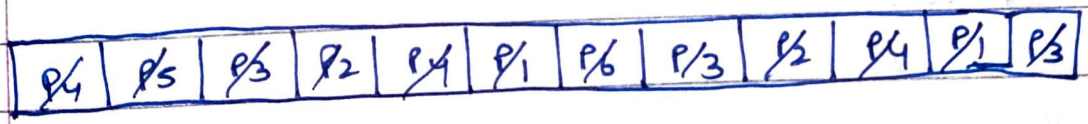
Average waiting time = $\frac{2 + 13 + 9 + 9.5 + 12}{5} = 9.1$

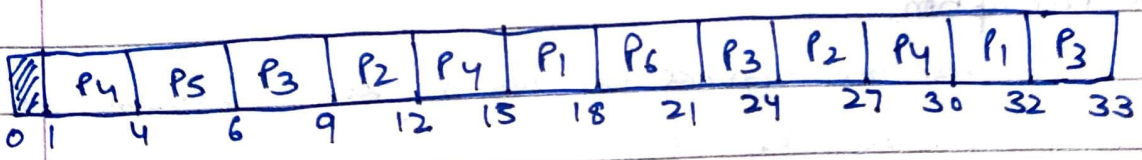
Q.

Process	Arrival time	Burst time
P_1	5	5
P_2	4	6
P_3	3	7
P_4	1	9
P_5	2	2
P_6	6	3

Sol.

Process	Arrival time	Burst time	completion time	Turnaround time	waiting time
P_4	1	~~9 6 3~~	30	29	20
P_5	2	~~2~~	6	4	2
P_3	3	~~7 4 1~~	33	30	23
P_2	4	~~6 3~~	27	23	17
P_1	5	~~8 2~~	32	27	22
P_6	6	~~3~~	21	15	12



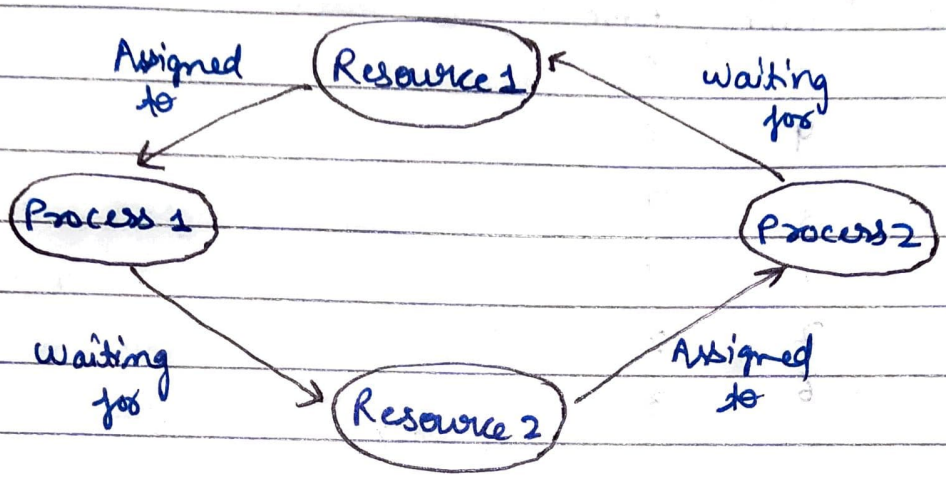


Average turn around time = $\frac{29 + 4 + 30 + 23 + 27 + 15}{6} = 21.33$

Average waiting time = $\frac{20 + 2 + 23 + 17 + 22 + 12}{6} = 16$

Deadlock

Deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other processes.



Deadlock can arise if following four conditions hold simultaneously:

1) Mutual Exclusion:
   One or more than one resources are non-sharable.

2) Hold and wait:
   A process is holding at least one resource and waiting for resources.

3) NO Preemption:
   A resource cannot be taken from a process unless the process releases the resource.

4) Circular wait:
   A set of processes are waiting for each other in circular form.

Methods for handling deadlock

There are three ways to handle deadlock

1) Deadlock prevention or avoidance
2) Deadlock detection and recovery
3) Ignore the problem all together

Deadlock prevention:

1) Avoid mutual exclusion

If a process could have been used by more than one process at the same time then the process would have never been waiting for any resource.

2) Avoid hold and wait

A process must be assigned all the necessary resources before the execution starts. A process must not wait for any resource once the execution has been started.

3) Avoid no preemption

Deadlock arises due to the fact that a process can't be stopped once it starts. However, if we take the resource away from the process which is causing deadlock then we can prevent deadlock.

4) Avoid Circular wait

we can assign a priority number to each of the resource. A process can't request for a lesser priority resource. This ensures that not a single process can request a resource which is being utilized by some other process and no cycle will be formed.

Avoidance Algorithm:

- Single instance of a resource type - Resource-Allocation graph
- Multiple instance of a resource type - Banker's algorithm

Page No.
Date

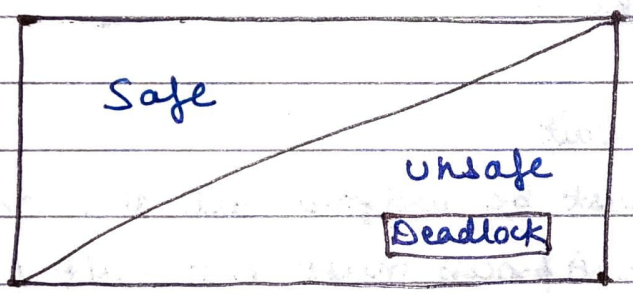
Deadlock Avoidance

Deadlock avoidance allows the three necessary conditions but makes judicious choice to assure that the deadlock is never reached.

Two approaches are used to avoid deadlock:

1. Do not start a process if its demands might lead to deadlock.
2. Do not grant an incremental resource request to a process if this allocation might lead to deadlock.

- System can be in one of the following states:



1. Safe state:

when the system can allocate resources to each process in some order and avoid a deadlock.

2. Unsafe state:

If the system did not follow the safe sequence of resource allocation from the beginning and it is now in a situation, which may lead to a deadlock.

3. Deadlock state:

If the system has some circular-wait condition existing for some processes, then it is in deadlock.