CPU Scheduling Algorithms

A Process Scheduler schedules different processes to be assigned to the CPU based on particular scheduling algorithms. There are six popular process scheduling algorithms which we are going to discuss in this chapter −

- First-Come, First-Served (FCFS) Scheduling
- Shortest-Job-Next (SJN) Scheduling
- Priority Scheduling
- Shortest Remaining Time
- Round Robin(RR) Scheduling
- Multiple-Level Queues Scheduling

These algorithms are either **non-preemptive or preemptive**. Non-preemptive algorithms are designed so that once a process enters the running state; it cannot be preempted until it completes its allotted time, whereas the preemptive scheduling is based on priority where a scheduler may preempt a low priority running process anytime when a high priority process enters into a ready state.

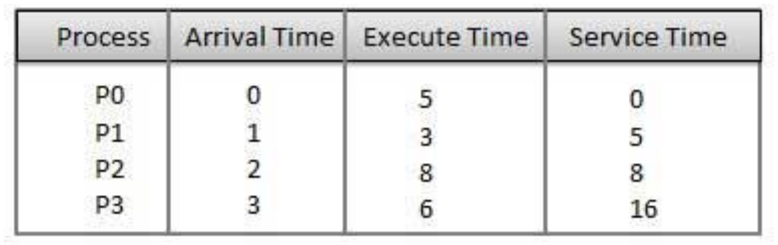
First Come First Serve (FCFS)

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

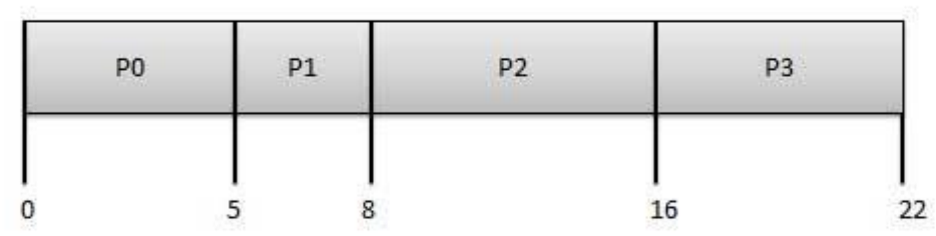
Convoy Effect in FCFS :

Convoy Effect is a situation where many processes, which need to use a resource for short time, are blocked by one process holding that resource for a long time.

This essentially leads to poor utilization of resources and hence poor performance.



Process	Arrival Time	Execute Time	Service Time
P0	0	5	0
P1	1	3	5
P2	2	8	8
P3	3	6	16



Wait time of each process is as follows −

Process	Turnaround Time	Wait Time : Turnaround Time - Burst Time
P0	5-0=5	5-5=0
P1	8-1=7	7-3=4
P2	16-2=14	14-8 = 6
P3	22-3=19	19 - 6 = 13

Average Wait Time: (0+4+6+13) / 4 = 5.75

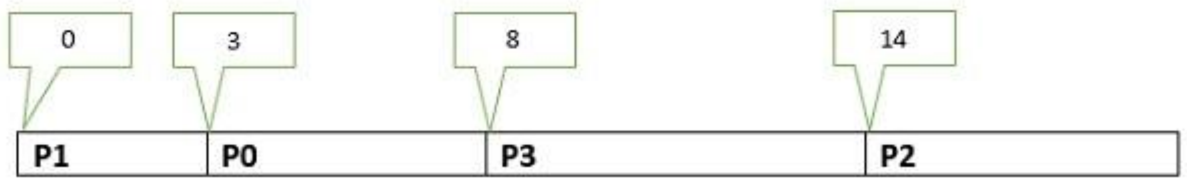
Shortest Job Next (SJN)/Shortest Job First

- This is also known as **shortest job first**, or SJF
- This is a non-preemptive, pre-emptive scheduling algorithm.
- Best approach to minimize waiting time.
- Easy to implement in Batch systems where required CPU time is known in advance.
- Impossible to implement in interactive systems where required CPU time is not known.

- The processer should know in advance how much time process will take.

Process	Execute/Burst Time	Service Time
P0	5	3
P1	3	0
P2	8	14
P3	6	8

Gantt Chart



Wait time of each process is as follows –

Process	Turnaround Time	Wait Time : Turnaround Time - Burst Time
P0	8	8-5 = 3
P1	3	3-3=0
P2	22	22 - 8 = 14
P3	14	14-6=8

Average Wait Time: (3+0+14+8) / 4 = 25/4=6.25

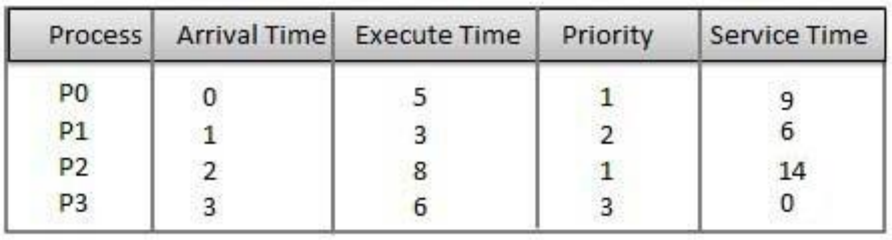
Shortest Remaining Time

- Shortest remaining time (SRT) is the preemptive version of the SJN algorithm.

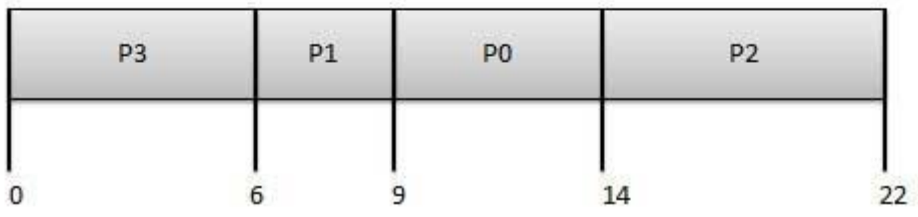
- The processor is allocated to the job closest to completion but it can be preempted by a newer ready job with shorter time to completion.
- Impossible to implement in interactive systems where required CPU time is not known.
- It is often used in batch environments where short jobs need to give preference.

Priority Based Scheduling

- Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems.
- Each process is assigned a priority. Process with highest priority is to be executed first and so on.
- Processes with same priority are executed on first come first served basis.
- Priority can be decided based on memory requirements, time requirements or any other resource requirement.



Process	Arrival Time	Execute Time	Priority	Service Time
P0	0	5	1	9
P1	1	3	2	6
P2	2	8	1	14
P3	3	6	3	0



Do Not Consider Arrival Time: All process arrives at Time t=0

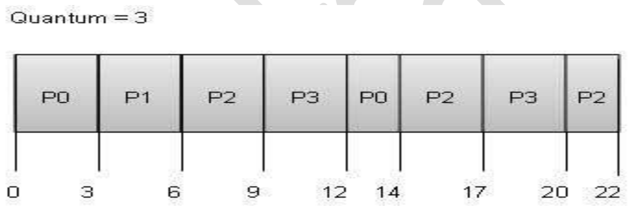
Process	Turnaround Time	Wait Time : Turnaround Time - Burst Time
P0	**14**	**14-5=9**
P1	**9**	**9-3=6**

P2	**22**	**22-8=14**
P3	**6**	**6-6= 0**

Average Wait Time: (9+6+14+0) / 4 = 29/4=6.25

Round Robin Scheduling

- Round Robin is the preemptive process scheduling algorithm.
- Each process is provided a fix time to execute, it is called a **quantum**.
- Once a process is executed for a given time period, it is preempted and other process executes for a given time period.
- Context switching is used to save states of preempted processes.



Wait time of each process is as follows –

Process	**Turnaround Time**	**Wait Time : Turnaround Time - Burst Time**
P0	**14-0=14**	**14-5=9**
P1	**6-1=5**	**5-3=2**
P2	**22-2=20**	**20-8=12**
P3	**20-3=17**	**17-6= 11**

Average Wait Time: (9+2+12+11) / 4 = 8.5

Multiple-Level Queues Scheduling

Multiple-level queues are not an independent scheduling algorithm. They make use of other existing algorithms to group and schedule jobs with common characteristics.

- Multiple queues are maintained for processes with common characteristics.
- Each queue can have its own scheduling algorithms.
- Priorities are assigned to each queue.

For example, CPU-bound jobs can be scheduled in one queue and all I/O-bound jobs in another queue. The Process Scheduler then alternately selects jobs from each queue and assigns them to the CPU based on the algorithm assigned to the queue.

Every algorithm supports a different class of process but in a generalized system, some process wants to be scheduled using a priority algorithm. While some process wants to remain in the system (interactive process) while some are background process when execution can be delayed.

In general round-robin algorithm with different time quantum is used for such maintenance. The Number of ready queue algorithm between the queue, algorithm inside a queue algo but the queues may change from system to system. There is a various class of scheduling algorithm which is created for situations like in which the processes are easily divided into the different groups. There is a common division between the foreground (interactive) process and background (batch process). There are two types processes have the different response time, the different requirement of resources so these processes need different scheduling algorithms. The foreground processes have priority (externally defined) over background processes.

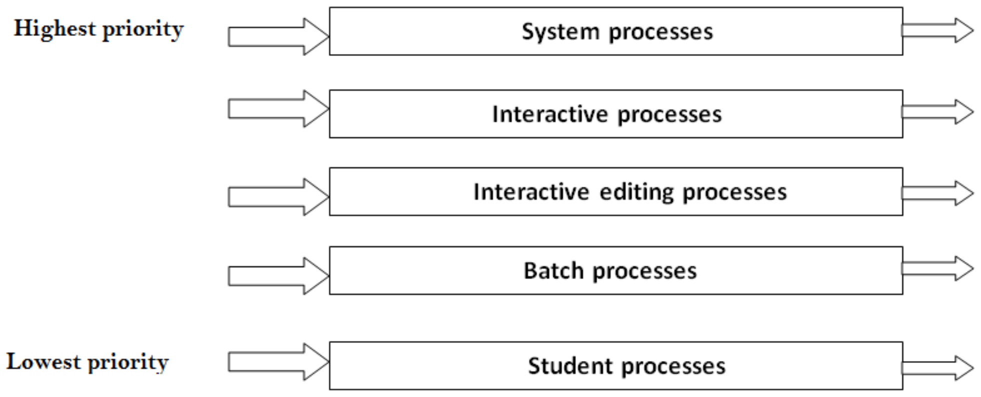
In the multilevel queue scheduling algorithm partition the ready queue has divided into seven separate queues. Based on some priority of the process; like memory size, process priority, or process type these processes are permanently assigned to one queue. Each queue has its own scheduling algorithm. For example, some queues are used for the foreground process and some for the background process.

The foreground queue can be scheduled by using a round-robin algorithm while the background queue is scheduled by a first come first serve algorithm.

It is said that there will be scheduled between the queues which are easily implemented as a fixed- priority preemptive scheduling. Let us take an example of a multilevel queue scheduling algorithm with five queues:

1. System process

2. Interactive processes
3. Interactive editing processes
4. Batch processes
5. Student processes



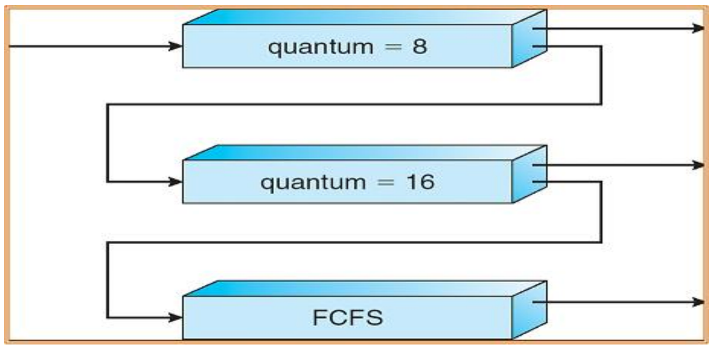
Multilevel feedback scheduling

Generally, we see in a **multilevel queue scheduling algorithm** processes are permanently stored in one queue in the system and do not move between the queue. There is some separate queue for foreground or background processes but the processes do not move from one queue to another queue and these processes do not change their foreground or background nature, these type of arrangement has the advantage of low scheduling but it is inflexible in nature.

Multilevel feedback queue scheduling it allows a process to move between the queues. The process are separate with different CPU burst time. If a process uses too much CPU time then it will be moved to the lowest priority queue. This idea leaves I/O bound and interactive processes in the higher priority queue. Similarly, the process which waits too long in a lower priority queue may be moved to a higher priority queue. This form of aging prevents starvation.

The **multilevel feedback queue scheduler** has the following parameters:

- The number of queues in the system.
- The scheduling algorithm for each queue in the system.
- The method used to determine when the process is upgraded to a higher-priority queue.
- The method used to determine when to demote a queue to a lower - priority queue.
- The method used to determine which process will enter in queue and when that process needs service.



(**Figure:** Multilevel feedback Queue Scheduling)

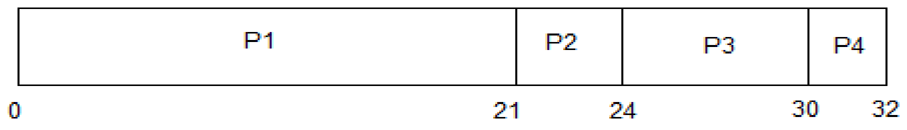
Multilevel Processor Queue scheduling

- If several identical processors are available then load sharing can occur.
- It would be possible to provide a separate queue for each processor. In this case one processor could be idle with empty queue, while another processor was very busy.
- To prevent such situation, we use one ready queue. All processes go into one queue and are scheduled onto any available processor.
- In one approach, each process is self scheduling. Each processor examines the ready queue and selects a process to execute. In this case we must ensure that two processor do not choose the same process and the processes are not lost from the queue.
- This problem can be avoided as appointing one processor as scheduler for the other processor creating as **Master-Slave Structure**.
- Master Processor will take all scheduling decisions; I/O processing and other system activities and other processor only execute user code.

First-Come, First-Served (FCFS) Scheduling

PROCESS	BURST TIME
P1	21
P2	3
P3	6
P4	2

The average waiting time will be = (0 + 21 + 24 + 30)/4 = 18.75 ms



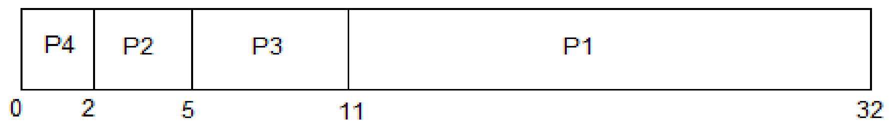
This is the GANTT chart for the above processes

Shortest Job First:

PROCESS	BURST TIME
P1	21
P2	3
P3	6
P4	2

In Shortest Job First Scheduling, the shortest Process is executed first.

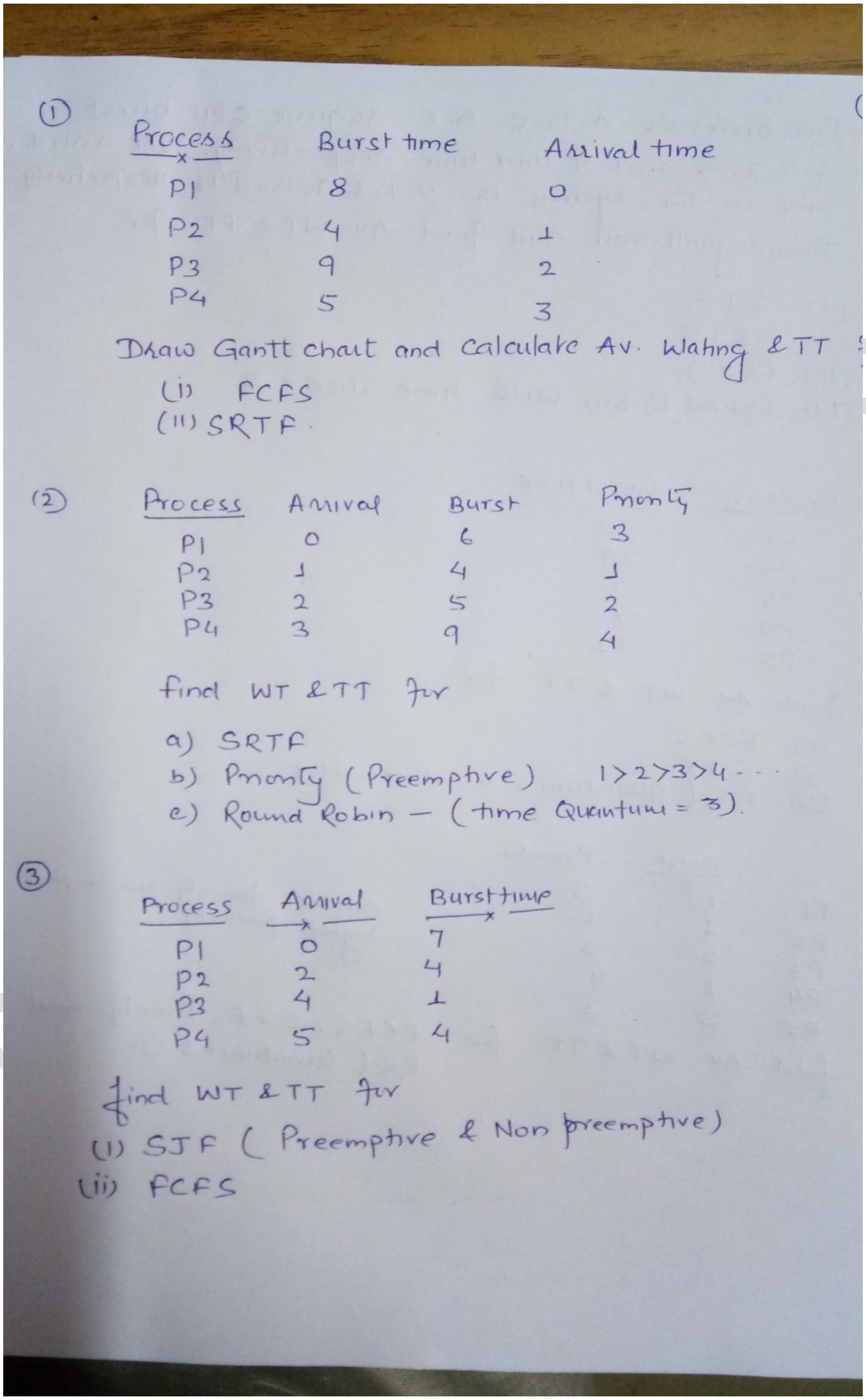
Hence the GANTT chart will be following :



Now, the average waiting time will be = (0 + 2 + 5 + 11)/4 = 4.5 ms

SRTF (Preemptive- SJF)

P1 (1 Sec) →P2 (3 Sec) →P4 (2 Sec) →P3 (6 Sec) →P1 (Remaining 20 Sec)



①

Process	Burst time	Arrival time
P1	8	0
P2	4	1
P3	9	2
P4	5	3

Draw Gantt chart and Calculate Av. Waiting & TT

(i) FCFS

(ii) SRTF.

(2)

Process	Arrival	Burst	Priority
P1	0	6	3
P2	1	4	1
P3	2	5	2
P4	3	9	4

find WT & TT for

a) SRTF

b) Priority (Preemptive) 1>2>3>4...

c) Round Robin – (time Quantum = 3).

③

Process	Arrival	Burst time
P1	0	7
P2	2	4
P3	4	1
P4	5	4

find WT & TT for

(i) SJF (Preemptive & Non preemptive)

(ii) FCFS

④ five processes A, B, C, D, E require CPU burst 3, 5, 2, 5 and 5 unit time respectively. The arriva time in the system is 0, 1, 3, 9, and 12 respectivel Draw gantt chart and find Av. WT & TT for

(i) SJF
(ii) SRTF
(iii) FCFS.
(IV) Round Robin with time Slice = 3

⑤

Process	Burst time
P1	10
P2	29
P3	3
P4	7
P5	12

find Av. WT & TT for

① FCFS
② SJF
③ RR (quantam = 10 milliseconds)

⑥

	Burst ↓	Priority
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

Smaller ~~priority~~ No implies highest priority.

find Av WT & TT for FCFS, SJF, Priority and RR (Quantum = 1).