Process Scheduling in an OS

In an operating system (OS), a process scheduler performs the important activity of scheduling a process between the ready queue and waiting queue and allocating them to the CPU. The OS assigns priority to each process and maintains these queues. The scheduler selects the process from the queue and loads it into memory for execution. There are two types of process scheduling: preemptive scheduling and non-preemptive scheduling.

Preemptive Scheduling

The scheduling in which a running process can be interrupted if a high priority process enters the queue and is allocated to the CPU is called **preemptive scheduling**. In this case, the current process switches from the running queue to ready queue, and the high priority process utilizes the CPU cycle.

Non-Preemptive Scheduling

The scheduling in which a running process cannot be interrupted by any other process is called **non-preemptive scheduling**. Any other process which enters the queue has to wait until the current process finishes its CPU cycle.

Dispatcher

Dispatcher module gives control of the CPU to the process selected by the short-term scheduler; this involves:

- switching context
- switching to user mode
- jumping to the proper location in the user program to restart that program

Dispatch latency – time it takes for the dispatcher to stop one process and start another running.

Scheduling Criteria

CPU utilization – keep the CPU as busy as possible. CPU utilization can range from 0 to 100 percent but it should range from 40 to 90 percent.

Throughput – # of processes that complete their execution per time unit

Turnaround time – amount of time to execute a particular process

Waiting time – amount of time a process has been waiting in the ready queue

Response time – amount of time it takes from when a request was submitted until the first response is produced, **not** output(for time-sharing environment).

Optimization Criteria

- Max CPU utilization
- Max throughput
- Min turnaround time
- Min waiting time
- Min response time

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

First Come First Serve (FCFS)

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

Example 1:

Process	Burst Time
P_1	24
P_2	3
P_3	3

Suppose that the processes arrive in the order: P_1 , P_2 , P_3

The Gantt Chart for the schedule is:

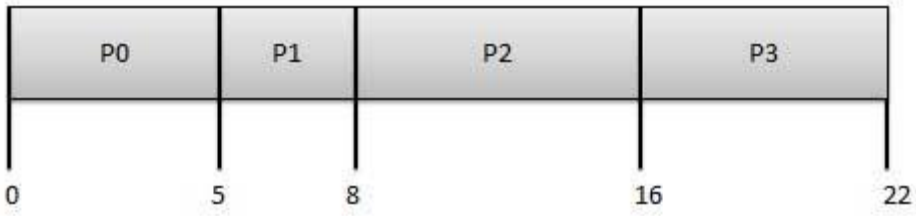
- Waiting time for P_1 = 0; P_2 = 24; P_3 = 27
- Average waiting time: (0 + 24 + 27)/3 = 17
- **Turnaround Time** of each process is as follows −

Process	Turnaround Time : Burst Time/Execute Time + Waiting Time
P1	0 +24 = 24
P2	24 + 3 = 27
P3	27 + 3 = 30

- Average Turnaround time: (24+27+30) / 3 = 27

Example 2:

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	Wait Time : Service Time - Arrival Time
P0	0 - 0 = 0
P1	5 - 1 = 4
P2	8 - 2 = 6
P3	16 - 3 = 13

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

- **Turnaround Time** of each process is as follows −

Process	Turnaround Time : Burst Time/Execute Time + Waiting Time
P0	5 + 0 = 5
P1	3 + 4 = 7
P2	8 + 6 = 14
P3	6 + 13 = 19

- Average Turnaround Time: (5+7+14+19) / 4 = 11.25