CPU Scheduling Criteria in OS

What is CPU scheduling?

CPU Scheduling is a process of determining which process will own CPU for execution while another process is in the waiting state.

CPU scheduling aims to make the system efficient, quick, and unbiased.

Types of CPU scheduling:

There are two types of scheduling methods:

1. **Preemptive scheduling**

In this scheduling, tasks are assigned with their priorities. Sometimes the lower priority task will be held till the high priority task finishes its execution and resume thereafter.

2. **Non-preemptive scheduling**

In this scheduling, the CPU is assigned to a specific process. This method is used for various hardware platforms because it doesn't need any additional hardware.

CPU Scheduling criteria:

CPU scheduling criteria is as follows:

- **CPU Utilization**

The main objective of CPU scheduling algorithm is to keep the CPU busy. CPU utilization varies from 49% to 90% depending on the load upon the system.

- **Throughput**

Throughput is the number of processes being executed and completed per unit time. Throughput varies depending on the length of each process.

- **Turnaround time**

The time elapsed from the time of submission of a process to the time of completion is known as the turnaround time. It is the total time spent by the system on waiting and executing.

- **Response time**

Response time is the time taken from the submission of a process request until the first response is produced.

Waiting time

Waiting time is the time spent by the process in the waiting queue.

CPU Scheduling algorithm

Types of scheduling algorithms are as follows:

1. First Come First Serve (FCFS)
2. Shortest-Job-First (SJF) Scheduling
3. Shortest Remaining Time
4. Priority Scheduling
5. Round Robin Scheduling
6. Multilevel Queue Scheduling

First Come First Serve

First Come First Serve or **FCFS** is the easiest and simplest CPU scheduling algorithm. When the process enters the ready queue, its Process Control Block or PCB is linked with the tail of the queue. Therefore, when the CPU becomes free, it is assigned to the process at the beginning of the queue.

In FCFS the process which requests the CPU gets the CPU allocation first.

Shortest Job first

Shortest Job First or **SJF** is a type of algorithm in which the process with the shortest execution time is selected for execution first. It can be preemptive or non-preemptive.

SJF significantly decreases the execution time for other processes that are waiting.

Shortest Remaining time

Shortest Remaining Time or **SRT** is an algorithm where the process that is closest to its completion is allocated first. SRT prevents a newer ready process from holding the completion of an older process.

Priority Scheduling

Priority Scheduling is the method where processes are scheduled based on preference. It helps the OS involve higher priority assignments. Processes with higher priority are executed first, and processes with equal priority are executed based on FCFS or round-robin algorithm.

Round-robin Scheduling

Round-robin Scheduling follows the principle of the round-robin method.

It is a hybrid model which is clock-driven and responds to an event within a specific time limit.

Multilevel Queue Scheduling

Multilevel Queue Scheduling separates the ready queue into various other queues based on specific properties of the processes, like the process priority, size of the memory, etc.

It is not an independent scheduling algorithm and needs other algorithms to schedule the jobs.

First Come First Serve Scheduling

In the "First come first serve" scheduling algorithm, as the name suggests, the process which arrives first, gets executed first, or we can say that the process which requests the CPU first, gets the CPU allocated first.

- First Come First Serve, is just like **FIFO**(First in First out) Queue data structure, where the data element which is added to the queue first, is the one who leaves the queue first.
- This is used in Batch Systems.
- It's **easy to understand and implement** programmatically, using a Queue data structure, where a new process enters through the **tail** of the queue, and the scheduler selects process from the **head** of the queue.
- A perfect real life example of FCFS scheduling is **buying tickets at ticket counter**.

Calculating Average Waiting Time

For every scheduling algorithm, **Average waiting time** is a crucial parameter to judge it's performance.

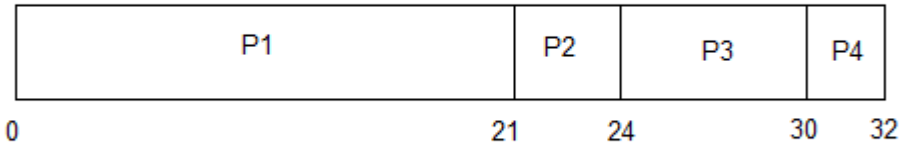
AWT or Average waiting time is the average of the waiting times of the processes in the queue, waiting for the scheduler to pick them for execution.

Lower the Average Waiting Time, better the scheduling algorithm.

Consider the processes P1, P2, P3, P4 given in the below table, arrives for execution in the same order, with **Arrival Time** 0, and given **Burst Time**, let's find the average waiting time using the FCFS scheduling algorithm.

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

The average waiting time will be 18.75 ms

For the above given proccesses, first **P1** will be provided with the CPU resources,

- Hence, waiting time for **P1** will be 0
- **P1** requires 21 ms for completion, hence waiting time for **P2** will be 21 ms
- Similarly, waiting time for process **P3** will be execution time of **P1** + execution time for **P2**, which will be (21 + 3) ms = 24 ms.
- For process **P4** it will be the sum of execution times of **P1**, **P2** and **P3**.

The **GANTT chart** above perfectly represents the waiting time for each process.

Problems with FCFS Scheduling

Below we have a few shortcomings or problems with the FCFS scheduling algorithm:

1. It is **Non Pre-emptive** algorithm, which means the **process priority** doesn't matter.

If a process with very least priority is being executed, more like **daily routine backup** process, which takes more time, and all of a sudden some other high priority process arrives, like **interrupt to avoid system crash**, the high priority process will have to wait, and hence in this case, the system will crash, just because of improper process scheduling.

2. Not optimal Average Waiting Time.
3. Resources utilization in parallel is not possible, which leads to **Convoy Effect**, and hence poor resource(CPU, I/O etc) utilization.

What is Convoy Effect?

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

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