Unit 2: CPU Scheduling

What is Process Scheduling?
Process Scheduling is the process of the process manager handling the removal of an active process from the CPU and selecting another process based on a specific strategy.

Process Scheduling is an integral part of Multi-programming applications. Such operating systems allow more than one process to be loaded into usable memory at a time and the loaded shared CPU process uses repetition time.

There are three types of: process schedulers
Long term or Job Scheduler
- Short term or CPU Scheduler
- Medium-term Scheduler
-

Why do we need to schedule processes?
- **Scheduling** is important in many different computer environments. One of the most important areas is scheduling which programs will work on the CPU. This task is handled by the Operating System (OS) of the computer and there are many different ways in which we can choose to configure programs.
- **Process Scheduling** allows the OS to allocate CPU time for each process. Another important reason to use a process scheduling system is that it keeps the CPU busy at all times. This allows you to get less response time for programs.
- Considering that there may be hundreds of programs that need to work, the OS must launch the program, stop it, switch to another program, etc. The way the OS configures the system to run another in the CPU is called "context switching". If the OS keeps context-switching programs in and out of the provided CPUs, it can give the user a tricky idea that he or she can run any programs he or she wants to run, all at once.
- So now that we know we can run 1 program at a given CPU, and we know we can change the operating system and remove another one using the context switch, how do we choose which programs we need. run, and with what program?
- That's where **scheduling** comes in! First, you determine the metrics, saying something like "the amount of time until the end". We will define this metric as "the time interval between which a function enters the system until it is completed". Second, you decide on a metrics that reduces metrics. We want our tasks to end as soon as possible.

What is the need for CPU scheduling algorithm?
CPU scheduling is the process of deciding which process will own the CPU to use while another process is suspended. The main function of the CPU scheduling is to ensure that whenever the CPU remains idle, the OS

has at least selected one of the processes available in the ready-to-use line.
In Multiprogramming, if the long-term scheduler selects multiple I / O binding processes then most of the time, the CPU remains an idle. The function of an effective program is to improve resource utilization.
If most operating systems change their status from performance to waiting then there may always be a chance of failure in the system. So in order to minimize this excess, the OS needs to schedule tasks in order to make full use of the CPU and avoid the possibility of deadlock.

Objectives of Process Scheduling Algorithm:

- Utilization of CPU at maximum level. **Keep CPU as busy as possible**.
- **Allocation of CPU should be fair**.
- **Throughput should be Maximum**. i.e. Number of processes that complete their execution per time unit should be maximized.
- **Minimum turnaround time**, i.e. time taken by a process to finish execution should be the least.
- There should be a **minimum waiting time** and the process should not starve in the ready queue.
- **Minimum response time.** It means that the time when a process produces the first response should be as less as possible.

What are the different terminologies to take care of in any CPU Scheduling algorithm?

- **Arrival Time:** Time at which the process arrives in the ready queue.
- **Completion Time:** Time at which process completes its execution.
- **Burst Time:** Time required by a process for CPU execution.
- **Turn Around Time:** Time Difference between completion time and arrival time.

Turn Around Time = Completion Time – Arrival Time

- **Waiting Time(W.T):** Time Difference between turn around time and burst time.

Waiting Time = Turn Around Time – Burst Time

Things to take care while designing a CPU Scheduling algorithm?
Different **CPU Scheduling algorithms** have different structures and the choice of a particular algorithm depends on a variety of factors. Many conditions have been raised to compare CPU scheduling algorithms.
The criteria include the following:

- **CPU utilization:** The main purpose of any CPU algorithm is to keep the CPU as busy as possible. Theoretically, CPU usage can range from 0 to 100 but in a real-time system, it varies from 40 to 90 percent depending on the system load.
- **Throughput:** The average CPU performance is the number of processes performed and completed during each unit. This is called throughput. The output may vary depending on the length or duration of the processes.
- **Turn round Time:** For a particular process, the important conditions are how long it takes to perform that process. The time elapsed from the time of process delivery to the time of completion is known as the

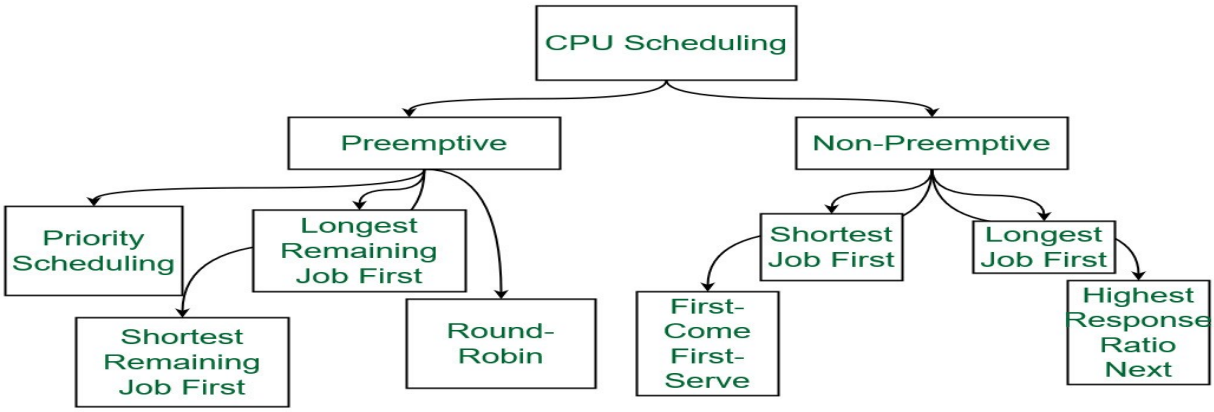
conversion time. Conversion time is the amount of time spent waiting for memory access, waiting in line, using CPU, and waiting for I / O.

- **Waiting Time:** The Scheduling algorithm does not affect the time required to complete the process once it has started performing. It only affects the waiting time of the process i.e. the time spent in the waiting process in the ready queue.
- **Response Time:** In a collaborative system, turn around time is not the best option. The process may produce something early and continue to computing the new results while the previous results are released to the user. Therefore another method is the time taken in the submission of the application process until the first response is issued. This measure is called response time.

What are the different types of CPU Scheduling Algorithms?

There are mainly two types of scheduling methods:

- Preemptive Scheduling: Preemptive scheduling is used when a process switches from running state to ready state or from the waiting state to the ready state.
- Non-Preemptive Scheduling: Non-Preemptive scheduling is used when a process terminates , or when a process switches from running state to waiting state.
-



Different types of CPU Scheduling Algorithms

Let us now learn about these CPU scheduling algorithms in operating systems one by one:

1. First Come First Serve:

FCFS considered to be the simplest of all operating system scheduling algorithms. First come first serve scheduling algorithm states that the process that requests the CPU first is allocated the CPU first and is implemented by using .**FIFO queue**

Characteristics of FCFS:

- FCFS supports non-preemptive and preemptive CPU scheduling algorithms.
- Tasks are always executed on a First-come, First-serve concept.
- FCFS is easy to implement and use.

- This algorithm is not much efficient in performance, and the wait time is quite high.

Advantages of FCFS:

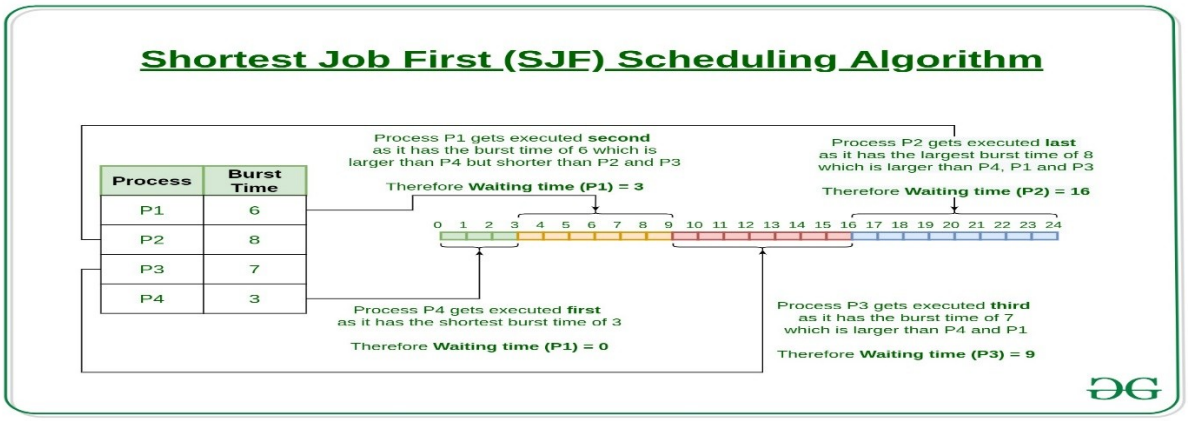
- Easy to implement
- First come, first serve method

Disadvantages of FCFS:

- FCFS suffers from **Convoy effect**.
- The average waiting time is much higher than the other algorithms.
- FCFS is very simple and easy to implement and hence not much efficient.

2. Shortest Job First(SJF):

Shortest job first (SJF) is a scheduling process that selects the waiting process with the smallest execution time to execute next. This scheduling method may or may not be preemptive. Significantly reduces the average waiting time for other processes waiting to be executed. The full form of SJF is Shortest Job First.



Characteristics of SJF:

- Shortest Job first has the advantage of having a minimum average waiting time among all operating system scheduling algorithms.
- It is associated with each task as a unit of time to complete.
- It may cause starvation if shorter processes keep coming. This problem can be solved using the concept of ageing.

Advantages of Shortest Job first:

- As SJF reduces the average waiting time thus, it is better than the first come first serve scheduling algorithm.
- SJF is generally used for long term scheduling

Disadvantages of SJF:

- One of the demerit SJF has is starvation.
- Many times it becomes complicated to predict the length of the upcoming CPU request

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on Shortest Job First.
e refer to our detailed article on the Longest job first scheduling.

3. Priority Scheduling:

Preemptive Priority CPU Scheduling Algorithm is a pre-emptive method of CPU scheduling algorithm that works **based on the priority** of a process. In this algorithm, the editor sets the functions to be as important,

meaning that the most important process must be done first. In the case of any conflict, that is, where there are more than one processor with equal value, then the most important CPU planning algorithm works on the basis of the FCFS (First Come First Serve) algorithm.

Characteristics of Priority Scheduling:

- Schedules tasks based on priority.
- When the higher priority work arrives while a task with less priority is executed, the higher priority work takes the place of the less priority one and
- The latter is suspended until the execution is complete.
- Lower is the number assigned, higher is the priority level of a process.

Advantages of Priority Scheduling:

- The average waiting time is less than FCFS
- Less complex

Disadvantages of Priority Scheduling:

- One of the most common demerits of the Preemptive priority CPU scheduling algorithm is the Starvation Problem. This is the problem in which a process has to wait for a longer amount of time to get scheduled into the CPU. This condition is called the starvation problem.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on Priority Preemptive Scheduling algorithm.

4. Round robin:

Round Robin is a CPU scheduling algorithm where each process is cyclically assigned a fixed time slot. It is the preemptive version of First come First Serve CPU Scheduling algorithm. Round Robin CPU Algorithm generally focuses on Time Sharing technique.

Characteristics of Round robin:

- It's simple, easy to use, and starvation-free as all processes get the balanced CPU allocation.
- One of the most widely used methods in CPU scheduling as a core.
- It is considered preemptive as the processes are given to the CPU for a very limited time.

Advantages of Round robin:

- Round robin seems to be fair as every process gets an equal share of CPU.
- The newly created process is added to the end of the ready queue.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on theRound robin Scheduling algorithm .

5. Shortest Remaining Time First:

Shortest remaining time first is the preemptive version of the Shortest job first which we have discussed earlier where the processor is allocated to the job closest to completion. In SRTF the process with the smallest amount of time remaining until completion is selected to execute.

Characteristics of Shortest remaining time first:

- SRTF algorithm makes the processing of the jobs faster than SJF algorithm, given it's overhead charges are not counted.

- The context switch is done a lot more times in SRTF than in SJF and consumes the CPU's valuable time for processing. This adds up to its processing time and diminishes its advantage of fast processing.

Advantages of SRTF:

- In SRTF the short processes are handled very fast.
- The system also requires very little overhead since it only makes a decision when a process completes or a new process is added.

Disadvantages of SRTF:

- Like the shortest job first, it also has the potential for process starvation.
- Long processes may be held off indefinitely if short processes are continually added.

To learn about how to implement this CPU scheduling algorithm, please refer to our detailed article on the shortest remaining time first.

Exercise:

1. Which of the following is false about SJF? S1: It causes minimum average waiting time S2: It can cause starvation (A) Only S1 (B) Only S2 (C) Both S1 and S2 (D) Neither S1 nor S2 Answer (D) S1 is true SJF will always give minimum average waiting time. S2 is true SJF can cause starvation.
2. Consider the following table of arrival time and burst time for three processes P0, P1 and P2.

Process	Arrival time	Burst Time
P0	0 ms	9 ms
P1	1 ms	4 ms
P2	2 ms	9 ms

1. The pre-emptive shortest job first scheduling algorithm is used. Scheduling is carried out only at arrival or completion of processes. What is the average waiting time for the three processes? (A) 5.0 ms (B) 4.33 ms (C) 6.33 (D) 7.33 Solution : Answer: – (A) Process P0 is allocated processor at 0 ms as there is no other process in the ready queue. P0 is preempted after 1 ms as P1 arrives at 1 ms and burst time for P1 is less than remaining time of P0. P1 runs for 4ms. P2 arrived at 2 ms but P1 continued as burst time of P2 is longer than P1. After P1 completes, P0 is scheduled again as the remaining time for P0 is less than the burst time of P2. P0 waits for 4 ms, P1 waits for 0 ms and P2 waits for 11 ms. So average waiting time is (0+4+11)/3 = 5.

2. Consider the following set of processes, with the arrival times and the CPU-burst times given in milliseconds

Process	Arrival time	Burst Time
P1	0 ms	5 ms
P2	1 ms	3 ms
P3	2 ms	3 ms
P4	4 ms	1 ms

1. What is the average turnaround time for these processes with the preemptive shortest remaining processing time first (SRPT) algorithm ? (A) 5.50 (B) 5.75 (C) 6.00 (D) 6.25 Answer (A) Solution: The following is Gantt Chart of execution

P1	P2	P4	P3	P1
1	4	5	8	12

1. Turn Around Time = Completion Time – Arrival Time Avg Turn Around Time = (12 + 3 + 6+ 1)/4 = 5.50
2. An operating system uses the Shortest Remaining Time First (SRTF) process scheduling algorithm. Consider the arrival times and execution times for the following processes:

Process	Arrival time	Burst Time
P1	20 ms	0 ms
P2	25 ms	15 ms
P3	10 ms	30 ms
P4	15 ms	45 ms

1. What is the total waiting time for process P2? (A) 5 (B) 15 (C) 40 (D) 55 Answer (B) At time 0, P1 is the only process, P1 runs for 15 time units. At time 15, P2 arrives, but P1 has the shortest remaining time. So P1 continues for 5 more time units. At time 20, P2 is the only process. So it runs for 10 time units At time 30, P3 is the shortest remaining time

process. So it runs for 10 time units At time 40, P2 runs as it is the only process. P2 runs for 5 time units. At time 45, P3 arrives, but P2 has the shortest remaining time. So P2 continues for 10 more time units. P2 completes its execution at time 55

Total waiting time for P2
= Completion time – (Arrival time + Execution time)
= 55 – (15 + 25)
= 15