Types of Scheduling Queues

We will look at the numerous types of Scheduling Queues used in computer systems in the following sub-topics.

Job Queue (In Disk)

This queue is known as the job queue, it contains all the processes or jobs in the list that are waiting to be processed. Job: When a job is created, it goes into the job queue and waits until it is ready for processing.

- Contains all submitted jobs.
- Processes are stored here in a wait state until they are ready to go to the execution stage.
- This is the first and most basic state that acts as a default storage of new jobs added to a scheduling system.
- Long Term Scheduler Picks a process from Job Queue and moves to ready queue.

Ready Queue (In Main Memory)

The Stand-by queue contains all the processes ready to be fetched from the memory, for execution. When the process is initiated, it joins the ready queue to wait for the CPU to be free. The operating system assigns a process to the executing processor from this queue based on the scheduling algorithm it implements.

- Contains processes (mainly their PCBs) waiting for the CPU to execute various processes it contains.
- They are controlled using a scheduling algorithm like FCFS, SJF, or Priority Scheduling.
- Short Term Scheduler picks a process from Ready Queue and moves the selected process to running state.

Block or Device Queues (In Main Memory)

The processes which are blocked due to unavailability of an I/O device are added to this queue. Every device has its own block queue.

Flow of Movement in the above Queues

The below diagram shows movements of processes in different queues.

- All processes are initially in the Job Queue.
- A new process is initially put in the Ready queue by scheduler. It waits in the ready queue until it is selected for execution(or dispatched). Once the process is assigned to the CPU and is executing, one of the following several events can occur:
  1) The process could issue an I/O request, and then be placed in a Device queue.
  2) The process could create a new subprocess and wait for its termination.
  3) The process could be removed forcibly from the CPU, as a result of an interrupt, and be put back in the ready queue.

Introduction of Process Management

Process Management for a single tasking or batch processing system is easy as only one process is active at a time. With multiple processes (multiprogramming or multitasking) being active, the process management becomes complex as a CPU needs to be efficiently utilized by multiple processes. Multiple active processes can may share resources like memory and may communicate with each other. This further makes things complex as an Operating System has to do process synchronization.

Please remember the main advantages of having multiprogramming are system responsiveness and better CPU utilization. We can run multiple processes in interleaved manner on a single CPU. For example, when the current process is getting busy with IO, we assign CPU to some other process.

CPU-Bound vs I/O-Bound Processes

A CPU-bound process requires more CPU time or spends more time in the running state. An I/O-bound process requires more I/O time and less CPU time. An I/O-bound process spends more time in the waiting state.

Process planning is an integral part of the process management operating system. It refers to the mechanism used by the operating system to determine which process to run next. The goal of process scheduling is to improve overall system performance by maximizing CPU utilization, minimizing throughput time, and improving system response time.

Process Management Tasks

Process management is a key part in operating systems with multi-programming or multitasking.

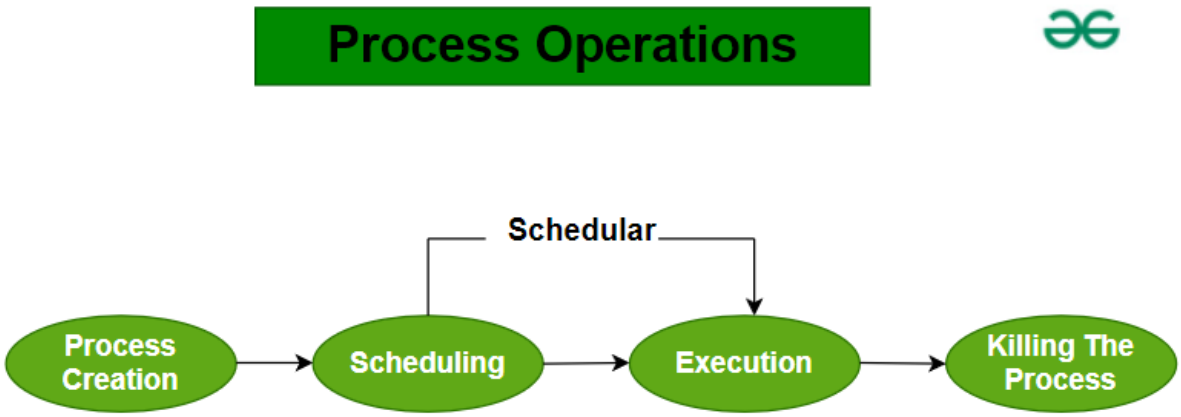
- Process Creation and Termination : Process creation involves creating a Process ID, setting up Process Control Block, etc. A process can be terminated either by the

operating system or by the parent process. Process termination involves clearing all resources allocated to it.

- CPU Scheduling : In a multiprogramming system, multiple processes need to get the CPU. It is the job of Operating System to ensure smooth and efficient execution of multiple processes.
- Deadlock Handling : Making sure that system does not reach a state where two or processes cannot proceed due to a cycling dependency on each other.
- Inter-Process Communication : Operating System provides facilities such as shared memory and message passing for cooperating processes to communicate.
- Process Synchronization : Process Synchronization is the coordination of execution of multiple processes in a multiprogramming system to ensure that they access shared resources (like memory) in a controlled and predictable manner.

Process Operations

Please remember a process goes through different states before termination and these state changes require different operations on processes by an operating system. These operations include process creation, process scheduling, execution and killing the process. Here are the key process operations:



Process Operations

Process Creation

Process creation in an operating system (OS) is the act of generating a new process. This new process is an instance of a program that can execute independently.

Scheduling

Once a process is ready to run, it enters the “ready queue.” The scheduler’s job is to pick a process from this queue and start its execution.

Execution

Execution means the CPU starts working on the process. During this time, the process might:

- Move to a waiting queue if it needs to perform an I/O operation.
- Get blocked if a higher-priority process needs the CPU.

Killing the Process

After the process finishes its tasks, the operating system ends it and removes its Process Control Block (PCB).

Context Switching of Process

The process of saving the context of one process and loading the context of another process is known as Context Switching. In simple terms, it is like loading and unloading the process from the running state to the ready state.

When Does Context Switching Happen?

Context Switching Happen:

- When a high-priority process comes to a ready state (i.e. with higher priority than the running process).
- An Interrupt occurs.
- User and kernel-mode switch (It is not necessary though)
- Preemptive CPU scheduling is used.

Context Switch vs Mode Switch

A mode switch occurs when the CPU privilege level is changed, for example when a system call is made or a fault occurs. The kernel works in more a privileged mode than a standard user task. If a user process wants to access things that are only accessible to the kernel, a mode switch must occur. The currently executing process need not be changed during a mode switch. A mode switch typically occurs for a process context switch to occur. Only the kernel can cause a context switch.

Process Scheduling Algorithms

The operating system can use different scheduling algorithms to schedule processes. Here are some commonly used timing algorithms:

- **First-Come, First-Served (FCFS):** This is the simplest scheduling algorithm, where the process is executed on a first-come, first-served basis. FCFS is non-preemptive, which means that once a process starts executing, it continues until it is finished or waiting for I/O.
- **Shortest Job First (SJF):** SJF is a proactive scheduling algorithm that selects the process with the shortest burst time. The burst time is the time a process takes to complete its execution. SJF minimizes the average waiting time of processes.
- **Round Robin (RR):** Round Robin is a proactive scheduling algorithm that reserves a fixed amount of time in a round for each process. If a process does not complete its execution within the specified time, it is blocked and added to the end of the queue. RR ensures fair distribution of CPU time to all processes and avoids starvation.
- **Priority Scheduling:** This scheduling algorithm assigns priority to each process and the process with the highest priority is executed first. Priority can be set based on process type, importance, or resource requirements.
- **Multilevel Queue:** This scheduling algorithm divides the ready queue into several separate queues, each queue having a different priority. Processes are queued based on their priority, and each queue uses its own scheduling algorithm. This scheduling algorithm is useful in scenarios where different types of processes have different priorities.

CPU Scheduling Criteria

CPU scheduling is essential for the system's performance and ensures that processes are executed correctly and on time. Different CPU scheduling algorithms have other properties and the choice of a particular algorithm depends on various factors. Many criteria have been suggested for comparing CPU scheduling algorithms.

What is CPU scheduling?

CPU Scheduling is a process that allows one process to use the CPU while another process is delayed due to unavailability of any resources such as I / O etc, thus making full use of the CPU. In short, CPU scheduling decides the order and priority of the processes to run and allocates the CPU time based on various parameters such as CPU usage, throughput, turnaround, waiting

time, and response time. The purpose of CPU Scheduling is to make the system more efficient, faster, and fairer.

Criteria of CPU Scheduling

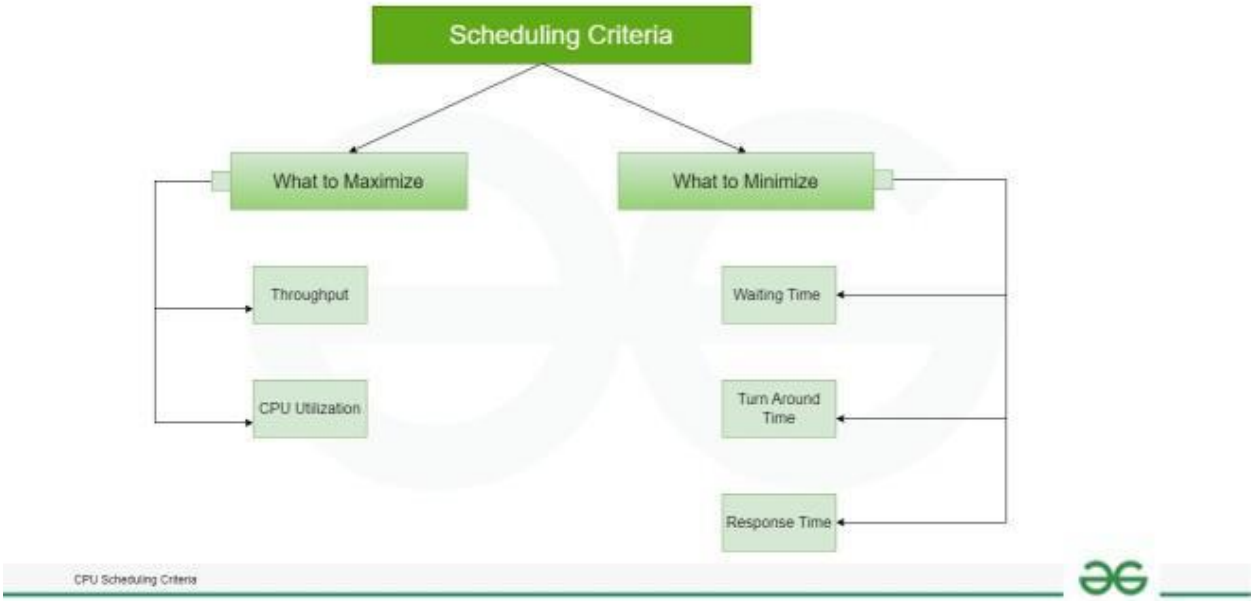
Now let's discuss CPU Scheduling has several criteria. Some of them are mentioned below.

1. CPU utilization

The main objective of any CPU scheduling algorithm is to keep the CPU as busy as possible. Theoretically, CPU utilization can range from 0 to 100 but in a real-time system, it varies from 40 to 90 percent depending on the load upon the system.

2. Throughput

A measure of the work done by the CPU is the number of processes being executed and completed per unit of time. This is called throughput. The throughput may vary depending on the length or duration of the processes.



CPU Scheduling Criteria

3. Turnaround Time

For a particular process, an important criterion is how long it takes to execute that process. The time elapsed from the time of submission of a process to the time of completion is known as the turnaround time. Turn-around time is the sum of times spent waiting to get into memory, waiting in the ready queue, executing in CPU, and waiting for I/O.

Turn Around Time = Completion Time – Arrival Time.

4. Waiting Time

A scheduling algorithm does not affect the time required to complete the process once it starts execution. It only affects the waiting time of a process i.e. time spent by a process waiting in the ready queue.

Waiting Time = Turnaround Time – Burst Time.

5. Response Time

In an interactive system, turn-around time is not the best criterion. A process may produce some output fairly early and continue computing new results while previous results are being output to the user. Thus another criterion is the time taken from submission of the process of the request until the first response is produced. This measure is called response time.

Response Time = CPU Allocation Time(when the CPU was allocated for the first) – Arrival Time

6. Completion Time

The completion time is the time when the process stops executing, which means that the process has completed its burst time and is completely executed.

7. Priority

If the operating system assigns priorities to processes, the scheduling mechanism should favor the higher-priority processes.

8. Predictability

A given process always should run in about the same amount of time under a similar system load.

Importance of Selecting the Right CPU Scheduling Algorithm for Specific Situations

It is important to choose the correct CPU scheduling algorithm because different algorithms have different priorities for different CPU scheduling criteria.Different algorithms have different strengths and weaknesses. Choosing the wrong CPU scheduling algorithm in a given situation can result in suboptimal performance of the system.

Example: Here are some examples of CPU scheduling algorithms that work well in different situations.

Round Robin scheduling algorithm works well in a time-sharing system where tasks have to be completed in a short period of time. SJF scheduling algorithm works best in a batch processing system where shorter jobs have to be completed first in order to increase throughput.Priority

scheduling algorithm works better in a real-time system where certain tasks have to be prioritized so that they can be completed in a timely manner.

Factors Influencing CPU Scheduling Algorithms

There are many factors that influence the choice of CPU scheduling algorithm. Some of them are listed below.

- The number of processes.
- The processing time required.
- The urgency of tasks.
- The system requirements.

Selecting the correct algorithm will ensure that the system will use system resources efficiently, increase productivity, and improve user satisfaction.

CPU Scheduling Algorithms

There are several CPU Scheduling Algorithms, that are listed below.

- First Come First Served (FCFS)
- Shortest Job First (SJF)
- Longest Job First (LJF)
- Priority Scheduling
- Round Robin (RR)
- Shortest Remaining Time First (SRTF)
- Longest Remaining Time First (LRTF)

Comparison of Different CPU Scheduling Algorithms in OS

A scheduling algorithm is used to estimate the CPU time required to allocate to the processes and threads. The prime goal of any CPU scheduling algorithm is to keep the CPU as busy as possible for improving CPU utilization.

Scheduling Algorithms

1. First Come First Serve(FCFS): As the name implies that the jobs are executed on a first come first serve basis. It is a simple algorithm based on FIFO that's first in first out. The process that

comes first in the ready queue can access the CPU first. If the arrival time is less, then the process will get the CPU soon. It can suffer from the convoy effect if the burst time of the first process is the longest among all the jobs.

2. **Shortest Job First (SJF)**: Also known as the shortest job first or shortest job next is a non-preemptive type algorithm that is easy to implement in batch systems and is best in minimizing the waiting time. It follows the strategies where the process that is having the lowest execution time is chosen for the next execution.

3. **Longest Job First Scheduling(LJF)**: The longest job first (LJF) is the non-preemptive version. This algorithm is also based upon the burst time of the processes. The processes are put into the ready queue according to their burst times and the process with the largest burst time is processed first.

4. **Longest Remaining Time First Scheduling (LRTF)**: The preemptive version of LJF is LRTF. Here the process with the maximum remaining CPU time will be considered first and then processed. And after some time interval, it will check if another process having more Burst Time arrived up to that time or not. If any other process has more remaining burst time, so the running process will get pre-empted by that process.

5. **Shortest Remaining Time First(SRTF)**: This algorithm is based on SJF and this is the preemptive version of SJF. In this scheduling algorithm, the process with the smallest remaining burst time is executed first and it may be preempted with a new job that has arrived with a shorter execution time.

6. **Round Robin(RR)**: It is a preemptive scheduling algorithm in which each process is given a fixed time called quantum to execute. At this time one process is allowed to execute for a quantum and then preempts and then another process is executed. In this way, there is context switching between processes to save states of these preempted processes.

7. **Priority Scheduling**: It is a non-preemptive algorithm that works in batch systems and, each process is assigned with a priority and the process with the highest priority is executed first. This can lead to starvation for other processes.

8. **Multiple Levels Queues Scheduling**: In this scheduling, multiple queues have their scheduling Algorithms and are maintained with the processes that possess the same characteristics. For this, priorities are assigned to each queue for the jobs to be executed.

9. **Multilevel-Feedback-Queue Scheduler**: It defines several queues and the scheduling algorithms for each queue. This algorithm is used to determine when to upgrade a process when to demote a process, and also determine the queue in which a process will enter and when that process needs service.

Note: *The SJF scheduling algorithm is hypothetical and un-implementable, as it is impossible to determine the burst time of any process without running it. SJF is a benchmarking algorithm as it provides minimum waiting time than any other scheduling algorithm*