Difference between scheduling algorithms and when to use each of them

What is Scheduling algorithms?

Scheduling algorithms are an integral part of operating systems and are responsible for determining the order in which processes or threads are executed on a computer's CPU. Different scheduling algorithms have varying characteristics and are suitable for different scenarios based on the system's requirements.

Some of different scheduling algorithms and when to use:

____

- **First-Come, First-Served (FCFS):**

Processes are executed in the order they arrive.

Suitable for scenarios where all processes have similar execution time, and there is no need to prioritize certain processes over others.

Not suitable when short processes are delayed by long ones (convoy effect).

- **Shortest Job Next (SJN) or Shortest Job First (SJF):**

The process with the shortest execution time is executed next.

Suitable when process execution times are known in advance or can be predicted accurately.

Minimizes average waiting time, but may lead to starvation for longer processes.

- **Round Robin (RR):**

Processes are executed in a cyclic order, with each process given a time slice (quantum) before switching to the next process.

Suitable for time-sharing systems where fairness and responsiveness are important.

Can lead to overhead due to frequent context switches.

- **Priority Scheduling:**

Each process is assigned a priority, and the process with the highest priority is executed next.

Suitable when processes have varying levels of importance or urgency.

Can lead to starvation for low-priority processes if not properly managed.

- **Multilevel Queue Scheduling:**

Processes are divided into multiple queues, each with its own priority level, and scheduling is done between queues.

Suitable for systems with a mix of interactive and batch processes, allowing for different scheduling policies for different queues.

- **Multilevel Feedback Queue Scheduling:**

Similar to multilevel queues, but processes can move between queues based on their behavior.

Suitable for dynamic environments where process behavior changes over time.

Provides a balance between responsiveness and fairness.