# Lecture 6

**–Priority Scheduling Algorithm**

Priority scheduling is one of the most common scheduling algorithms to schedule tasks based on their priority. Each task is assigned a priority value based on specific criteria.
The task with the highest priority is selected for execution first. If there are multiple tasks sharing the same priority, they are scheduled in the order they arrived, following a First-Come, First-Served approach.

Priority Scheduling can be implemented in two ways:

- Non-Preemptive Priority Scheduling
- Preemptive Priority Scheduling

Non-Preemptive Priority Scheduling Algorithm

In Non-Preemptive Priority Scheduling, the machine is not taken away from the running task. Even if a higher-priority task arrives, the currently running task will complete first; that means a high-priority task must wait until the currently running task finishes.

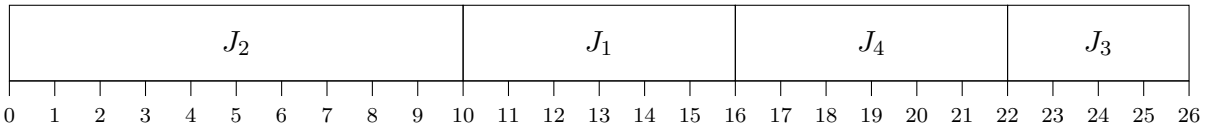
**Examples of Non-Preemptive Priority Scheduling**

Example 1:

Suppose that we have a set of four jobs that have arrived at the same time in the order $J_1$, $J_2$, $J_3$ and $J_4$. The processing time in milliseconds and the priorities of the tasks is given by the following table

| $J_i$ | $p_i$ | Priority |
|---|---|---|
| $J_1$ | 6 | 2 |
| $J_2$ | 10 | 1 |
| $J_3$ | 4 | 3 |
| $J_4$ | 6 | 2 |

Considering that a lower priority value means higher priority, let us perform non-preemptive priority scheduling on it. We will draw the Gantt chart and find the average of the waiting time, completion time and Flowtime.



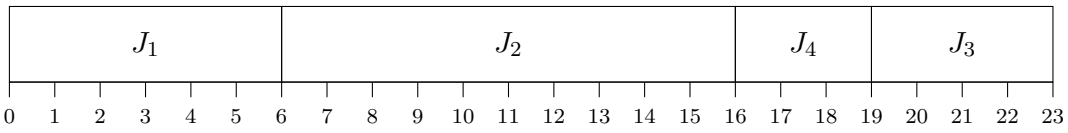
| $J_i$ | $C_i$ | $W_i$ | $F_i$ |
|---|---|---|---|
| $J_1$ | 16 | 10 | 16 |
| $J_2$ | 10 | 0 | 10 |
| $J_3$ | 26 | 22 | 26 |
| $J_4$ | 22 | 16 | 22 |
| Average | 18.5 | 12 | 18.5 |

Example 2:
In this example, we consider a situation when the tasks arrive at different times. Suppose we have set of four tasks whose arrival times, processing times and priorities are as follows

| $J_i$ | $r_i$ | $p_i$ | Priority |
|---|---|---|---|
| $J_1$ | 0 | 6 | 2 |
| $J_2$ | 4 | 10 | 1 |
| $J_3$ | 4 | 4 | 2 |
| $J_4$ | 8 | 3 | 1 |

Let us draw the Gantt chart and find the average of the waiting time, completion time and Flowtime using non-preemptive priority scheduling, considering lower priority value as higher priority.

At time 0ms, only $J_1$ is there, and so it is assigned to the machine. $J_1$ completes execution at 6ms, and at that time, $J_2$ and $J_3$ have arrived. $J_2$ has higher priority and hence is assigned to the machine. $J_2$ completes execution at time 10ms. By that time, $J_4$ has arrived, having priority value 1, and so it is assigned to the machine. Once $J_4$ completes execution, $J_3$ is assigned to the machine. So the order of execution is $J_1$, $J_2$, $J_4$, $J_3$ as shown in the following Gantt chart.



| $J_i$ | $C_i$ | $W_i$ | $F_i$ |
|---|---|---|---|
| $J_1$ | 6 | 0 | 6 |
| $J_2$ | 16 | 2 | 12 |
| $J_3$ | 23 | 15 | 19 |
| $J_4$ | 19 | 8 | 11 |
| Average | 16 | 6.25 | 12 |

Preemptive Priority Scheduling

In Preemptive Priority Scheduling, the machine can be taken away from the currently running task if a new task with a higher priority arrives. For example, a low-priority task is running, and a high-priority task arrives; the machine immediately switches to the high-priority task.

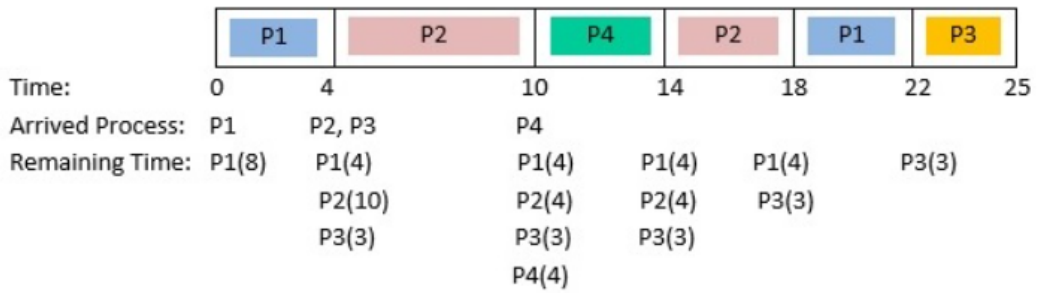
**Example of Preemptive Priority Scheduling**

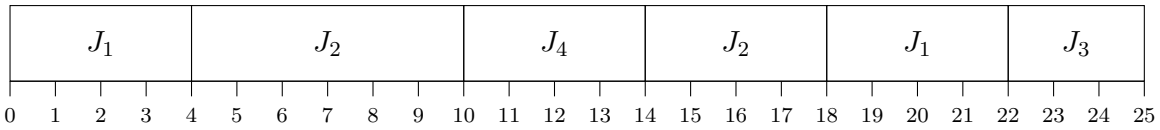
Let us consider the following set of jobs whose arrival times, processing times and priorities are give in the following table

| $J_i$ | $r_i$ | $p_i$ | Priority |
|---|---|---|---|
| $J_1$ | 0 | 8 | 3 |
| $J_2$ | 4 | 10 | 2 |
| $J_3$ | 4 | 3 | 3 |
| $J_4$ | 10 | 4 | 1 |

Gantt Chart

At time 0ms, $J_1$ is the only job and so it starts executing. At time 4ms, $J_2$ and $J_3$ arrive. Since $J_2$ has higher priority than $J_1$, $J_2$ preempts $J_1$. At 10ms, $J_4$ arrives, which has higher priority than $J_2$ and so pre-empts $J_2$. $J_4$ completes execution at 14ms and leaves. The jobs in the system are $J_1$, $J_2$, and $J_3$, among which $J_2$ has the highest priority and so is assigned to the machine. At 18ms, $J_2$ completes execution, and so $J_1$ and $J_3$ are the jobs in the system. Since both jobs are of the same priority, the scheduler selects $J_1$ by the FCFS method. When $J_1$ completes execution, finally $J_3$ executes. The following Gantt chart shows the order of execution






| $J_i$ | $C_i$ | $W_i$ | $F_i$ |
|---|---|---|---|
| $J_1$ | 22 | 14 | 22 |
| $J_2$ | 18 | 4 | 14 |
| $J_3$ | 25 | 18 | 21 |
| $J_4$ | 14 | 0 | 4 |
| Average | 19.75 | 9 | 15.25 |