Process and Threads Management: Scheduling Algorithms

Outline

- What is scheduling
- Objectives of scheduling
- Types of scheduler
- Scheduling algorithms
  - First Come First Served (FCFS)
  - Shortest Job First (SJF)
  - Shortest Remaining Time Next (SRTN)
  - Round Robin (RR)
  - Priority
    - Non-Preemptive Priority
    - Preemptive Priority
- Real Time Operating System

What is process scheduling?

What is process scheduling?

- Process scheduling is the **activity of the process manager** that handles **suspension of running process from CPU and selection of another process** on the basis of a particular strategy.
- The **part of operating system** that **makes** the **choice** is called **scheduler**.
- The **algorithm used by** this **scheduler** is called scheduling **algorithm**.
- Process scheduling is an essential part of a multiprogramming operating systems.

Objectives (goals) of scheduling

Objectives (goals) of scheduling

- **Fairness**: giving each process a fair share of the CPU.
- **Balance**: keeping all the parts of the system busy (Maximize).

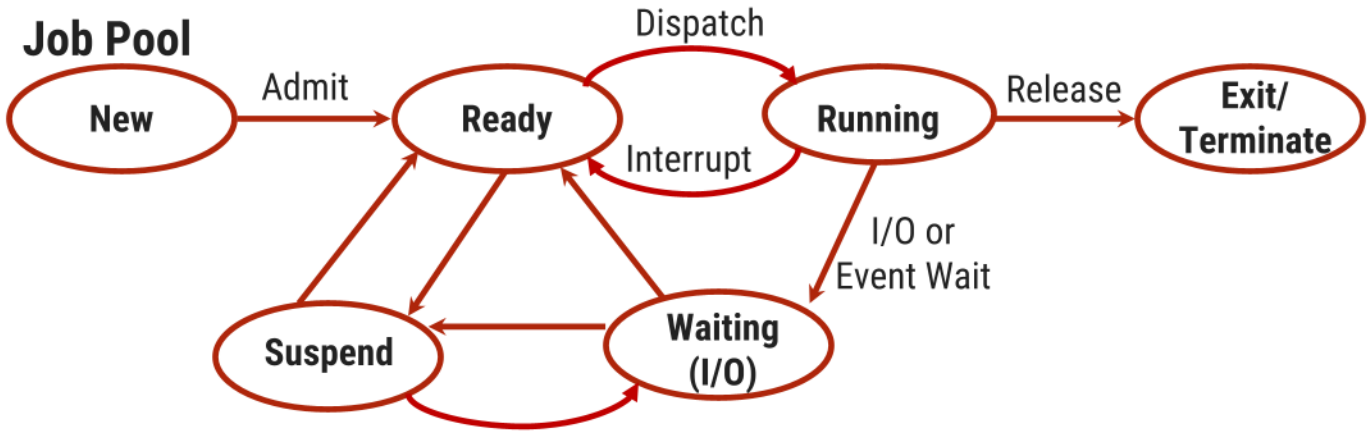
- **Throughput**: no of processes that are completed per time unit (Maximize).
- **Turnaround time**: time to execute a process from submission to completion (Minimize).
  - **Turnaround time** = Process finish time – Process arrival time

- **CPU utilization**: percent of time that the CPU is busy in executing a process.
  - keep CPU as busy as possible (Maximized).

- **Response time**: time between issuing a command and getting the result (Minimized).
- **Waiting time**: amount of time a process has been waiting in the ready queue (Minimize).
  - **Waiting time** = Turnaround time – Actual execution time

Revisit Process State Diagram

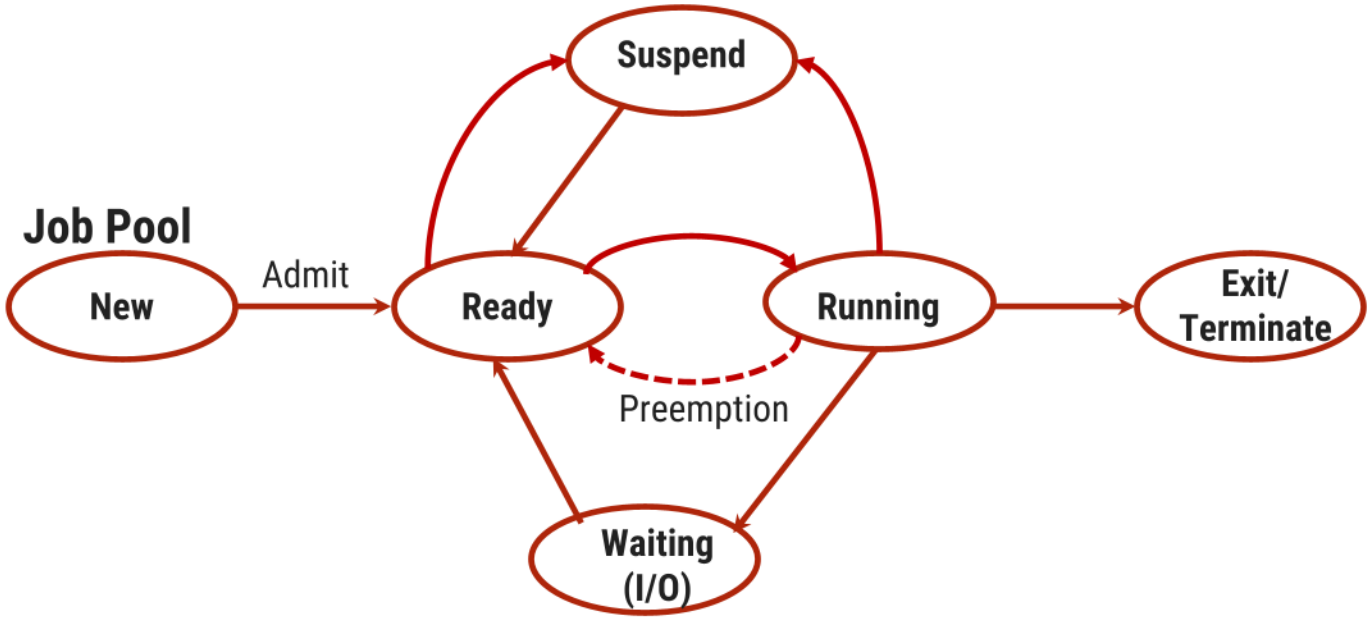
Five State Process Model and Transitions



- **Job Pool/New:** - Collection of programs that are required for execution or we want for execution. A new process is then created.
- **Suspend:** - The process can be suspended when the OS finds illegal access, due to the arrival of a high-priority process. A user process can be suspended when it demands more memory.
- **Ready** – Processes are waiting for CPU
- **Running** – Process is using the CPU
- **Waiting or I/O**– Processing is waiting for I/O or external event.
- **Exit (Terminated)** – Process has finished execution.

Note:
- The degree of multiprogramming decreases upon termination.
- OS brings a new process in the Ready state to maintain the degree of multi-programming.

Simplified Process State Diagram

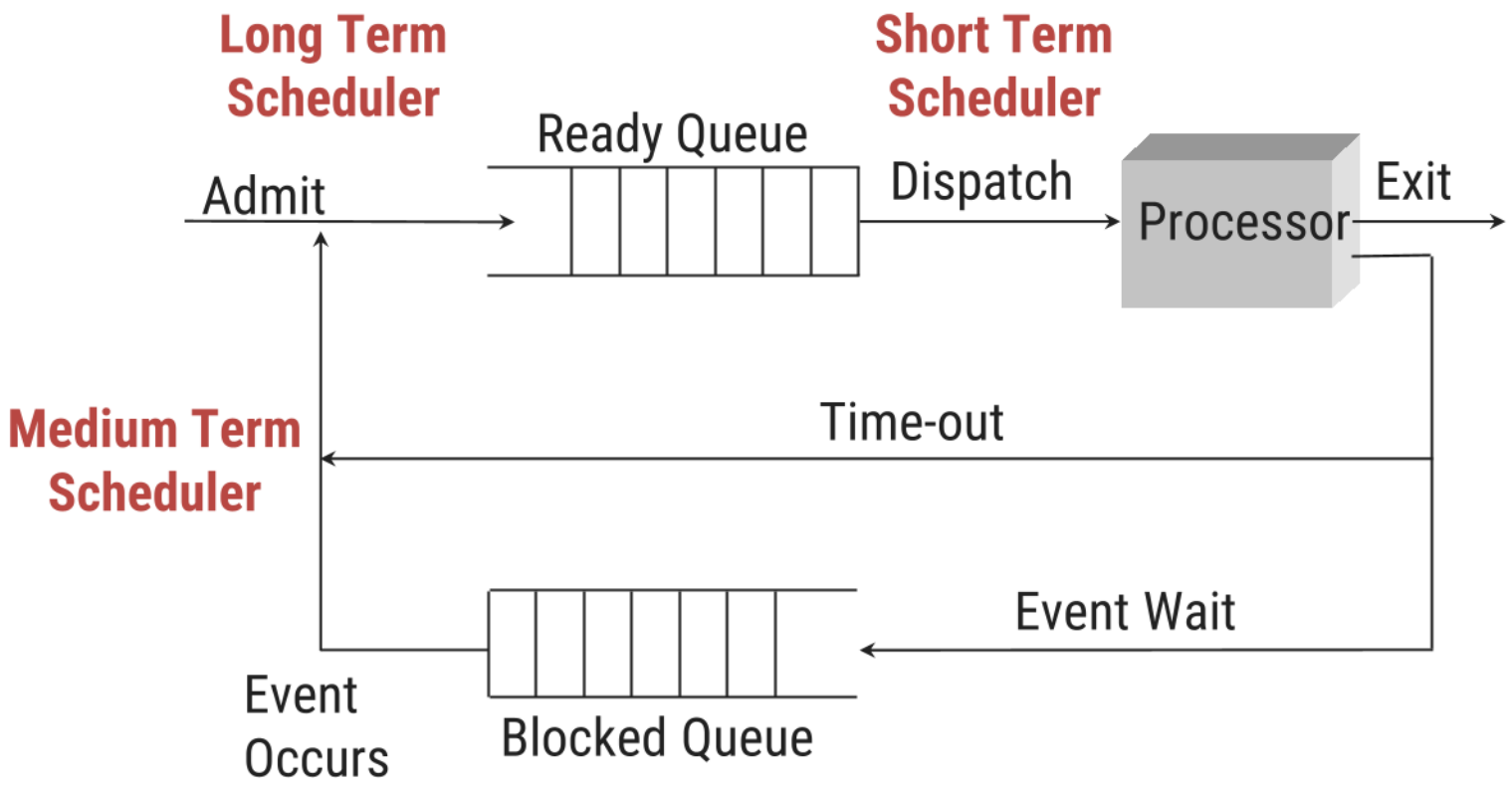


- **Context Switching**–It is an activity where the process that is in the running state is removed from it, and a new process from the ready state is to be loaded in the running state.

- **Dispatcher** – It is the routine of the OS that is supposed to perform context switching activity.

Types of schedulers

Types of schedulers



Types of schedulers

Long-Term Scheduler	Short-Term Scheduler	Medium-Term Scheduler
It is a job scheduler.	It is a CPU scheduler.	It is a process swapping scheduler.
It selects processes from pool and loads them into memory for execution.	It selects those processes which are ready to execute.	It can re-introduce the process into memory and execution can be continued.
Speed is lesser than short term scheduler.	Speed is fastest among other two schedulers.	Speed is in between both short and long term scheduler.
It is almost absent or minimal in time sharing system.	It is also minimal in time sharing system.	It is a part of time sharing systems.

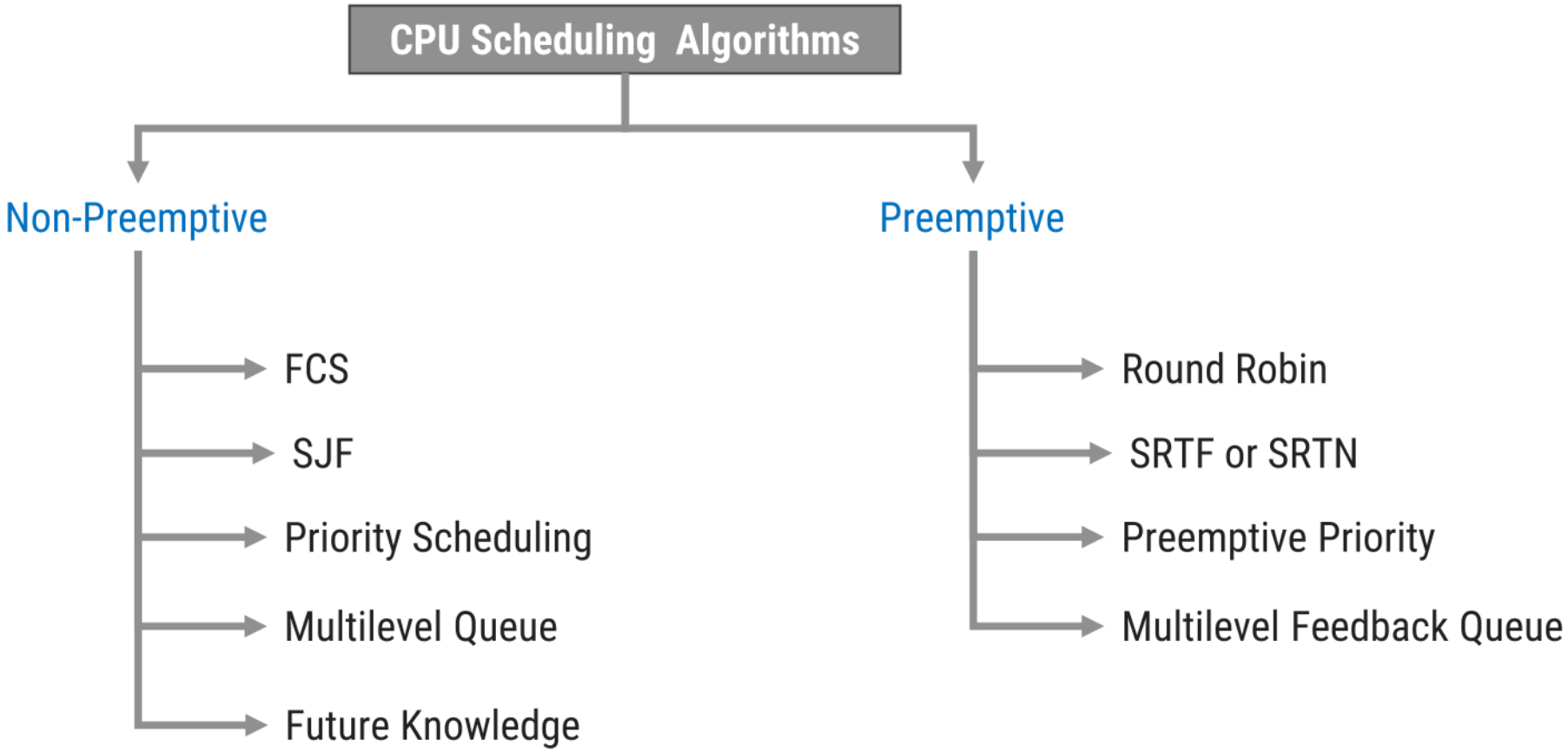
Scheduling algorithms

Scheduling algorithms

1. First Come First Served (FCFS)
2. Shortest Job First (SJF)
3. Shortest Remaining Time Next/First (SRTN/F)
4. Round Robin (RR)
5. Priority
    I. Preemptive
    II. Non-Preemptive

CPU Scheduling Algorithms: Classification

- **Note:** Scheduling is required when a single resource (CPU) is shared by multiple processes.
- **Note:** Short-term scheduler is also called the CPU scheduler.



CPU Scheduling Algorithms: Note Points

- **Non-Preemptive:** The OS takes a processor from a process if the process requires I/O, illegal access (Suspend), or termination; otherwise, the OS will not take the processor. [No Forceful withdrawal of the CPU].
- **Preemptive:** The OS takes the processor from the process forcefully when a process of high priority arrives, even if the currently executing process does not require the I/O, suspend, or terminate. [Forceful withdrawal of CPU].

- **Turn Around Time (TAT):** It is the time period for which the process is active. Or it is the time from when the process is created (first entered in the Ready state) to the time when the process is terminated.
- **Execution Time (ET):** It is the time period for which the process is holding the CPU (time period for which the process is in the Running state).

- **Waiting Time (WT): Turn Around Time – Execution Time.**
  - ↪ **Waiting time = I/O wait + Suspend + Ready = Other than running state time.**

- **Response Time (RT):** It is the time from when the process is entered into the ready state for the first time to when the process is entered into the running state.

CPU Scheduling Algorithms: Note Points

- **CPU Utilization:** It is unitless.

$$\text{Utilization} = \frac{Useful\ time}{Useful\ time + Ideal\ Time + Overhead\ Time} = \frac{Useful\ time}{Total}$$

- **Overhead time:** Time required in context switching.
- **Ideal time:** Time for which the processor is free.

- **Note**: Both the above times should be minimum for CPU utilization.

- **Throughput:** Number of processes completed per unit of time. (Always Calculated on average time)

Example Question

- Consider the following details

Process	Burst Time	Priority
P0	6	3
P1	8	2
P2	4	1
P3	2	4

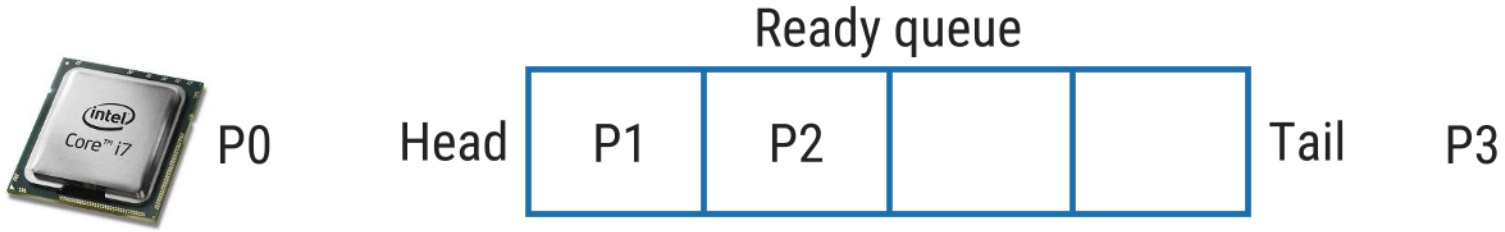
Assuming all the processes are available in a ready state at time 0, & their order of arrival is P0, P1, P2, P3. Consider a lower priority no. means higher priority. Calculate average turnaround time, average waiting time, & average response time using FCS, SJF, and & priority algorithms.

***Note:** CPU burst time is not the execution time. It is the time required by the process for a single stage (chance) as the process executes in multiple stages.*

First Come First Served (FCFS)

First Come First Served (FCFS)

- Selection criteria:
  - The **process that request first is served first**.
  - It means that **processes are served in the exact order of their arrival**.



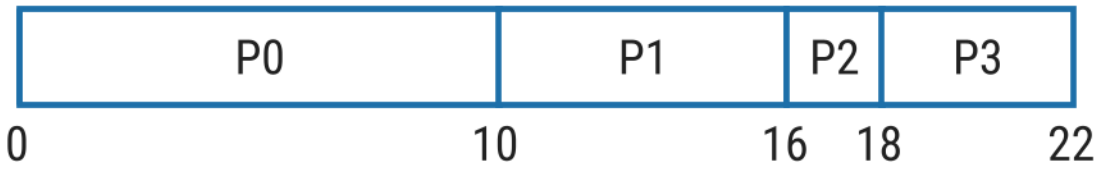
- Decision Mode:
  - **Non preemptive**: **Once a process is selected, it runs until it is blocked** for an I/O or some other event or it is terminated.

- Implementation:
  - This strategy can be easily **implemented by using FIFO** (First In First Out) queue.
  - When CPU becomes free, a process from the first position in a queue is selected to run.

First Come First Served (FCFS)

Process	Arrival Time (T0)	Burst Time (ΔT)	Finish Time (T1)	Turnaround Time (TAT = T1 - T0)	Waiting Time (WT = TAT - ΔT)
P0	0	10	10	10	0
P1	1	6	16	15	9
P2	3	2	18	15	13
P3	5	4	22	17	13

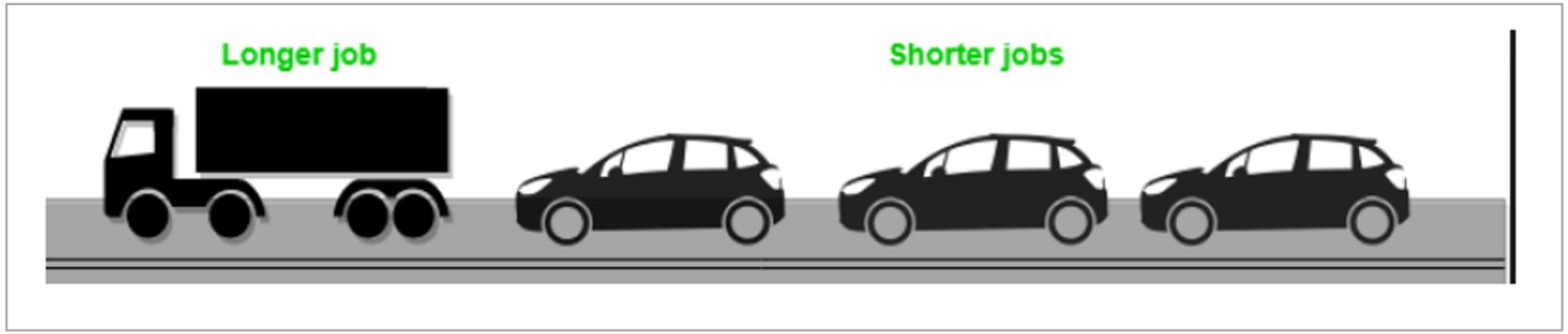
- Gantt Chart



- Average Turnaround Time: (10+15+15+17)/4 = 14.25 ms.
- Average Waiting Time: (0+9+13+13)/4 = 8.75 ms.

First Come First Served (FCFS)

- Advantages
  - **Simple** and **fair**.
  - **Easy to understand** and **implement**.
  - Every process will get a chance to run, so **starvation doesn't occur**.
    - Starvation is the problem that occurs when high priority processes keep executing and low priority processes get blocked for indefinite time.
- Disadvantages
  - **Not efficient** because average waiting time is too high.
  - **Convoy effect is possible**. All small I/O bound processes wait for one big CPU bound process to acquire CPU.

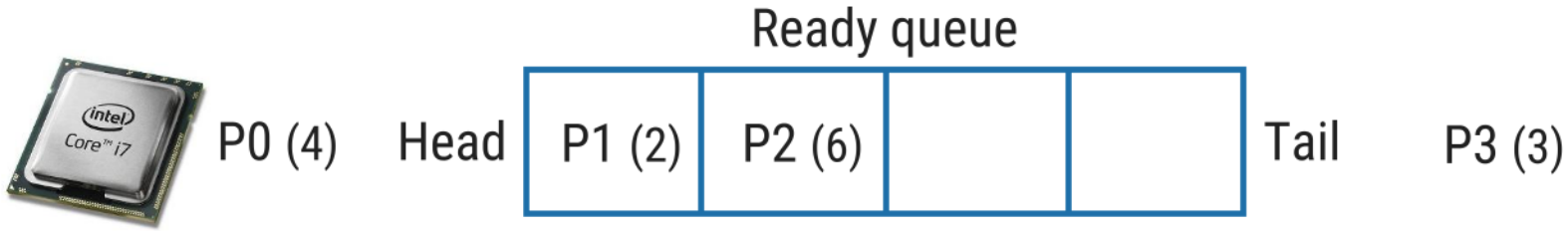


  - **CPU utilization may be less efficient** especially when a CPU bound process is running with many I/O bound processes.

Shortest Job First (SJF)

Shortest Job First (SJF)

- Selection criteria:
  - The process, that **requires shortest time to complete execution, is served first**.



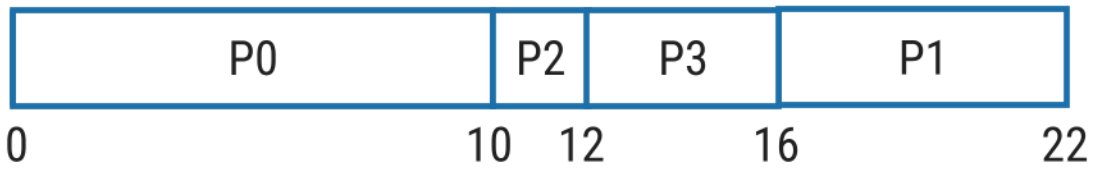
- Decision Mode:
  - **Non preemptive**: **Once a process is selected, it runs until it is blocked** for an I/O or some other event or it is terminated.

- Implementation:
  - This strategy can be easily **implemented by using sorted FIFO** (First In First Out) queue.
  - All processes in a queue are **sorted in ascending order based on their required CPU bursts**.
  - When CPU becomes free, a process from the first position in a queue is selected to run.

Shortest Job First (SJF)

Process	Arrival Time (T0)	Burst Time (ΔT)	Finish Time (T1)	Turnaround Time (TAT = T1 - T0)	Waiting Time (WT = TAT - ΔT)
P0	0	10	10	10	0
P1	1	6	22	21	15
P2	3	2	12	9	7
P3	5	4	16	11	7

- Gantt Chart



- Average Turnaround Time: (10+21+9+11)/4 = 12.75 ms.
- Average Waiting Time: (0+15+7+7)/4 = 7.25 ms.

Shortest Job First (SJF)

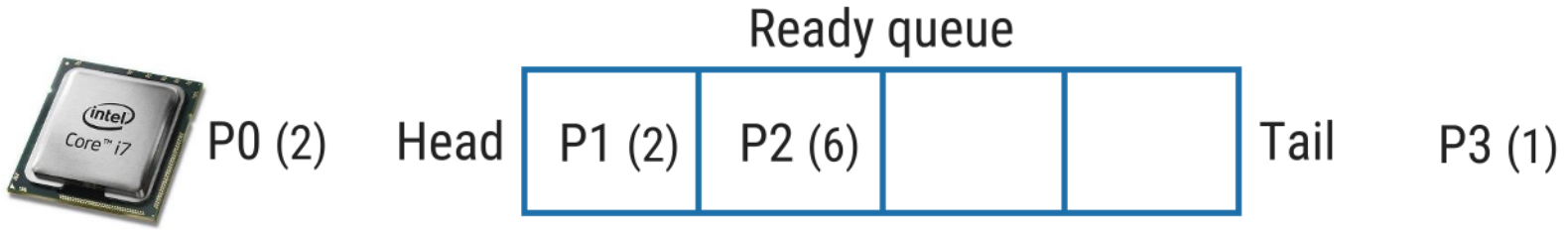
- Advantages
  - **Less waiting time.**
  - **Good response for short processes.**

- Disadvantages
  - It is **difficult to estimate time required** to complete execution.
  - **Starvation is possible for long process**. Long process may wait forever.
    - Starvation is the problem that occurs when high priority processes keep executing and low priority processes get blocked for indefinite time.

Shortest Remaining Time Next (SRTN)

Shortest Remaining Time Next (SRTN)

- Selection criteria:
  - The process, **whose remaining run time is shortest, is served first**. This is a **preemptive version of SJF** scheduling.

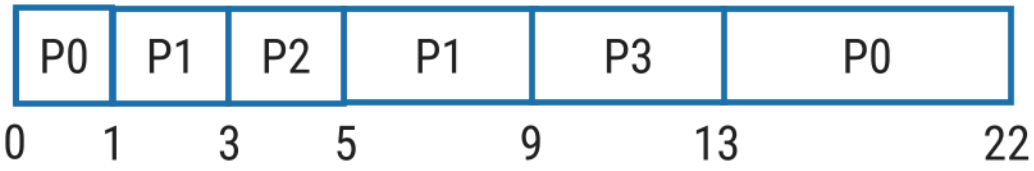


- Decision Mode:
  - **Preemptive**: **When a new process arrives, its total time is compared to the current process remaining run time**.
  - If the **new process needs less time to finish than the current process, the current process is suspended and the new job is started**.
- Implementation:
  - This strategy can also be implemented by using **sorted FIFO queue**.
  - All processes in a queue are **sorted in ascending order on their remaining run time**.
  - When CPU becomes free, a process from the first position in a queue is selected to run.

Shortest Remaining Time Next (SRTN)

Process	Arrival Time (T0)	Burst Time (ΔT)	Finish Time (T1)	Turnaround Time (TAT = T1 - T0)	Waiting Time (WT = TAT - ΔT)
P0	0	10	22	22	12
P1	1	6	9	8	2
P2	3	2	5	2	0
P3	5	4	13	8	4

- Gantt Chart



Proce	Proce	Process	Process	Process	Remaining Time
P1	P0	P0	P0	P0	9
P0	P2	P1	P3	4	
	P1	P3	4		

- Average Turnaround Time: 10 ms
- Average Waiting Time: 4.5 ms

Shortest Remaining Time Next (SRTN)

- Advantages
  - **Less waiting time.**
  - **Quite good response for short processes.**

- Disadvantages
  - It is **difficult to estimate time required** to complete execution.
  - **Starvation is possible for long process**. Long process may wait forever.
    - Starvation is the problem that occurs when high priority processes keep executing and low priority processes get blocked for indefinite time.
  - **Context switch overhead is there**.

Round Robin (RR)

Round Robin (RR)

- Selection criteria:
  - Each selected process is **assigned a time interval, called time quantum or time slice**.
  - Process is **allowed to run only for this time interval**.
  - Here, two things are possible:
  - First, **process is either blocked or terminated before the quantum has elapsed**. In this case the **CPU switching is done** and **another process is scheduled** to run.

Ready queue & Quantum = 3

P0 (2) Head | P1 (2) | P2 (6) | | | Tail P3 (1)

  - Second, **process needs CPU burst longer than time quantum**. In this case, process is **running at the end of the time quantum**.
  - Now, it will be **preempted** and **moved to the end of the queue**.
  - CPU will be **allocated to another process**.
  - Here, **length of time quantum is critical to determine**.

Ready queue & Quantum = 3

P0 (4) Head | P1 (2) | P2 (6) | | | Tail P3 (1)

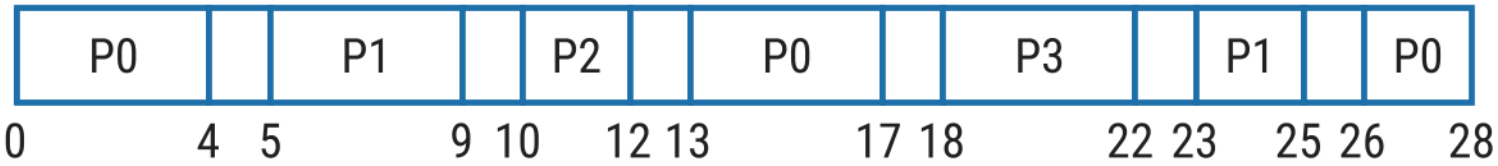
Round Robin (RR)

- Decision Mode:
  - **Preemptive**: **When a new process arrives, its total time is compared to the current process remaining run time**.
  - **Selection of new job is as per FCFS** scheduling algorithm.

- Implementation:
  - This strategy can be implemented by using **circular FIFO queue**.
  - If any process comes, or process releases CPU, or process is preempted. It is moved to the end of the queue.
  - When CPU becomes free, a process from the first position in a queue is selected to run.

Round Robin (RR)

Process	Arrival Time (T0)	Burst Time (ΔT)	Finish Time (T1)	Turnaround Time (TAT = T1 - T0)	Waiting Time (WT = TAT - ΔT)
P0	0	10	28	28	18
P1	1	6	25	24	18
P2	3	2	12	9	7
P3	5	4	22	17	13



- Gantt Chart
  - Quantum time is 4 ms &
  - Context switch overhead is 1 ms
- Avg. Turnaround Time: 19.5 r
- Avg. Waiting Time: 14 ms

Process	Re	Process	Process	Ren	Process	Rer	Process	Process	Remaining Time
P1	6	P2	P0	6	P3	4	P1	P0	2
P2	2	P0	P3	4	P1	2	P0	2	
P0	6	P3	P1	2	P0	2			
		P1	2						

Round Robin (RR)

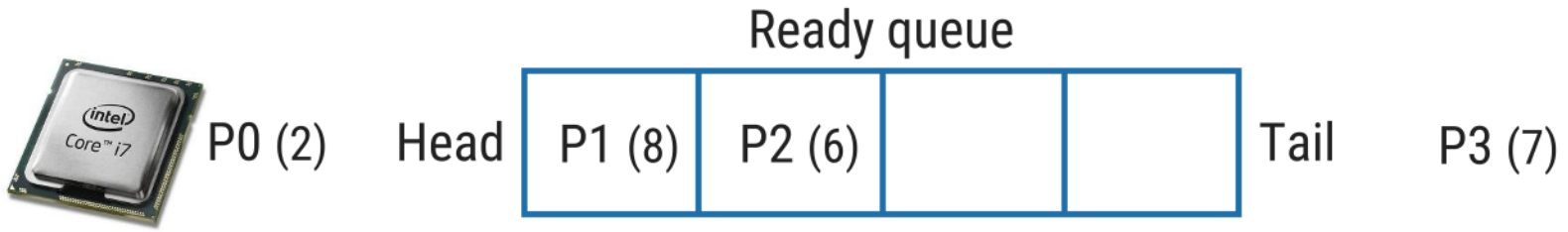
- Advantages
  - **Simplest, fairest and most widely used algorithms.**

- Disadvantages
  - **Context switch overhead is there**.
  - **Determination of time quantum is too critical**.
    - If it is too short, it causes frequent context switches and lowers CPU efficiency.
    - If it is too long, it causes poor response for short interactive process.

Priority
(Non-Preemptive Priority)

Non-Preemptive Priority

- Selection criteria:
  - The process, that **has highest priority, is served first**.

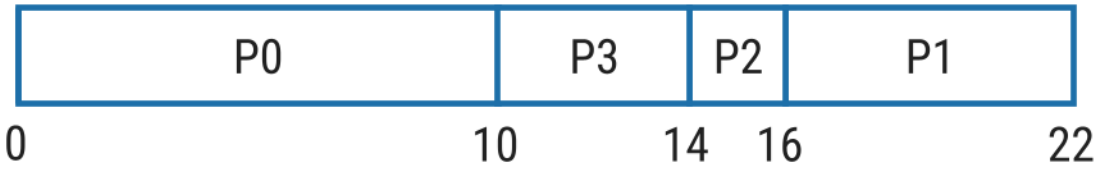


- Decision Mode:
  - **Non preemptive**: **Once a process is selected, it runs until it is blocked** for an I/O or some other event or it is terminated.
- Implementation:
  - This strategy can also be implemented by using **sorted FIFO queue**.
  - All processes in a queue are **sorted based on their priority with highest priority process at front end**.
  - When CPU becomes free, a process from the first position in a queue is selected to run.

Non-Preemptive Priority

Process	Arrival Time (T0)	Burst Time (ΔT)	Priority	Finish Time (T1)	Turnaround Time (TAT = T1 - T0)	Waiting Time (WT = TAT - ΔT)
P0	0	10	5	10	10	0
P1	1	6	4	22	21	15
P2	3	2	2	16	13	11
P3	5	4	0	14	9	5

- Gantt Chart (small values for priority means higher priority of a process)



- Avg. Turnaround Time: 13.25 ms
- Avg. Waiting Time: 7.75 ms

Non-Preemptive Priority

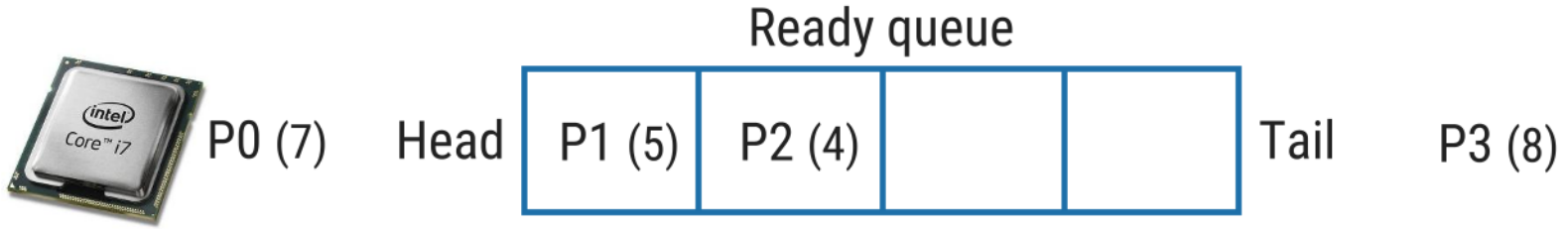
- Advantages
  - **Priority is considered so critical processes can get even better response time.**

- Disadvantages
  - **Starvation is possible for low priority processes.** It can **be overcome by using** technique called **'Aging'.**
  - **Aging: gradually increases the priority of processes that wait in the system for a long time.**

Priority
(Preemptive Priority)

Preemptive Priority

- Selection criteria:
  - The process, that **has highest priority, is served first**.

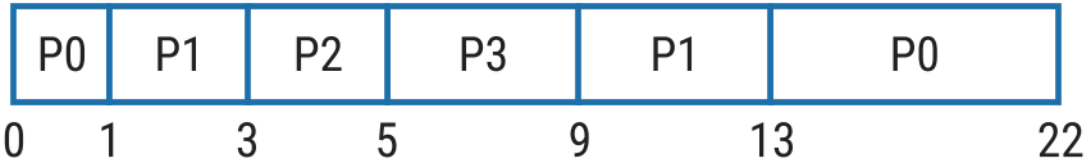


- Decision Mode:
  - **Preemptive: When a new process arrives, its priority is compared with current process priority.**
  - If the **new process has higher priority than the current, the current process is suspended and new job is started.**
- Implementation:
  - This strategy can also be implemented by using **sorted FIFO queue**.
  - All processes in a queue are **sorted based on their priority with highest priority process at front end**.
  - When CPU becomes free, a process from the first position in a queue is selected to run.

Preemptive Priority

Process	Arrival Time (T0)	Burst Time (ΔT)	Priority	Finish Time (T1)	Turnaround Time (TAT = T1 - T0)	Waiting Time (WT = TAT - ΔT)
P0	0	10	5	22	22	12
P1	1	6	4	13	12	6
P2	3	2	2	5	2	0
P3	5	4	0	9	4	0

- Gantt Chart
  - small values means higher priority
- Avg. Turnaround Time: 10 ms
- Avg. Waiting Time: 4.5 ms



Proces	Proces	Process	Process	Process	Priority
P1	P0	P0	P0	P0	5
P0	P2	P1	P1	4	
	P1	P3	0		

Preemptive Priority

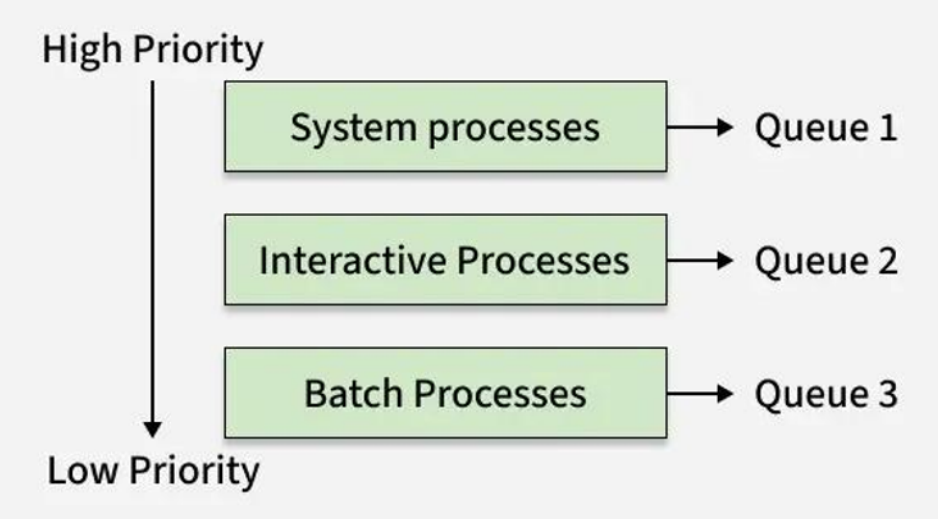
- Advantages
  - **Priority is considered so critical processes can get even better response time.**

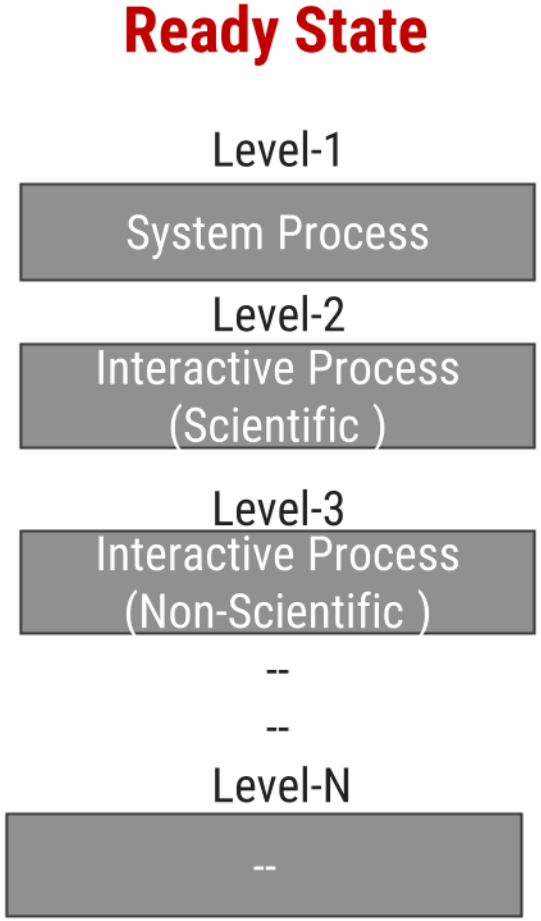
- Disadvantages
  - **Starvation is possible for low priority processes.** It can **be overcome by using** technique called **'Aging'.**
  - **Aging: gradually increases the priority of processes that wait in the system for a long time.**
  - **Context switch overhead is there**.

Multi Level Queue and Multilevel feedback Queue

Multilevel Queue (MLQ)

- **Multi-level Queue (MLQ)** scheduling is an OS technique that divides the ready queue into separate queues (e.g., system, interactive, batch), each with its own scheduling algorithm (like Round Robin or FCFS) and priority, with higher queues always running before lower ones, providing better responsiveness for interactive tasks by separating them from background jobs, though processes don't move between queues.





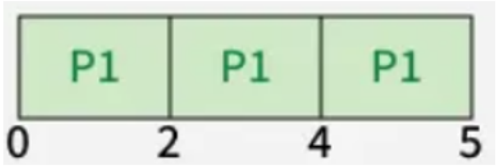
- It is based on priority. Level-1 is the highest priority, and execution of the process in these queues is non-preemptive.
- Here, starvation may occur due to non-preemption.

Multilevel Queue: Example

- Consider the table below of four processes under Multilevel queue scheduling. We have 3 processes:
  - P1: Uses Round Robin (RR) in Queue 1
  - P2: Uses First-Come-First-Serve (FCFS) in Queue 2
  - P3: Uses Shortest Job First (SJF) in Queue 3.

Process	Queue(scheduling)	Arrival Time	Burst Time
P1	Queue 1 (RR , q=2)	0 ms	5 ms
P2	Queue 2 (FCFS)	1 ms	4 ms
P3	Queue 3 (SJF)	2 ms	3 ms

- **Sol.:**



At time=0 ms.
P1 arrives (Queue 1: RR).
CPU executes P1 for 2 ms (time quantum = 2).
Remaining burst of P1=5-2=3 ms.

At time=2 ms.
At **t=1**,P2 arrived (Queue 2: FCFS).
At t=2, P3 arrived (Queue 3: SJF).
Still, Queue 1 has P1 (3 ms left) > highest priority.
CPU executes P1 again for 2 ms.
Remaining burst of P1=3-2=1ms.

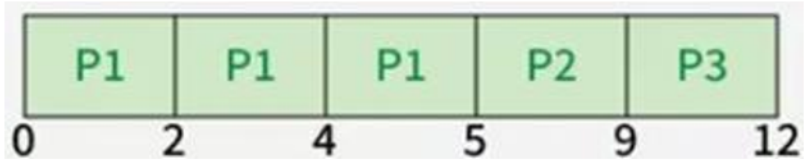
At time=4 ms.
Queue 1 still has P1 (1 ms left).
CPU executes P1 for final 1 ms.
P1 finishes at t = 5 ms.

Multilevel Queue: Example (Cont..)

- Consider the table below of four processes under Multilevel queue scheduling. We have 3 processes:
  - P1: Uses Round Robin (RR) in Queue 1
  - P2: Uses First-Come-First-Serve (FCFS) in Queue 2
  - P3: Uses Shortest Job First (SJF) in Queue 3.

Process	Queue(scheduling)	Arrival Time	Burst Time
P1	Queue 1 (RR , q=2)	0 ms	5 ms
P2	Queue 2 (FCFS)	1 ms	4 ms
P3	Queue 3 (SJF)	2 ms	3 ms

- **Sol.:**



At time=5 ms

Now Queue 1 is empty.
Next highest priority = Queue 2 (P2, FCFS).
CPU executes P2 fully (since FCFS is non-preemptive).

Burst =4 ms.
P2 finishes at t =9 ms.

At time=9 ms

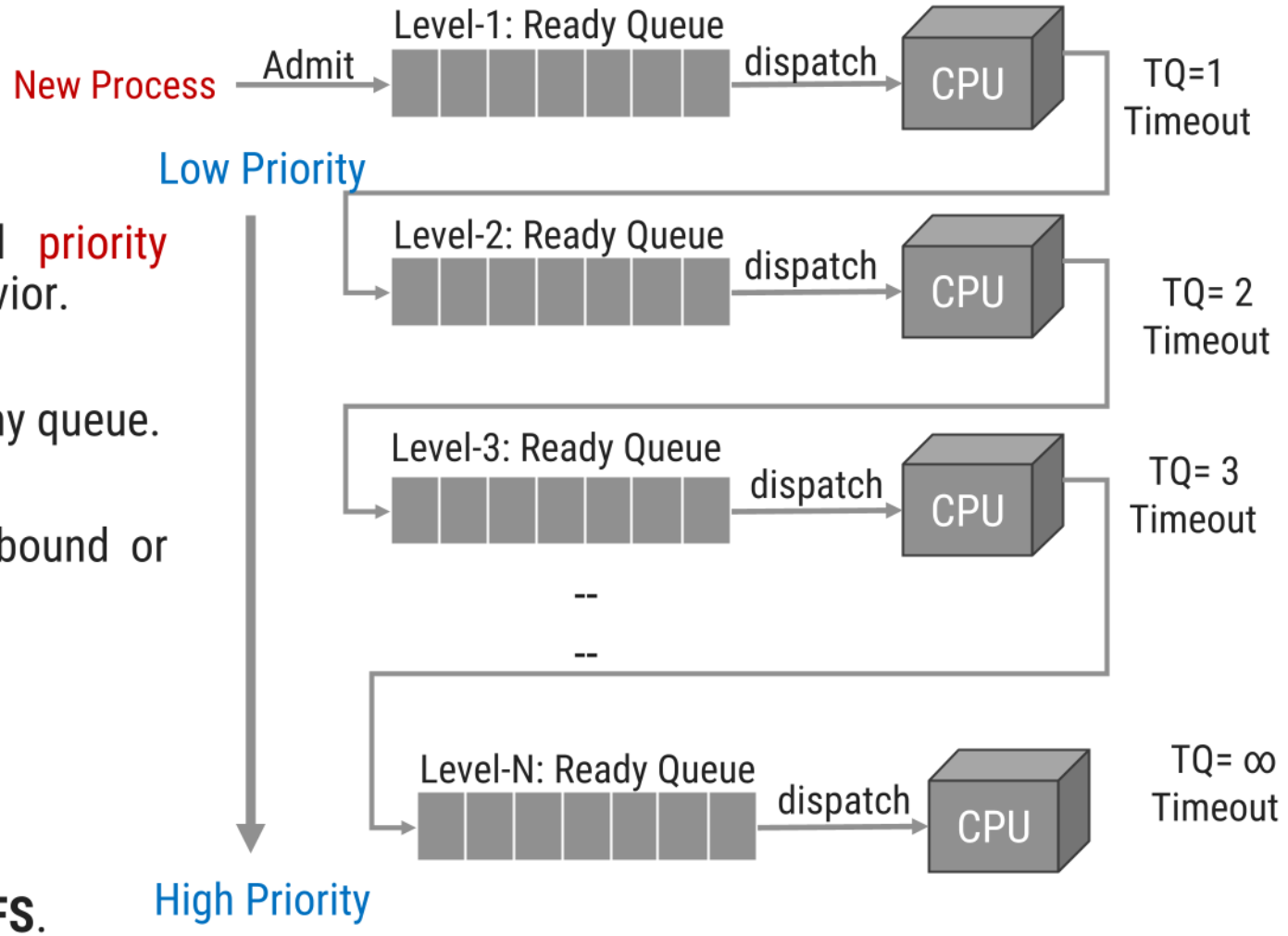
Queue 3 (P3) is now ready.
Scheduling = SJF, only one process.
CPU executes P3 fully (3 ms).

P3 finishes at t = 12 ms.

Multilevel Queue Feedback Queue (MLFQ)

- It is an extension of MLQ where processes can move between the queues; thus, it is much more efficient.

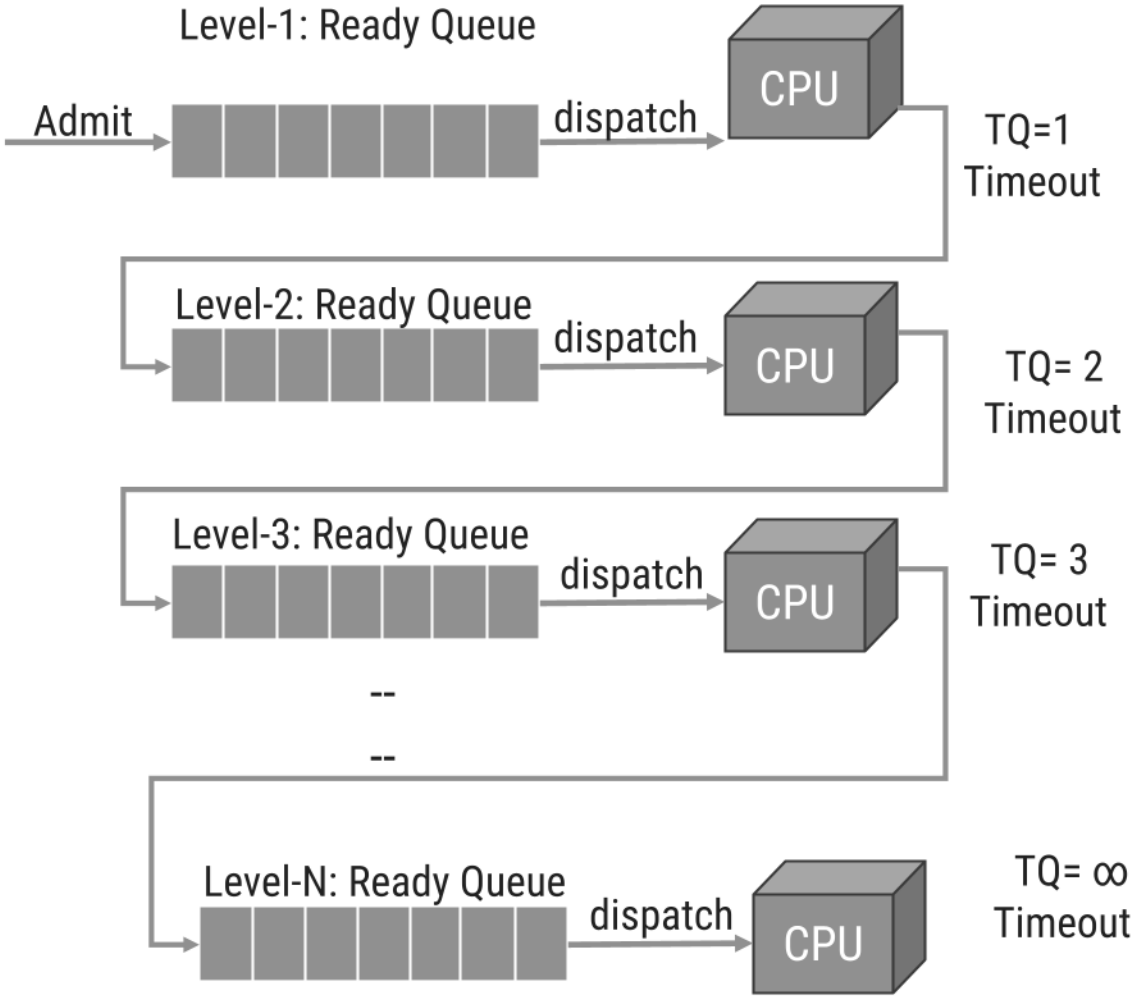
- **Characteristics of MLFQ:**
  - Each queue has its own time slice.
  - Processes can move between queues and priority changes based on their CPU usage or I/O behavior.
  - Processes are NOT permanently assigned to any queue.
  - CPU-heavy processes move down, while I/O-bound or short processes move up.
  - High-priority queues have shorter slices.
  - Processes using their slice may get demoted.
  - If there is only one level, then it works as **FCFS**.
  - The last level also works as FCFS.



Multilevel Queue Feedback Queue (MLFQ)

- It is an extension of MLQ where processes can move between the queues; thus, it is much more efficient.

- **Note Points:**
  - Priority of long jobs decreases, and more preference is given to short jobs.
  - Here, the response time of the processes can be fixed for the multi-programming system.
  - MLFQ dynamically adjusts the priority of processes based on their behavior.
  - Preemption is allowed, to ensure urgent or interactive tasks get CPU quickly.
  - Meaning higher-priority processes can interrupt lower-priority ones. Lower-priority processes resume later when the CPU is free.



Real Time Operating System

Real Time Operating System

- A real-time system is one in which **time plays an essential role**.
- Real time computing may be defined as that type of computing in which the **correctness of the system depends not only on the logical result of the computation but also on the time at which the results are produced**.
- Some of the real-time systems are **patient monitoring in a hospital intensive-care unit, the autopilot in an aircraft and robot control in an automated factory**.
- In all these cases, having the right answer but having it too late is often just as bad as not having it at all.
- Real time task may be classified as hard and soft.
  - A **hard real time task is one that must meet its deadline**; otherwise it will cause unacceptable damage or a fatal error to the system.
  - A **soft real time task has an associated deadline that is desirable but not mandatory**; it will not cause unacceptable damage or a fatal error on missing deadline.

Real Time Operating System

- The events that a real-time system may have to respond to can be further categorized as **periodic (occurring at regular intervals)** or **aperiodic (occurring unpredictably)**.
- A system may have to respond to multiple periodic event streams. Depending on how much time each event requires for processing, it may not even be possible to handle them all.
- Real-time scheduling algorithms can be **static or dynamic**.
  - Static: The former make their scheduling decisions before the system starts running.
  - Dynamic: The latter make their scheduling decisions at run time.
  - Static scheduling only works when there is perfect information available in advance about the work to be done and the deadlines that have to be met.
  - Dynamic scheduling algorithms do not have these restrictions.

Exercise

1. Five batch jobs A to E arrive at same time. They have estimated running times 10,6,2,4 and 8 minutes. Their priorities are 3,5,2,1 and 4 respectively with 5 being highest priority. For each of the following algorithm determine mean process turnaround time. Ignore process swapping overhead. Quantum time is 2 minute.
   - Round Robin, Priority Scheduling, FCFS, SJF.

2. Suppose that the following processes arrive for the execution at the times indicated. Each process will run the listed amount of time. Assume preemptive scheduling.

Process	Arrival Time (ms)	Burst Time (ms)
P1	0.0	8
P2	0.4	4
P3	1.0	1

   - What is the turnaround time for these processes with Shortest Job First scheduling algorithm?

Exercise

3. Consider the following set of processes with length of CPU burst time given in milliseconds.

Process	Burst Time	Priority
P1	10	3
P2	1	1
P3	2	3
P4	1	4
P5	5	2

   - Assume arrival order is: P1, P2, P3, P4, P5 all at time 0 and a smaller priority number implies a higher priority. Draw the Gantt charts illustrating the execution of these processes using preemptive priority scheduling.

Practice Questions

1. Define term Scheduler, Scheduling and Scheduling Algorithm with example.
2. Define terms. 1) Throughput 2) Waiting Time 3) Turnaround Time 4) Response Time 5) Granularity 6) Short Term Scheduler 7) CPU Utilization
3. What is scheduler? Explain queuing diagram representation of process scheduler with figure.
4. Write various scheduling criteria.
5. Consider Five Processes P1 to P5 arrived at same time. They have estimated running time 10, 2, 6, 8 and 4 seconds, respectively. Their Priorities are 3, 2, 5, 4 and 1, respectively with 5 being highest Priority. Find the average turnaround time and average waiting time for Round Robin (quantum time=3) and Priority Scheduling algorithm.
6. Consider the processes P1, P2, P3, P4 with burst time is 21, 3, 6 and 2 respectively, arrives for execution in the same order, with arrival time 0, draw GANTT chart and find the average waiting time using the FCFS and SJF scheduling algorithm.

Thank You