CPU Scheduling

What is CPU Scheduling?

- CPU scheduling is a process which allows one process to use the CPU while the execution of another process is on hold(in waiting state).
- The aim of CPU scheduling is to make the system efficient, fast and fair.
- Whenever the CPU becomes idle, the operating system must select one of the processes in the **ready queue** to be executed.
- The selection process is carried out by the short-term scheduler (or CPU scheduler).
- The scheduler selects from among the processes in memory that are ready to execute, and allocates the CPU to one of them.

CPU Scheduling: Dispatcher

- Another component involved in the CPU scheduling function is the **Dispatcher**.
- The dispatcher is the module that gives control of the CPU to the process selected by the **short-term scheduler**.

Types of CPU Scheduling

CPU scheduling decisions may take place under the following four circumstances:

- When a process switches from the **running** state to the **waiting** state(for I/O request or invocation of wait for the termination of one of the child processes).
- When a process switches from the **running** state to the **ready** state (for example, when an interrupt occurs).
- When a process switches from the **waiting** state to the **ready** state(for example, completion of I/O).
- When a process **terminates**.
- In circumstances 1 and 4, there is no choice in terms of scheduling. A new process(if one exists in the ready queue) must be selected for execution. There is a choice, however in circumstances 2 and 3.
- When Scheduling takes place only under circumstances 1 and 4, we say the scheduling scheme is **non-preemptive**; otherwise the scheduling scheme is **preemptive**.

Non-Preemptive Scheduling

- Under non-preemptive scheduling, once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state.

Preemptive Scheduling

- In this type of Scheduling, the tasks are usually assigned with priorities.
- At times it is necessary to run a certain task that has a higher priority before another task although it is running.
- Therefore, the running task is interrupted for some time and resumed later when the priority task has finished its execution.

CPU Scheduling Criteria

There are many different criterias to check when considering the **"best"** scheduling algorithm, they are:

CPU Utilization

- To make out the best use of CPU and not to waste any CPU cycle, CPU would be working most of the time(Ideally 100% of the time). Considering a real system, CPU usage should range from 40% (lightly loaded) to 90% (heavily loaded.)

Throughput

- It is the total number of processes completed per unit time or rather say total amount of work done in a unit of time. This may range from 10/second to 1/hour depending on the specific processes.

Turnaround Time

- It is the amount of time taken to execute a particular process, i.e. The interval from time of submission of the process to the time of completion of the process(Wall clock time).

Waiting Time

- The sum of the periods spent waiting in the ready queue amount of time a process has been waiting in the ready queue to acquire get control on the CPU.

Load Average

- It is the average number of processes residing in the ready queue waiting for their turn to get into the CPU.

Response Time

- Amount of time it takes from when a request was submitted until the first response is produced. Remember, it is the time till the first response and not the completion of process execution(final response).

- **CPU Burst:** The time required by Process for execution.

Optimization Criteria

- Max CPU Utilization
- Max Throughput
- Min Turnaround time
- Min Waiting time
- Min response time

Scheduling Algorithms

To decide which process to execute first and which process to execute last to achieve maximum CPU utilisation, computer scientists have defined some algorithms, they are:

- **First Come First Serve(FCFS) Scheduling**
- **Shortest-Job-First(SJF) Scheduling**
- **Priority Scheduling**
- **Round Robin(RR) Scheduling**

First Come First Serve (FCFS) Scheduling

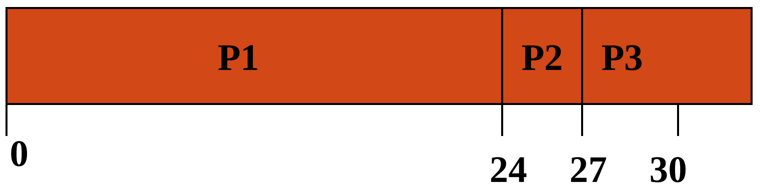
- Policy: Process that requests the CPU *FIRST* is allocated the CPU *FIRST*.
  - FCFS is a non-preemptive algorithm.
- Implementation - using FIFO queues
  - incoming process is added to the tail of the queue.
  - Process selected for execution is taken from head of queue.
- Performance metric - Average waiting time in queue.
- Gantt Charts are used to visualize schedules.

First-Come, First-Served(FCFS) Scheduling

- Example

Process	Burst Time
P1	24
P2	3
P3	3

Gantt Chart for Schedule



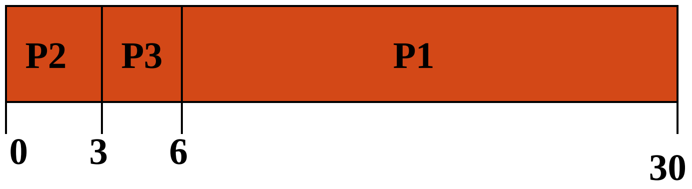
- Suppose the arrival order for the processes is
  - P1, P2, P3
- Waiting time
  - P1 = 0;
  - P2 = 24;
  - P3 = 27;
- Average waiting time
  - (0+24+27)/3 = 17

FCFS Scheduling (cont.)

- Example

Process	Burst Time
P1	24
P2	3
P3	3

Gantt Chart for Schedule



- Suppose the arrival order for the processes is
  - P2, P3, P1
- Waiting time
  - P1 = 6; P2 = 0; P3 = 3;
- Average waiting time
  - (6+0+3)/3 = 3 , better..
- *Convoy Effect*:
  - short process behind long process, e.g. 1 CPU bound process, many I/O bound processes.

What is Convoy Effect?

- Convoy Effect is a situation where many processes, who need to use a resource for short time are blocked by one process holding that resource for a long time.
- This essentially leads to poor utilization of resources and hence poor performance.

Shortest-Job-First(SJF) Scheduling

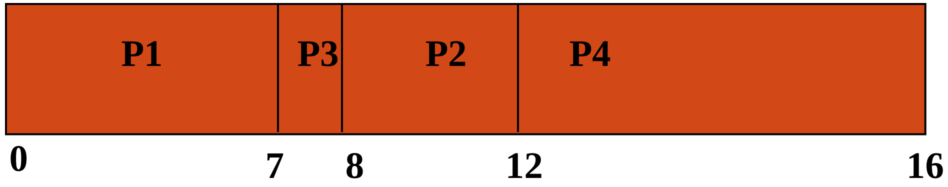
- Associate with each process the length of its next CPU burst. Use these lengths to schedule the process with the shortest time.
- Two Schemes:
  - **Scheme 1: Non-preemptive**
    - Once CPU is given to the process it cannot be preempted until it completes its CPU burst.
  - **Scheme 2: Preemptive**
    - If a new CPU process arrives with CPU burst length less than remaining time of current executing process, preempt. Also called Shortest-Remaining-Time-First (SRTF).
  - SJF is optimal - gives minimum average waiting time for a given set of processes.

Non-Preemptive SJF Scheduling

- Example

Process	Arrival Time	Burst Time
P1	0	7
P2	2	4
P3	4	1
P4	5	4

Gantt Chart for Schedule



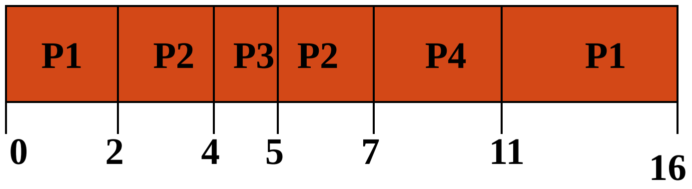
Average waiting time = [0+(8-2)+(7-4)+(12-5)]/4 = 4

Preemptive SJF Scheduling(SRTF)

- Example

Process	Arrival Time	Burst Time
P1	0	7
P2	2	4
P3	4	1
P4	5	4

Gantt Chart for Schedule



Average waiting time = (11-2)+(5-2-2)+(4-4)+(5-7))/4 = 3

Priority Scheduling

- Priority is assigned for each process.
- Process with highest priority is executed first and so on.
- Processes with same priority are executed in FCFS manner.
- Priority can be decided based on memory requirements, time requirements or any other resource requirement.

Priority Scheduling

PROCESS	BURST TIME	PRIORITY
P1	21	2
P2	3	1
P3	6	4
P4	2	3

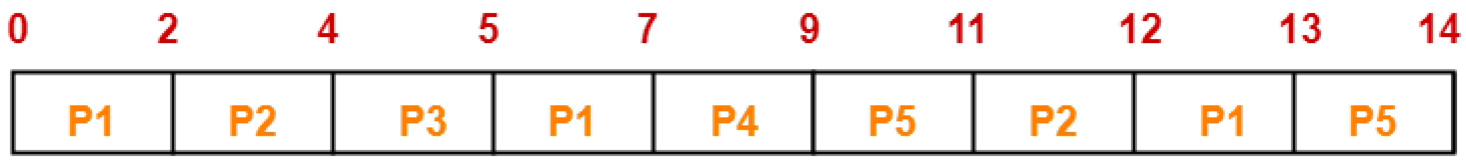
The GANTT chart for following processes based on Priority scheduling will be,

The average waiting time will be, (0 + 3 + 24 + 26)/4 = 13.25 ms

Round Robin Scheduling

- A fixed time is allotted to each process, called **quantum**, for execution.
- Once a process is executed for given time period that process is preempted and other process executes for given time period.
- Context switching is used to save states of preempted processes.

Process Id	Arrival time	Burst time
P1	0	5
P2	1	3
P3	2	1
P4	3	2
P5	4	3



Gantt Chart

Process Id	Exit time	Turn Around time	Waiting time
P1	13	13 – 0 = 13	13 – 5 = 8
P2	12	12 – 1 = 11	11 – 3 = 8
P3	5	5 – 2 = 3	3 – 1 = 2
P4	9	9 – 3 = 6	6 – 2 = 4
P5	14	14 – 4 = 10	10 – 3 = 7

- Average waiting time = (8 + 8 + 2 + 4 + 7) / 5 = 29 / 5 = 5.8 unit