OPERATING SYSTEMS (THEORY)

LECTURE - 6

K.ARIVUSELVAN

Assistant Professor (Senior) – (SITE)

VIT University

SCHEDULING ALGORITHMS

SCHEDULING ALGORITHMS

Policies:

(1) Pre emptive:

- Once the CPU is assigned to a process, the CPU can release the processes even in the Middle of the execution

(2) Non - Pre emptive:

- Once the CPU assigned to a process the processor do not release, until the completion of that process

SCHEDULING ALGORITHMS

- First Come First Serve [FCFS]
- Shortest Job First [SJF]
- Shortest Remaining Time First [SRTF]
- Priority Scheduling
- Round Robin Scheduling
- Highest Response Ratio Next [HRRN]
- Multi Level Feedback Queue

First Come First Serve [FCFS]

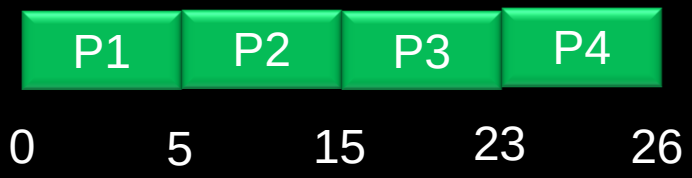
- Simplest of all scheduling algorithms
- Process requests the CPU first is given the CPU
- Implemented using FIFO Queue
- Once a process is given the CPU, it keeps till the completion

(Non – Preemptive)

EXAMPLE - 1

Arrival Time	Process	CPU Time (or) Burst Time
0	P1	5
0	P2	10
0	P3	8
0	P4	3

Time required by CPU to execute job



Gantt Chart

Average Waiting Time:

Waiting Time = **Starting Time – Arrival** Time

P1 => 0 – 0 = 0

P2 => 5 – 0 = 5

P3 => 15 – 0 = 15

P4 => 23 – 0 = 23

Average waiting Time = 0 + 5 + 15 + 23 / 4

= 43 / 4

=> 10.75ms

Average Turn Around Time:

Turn around time = Finished Time – Arrival Time

P1 = 5 – 0 = 5

P2 = 15 – 0 = 15

P3 = 23 – 0 = 23

P4 = 26 – 0 = 26

Average Turn around Time = 5 + 15 + 23 + 26 / 4

= 69 / 4

=> 17.25ms

Average Response Time:

Response Time = First response – arrival Time

P1 => 0 – 0 = 0

P2 => 5 – 0 = 5

P3 => 15 – 0 = 15

P4 => 23 – 0 = 23

Average Response Time = 0 + 5 + 15 + 23 / 4

= 43 / 4

=> 10.75ms

Average Waiting Time = 10.75

Average Turn around Time = 17.25

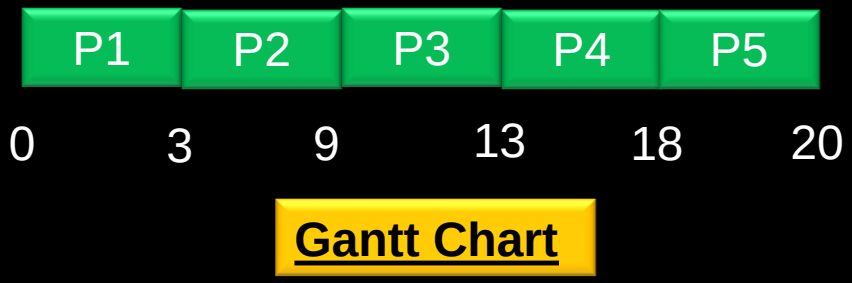
Average Response Time = 10.75

Cons:

- **High Turn around & Waiting time**
- **Low rate of CPU utilization { Bcs of non-preemption}**
- **Short job have to wait for long time, when the CPU is allocate to long jobs**

EXAMPLE - 2

Arrival Time	Process	CPU Time (or) Burst Time
0	P1	3
2	P2	6
4	P3	4
6	P4	5
8	P5	2



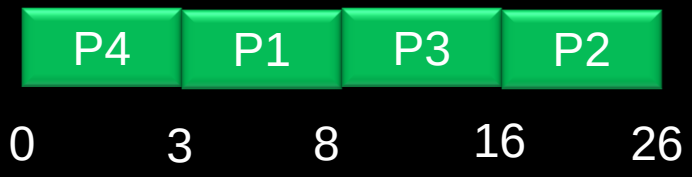
Gantt Chart

Shortest Job First Scheduling [SJF]

- Scheduling is done on the basis of a process having Shortest execution time
- CPU is assigned to the process having smallest Burst Time
- If two process have the same CPU time, then FFCS is used
- SJF Follows Non-Preemptive policy

EXAMPLE - 1

Arrival Time	Process	CPU Time (or) Burst Time
0	P1	5
0	P2	10
0	P3	8
0	P4	3 ← Shortest CPU time



Gantt Chart

Pros:

- Moves a shorter process for execution before a longer process

- Having Least Average waiting time and Turn around time

Cons:

- Big jobs are waiting some much time for CPU (‘AGING’)

Shortest Remaining Time First [SRTF]

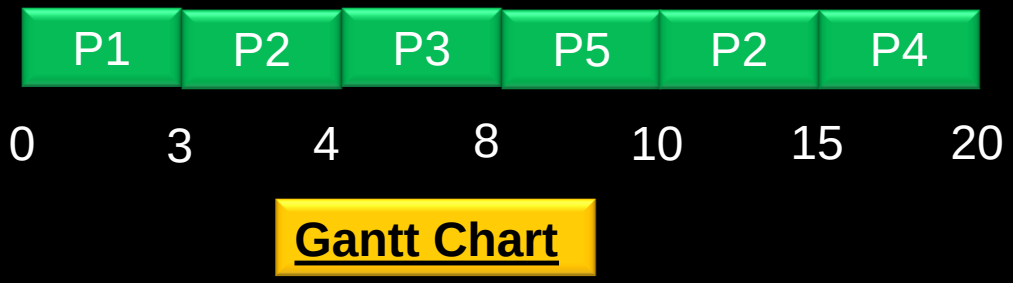
- Policy => Preemptive Scheduling

- Scheduler compare the remaining time of executing process & new process

- Scheduler always selects the process that has Shortest Remaining Time

EXAMPLE - 1

Arrival Time	Process	CPU Time (or) Burst Time
0	P1	3
2	P2	6
4	P3	4
6	P4	5
8	P5	2



Gantt Chart

Priority Scheduling

- Priority is associated with each process
- Scheduler always picks up the highest priority process for execution from Ready Queue
- Priority scheduling can be either :

=> Preemptive

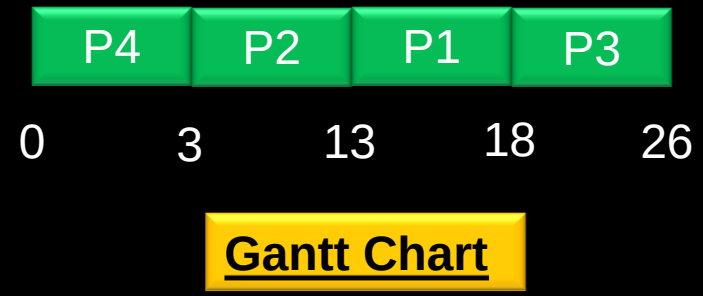
=> Non Preemptive

Cons:

- Low priority jobs are waiting for the CPU for the longest period of the time

EXAMPLE

Priority	Process	CPU Time (or) Burst Time
3	P1	5
2	P2	10
4	P3	8
1	P4	3



Gantt Chart

Round Robin Scheduling

- Designed specially for Time sharing system

- CPU time is divided into Time Slices (or) Time Quantum, Each process is allocated a small time slice

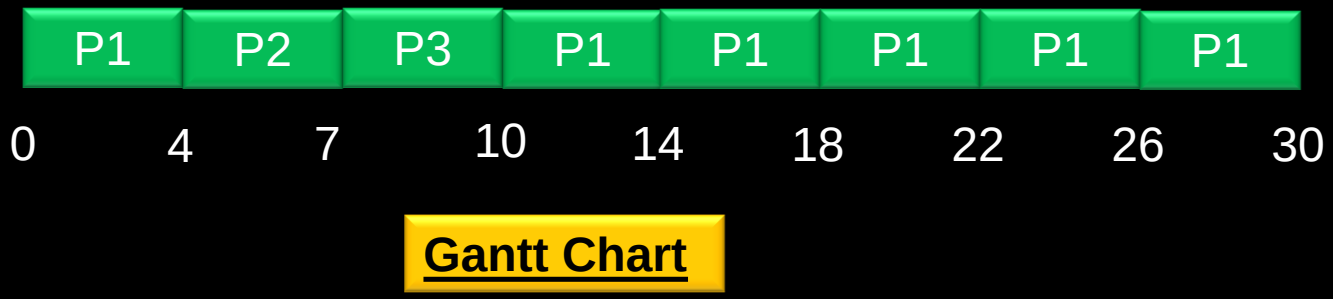
- Time Quantum is generally 10 to 100 ms

- Decision mode: preemptive

EXAMPLE

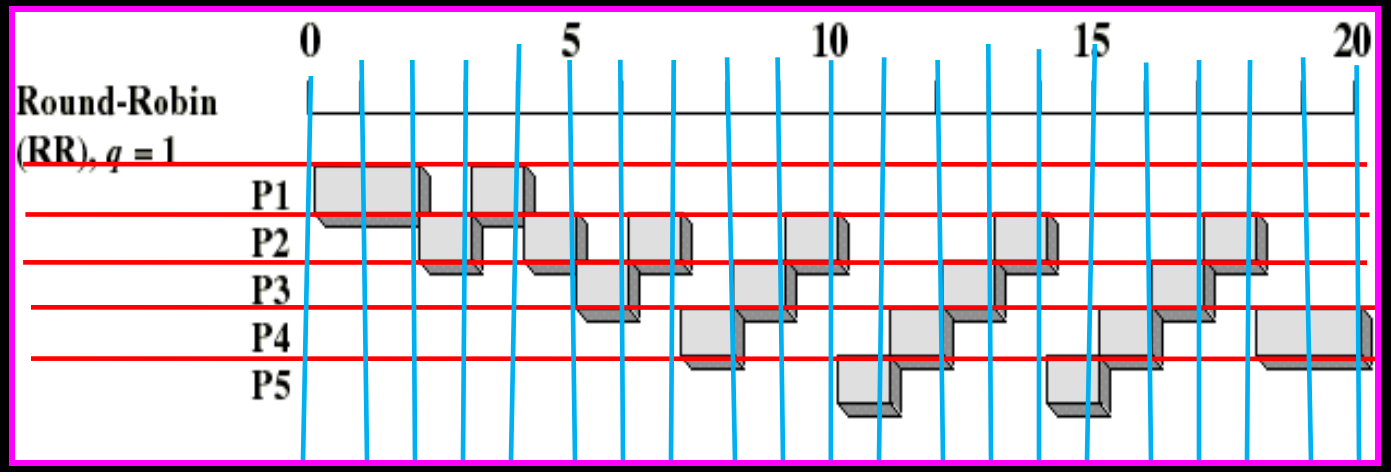
Process	CPU Time (or) Burst Time
P1	24
P2	3
P3	3

Time Quantum = 4 ms



Gantt Chart

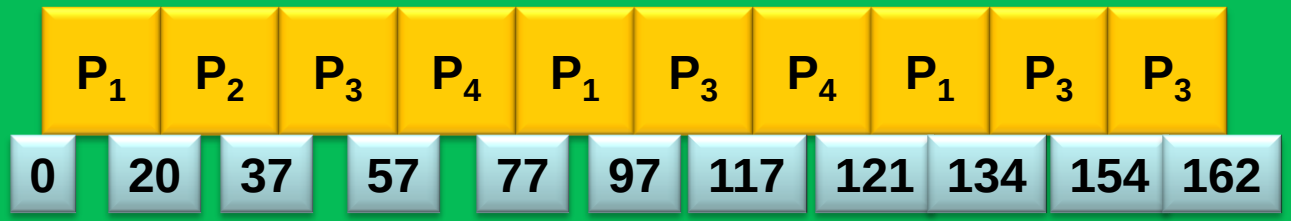
Process	Arrival Time	Service Time
1	0	3
2	2	6
3	4	4
4	6	5
5	8	2



Example of RR with Time Quantum = 20

Process	Burst Time
P_1	53
P_2	17
P_3	68
P_4	24

- The **Gantt chart** is:



- Typically, **higher average turnaround than SJF**, but **better *response*.**

Highest Response Ratio Next [HRRN]

- Process having the highest response ratio, execute job first

- Once a job gets the CPU it runs upto completion

Response Ratio = W + S / S

W = Time spent waiting for the processor

S = Total service time required by the process

Multilevel Feedback Queue

- Preemptive scheduling with dynamic priorities

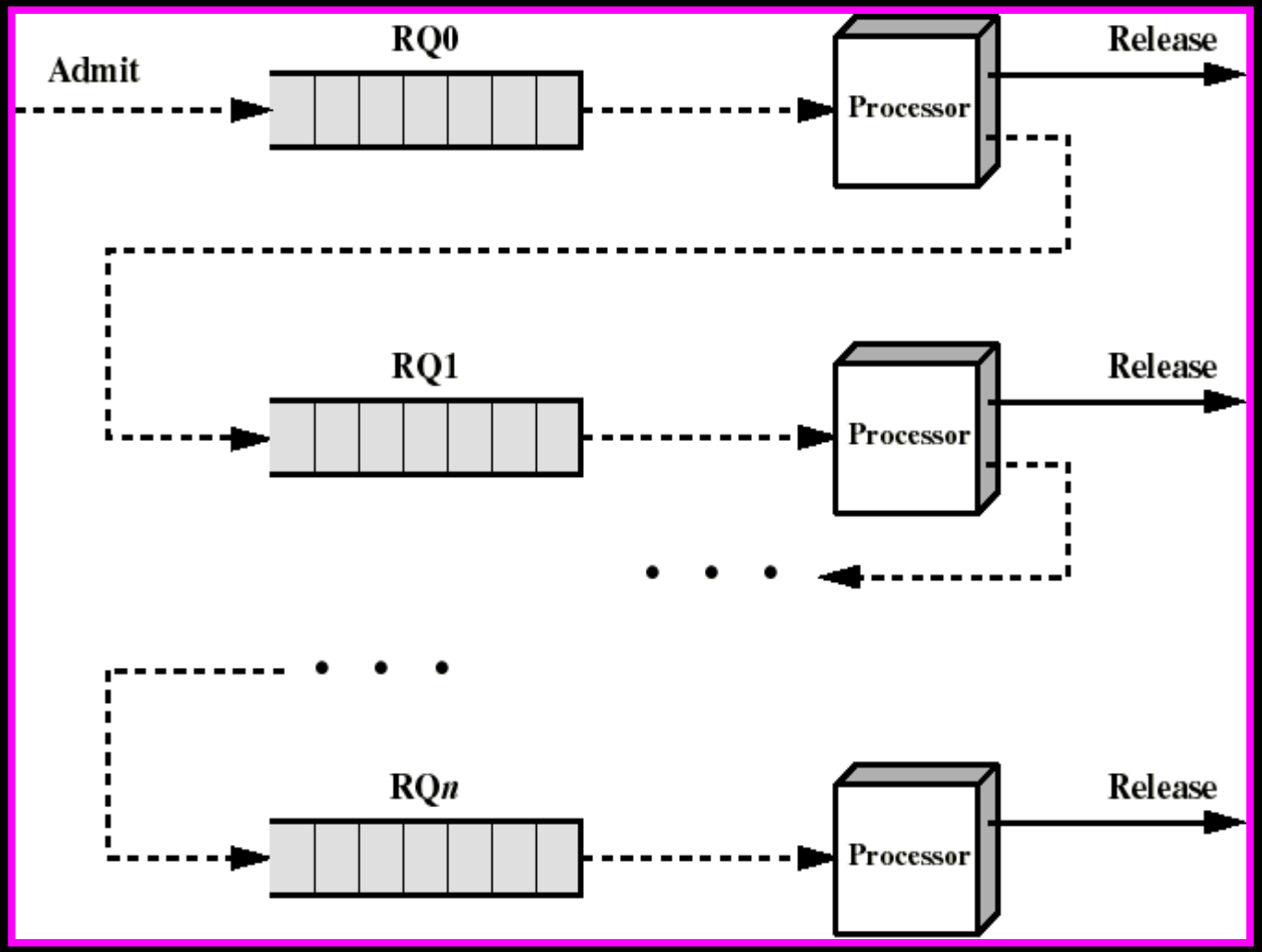
- A process can move between the various queues

- Multilevel-feedback-queue scheduler defined by the following parameters:

  => number of queues

  => scheduling algorithms for each queue

  => method used to determine which queue a process will enter when that process needs service



Example of Multilevel Feedback Queue

Three queues:

*Q*0 – RR with time quantum 8 milliseconds

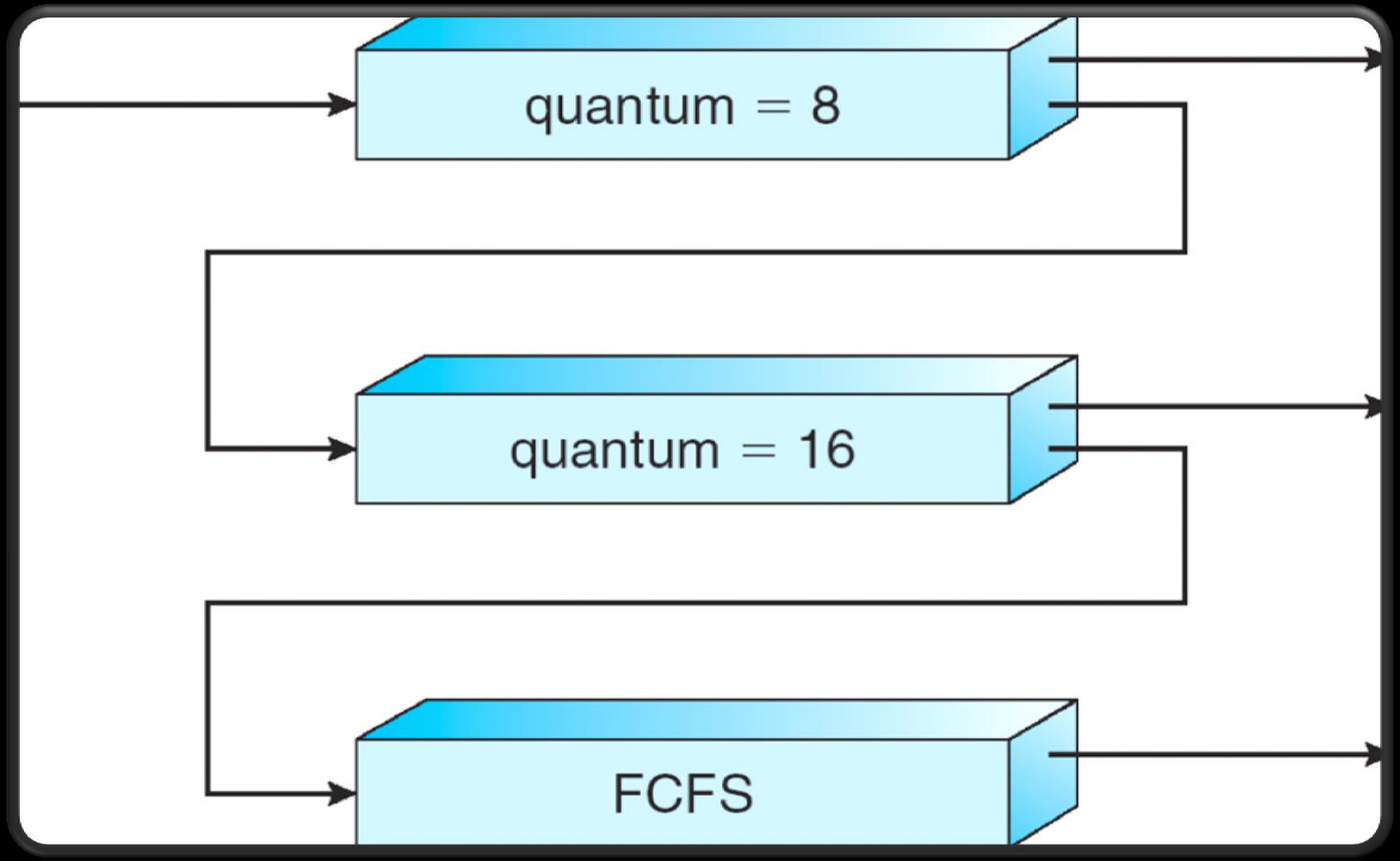
*Q*1 – RR with time quantum 16 milliseconds

*Q*2 – FCFS

Scheduling:

- A new job enters queue *Q0* which is served RR. When it gets CPU, job receives 8 milliseconds. If it does not finish in 8milliseconds, job is moved to queue *Q*1.

- At *Q*1 job is again served RR and receives 16 additional milliseconds. If it still does not complete, it is preempted and moved to queue *Q*2.



Algorithms Comparison

- Which one is best?

- The answer depends on:

=> on the system workload (extremely variable)

=> hardware support for the dispatcher

=> relative weighting of performance criteria (response time, CPU utilization, throughput...).

Hence the answer depends on too many factors to give any...