Lecture - 4

CPU Scheduling –

→ maximum CPU utilization obtained with multiprogramming.

→ 2 types of CPU scheduling,

(I) non-preemtive → [no process will be terminated until it finishes the task.]
↳ [no context switching]

(II) preemtive → [terminate processes if needed.]
↳ [multiple context switching]

{{context switching is called Preemtive}}

- CPU utilization: keeps the CPU as [busy] as possible.
- Turnaround time: time of a process from the ~~arrival~~ [arrival in the ready queue] to the [termination].

~~waiting~~

- Waiting time: time of a process to wait in ready queue. Time spent in the Ready queue

- Response time: Time spent from arrival in the ready queue to get the first CPU allocation.

- Throughput: # of processes that complete their execution per unit time.

⇒ Scheduling Algorithm Optimization Criteria

- → Max CPU utilization
- → Max throughput
- → Min turnaround time.
- → Min waiting time.
- → Min response time.

Algo 1: First Come First Serve (FCFS)

↳ only non-preemtive

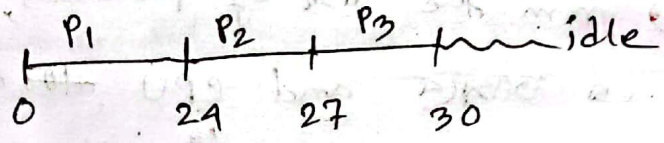
↳ no context switching

↳ gives CPU time serially

FCFS scheduling:

process	Brust time	Arrival time (if not given)
P_1	24	0
P_2	3	0
P_3	3	0

the Graanft chart:



time

0 ← P_1(24), P_2(3), P_3(3)

24 ← P_2(3), P_3(3)

27 ← P_3(3)

30 ← empty

Turnaround time	waiting time	response time	# of context switching
$P_1 = (24-0) = 24$	$P_1 = (0-0) = 0$	$P_1 = (0-0) = 0$	0
$P_2 = (27-0) = 27$	$P_2 = (24-0) = 24$	$P_2 = (24-0) = 24$	
$P_3 = (30-0) = 30$	$P_3 = (27-0) = 27$	$P_3 = (27-0) = 27$	
Avg = 27	Avg = 17	Avg = 17	

So, for non-preemtive, waiting time and response time is Same.

→ Convoy Effect: Long Process অনেক Time নেয়, so small Process এ পর্যায়ে RAM থেকে Remove হতে পারে। (only in non-preemtive)

Algo-2 : Shortest Job First (SJF)

↳ non-preemtive
↳ preemtive [SRJF / SRTF / SRPF]

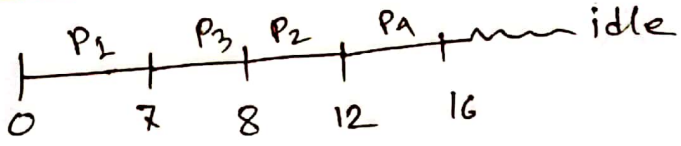
for non-preemtive, from the list of processes, short process আগে RAM এ আসবে and CPU allocation পাবে। the process will finish execution then আবার shortest process select হবে।

for preemtive, allocation এর আগে দেখবে কে shortest. Then ওই process CPU allocation পাবে। Then অন্য process RAM এ আসলেন context switch হবে and ও check করবে shortest কিনা।

SJF Scheduling: (non-preemtive)

process	arrival time	Brust time
P_1	0	7
P_2	2	4
P_3	4	1
P_4	5	4

Granth Chart:



RAM

0 ← P_1 (7)

7 ← P_2 (4), P_3 (1), P_4 (4)

8 ← P_2 (4) ; P_4 (4)

12 ← P_4 (4)

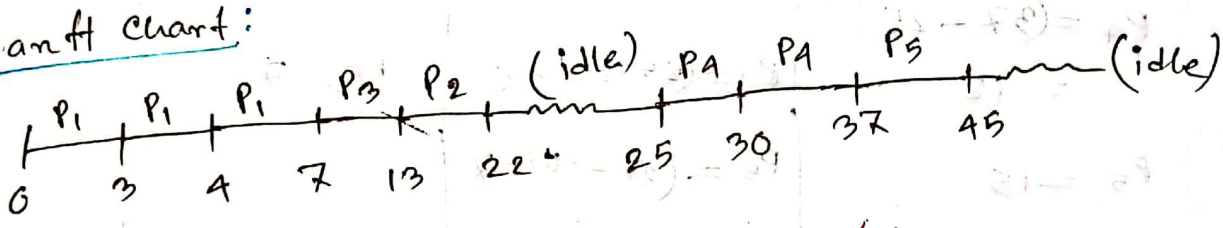
16 ← empty

Turnaround time	waiting time	Response time	# of context switch
$P_1 = (7-0) = 7$	$P_1 = (0-0) = 0$	$P_1 = 0$	0
$P_2 = (16-2) = 14$	$P_2 = (12-2) = 10$	$P_2 = 10$	
$P_3 = (8-4) = 4$	$P_3 = (7-4) = 3$	$P_3 = 3$	
$P_4 = (12-5) = 7$	$P_4 = (8-5) = 3$	$P_4 = 3$	
Avg = 8	Avg = 4	Avg = 4	

SRJF scheduling : (preemtive)

process	arrival time	brust time
P_1	0	7
P_2	3	9
P_3	4	6
P_4	25	12
P_5	30	8

Grantt chart:



0 ← P_1(7)

3 ← P_1(4), P_2(9)

4 ← P_1(3), P_2(9), P_3(6)

7 ← P_2(9), P_3(6)

13 ← P_2(9)

25 ← P_4(12)

30 ← P_4(7), P_5(8)

37 ← P_5(8)

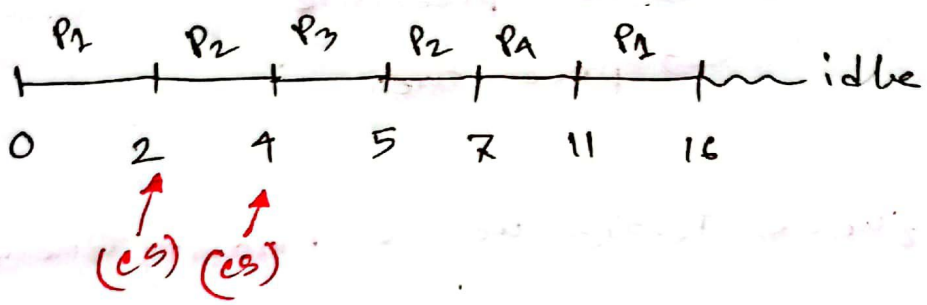
45 ← Empty

Turnaround time	waiting time	Response time	# of cs
$P_1 = (7-0) = 7$ $P_2 = 19$ $P_3 = 9$ $P_4 = (37-25) = 12$ $P_5 = 15$	$P_1 = (0-0) + (3-3) + (4-4) = 0$ $P_2 = (13-3) = 10$ $P_3 = (7-4) = 3$ $P_4 = (25-25) = 0$ $P_5 = (37-30) = 7$	$P_1 = 0$ $P_2 = (13-3) = 10$ $P_3 = 3$ $P_4 = 0$ $P_5 = 7$	0
avg = 12.5	Avg = 4	avg = 4	

⇒

Process	arrival time	Brust time
P_1	0	7
P_2	2	4
P_3	4	1
P_4	5	4

Gantt Chart:



0 ← P_1 (7)

2 ← P_1 (5), P_2 (4)

4 ← P_1 (5), P_2 (2), P_3 (1)

5 ← P_1 (5), P_2 (2), P_4 (4)

7 ← P_1 (5), P_4 (4)

11 ← P_1 (5)

16 ← empty.

Turnaround time	# of context switch
$P_1 = (16-0) = 16$ $P_2 = (7-2) = 5$ $P_3 = (5-4) = 1$ $P_4 = (11-5) = 6$	2
Avg = 7	

waiting time	response time
$P_1 = (0-0) + (11-2) = 9$ $P_2 = (2-2) + (5-4) = 1$ $P_3 = (4-4) = 0$ $P_4 = (7-5) = 2$	$P_1 = (0-0) = 0$ $P_2 = (2-2) = 0$ $P_3 = (4-4) = 0$ $P_4 = (7-5) = 2$
Avg = 3	Avg = 0.5

Algo - 3 : Priority Queue

↳ non-preemtive.
↳ preemtive.

ready queue হিসেবে we use Priority Queue here.

priority can be given
↳ internally
↳ externally

factor → time interval, mem requirement, the number of open files.

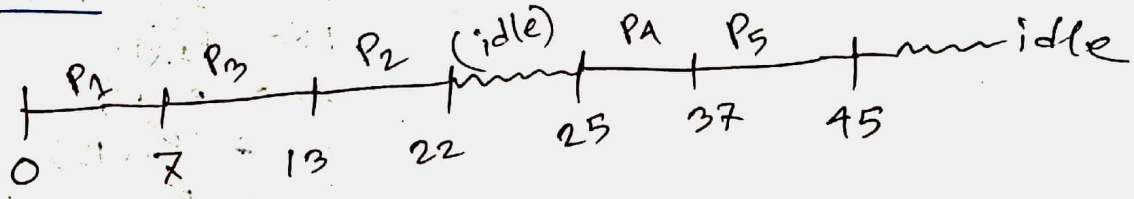
drawback → starvation (high priority process gets the CPU allocation first. So, low priority processes get terminated from the RAM before executing.)

solution → Aging.

Non-preemtive Priority Scheduling:

process	arrival time	Brust time	Priority
P_1	0	7	5
P_2	3	9	3
P_3	4	6	2
P_4	25	12	4
P_5	30	8	1

Gantt chart:



# of context switching = 0

calculate the times

$0 \leftarrow P_1^5$ (7)

$7 \leftarrow P_2^3(9), P_3^2(6)$

$13 \leftarrow P_2^3$ (9)

$22 \leftarrow$ empty

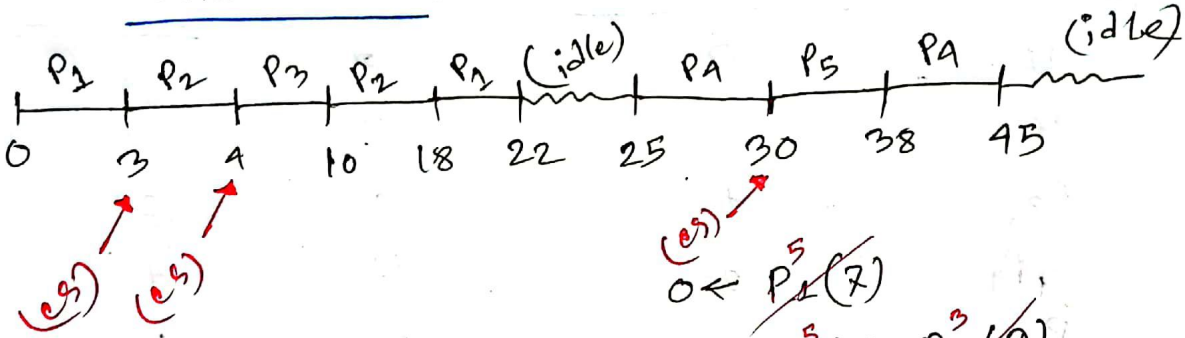
$25 \leftarrow P_4^4$ (12)

$37 \leftarrow P_5^1$ (8)

$45 \leftarrow$ empty

Preemtive Priority Scheduling:

Gantt Chart:



0 ← P_1^5(7)

3 ← P_1^5(4), P_2^3(9)

4 ← P_1^5(4), P_2^3(8), P_3^2(6)

10 ← P_1^5(4), P_2^3(8)

18 ← P_1^5(4)

22 ← empty

25 ← P_4^1(12)

30 ← P_4^1(7), P_5^1(8)

38 ← P_4^1(7)

45 ← empty

Turnaround time	waiting time (RS)	response time	# of context switching
$P_1 = (22-0) = 22$	$P_1 = (0-0) + (18-3) = 15$	$P_1 = (0-0) = 0$	3
$P_2 = (18-3) = 15$	$P_2 = (3-3) + (10-4) = 6$	$P_2 = (3-3) = 0$	
$P_3 = (10-4) = 6$	$P_3 = (4-4) = 0$	$P_3 = (4-4) = 0$	
$P_4 = (45-25) = 20$	$P_4 = (25-25) + (38-30) = 8$	$P_4 = (25-25) = 0$	
$P_5 = (38-30) = 8$	$P_5 = (30-30) = 0$	$P_5 = (30-30) = 0$	
Avg = 14.2	Avg = 5.8	Avg = 0	

Algo-4 : Round Robin (RR)

* have to maintain the serial in the RAM

↳ only preemtive

↳ sets a Time Quantum (T_q)

This algo is more suitable because the context switching occurs in every time quantum. After the T_q overs, the algo makes context switching and gives another process CPU time. For this reason there is no Starvation occurs and every processes get CPU time.

→ Context switch can be count by observing T_q period in the Graph.

→ no process needs to wait more than $(n-1)_q$ time.

→ if T_q is Large, then No context switch.
↳ acts like FCFS

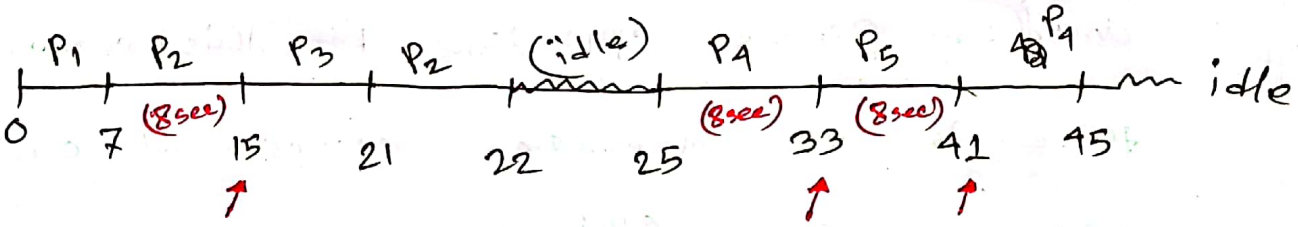
→ if T_q is small, then Rapid context switch and CPU overhead increases.

Round Robin Scheduling:

Process	Arrival time	Brust time
P_1	0	7
P_2	3	9
P_3	4	6
P_4	25	12
P_5	30	8

$T_2 = 8$

Gantt chart:



0 ← P_1 (7)

7 ← P_2 (9), P_3 (6)

15 ← P_3 (6), P_2 (1)

21 ← P_2 (1)

22 ← empty

25 ← P_4 (12)

33 ← P_5 (8), P_4 (4)

41 ← P_4 (4)

45 ← empty

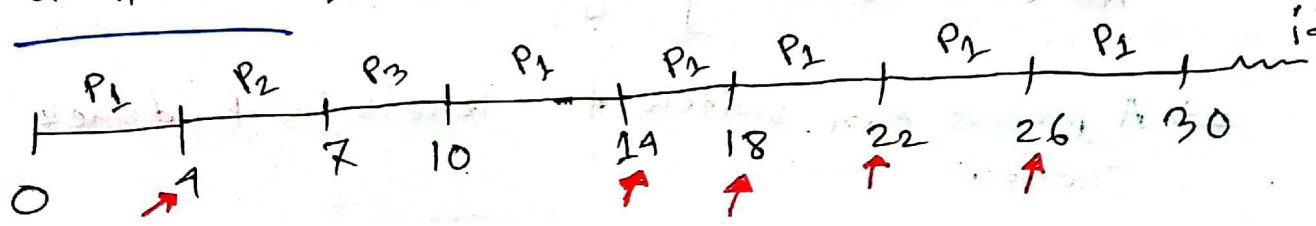
Turnaround time	waiting time	response time
$P_1 = (7-0) = 7$ $P_2 = (22-3) = 19$ $P_3 = (21-4) = 17$ $P_4 = (45-25) = 20$ $P_5 = (41-30) = 11$	$P_1 = (0-0) = 0$ $P_2 = (7-3) + (21-15)$ $= 10$ $P_3 = (15-4) = 11$ $P_4 = (25-25) + (41-33)$ $= 8$ $P_5 = (33-30) = 3$	$P_1 = (0-0) = 0$ $P_2 = (7-3) = 4$ $P_3 = (15-4) = 11$ $P_4 = (25-25) = 0$ $P_5 = (33-30) = 3$
Avg = 14.8	Avg = 6.4	Avg = 3.6

⇒

process	Brust time	Arrival time
P_1	24	0
P_2	3	6
P_3	3	0

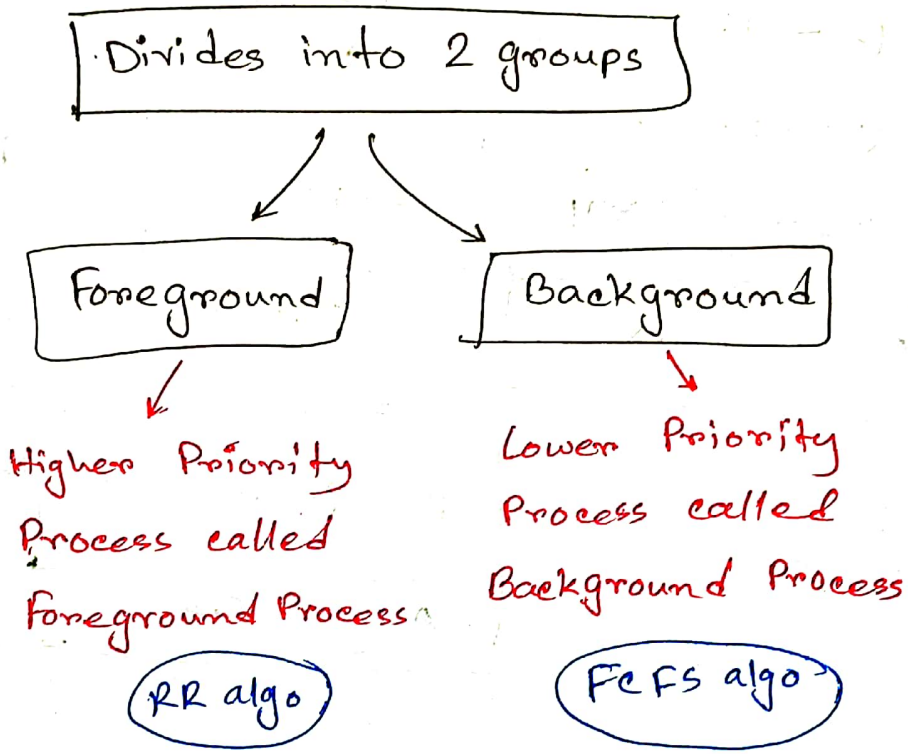
$(T_Q = 4)$

Gantt Chart:

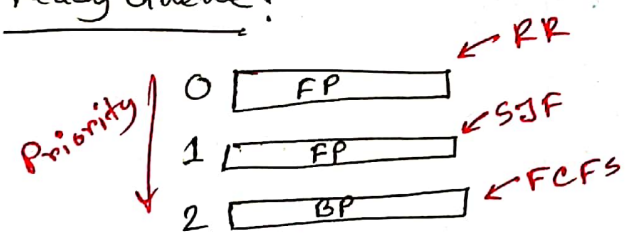


Multilevel Queue Scheduling:

↳ divide the Ready Queue to multiple levels.



Ready Queue:



→ Why using multilevel? Ans: bcos, every queue has its own scheduling algorithm.

⟹ A process can switch the levels in Feedback queue.