CPU Scheduling

CPU Scheduler

- Short-term scheduler selects from among the processes in ready queue, and allocates the CPU to one of them
  - Queue may be ordered in various ways
- CPU scheduling decisions may take place when a process:
  1. Switches from running to waiting state
  2. Switches from running to ready state
  3. Switches from waiting to ready
  4. Terminates
- Scheduling under 1 and 4 is nonpreemptive
- All other scheduling is preemptive

Dispatcher

- Dispatcher module gives control of the CPU to the process selected by the short-term scheduler; this involves:
  - switching context
  - switching to user mode
  - jumping to the proper location in the user program to restart that program
- Dispatch latency – time it takes for the dispatcher to stop one process and start another running

Types of Scheduling Algorithms

- PREEMPTIVE SCHEDULING: The resources are allocated to a process for a limited time.
- Process can be interrupted in between.

- NON PREEMPTIVE SCHEDULING: Process can not be interrupted till it terminates or switches to waiting state.

Scheduling Criteria

- CPU utilization – keep the CPU as busy as possible
- Throughput – # of processes that complete their execution per time unit
- Turnaround time – amount of time to execute a particular process
- Waiting time – amount of time a process has been waiting in the ready queue
- Response time – amount of time it takes from when a request was submitted until the first response is produced, not output

Scheduling Algorithm Optimization Criteria

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

Parameter related to Process

- Arrival Time: arrives in the ready queue
- Burst Time : amount of time required to finish process.
- Completion Time: amount of time up to termination time.
- Turnaround Time (TAT): amount of time taken to fulfill a request.

$$TAT = CT - AT$$

- Waiting Time: The time processes spend in the Ready Queue Waiting their turn to get on the CPU.

$$WT = TAT - BT$$

- Response Time: It is the time taken in an interactive program.

First- Come, First-Served (FCFS) Scheduling

- Jobs are executed on first come, first serve basis.
- It is a non-preemptive
- Easy to understand and implement.
- Its implementation is based on FIFO queue.
- Poor in performance as average wait time is high.

First- Come, First-Served (FCFS) Scheduling

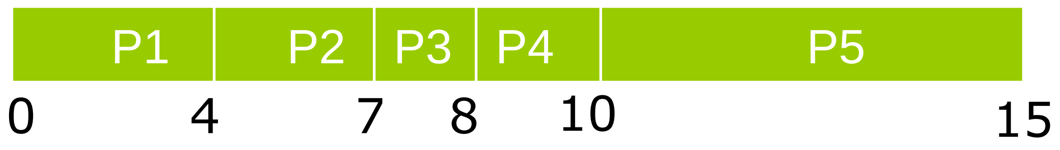
- Solve given example

P.No	A.T	B.T
1	0	4
2	1	3
3	2	1
4	3	2
5	4	5

First- Come, First-Served (FCFS) Scheduling

P.No	A.T	B.T
1	0	4
2	1	3
3	2	1
4	3	2
5	4	5

Gantt chart

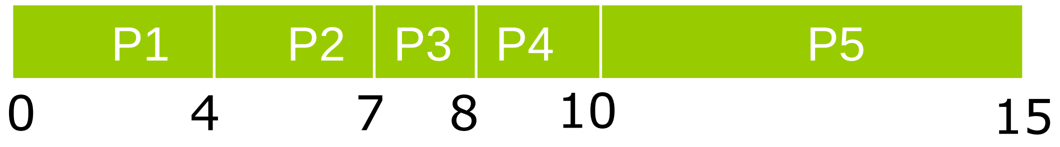


First- Come, First-Served (FCFS) Scheduling

Calculate Completion Time,

P.No	A.T	B.T	CT	TAT = (CT- AT)	WT = (TAT – B.T)
1	0	4	4	4	0
2	1	3	7	6	3
3	2	1	8	6	5
4	3	2	10	7	5
5	4	5	15	11	6

Gantt chart



- Avg TAT = (4+6+6+7+11)/5 = ?
- Avg WT = (0+3+5+5+6)/5 = ?

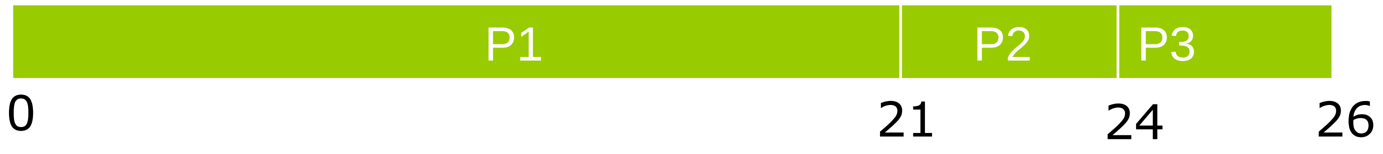
Convoy effect

- Convoy effect - short process behind long process.
- As FCFS is a non-preemptive scheduling algorithm, the CPU will be allocated to a process until it get finished, that means other processes have to wait for their turn.
- This lead to a situation called convoy effect.
- Example: VIP visit and other cars passing behind them (normal person in waiting)

Convoy effect

P.No	A.T	B.T
1	0	21
2	1	3
3	1	2

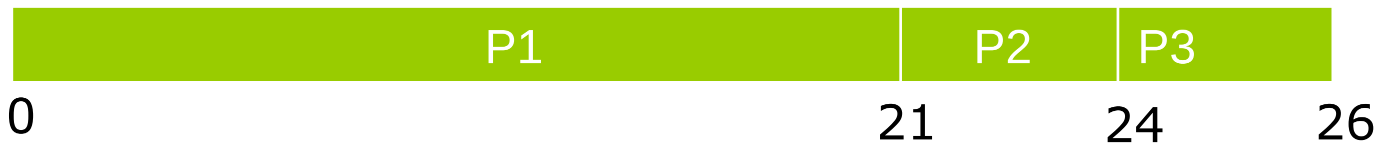
Gantt chart



Convoy effect

P.No	A.T	B.T	C.T	TAT	WT
1	0	21	21	21	0
2	1	3	24	23	20
3	1	2	26	25	23

Gantt chart



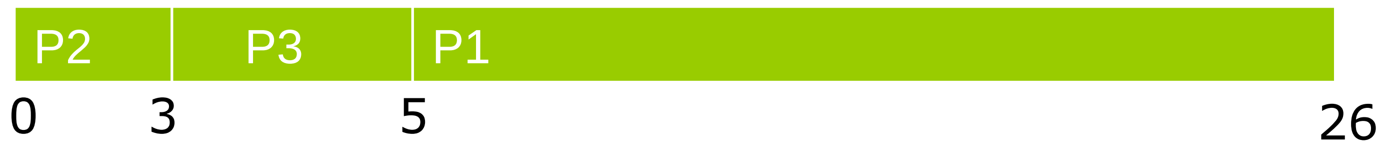
Avg WT =(0+20+23)/3 = 43/3

Convoy effect

Same process in different order

P.No	A.T	B.T	CT	TAT	WT
1	1	21	26	25	4
2	0	3	3	3	0
3	0	2	5	5	3

Gantt chart



Avg WT =(4+0+3)/3 = 7/3

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
- SJF is optimal – gives minimum average waiting time for a given set of processes
  - The difficulty is knowing the length of the next CPU request

Shortest-Job-First (SJF) Scheduling

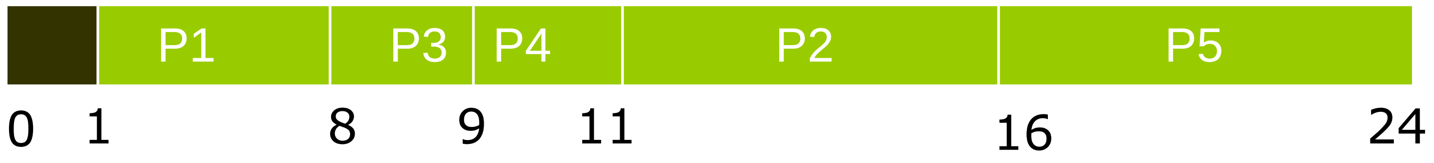
- Solve given example : non preemptive

P.No	A.T	B.T
1	1	7
2	2	5
3	3	1
4	4	2
5	5	8

Shortest-Job-First (SJF) Scheduling

P.No	A.T	B.T
1	1	7
2	2	5
3	3	1
4	4	2
5	5	8

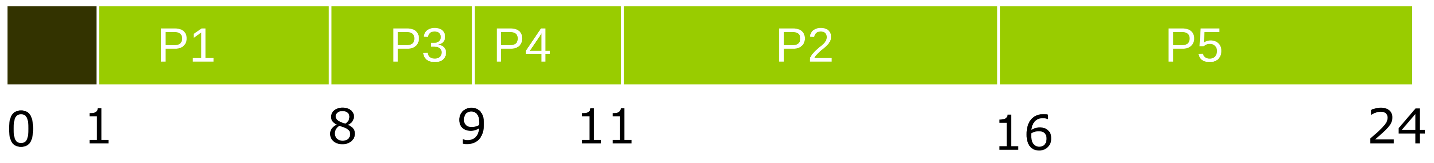
Gantt chart



Shortest-Job-First (SJF) Scheduling

P.No	A.T	B.T	CT	TAT	WT
1	1	7	8	7	0
2	2	5	16	14	9
3	3	1	9	6	5
4	4	2	11	7	5
5	5	8	24	19	11

Gantt chart



AVG TAT: ?

AVG WT: ?

shortest-remaining-time-first

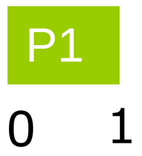
- Shortest remaining time scheduling is the preemptive

P.No	A.T	B.T
1	0	7
2	1	5
3	2	3
4	3	1
5	4	2
6	5	1

shortest-remaining-time-first

P.No	A.T	B.T
1	0	7
2	1	5
3	2	3
4	3	1
5	4	2
6	5	1

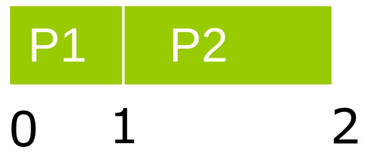
Gantt chart



shortest-remaining-time-first

P.No	A.T	B.T
1	0	~~7~~, 6
2	1	5
3	2	3
4	3	1
5	4	2
6	5	1

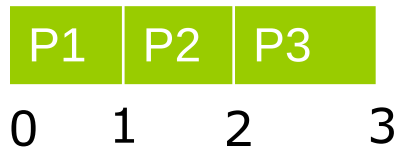
Gantt chart



shortest-remaining-time-first

P.No	A.T	B.T
1	0	6
2	1	4
3	2	3
4	3	1
5	4	2
6	5	1

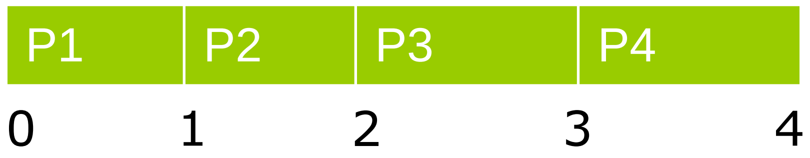
Gantt chart



shortest-remaining-time-first

P.No	A.T	B.T
1	0	6
2	1	4
3	2	2
4	3	1
5	4	2
6	5	1

Gantt chart

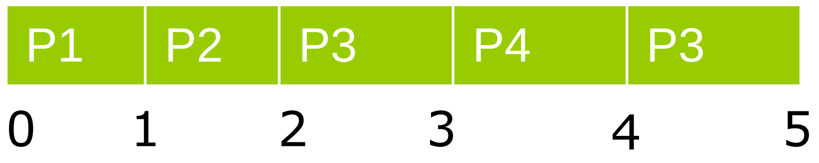


shortest-remaining-time-first

P.No	A.T	B.T
1	0	6
2	1	4
3	2	2
5	4	2
6	5	1

NOW p3 and p5 both have same time : pick based on A.T, so p3 is picked

Gantt chart

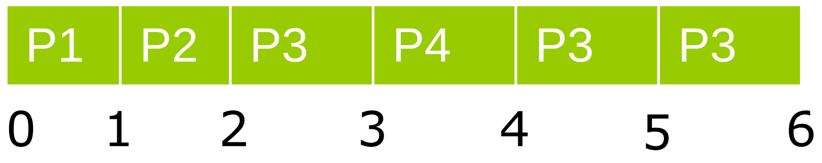


shortest-remaining-time-first

P.No	A.T	B.T
1	0	6
2	1	4
3	2	1
5	4	2
6	5	1

NOW p3 and p6 both have same time : pick based on A.T, so p3 is picked

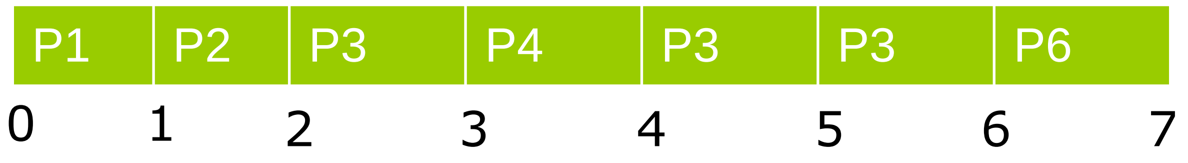
Gantt chart



shortest-remaining-time-first

P.No	A.T	B.T
1	0	6
2	1	4
5	4	2
6	5	1

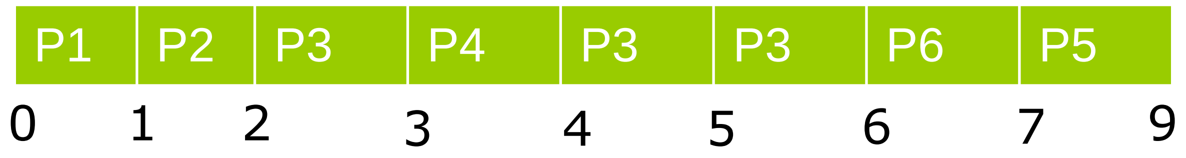
Gantt chart



shortest-remaining-time-first

P.No	A.T	B.T
1	0	6
2	1	4
5	4	2

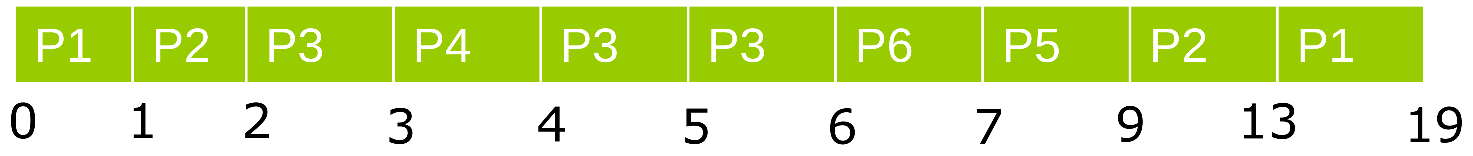
Gantt chart



shortest-remaining-time-first

P.No	A.T	B.T
1	0	6
2	1	4

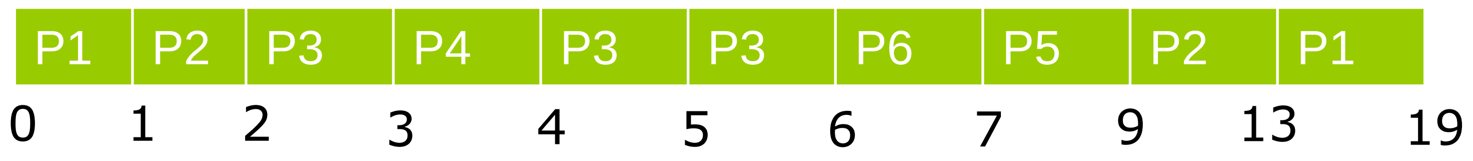
Gantt chart



shortest-remaining-time-first

P.No	A.T	B.T	CT	TAT	WT
1	0	7	19	19	12
2	1	5	13	12	7
3	2	3	6	4	1
4	3	1	4	1	0
5	4	2	9	5	3
6	5	1	7	2	1

Gantt chart



AVG TAT:?

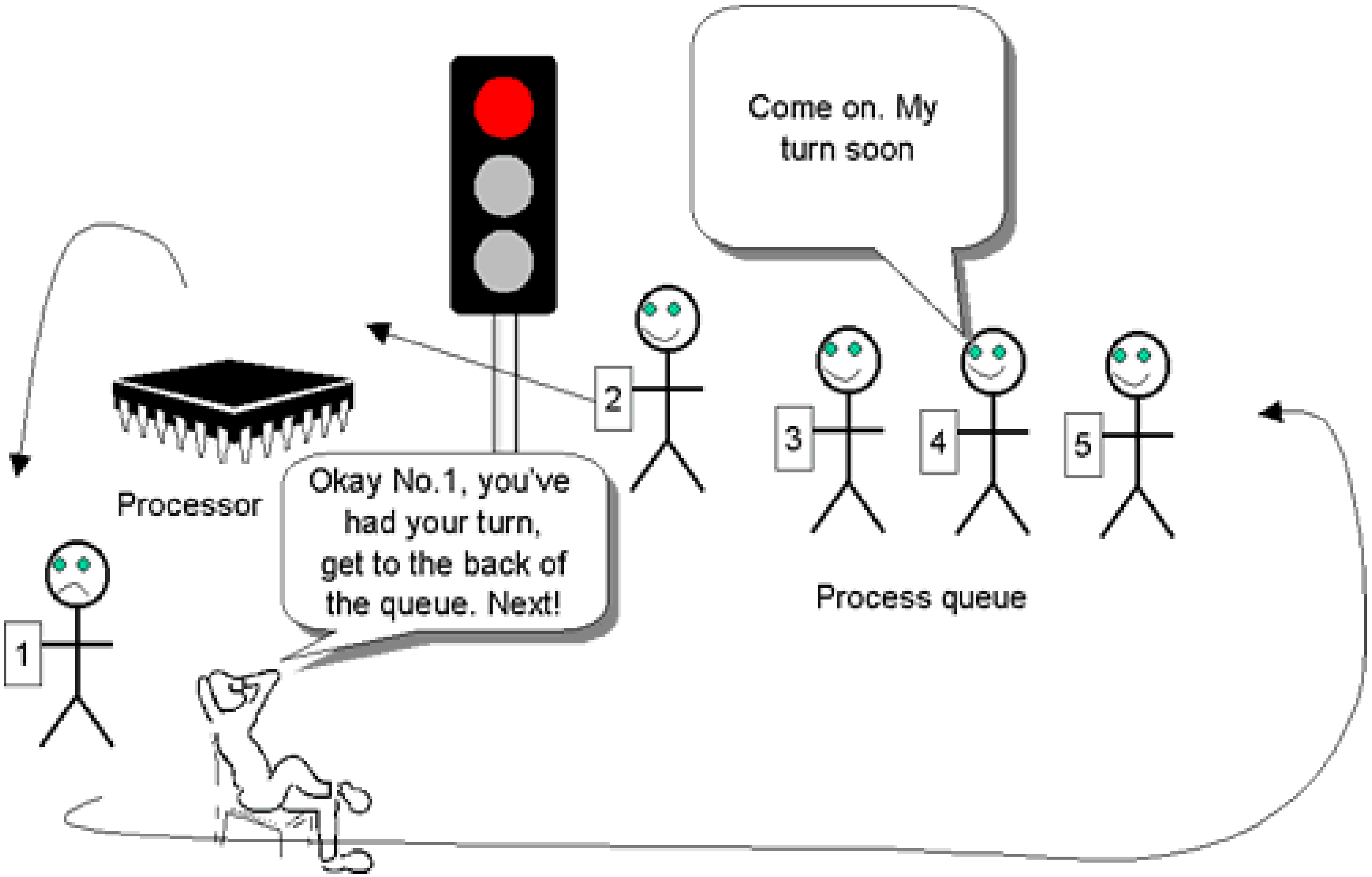
AVG WT:?

Round Robin (RR)

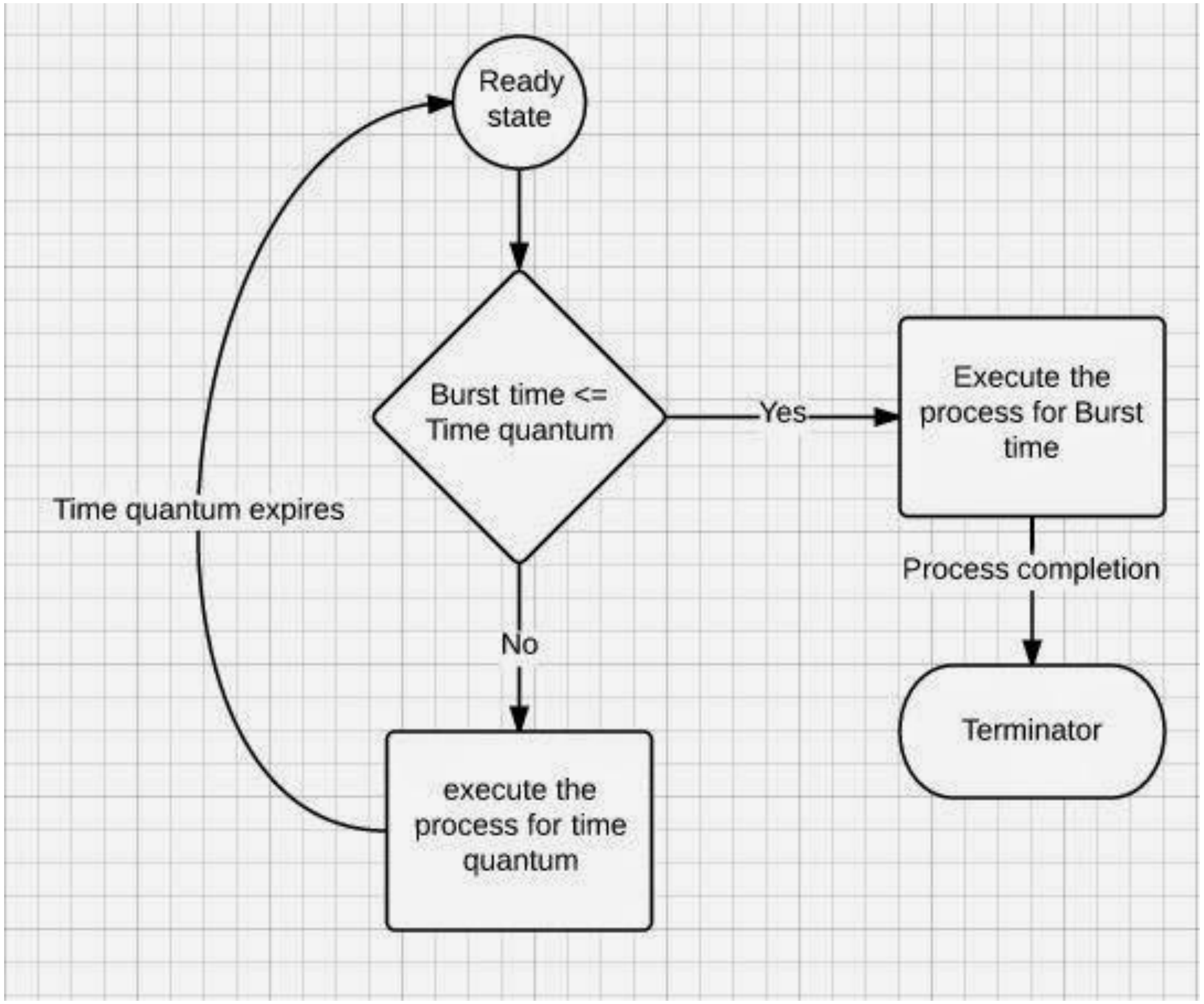
- Each process gets a small unit of CPU time (time quantum *q*)
- After this time has elapsed, the process is preempted and added to the end of the ready queue.
- Timer interrupts every quantum to schedule next process

- Performance
  - *q* large $\Rightarrow$ FIFO
  - *q* small $\Rightarrow$ *q* must be large with respect to context switch, otherwise overhead is too high

Round Robin (RR)


Come on. My turn soon
Okay No.1, you've had your turn, get to the back of the queue. Next!
Processor
Process queue
1
2
3
4
5

Round Robin (RR) Algorithm


Ready state
Burst time <= Time quantum
Yes
Execute the process for Burst time
Process completion
Terminator
No
execute the process for time quantum
Time quantum expires

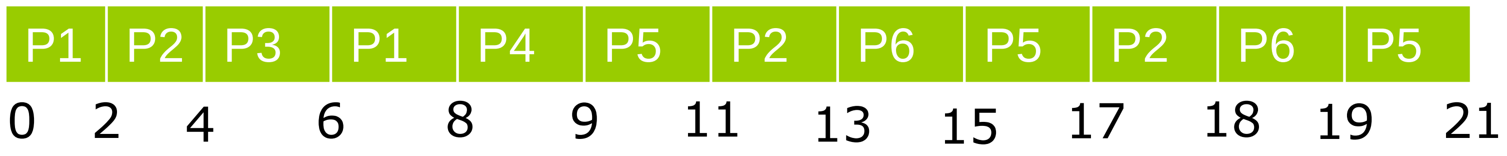
Round Robin (RR) Example

TQ= 2

P.No	A.T	B.T
1	0	4
2	1	5
3	2	2
4	3	1
5	4	6
6	6	3

Always look at arrival time (maintain queue for next process selection) and TQ both for process selection

Gantt chart

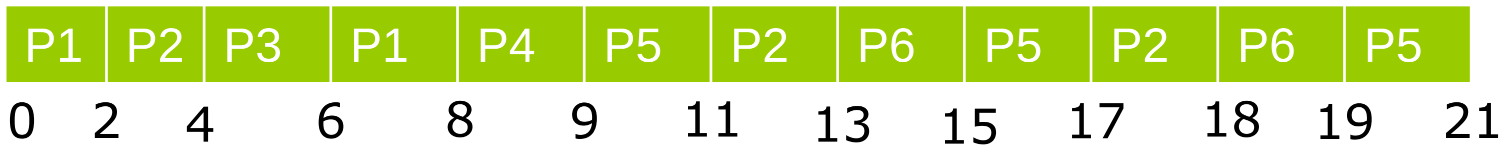


Round Robin (RR) Example

TQ= 2

P.No	A.T	B.T	CT	TAT	WT
1	0	4	8	8	4
2	1	5	18	17	12
3	2	2	6	4	2
4	3	1	9	6	5
5	4	6	21	17	11
6	6	3	19	13	10

Gantt chart



AVG TAT:?

AVG WT:?

Round Robin (RR) Example

What if we have TQ= 4 and TQ= 1 ?

P.No	A.T	B.T
1	0	4
2	1	5
3	2	2
4	3	1
5	4	6
6	6	3

TQ=1 provides more context switching

TQ= 4 might lead to starvation for few processes as per TQ increase

Longest-Job-First (LJF) Scheduling

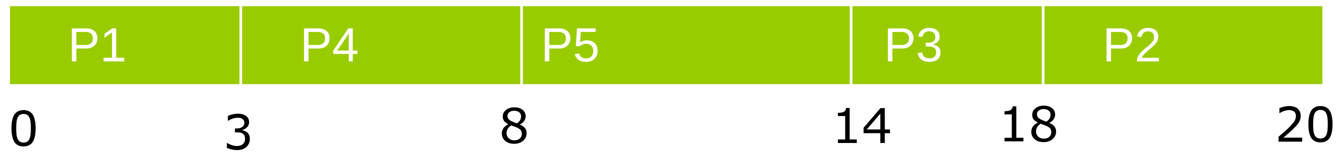
- Process with longest B.T selected
- Solve given example : non preemptive

P.No	A.T	B.T
1	0	3
2	1	2
3	2	4
4	3	5
5	4	6

Longest-Job-First (LJF) Scheduling

P.No	A.T	B.T
1	0	3
2	1	2
3	2	4
4	3	5
5	4	6

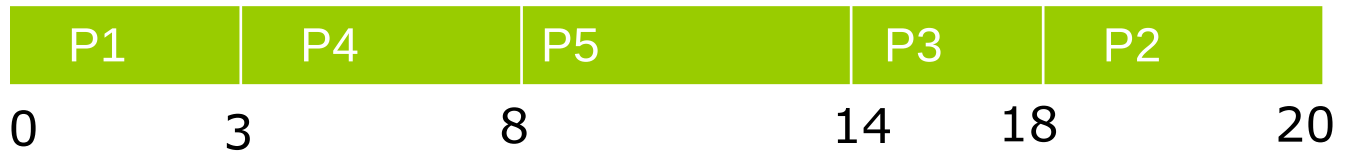
Gantt chart



Longest-Job-First (LJF) Scheduling

P.No	A.T	B.T	CT	TAT	WT
1	0	3	3	3	0
2	1	2	20	19	17
3	2	4	18	16	12
4	3	5	8	5	0
5	4	6	14	10	4

Gantt chart



AVG TAT: ?

AVG WT: ?

HRRN

- Highest Response Ratio Next (Non-Preemptive)
- Given n processes with their Arrival times and Burst times, the task is to find average waiting time and average turn around time using HRRN scheduling algorithm.
- The name itself states that we need to find the response ratio of all available processes and select the one with the highest Response Ratio.
- A process once selected will run till completion.
- Response Ratio = (W + S)/S
- Here, W is the waiting time of the process so far and S (service) is the Burst time of the process.

Performance of HRRN **–**

- Shorter Processes are favored.
- Aging without service increases ratio, longer jobs can get past shorter jobs.

HRRN

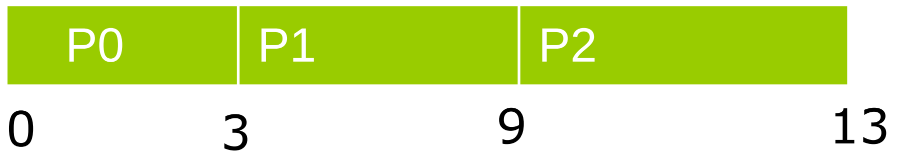
P.No	A.T	B.T
0	0	3
1	2	6
2	4	4
3	6	5
4	8	2

- P0 will run first till completion
- Only P1 is available after p0 completion.
- Run p1 till completion
- Now all p2 p3 and p4 available so which one should select for run based on response ratio
- P2 = come at 4 and wait till 9, so wait time is w2= 9-4=5
- So RR(p2) = W+S/S = 5+4/4=2.25
- P3 = w3 = 3
- So RR(p3) = 3+5/5=1.6
- P4 = w4 =1
- So , RR(p4) = 1+2/2 = 1.5
- So, p2 will pick based on highest RR.

(HRRN) Scheduling

P.No	A.T	B.T
0	0	3
1	2	6
2	4	4
3	6	5
4	8	2

Gantt chart

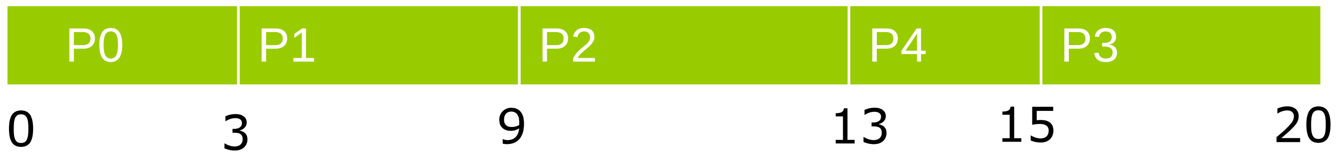


- Now again calculate RR
- At P3 and P4 RR(p3) = 7+5/5 =2.54
- RR(p4) = 5+2/2=3.5
- So, p4 selected based on highest RR

(HRRN) Scheduling

P.No	A.T	B.T
0	0	3
1	2	6
2	4	4
3	6	5
4	8	2

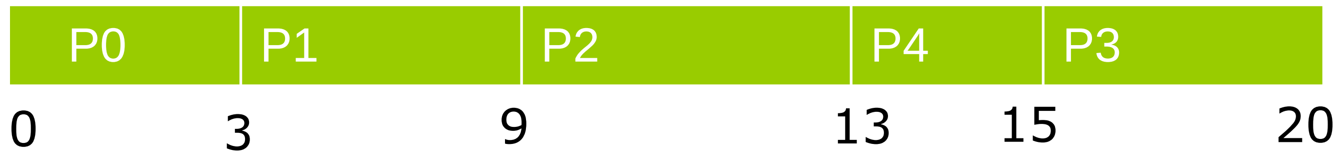
Gantt chart



(HRRN) Scheduling

P.No	A.T	B.T	CT	TAT	WT	RT
0	0	3	3	3	0	0
1	2	6	9	7	1	1
2	4	4	13	9	5	5
3	6	5	20	14	9	9
4	8	2	15	7	5	5

Gantt chart



Priority Scheduling

- A priority number (integer) is associated with each process
- The CPU is allocated to the process with the highest priority
  - Preemptive
  - Non preemptive
- Problem ≡ Starvation – low priority processes may never execute
- Solution ≡ Aging – as time progresses increase the priority of the process

Priority Scheduling

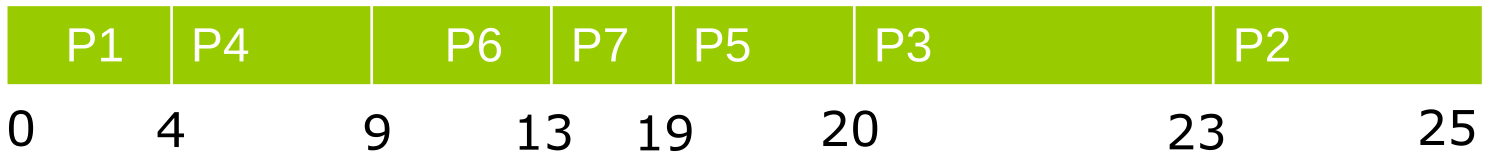
- Solve given example : non preemptive
- highest number = highest priority

P.No	Priority	A.T	B.T
1	2	0	4
2	4	1	2
3	6	2	3
4	10	3	5
5	8	4	1
6	12	5	4
7	9	6	6

Priority Scheduling

P.No	Priority	A.T	B.T
1	2	0	4
2	4	1	2
3	6	2	3
4	10	3	5
5	8	4	1
6	12	5	4
7	9	6	6

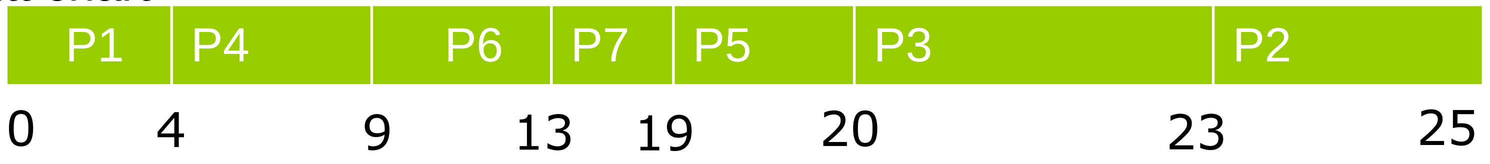
Gantt chart



Priority Scheduling

P.No	Priority	A.T	B.T	CT	TAT	WT
1	2	0	4	4	4	0
2	4	1	2	25	24	22
3	6	2	3	23	21	18
4	10	3	5	9	6	1
5	8	4	1	20	16	15
6	12	5	4	13	8	4
7	9	6	6	19	13	7

Gantt chart



Priority Scheduling

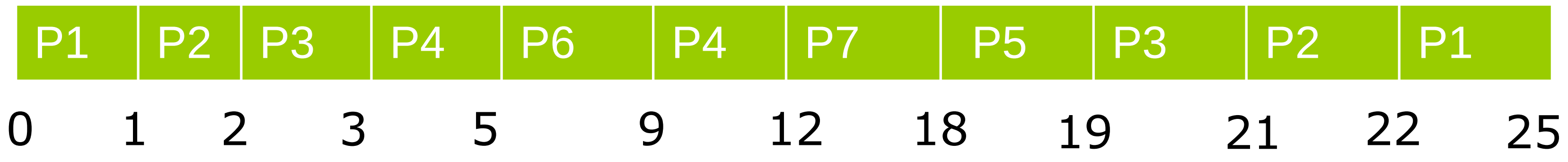
- Solve given example : preemptive
- highest number = highest priority

P.No	Priority	A.T	B.T
1	2	0	4
2	4	1	2
3	6	2	3
4	10	3	5
5	8	4	1
6	12	5	4
7	9	6	6

Priority Scheduling

P.No	Priority	A.T	B.T
1	2	0	4
2	4	1	2
3	6	2	3
4	10	3	5
5	8	4	1
6	12	5	4
7	9	6	6

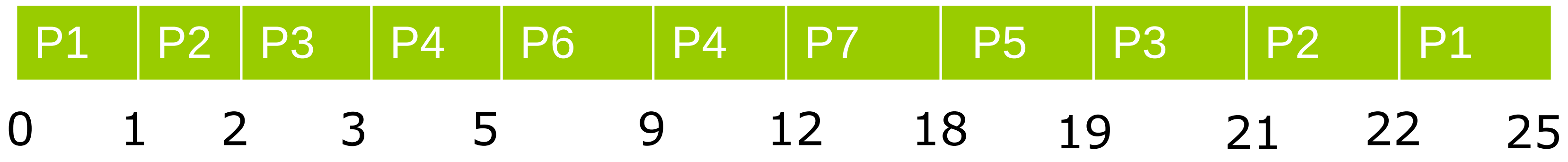
Gantt chart



Priority Scheduling

P.No	Priority	A.T	B.T	CT	TAT	WT	RT
1	2	0	4	25	25	21	0
2	4	1	2	22	21	19	0
3	6	2	3	21	19	16	0
4	10	3	5	12	9	4	0
5	8	4	1	19	15	14	14
6	12	5	4	9	4	0	0
7	9	6	6	18	12	6	6

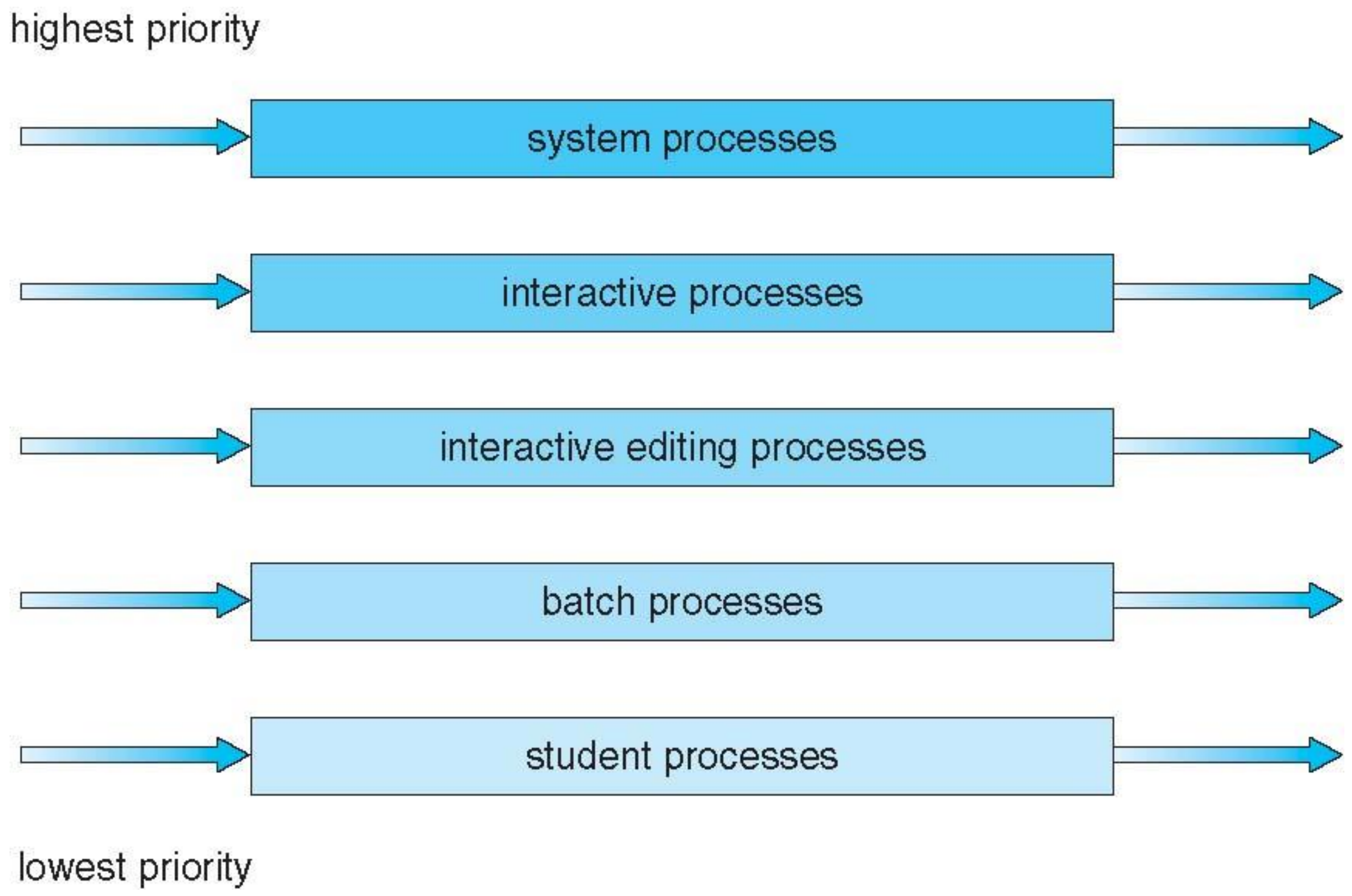
Gantt chart



Multilevel Queue

- Ready queue is partitioned into separate queues, eg:
  - foreground (interactive)
  - background (batch)
- Process permanently in a given queue
- Each queue has its own scheduling algorithm:
  - foreground – RR
  - background – FCFS
- Scheduling must be done between the queues:
  - Fixed priority scheduling; (i.e., serve all from foreground then from background). Possibility of starvation.
  - Time slice – each queue gets a certain amount of CPU time which it can schedule amongst its processes; i.e., 80% to foreground in RR
  - 20% to background in FCFS

Multilevel Queue Scheduling

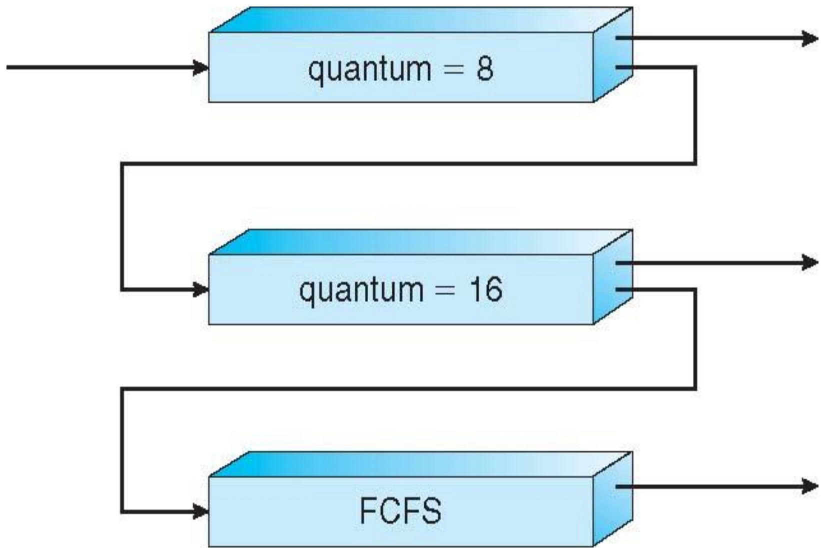


Multilevel Feedback Queue

- A process can move between the various queues; aging can be implemented this way
- Multilevel-feedback-queue scheduler defined by the following parameters:
  - number of queues
  - scheduling algorithms for each queue
  - method used to determine when to upgrade a process
  - method used to determine when to demote a process
  - 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 time quantum 16 milliseconds
  - Q_2 – FCFS
- Scheduling
  - A new job enters queue Q_0 which is served FCFS
    - When it gains CPU, job receives 8 milliseconds
    - If it does not finish in 8 milliseconds, job is moved to queue Q_1
  - At Q_1 job is again served FCFS and receives 16 additional milliseconds
    - If it still does not complete, it is preempted and moved to queue Q_2



Q&A

- Consider a system which has a CPU bound process,which require the burst time of 40 seconds.The multilevel Feed Back Queue scheduling algorithm is used and the queue time quantum '2' seconds and in each level it is incremented by '5' seconds.Then how many times the process will be interrupted and on which queue the process will terminate the execution?

  (a) 5,4 (b) 4,5 (c) 3,4 (d) 4,3

- Process P needs 40 Seconds for total execution.
  Multilevel Feedback queue:
  Queue1 = 2
  Queue2 = 7 (2+5)
  Queue3 = 12 (7+5)
  Queue4 = 17 (12+5)
  Queue5 = 2 (left time)
- hence you can see process is interrupted 4 times, and on 5th queue
  process completed its execution.
  Hence, (b) is correct option

Consider a set of 5 processes whose arrival time. CPU time needed and the priority are given below
smaller the number, higher the priority.
If the CPU scheduling policy FCFS, the average waiting time will be

Process	Arrival Time (in ms)	CPU Time Needed (in ms)	Priority
P1	0	10	5
P2	0	5	2
P3	2	3	1
P4	5	20	4
P5	10	2	3

smaller the number, higher the priority.
If the CPU scheduling policy FCFS, the average waiting time will be

- A) 12.8 ms
- B) 8.4 ms
- C) 6.55 ms
- D) 11.11 ms

- A
- According to FCFS process solve are p1 p2 p3 p4 p5 so
- for p1 waiting time =0 process time=10 then
- for p2 waiting time = (process time of p1-arrival time of p2)=10-0=10 then
- for p3 waiting time = (pr. time of (p1+p2)-arrival time of p3)=(10+5)-2=13 and
- same for p4 waiting time=18-5=13
- same for p5 waiting time=38-10=28
- So total average waiting time=(0+10+13+13+28)/5=12.8
- So answer is 'A'.
- 0 + 10 + (15- 2) + (18 5) + (38- 10) divided by 5, i.e. 12.8 ms.
- Note : Here we will not see priority, we only see who comes first. And here both p1 andp2 came simultaneously and so we take p1 first and it gives answer which matches in option.

- For the processes listed in the following table, which of the following scheduling schemes will give the lowest average turnaround time?

P.No	A.T	B.T
A	0	3
B	1	6
C	4	4
D	6	2

a) FCFS
b) Non- preemptive SJF
c) Shortest remaining first
d) RR 2

- For the processes listed in the following table, which of the following scheduling schemes will give the lowest average turnaround time?

P.No	A.T	B.T
A	0	3
B	1	6
C	4	4
D	6	2

a) FCFS (7.25)
b) Non- preemptive SJF (6.75)
c) Shortest remaining first (6.25)
d) RR 2 (8.25)

End of Chapter