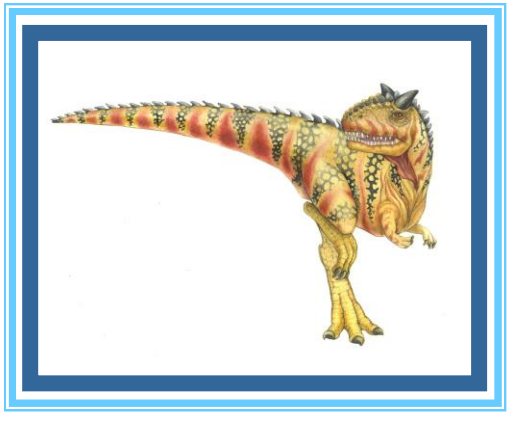
Operating Systems
Chapter 5: CPU Scheduling



Lecturer: Dalya Samer

Chapter 5 Outlines

- Basic Concepts
- Scheduling Criteria
- Scheduling Algorithms

Basic Concepts

- **CPU scheduling** is the central in multi-programming system.

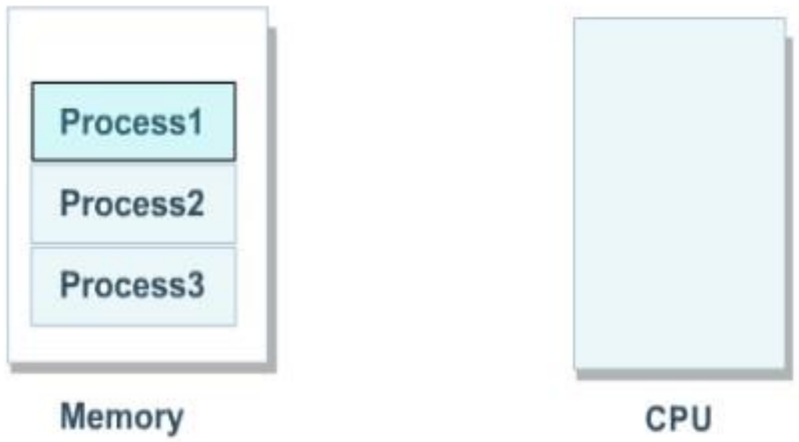
- Maximum CPU utilization obtained with multiprogramming (**prevent CPU from being idle**).

- Processes residing in the main memory is selected by the Scheduler that is:

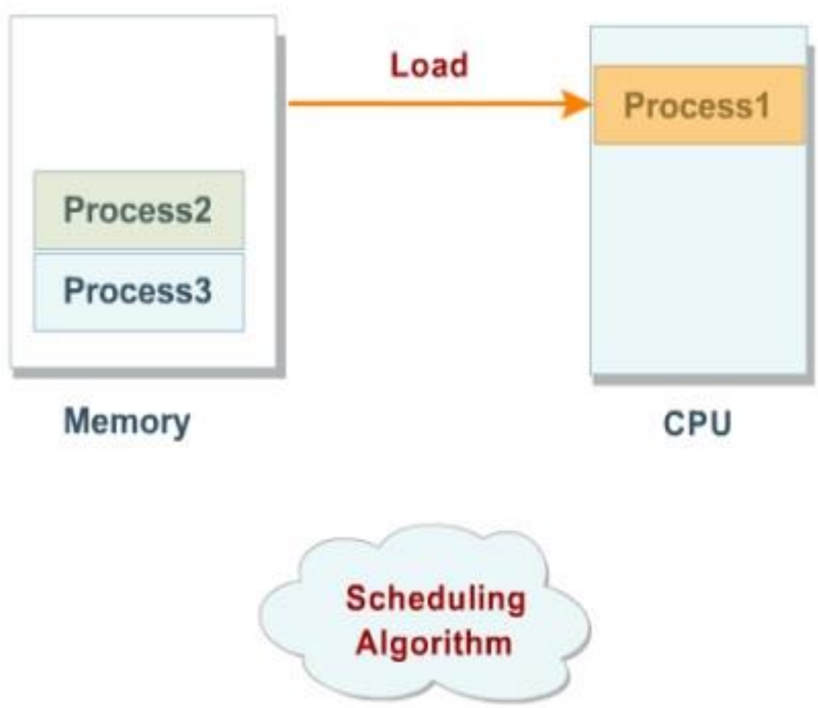
  ➢ Concerned with deciding a policy about which process is to be selected.

  ➢ Process selection based on a scheduling algorithm.

Basic Concepts

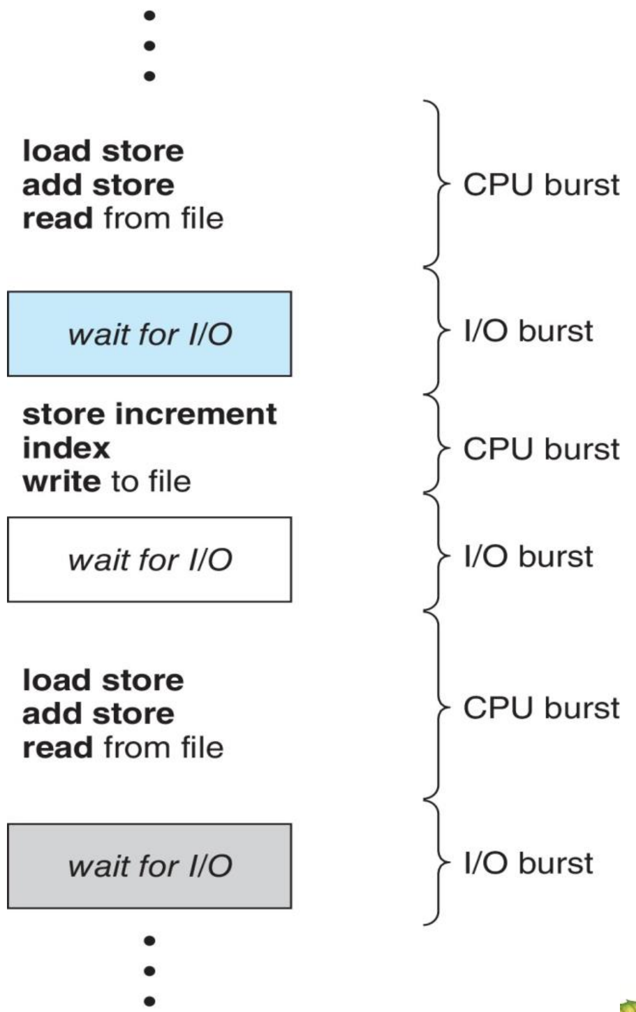


Basic Concepts



Basic Concepts

- Process execution consists of a **cycle** of CPU execution and I/O wait.
- Processes alternate between these two states. Process execution begins with a **CPU burst**. That is followed by an **I/O burst**, which is followed by another CPU burst, then another I/O burst, and so on.
- CPU bursts vary greatly from process to process and from computer to computer.



Basic Concepts

- **Schedulers**

- **Long-term scheduler** chooses some of them to go to memory (**ready queue**).

- Then, **short-term scheduler** (or **CPU scheduler**) chooses from ready queue a job to run on CPU.

- **Medium-term scheduler** may move (**swap**) some partially-executed jobs from memory to disk (to enhance performance).

Basic Concepts

- **CPU Scheduler**

• 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**.

Basic Concepts

- 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

Basic Concepts

- **Scheduling can be:**

- **Non-preemptive**

  ➢ Once a process is allocated the CPU, it **does not** leave until terminate.

- **Preemptive**

  ➢ OS can force (preempt) a process from CPU at anytime.

  ✓ Say, to allocate CPU to another higher-priority process.

Basic Concepts

Non-preemptive and Preemptive

Which is harder to implement? and why?

Basic Concepts

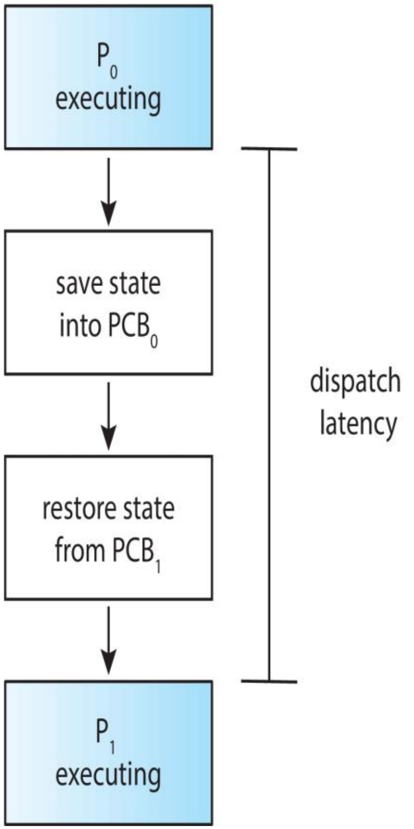
- **Non-preemptive and Preemptive**

- **Preemptive is harder:** Need to maintain consistency of data shared between processes, and more importantly, kernel data structures (e.g., I/O queues).

Basic Concepts

- **Dispatcher** module gives control of the CPU to the process selected by the CPU scheduler; this involves:
  - Switching context
  - Switching to user mode
  - Jumping to the proper location in the user program to restart that program

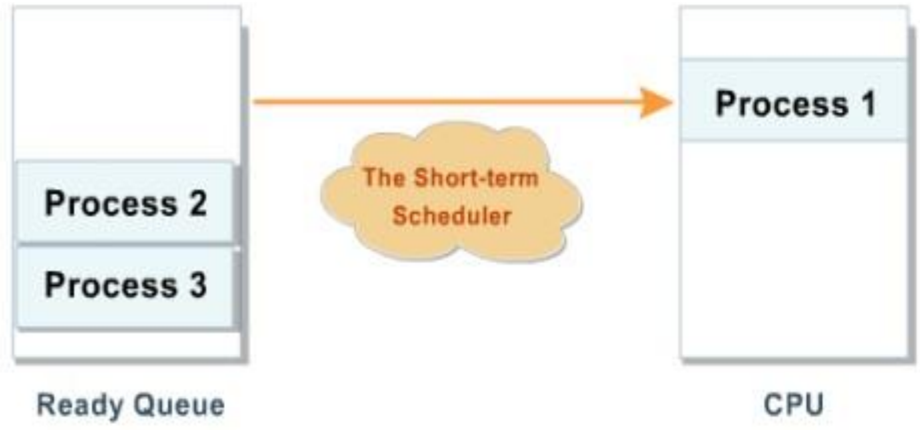
- **Dispatch latency** – time it takes for the dispatcher to stop one process and start another running



Basic Concepts

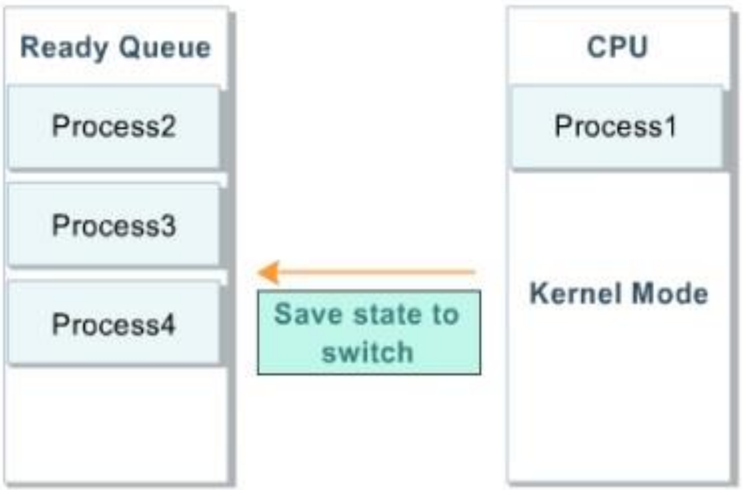
Basic Concepts

Dispatcher



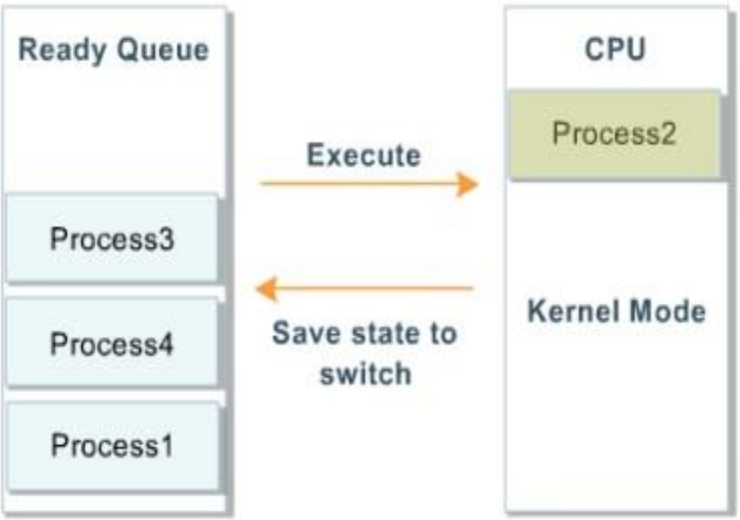
Basic Concepts

Dispatcher



Basic Concepts

Dispatcher



Scheduling Criteria

- **CPU utilization** – keep the CPU as busy as possible
- **Throughput** – no. 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.

Scheduling Criteria

Scheduling Algorithm Optimization Criteria

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

Scheduling Algorithms

- There are many different CPU-scheduling algorithms:

1. First Come, First Served (FCFS).
2. Shortest Job First (SJF).
   - ➢ Preemptive SJF.
   - ➢ Non-Preemptive SJF.
3. Priority.
4. Round Robin.
5. Multilevel queues.

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The Gantt Chart for the schedule is:

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The **Gantt Chart** for the schedule is:

Waiting Time		
P_1	P_2	P_3
0	24	27

Average waiting time: $(0 + 24 + 27)/3 = 17$

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The **Gantt Chart** for the schedule is:

Turnaround Time		
P_1	P_2	P_3
24	27	30

Average turnaround time: (24 + 27 + 30)/3 = 27

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The Gantt Chart for the schedule is:

Response Time		
P_1	P_2	P_3
0	24	27

Average response time: $(0 + 24 + 27)/3 = 17$

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_2 , P_3 , P_1
  The Gantt Chart for the schedule is:

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_2 , P_3 , P_1
  The Gantt Chart for the schedule is:

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_2 , P_3 , P_1
  The Gantt Chart for the schedule is:

Waiting Time		
P_1	P_2	P_3
6	0	3

Average waiting time: $(6 + 0 + 3)/3 = 3$

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_2 , P_3 , P_1
  The Gantt Chart for the schedule is:

Waiting Time		
P_1	P_2	P_3
6	0	3

Average waiting time: $(6 + 0 + 3)/3 = 3$

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_2 , P_3 , P_1
  The Gantt Chart for the schedule is:

Turnaround Time		
P_1	P_2	P_3
30	3	6

Average turnaround time: (30 + 3 + 6)/3 = 13

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_2 , P_3 , P_1
  The Gantt Chart for the schedule is:

Turnaround Time		
P_1	P_2	P_3
30	3	6

Average turnaround time: (30 + 3 + 6)/3 = 13

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_2 , P_3 , P_1
  The Gantt Chart for the schedule is:

Response Time		
P_1	P_2	P_3
6	0	3

Average response time: $(6 + 0 + 3)/3 = 3$

Scheduling Algorithms

1. First-Come, First-Served (FCFS) Scheduling

- FCFS is fair in the formal sense or human sense of fairness.
- but it is unfair in the sense that long jobs take priority over short jobs and unimportant jobs make important jobs wait.
- One of the major drawbacks of this scheme is that the waiting time and the average turnaround time is often quite long.

Scheduling Algorithms

2. Shortest-Job-First (SJF) Scheduling

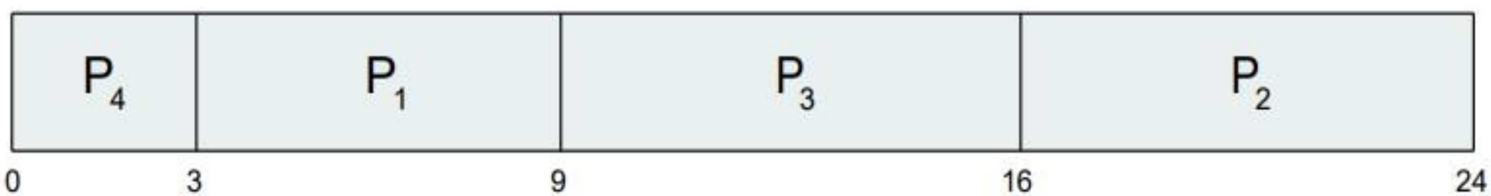
- Associate with each process the length of its next CPU burst.
  - Use these lengths to schedule the process with the shortest time.
- SJF is optimal – gives minimum average waiting time for a given set of processes.
  - The difficulty is knowing the length of the next CPU request.
  - Could ask the user.

Scheduling Algorithms

2.1 Shortest-Job-First (SJF) Scheduling

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3

- SJF scheduling chart

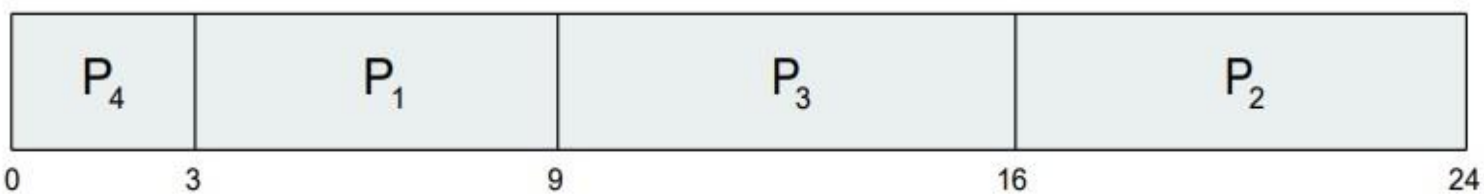


Scheduling Algorithms

2.1 Shortest-Job-First (SJF) Scheduling

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3

- SJF scheduling chart



Waiting Time			
P_1	P_2	P_3	P_4
3	16	9	0

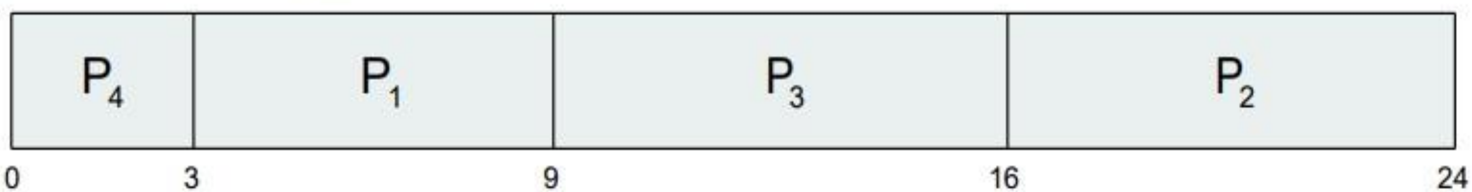
Average waiting time: $(3 + 16 + 9 + 0)/4 = 7$

Scheduling Algorithms

2.1 Shortest-Job-First (SJF) Scheduling

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3

- SJF scheduling chart



Turnaround Time			
P_1	P_2	P_3	P_4
9	24	16	3

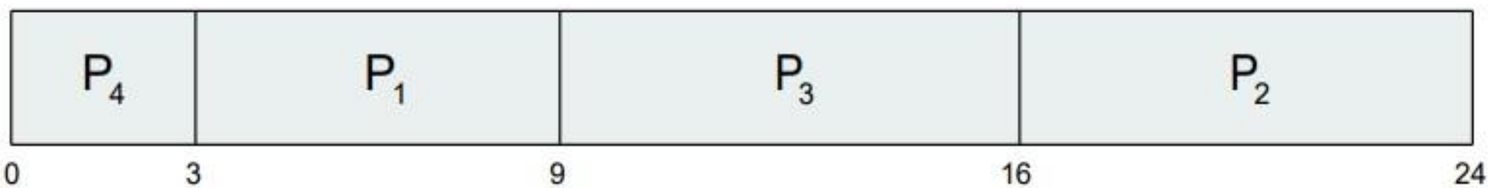
Average Turnaround time: $(9 + 24 + 16 + 3)/4 = 13$

Scheduling Algorithms

2.1 Shortest-Job-First (SJF) Scheduling

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3

- SJF scheduling chart



Response Time			
P_1	P_2	P_3	P_4
3	16	9	0

Average Response time: (3 + 16 + 9 + 0)/4 = 7

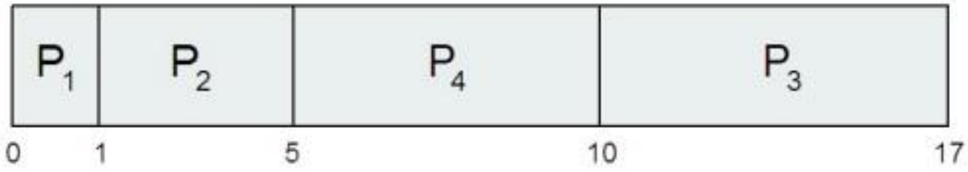
Scheduling Algorithms

2.1 Shortest-Job-First (SJF) (Non-Preemptive SJF)

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	1
P_2	1	4
P_3	2	7
P_4	3	5

- *Non-Preemptive* SJF Gantt Chart



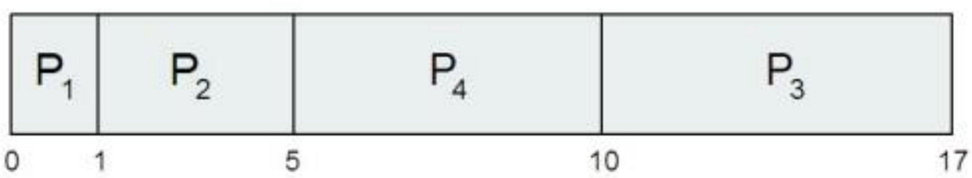
Scheduling Algorithms

2.1 Shortest-Job-First (SJF) (Non-Preemptive SJF)

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	1
P_2	1	4
P_3	2	7
P_4	3	5

- *Non-Preemptive* SJF Gantt Chart



Waiting Time			
P_1	P_2	P_3	P_4
(0 – 0)	(1 – 1)	(10 – 2)	(5 – 3)

Average waiting time: (0 + 0 + 8 + 2)/4 = 2.5 msec

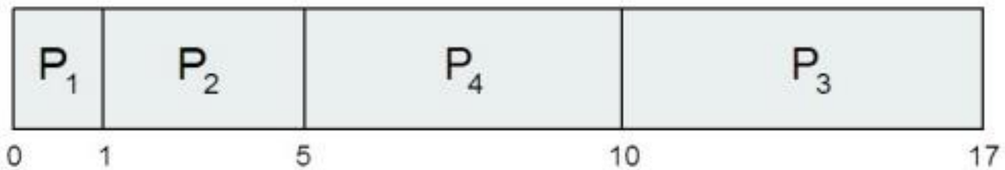
Scheduling Algorithms

2.1 Shortest-Job-First (SJF) (Non-Preemptive SJF)

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	1
P_2	1	4
P_3	2	7
P_4	3	5

- *Non-Preemptive* SJF Gantt Chart



Turnaround Time			
P_1	P_2	P_3	P_4
(1 – 0)	(5 – 1)	(17 – 2)	(10 – 3)

Average Turnaround time: (1 + 4 + 15 + 7)/4 = 6.75 msec

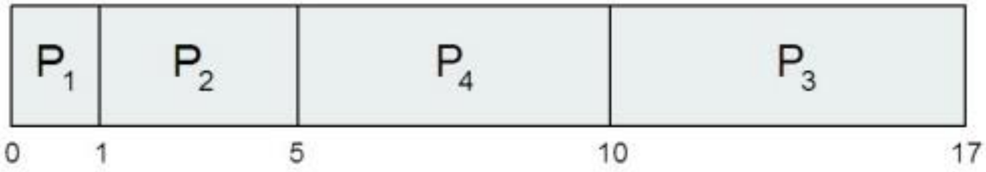
Scheduling Algorithms

2.1 Shortest-Job-First (SJF) (Non-Preemptive SJF)

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	1
P_2	1	4
P_3	2	7
P_4	3	5

- *Non-Preemptive* SJF Gantt Chart



Response Time			
P_1	P_2	P_3	P_4
(0 – 0)	(1 – 1)	(10 – 2)	(5 – 3)

Average Response time: (0 + 0 + 8 + 2)/4 = 2.5 msec

Scheduling Algorithms

2.2 Shortest-remaining-time-first (Preemptive SJF)

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	***Arrival*** **Time**	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- ***Preemptive*** **SJF** Gantt Chart

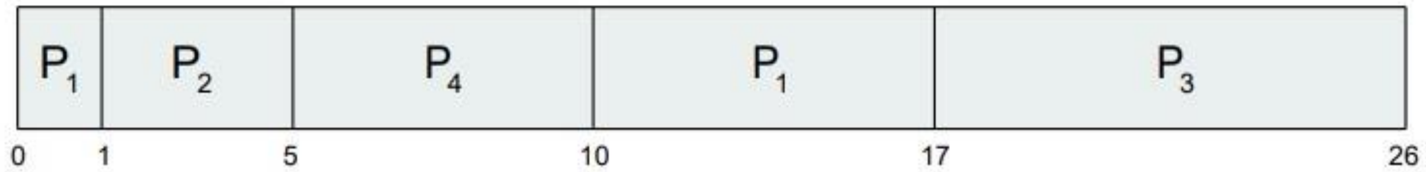
Scheduling Algorithms

2.2 Shortest-remaining-time-first (Preemptive SJF)

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	***Arrival*** **Time**	Burst Time
P_1	0	~~8~~ ~~7~~
P_2	1	~~4~~ ~~3~~ ~~2~~
P_3	2	~~9~~
P_4	3	~~5~~

- ***Preemptive*** SJF Gantt Chart



Scheduling Algorithms

2.2 Shortest-remaining-time-first (Preemptive SJF)

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- *Preemptive* SJF Gantt Chart

Waiting Time			
P_1	P_2	P_3	P_4
10 – 1	1 – 1	17 – 2	5 – 3
= 9	**= 0**	**= 15**	**= 2**

Average waiting time = [9+0+15+2]/4 = 26/4 = 6.5 msec

Scheduling Algorithms

2.2 Shortest-remaining-time-first (Preemptive SJF)

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- *Preemptive* SJF Gantt Chart

Turnaround Time			
P_1	P_2	P_3	P_4
17 – 0	5 – 1	26 – 2	10 – 3
= 17	**= 4**	**= 24**	**= 7**

Average turnaround time = [17+4+24+7]/4 = 52/4 = 13 msec

Scheduling Algorithms

2.2 Shortest-remaining-time-first (Preemptive SJF)

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- *Preemptive* SJF Gantt Chart

Response Time			
P_1	P_2	P_3	P_4
0 – 0	1 – 1	17 – 2	5 – 3
= 0	**= 0**	**= 15**	**= 2**

Average response time = [0+0+15+2]/4 = 17/4 = 4.25 msec

Scheduling Algorithms

3. Priority Scheduling

- A priority number (integer) is associated with each process
- The CPU is allocated to the process with the highest priority (smallest integer> highest priority)
    - Preemptive
    - Nonpreemptive
- SJF is priority scheduling where priority is the inverse of predicted next CPU burst time
- Problem>**Starvation** – low priority processes may never execute
- Solution>**Aging** – as time progresses increase the priority of the process

Scheduling Algorithms

3. Priority Scheduling

Process	Burst Time	Priority
P_1	10	3
P_2	1	1 (Highest priority)
P_3	2	4
P_4	1	5
P_5	5	2

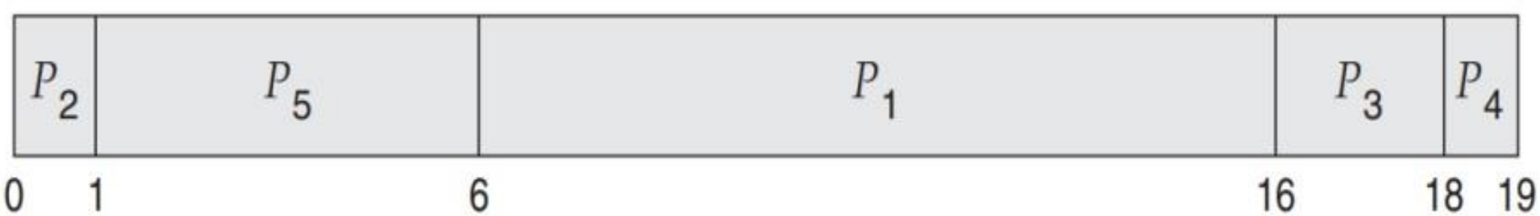
- Priority scheduling Gantt Chart

Scheduling Algorithms

3. Priority Scheduling

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- Priority scheduling Gantt Chart



Scheduling Algorithms

3. Priority Scheduling

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- Priority scheduling Gantt Chart

Waiting Time				
P_1	P_2	P_3	P_4	P_5
6	0	16	18	1

Average Waiting time = [6+0+16+18+1]/5 = 8.2 msec

Scheduling Algorithms

3. Priority Scheduling

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- Priority scheduling Gantt Chart

Turnaround Time				
P_1	P_2	P_3	P_4	P_5
16	1	18	19	6

Average Turnaround time = [16+1+18+19+6]/5 = 12 msec

Scheduling Algorithms

3. Priority Scheduling

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- Priority scheduling Gantt Chart

Response Time				
P_1	P_2	P_3	P_4	P_5
6	0	16	18	1

Average Response time = [6+0+16+18+1]/5 = 8.2 msec

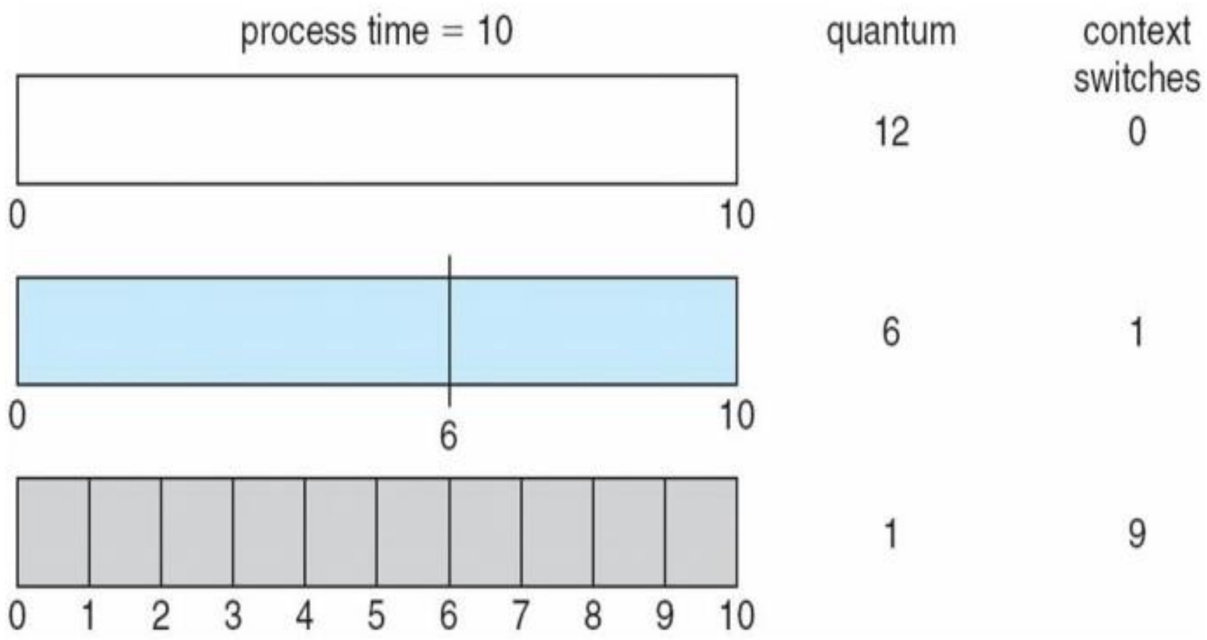
Scheduling Algorithms

4. Round Robin (RR) Scheduling

- Each process gets a small unit of CPU time (time quantum q), usually 10-100 milliseconds. After this time has elapsed, the process is preempted and added to the end of the ready queue.
- If there are n processes in the ready queue and the time quantum is q, then each process gets $1/n$ of the CPU time in chunks of at most q time units at once.
- No process waits more than $(n - 1)q$ time units.

Scheduling Algorithms

4. Round Robin (RR) Scheduling



Scheduling Algorithms

4. Round Robin (RR) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

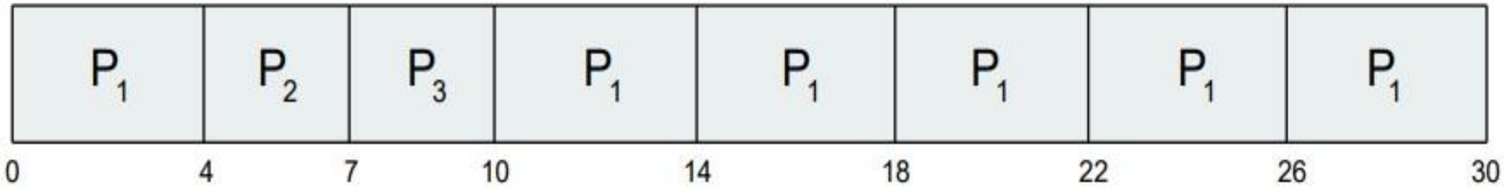
- All the processes **arrive** at the same time **0**.
- Round Robin (RR) scheduling of quantum: **4** ms

Scheduling Algorithms

4. Round Robin (RR) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- All the processes **arrive** at the same time **0**.
- Round Robin (RR) scheduling of quantum: **4** ms



Scheduling Algorithms

4. Round Robin (RR) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- All the processes **arrive** at the same time **0**.
- Round Robin (RR) scheduling of quantum: **4** ms

P_1	P_2	P_3	P_1	P_1	P_1	P_1	P_1
0–4	4–7	7–10	10–14	14–18	18–22	22–26	26–30

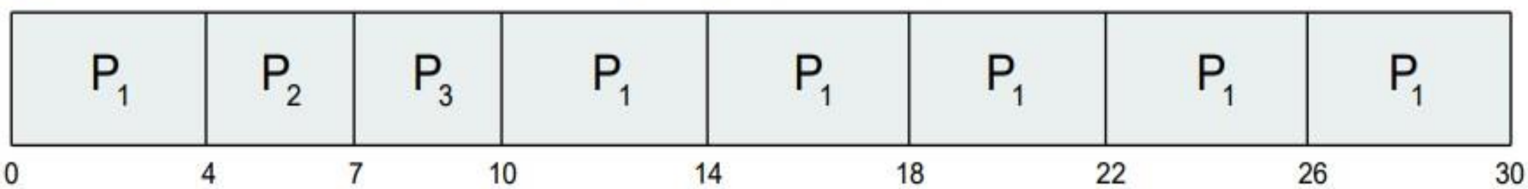
- # of context switches = 7

Scheduling Algorithms

4. Round Robin (RR) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- All the processes **arrive** at the same time **0**.
- Round Robin (RR) scheduling of quantum: **4** ms



Waiting Time		
P_1	P_2	P_3
0 + (10 – 4)	4	7

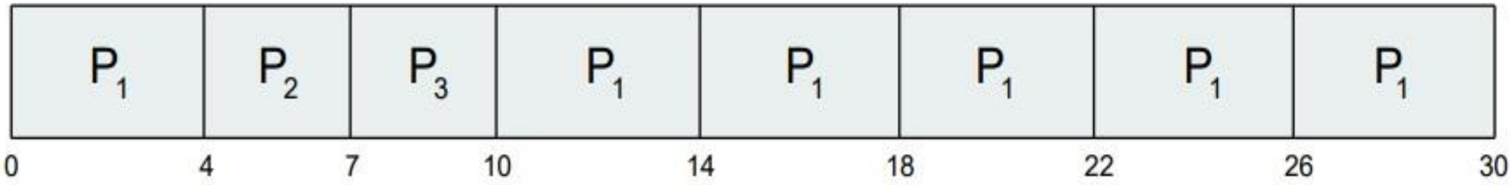
Average waiting time: (6 + 4 + 7)/3 = 5.667 ms

Scheduling Algorithms

4. Round Robin (RR) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- All the processes **arrive** at the same time **0**.
- Round Robin (RR) scheduling of quantum: **4** ms



Turnaround Time		
P_1	P_2	P_3
30	7	10

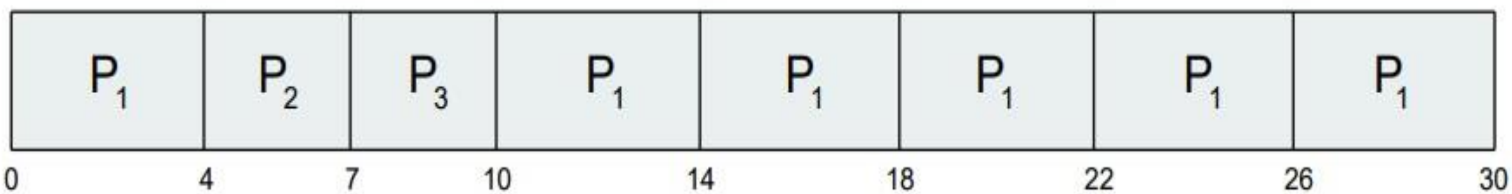
Average Turnaround time: $(30 + 7 + 10)/3 = 15.667$ ms

Scheduling Algorithms

4. Round Robin (RR) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- All the processes **arrive** at the same time **0**.
- Round Robin (RR) scheduling of quantum: **4** ms



Response Time		
P_1	P_2	P_3
0	4	7

Average Response time: (0 + 4 + 7)/3 = 3.667 ms

Scheduling Algorithms

5. Multilevel Queue Scheduling

- Ready queue is partitioned into separate queues, ex:
  - **foreground** (interactive)
  - **background** (batch)
- Process permanently in a given queue
- Each queue has its own scheduling algorithm:
  - foreground – RR.
  - background – FCFS

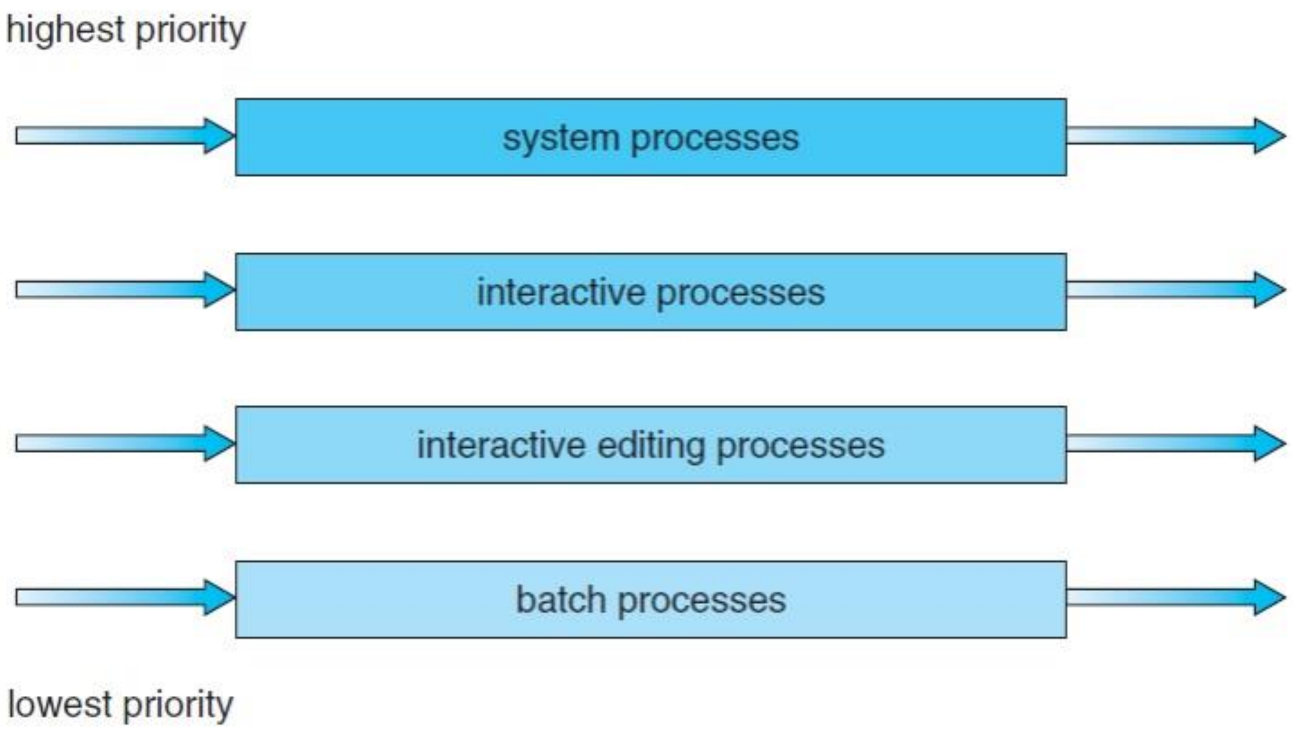
Scheduling Algorithms

5. Multilevel Queue Scheduling

- 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.

Scheduling Algorithms

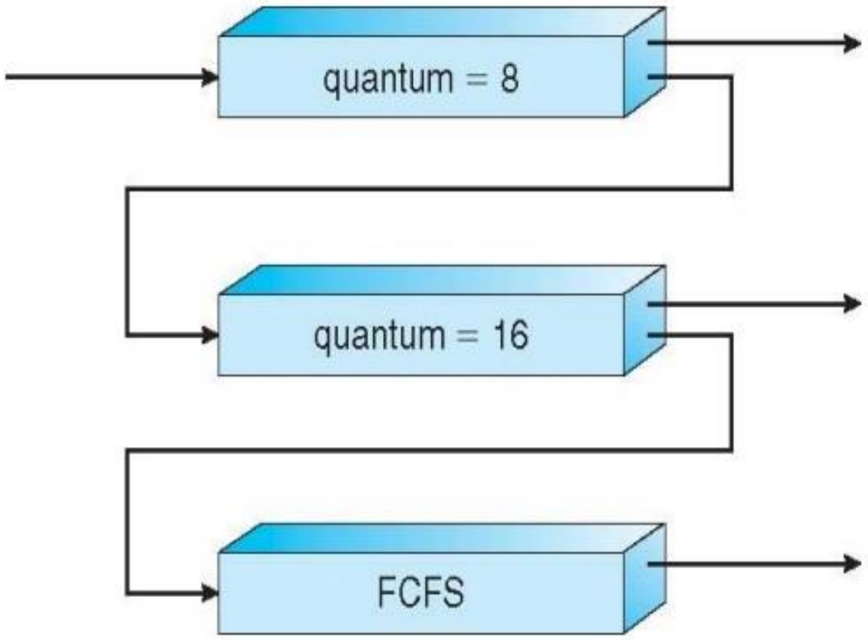
5. Multilevel Queue Scheduling



Scheduling Algorithms

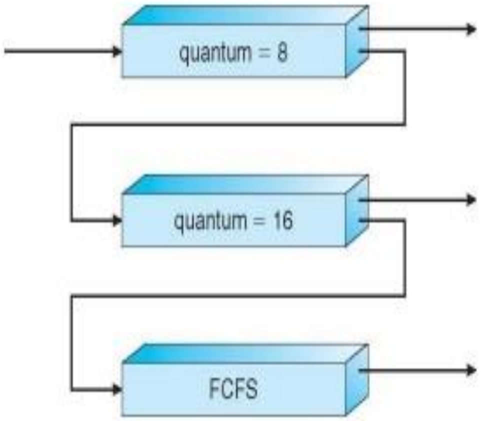
5. Multilevel Queue Scheduling

- Three queues:
  - Q_0 – **RR** with time quantum **8** milliseconds
  - Q_1 – **RR** time quantum **16** milliseconds
  - Q_2 – **FCFS**



Scheduling Algorithms

5. Multilevel Queue Scheduling



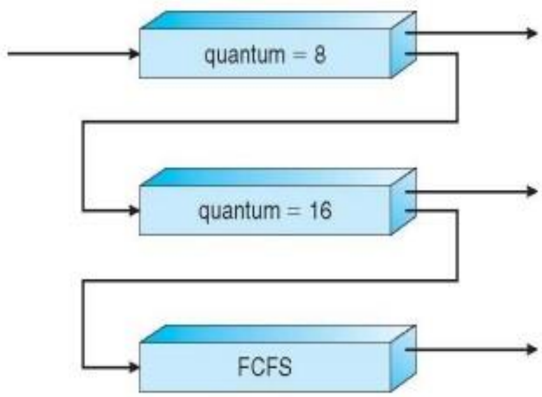
- Scheduling
- A new job enters queue Q_0 which is served FCFS
  - When it gains CPU, job receives 8 milliseconds.
  - If it does not finish in 8 milliseconds, job is moved to queue Q_1.
- At Q_1 job is again served FCFS and receives 16 additional milliseconds
  - If it still does not complete, it is preempted and moved to queue Q_2.

5. Multilevel Queue Scheduling

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	***Arrival*** **Time**	Burst Time
P_1	0	7
P_2	1	60
P_3	2	20
P_4	3	40

Using multi-processors or multi-core processor

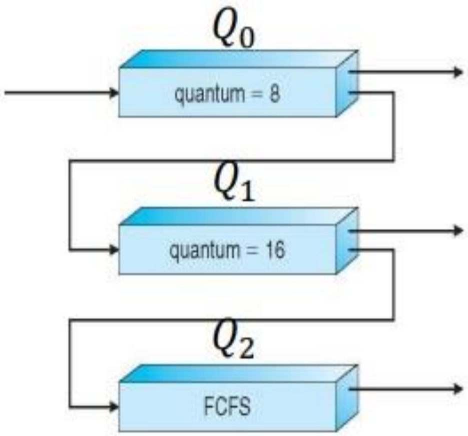


Multilevel Queue Fixed priority
non-preemptive

Scheduling Algorithms

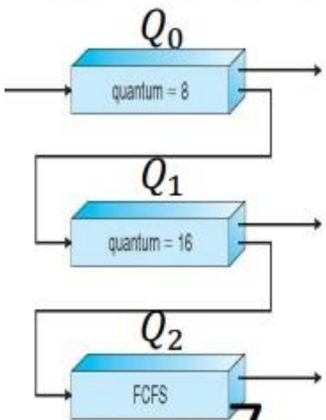
5. Multilevel Queue Scheduling

Process	*Arrival* Time	Burst Time
P_1	0	7
P_2	1	60
P_3	2	20
P_4	3	40

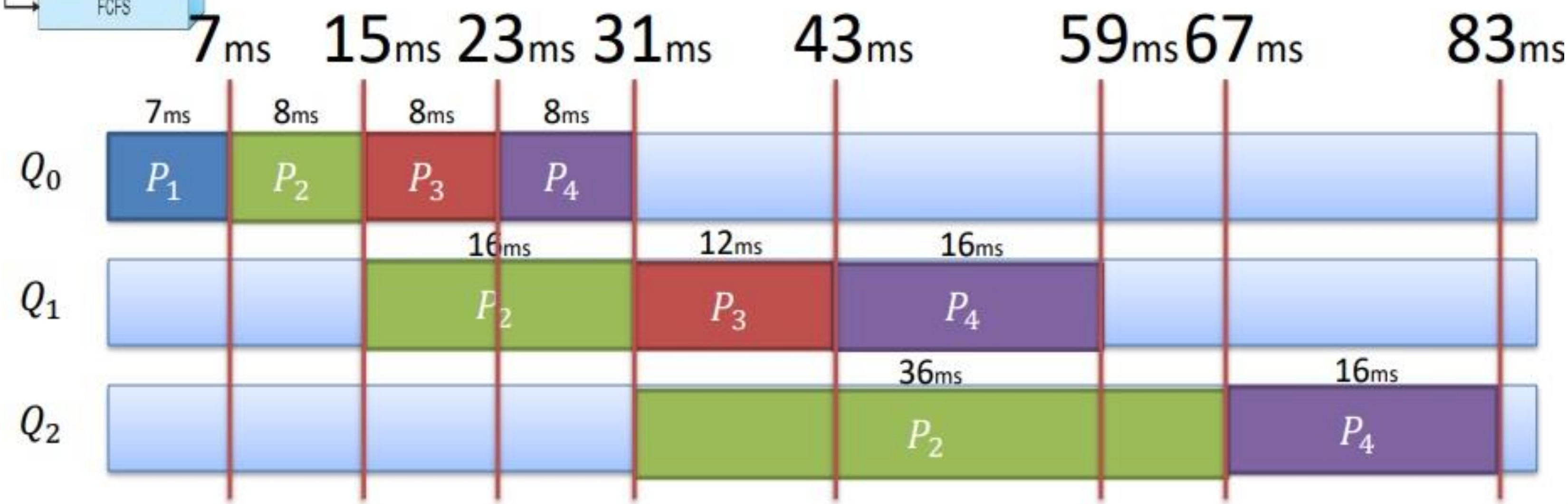


Scheduling Algorithms

5. Multilevel Queue Scheduling

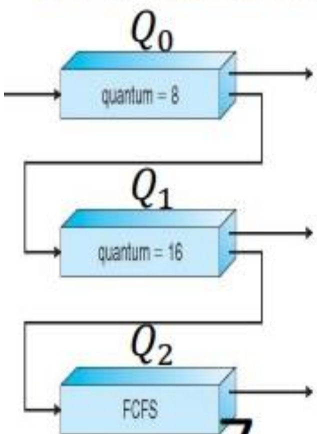


Process	***Arrival*** Time	Burst Time
P_1	0	7
P_2	1	60 52 44 36 24 8
P_3	2	20 12
P_4	3	40 32 16



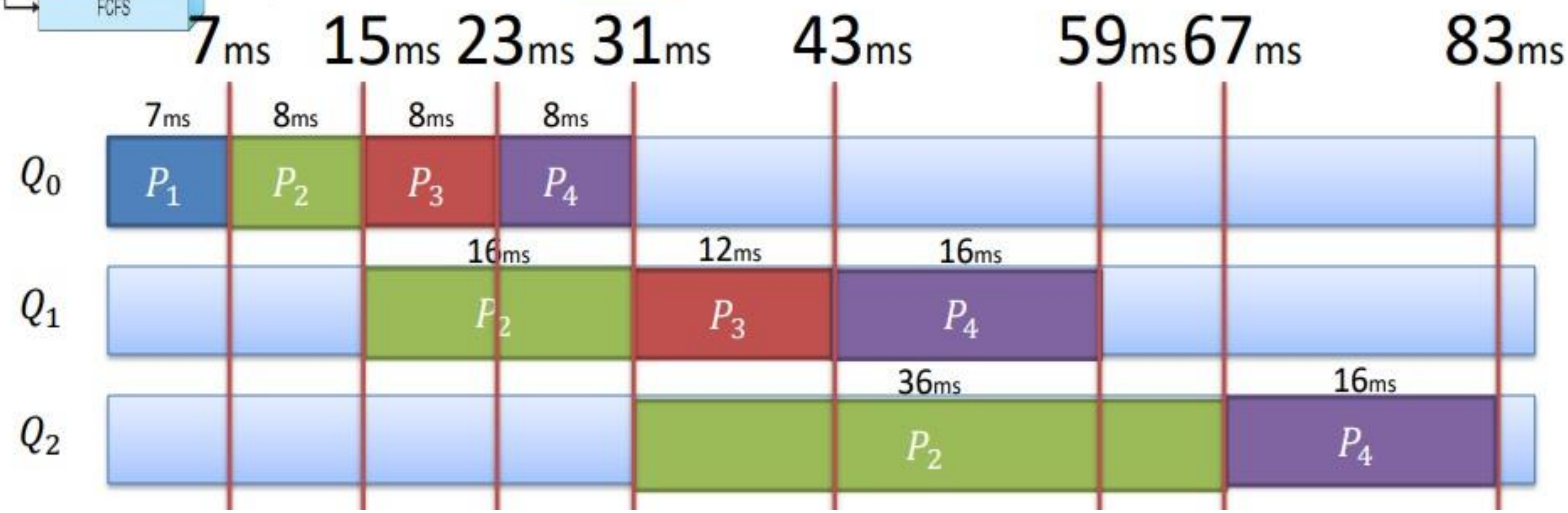
Scheduling Algorithms

5. Multilevel Queue Scheduling



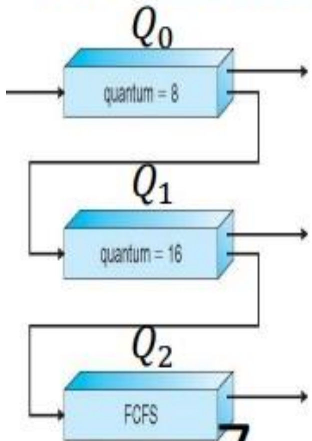
Process	Arrival Time
P_1	0
P_2	1
P_3	2
P_4	3

Waiting Time			
P_1	P_2	P_3	P_4
0	$7-1$	$15+8-2$	$23+12+8-3$
= 0	**= 6**	**= 21**	**= 40**



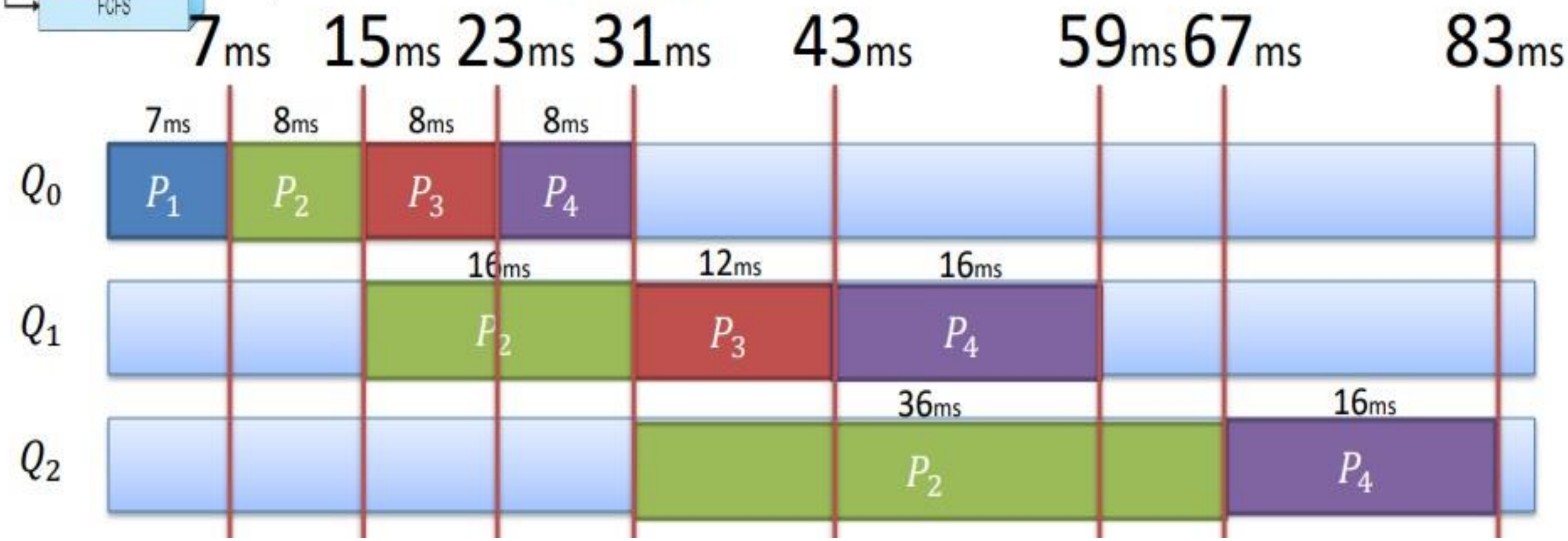
Scheduling Algorithms

5. Multilevel Queue Scheduling



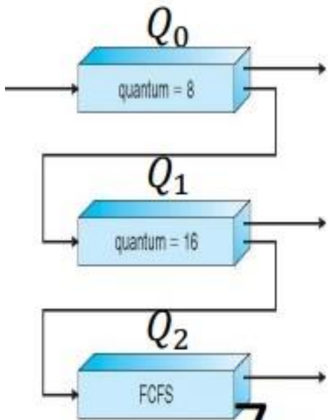
Process	*Arrival* Time
P_1	0
P_2	1
P_3	2
P_4	3

Turnaround Time			
P_1	P_2	P_3	P_4
$7 - 0$	$67 - 1$	$43 - 2$	$83 - 3$
$= 7$	$= 66$	$= 41$	$= 80$



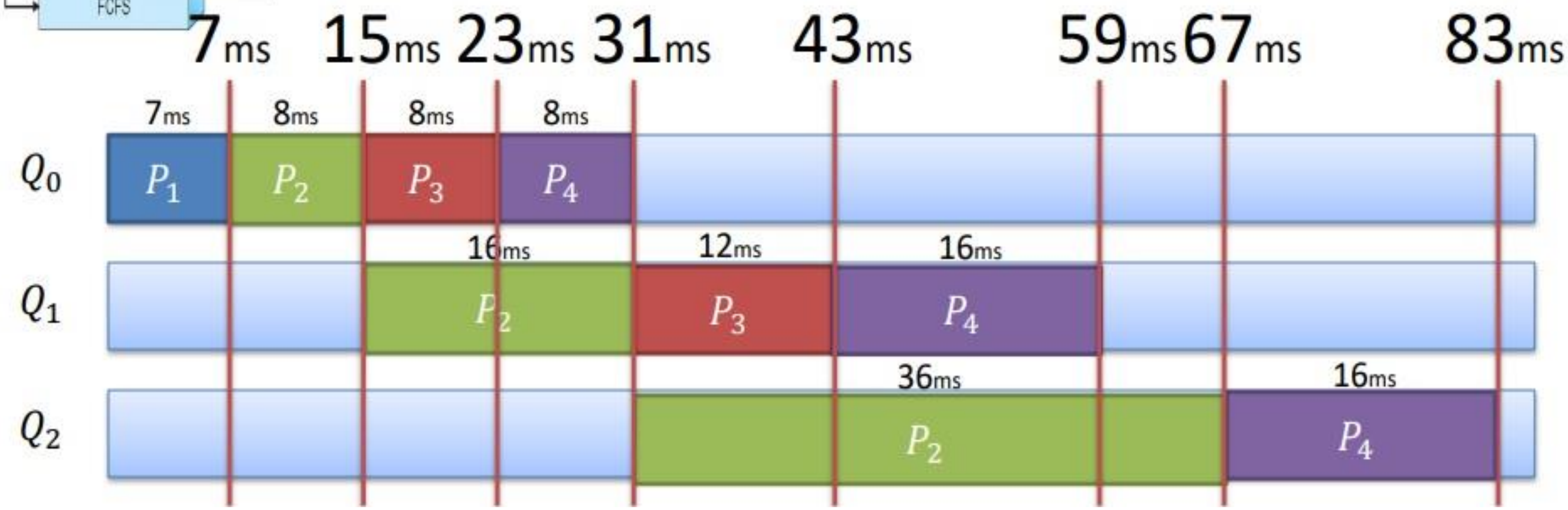
Scheduling Algorithms

5. Multilevel Queue Scheduling



Process	*Arrival* Time
P_1	0
P_2	1
P_3	2
P_4	3

Response Time			
P_1	P_2	P_3	P_4
$0 - 0$	$7 - 1$	$15 - 2$	$23 - 3$
= 0	**= 6**	**= 13**	**= 20**

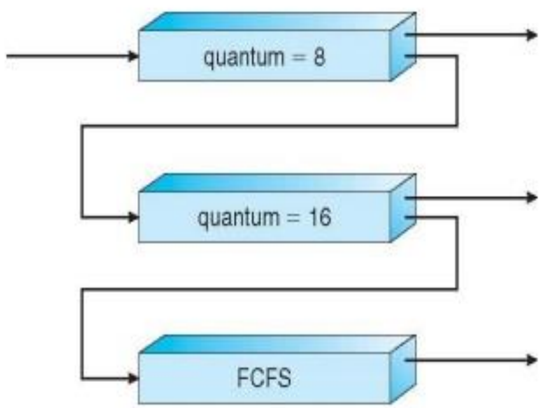


5. Multilevel Queue Scheduling

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	7
P_2	1	60
P_3	2	20
P_4	3	40

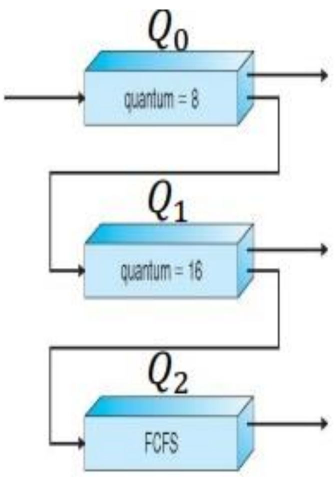
Using single-core processor



Multilevel Queue Fixed priority
non-preemptive

Scheduling Algorithms

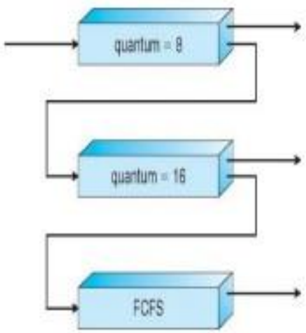
5. Multilevel Queue Scheduling



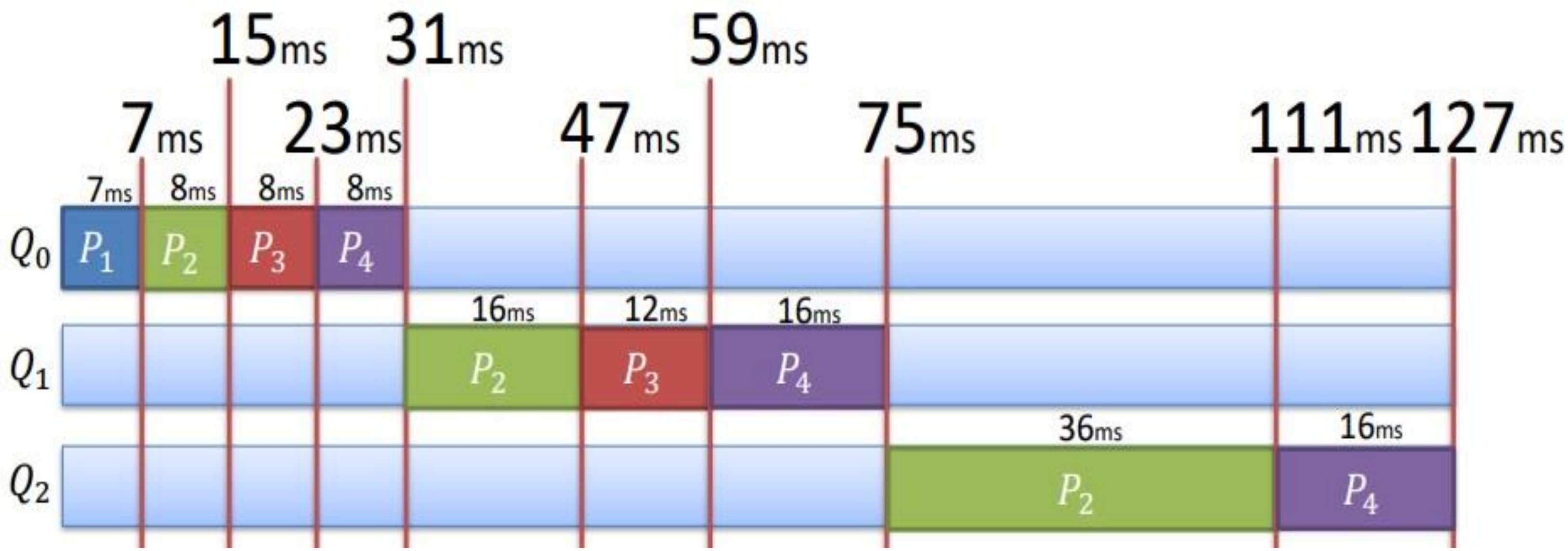
Process	***Arrival*** **Time**	Burst Time
P_1	0	7
P_2	1	60
P_3	2	20
P_4	3	40

Scheduling Algorithms

5. Multilevel Queue Scheduling

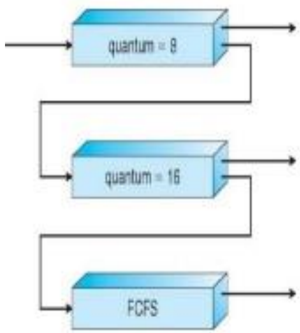


Process	***Arrival*** Time	Burst Time
~~P_1~~	0	~~7~~
~~P_2~~	1	~~60~~ ~~52~~ ~~36~~
~~P_3~~	2	~~20~~ ~~12~~
~~P_4~~	3	~~40~~ ~~32~~ ~~16~~

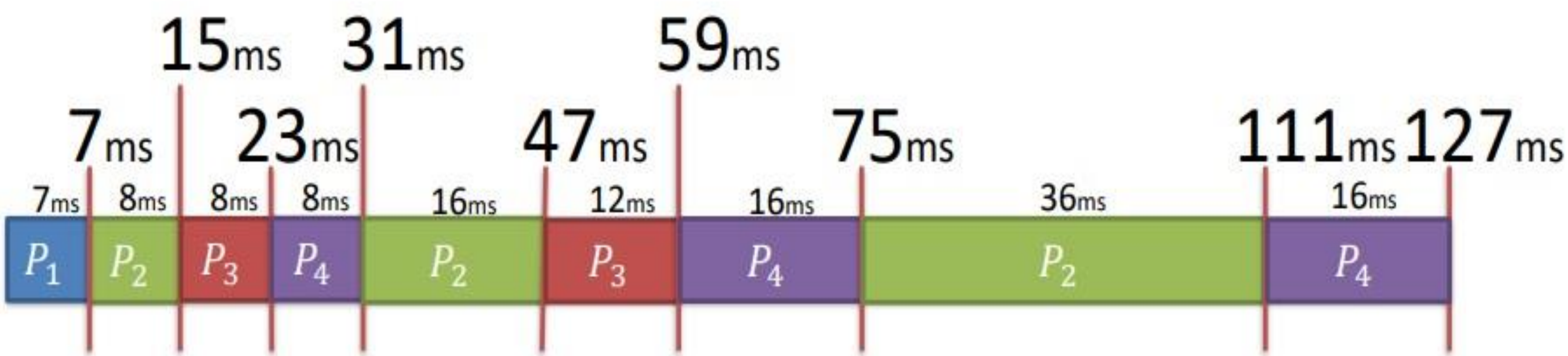


Scheduling Algorithms

5. Multilevel Queue Scheduling

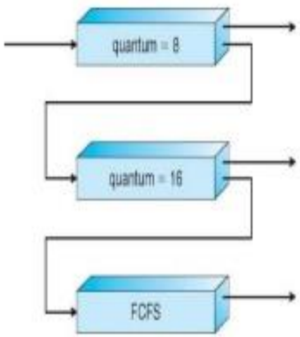


Process	*Arrival* Time	Burst Time
~~P_1~~	0	~~7~~
~~P_2~~	1	~~60~~ ~~52~~ ~~36~~
~~P_3~~	2	~~20~~ ~~12~~
~~P_4~~	3	~~40~~ ~~32~~ ~~16~~



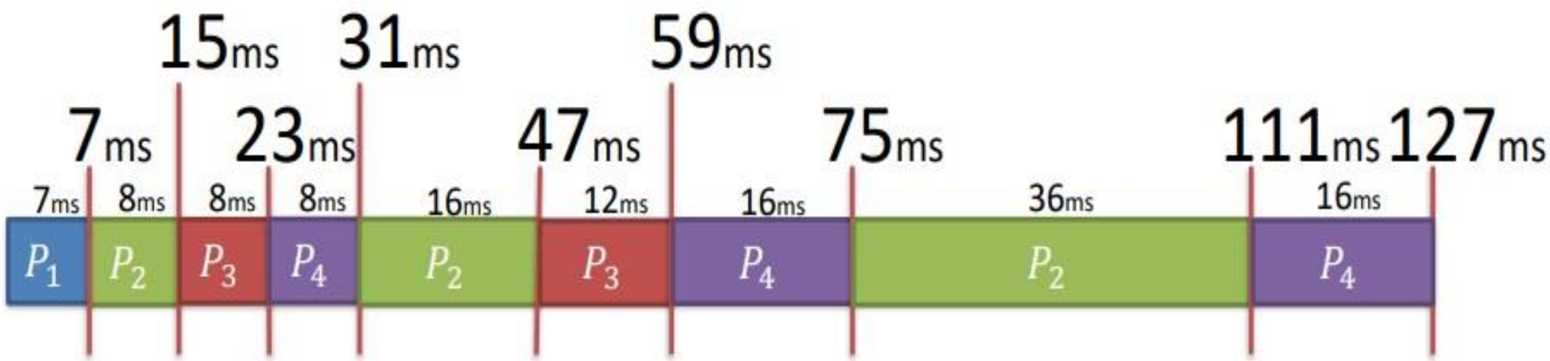
Scheduling Algorithms

5. Multilevel Queue Scheduling



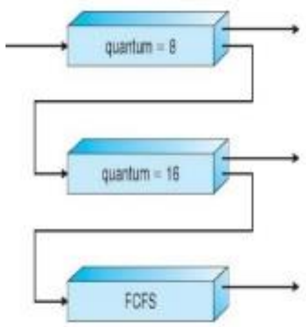
Process	*Arrival* Time
P_1	0
P_2	1
P_3	2
P_4	3

Waiting Time			
P_1	P_2	P_3	P_4
0	6 + 16 + 28	13 + 24	20 + 28 + 36
= 0	= 50	= 37	= 84



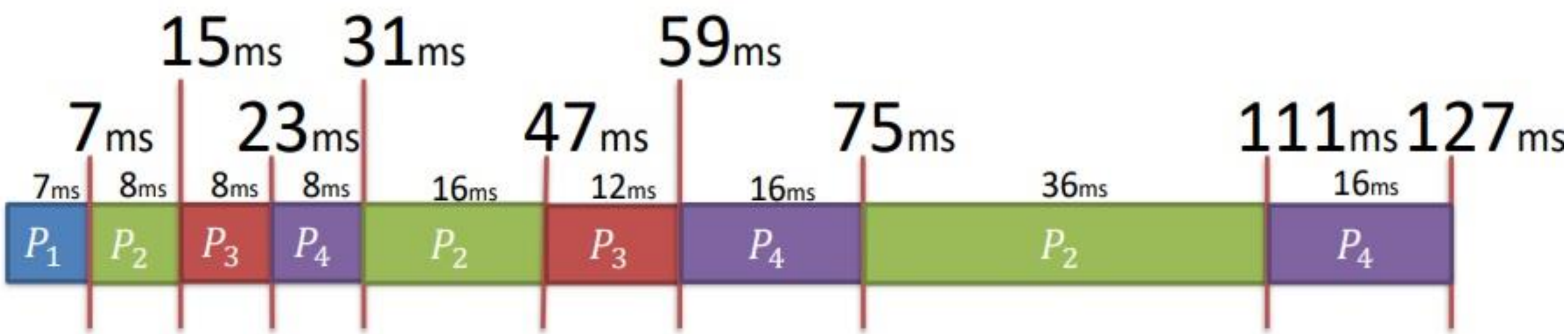
Scheduling Algorithms

5. Multilevel Queue Scheduling



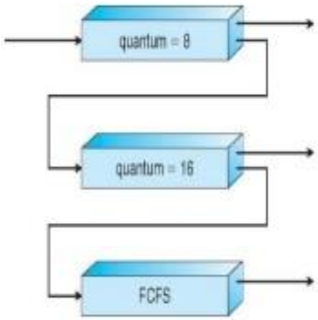
Process	*Arrival* Time
P_1	0
P_2	1
P_3	2
P_4	3

Turnaround Time			
P_1	P_2	P_3	P_4
7 – 0	111 – 1	59 – 2	127 – 3
= 7	**= 110**	**= 57**	**= 124**



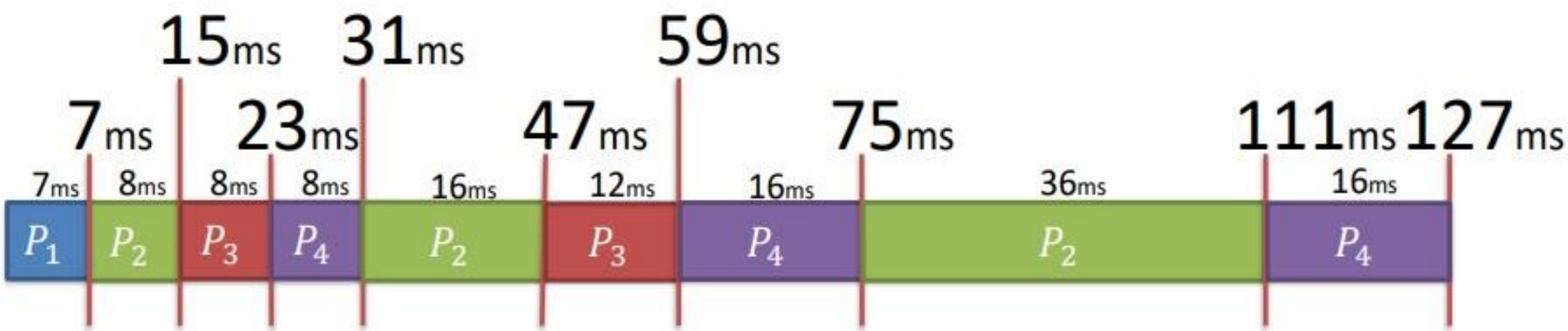
Scheduling Algorithms

5. Multilevel Queue Scheduling



Process	*Arrival* Time
P_1	0
P_2	1
P_3	2
P_4	3

Response Time			
P_1	P_2	P_3	P_4
0 – 0	7 – 1	15 – 2	23 – 3
= 0	**= 6**	**= 13**	**= 20**



End of Chapter 5