OPERATING SYSTEMS

Process Management 6

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

OPERATING SYSTEMS

Course Syllabus - Unit 1

UNIT 1: Introduction and Process Management

Operating-System Structure & Operations, Kernel Data Structures,Computing Environments, Operating-System Services, OperatingSystem Design and Implementation. Process concept: Process in memory, Process State, Process Control Block, Process Creation and Termination, CPU Scheduling and Scheduling Algorithms, IPC - Shared Memory & Message Passing, Pipes - Named and Ordinary. Case Study: Linux/Windows Scheduling Policies.

OPERATING SYSTEMS

Course Outline

Class No.	Chapter Title / Reference Literature	Topics to be covered	% of Portions covered	
			Reference chapter	Cumulative
1	1.1-1.2	What Operating Systems Do, Computer-System Organization?	1	21.4
2	1.3,1.4,1.5	Computer-System Architecture, Operating-System Structure & Operations	1	
3	1.10,1.11	Kernel Data Structures, Computing Environments	1	
4	2.1,2.6	Operating-System Services, Operating System Design and Implementation	2	
5	3.1-3.3	Process concept: Process in memory, Process State, Process Control Block, Process Creation and Termination	3	
6	5.1-5.2	CPU Scheduling: Basic Concepts, Scheduling Criteria	5	
7	5.3	Scheduling Algorithms: First-Come, First-Served Scheduling, Shortest-Job-First Scheduling	5	
8	5.3	Scheduling Algorithms: Shortest-Job-First Scheduling (Pre-emptive), Priority Scheduling	5	
9	5.3	Round-Robin Scheduling, Multi-level Queue, Multi-Level Feedback Queue Scheduling	5	
10	5.5,5.6	Multiple-Processor Scheduling, Real-Time CPU Scheduling	5	
11	5.7	Case Study: Linux/Windows Scheduling Policies	5	
12	3.4,3.6.3	IPC – Shared Memory & Message Passing, Pipes – Named and Ordinary	3,6	

Topics Outline

- Preemptive Priority Scheduling (PPS)
- Round Robin Scheduling (RR)
- Multilevel Queues
- Multilevel Feedback Queue

CPU Scheduling: 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 typically used for time-sharing environment

CPU Scheduling Algorithm Optimisation Criteria

- Maximize CPU utilization
- Maximize Throughput
- Minimize TurnAround Time
- Minimize Waiting Time
- Minimize Response Time

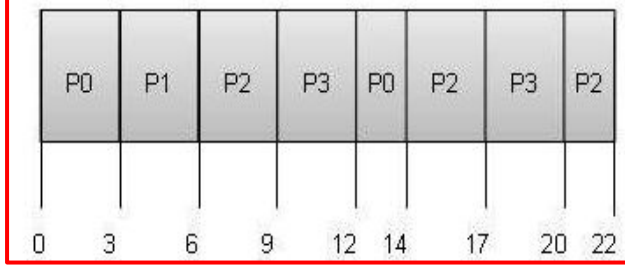
Sample Schemas and Tools for solving Scheduling Problems

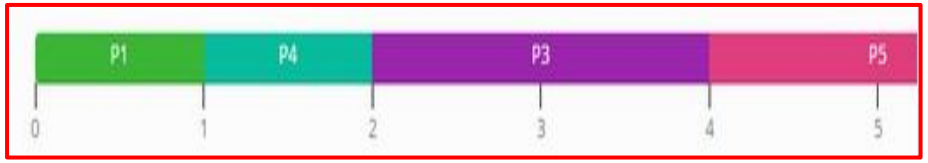
Process	Arrival Time	Execute Time	Service Time

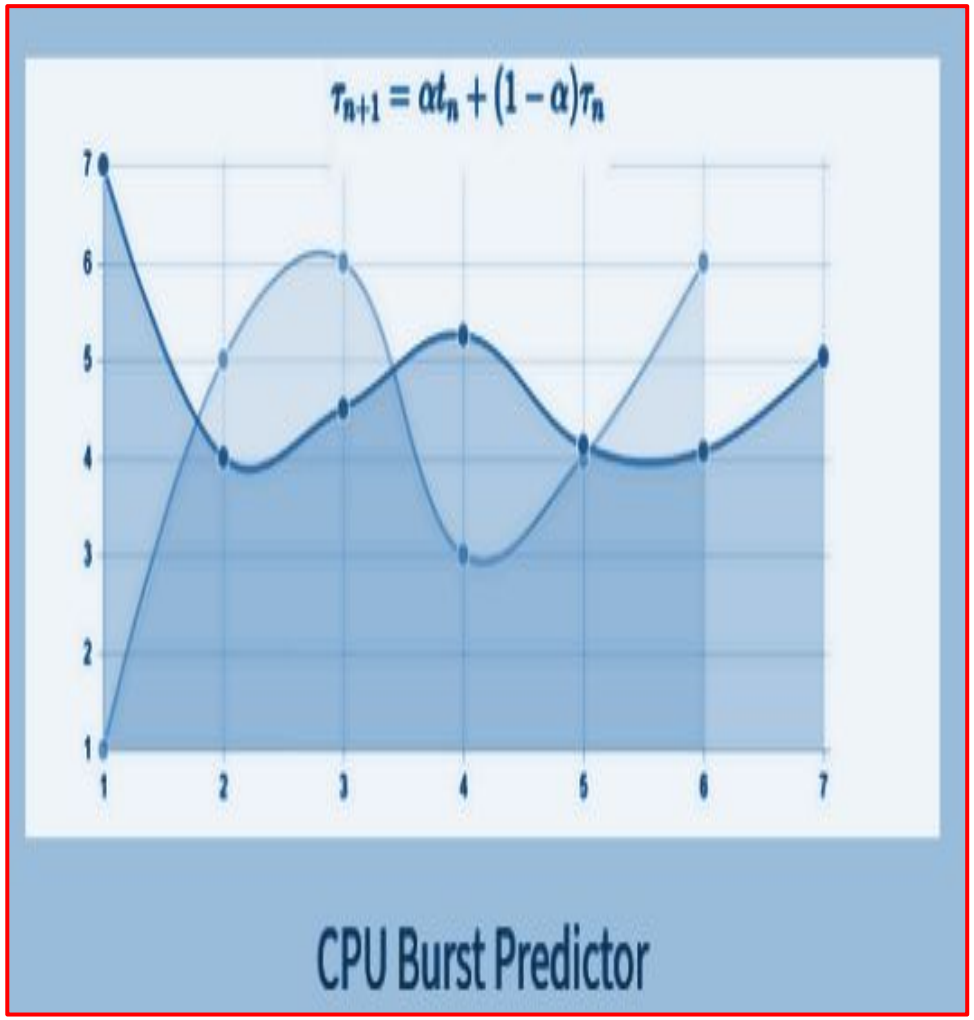
Process	Execution time	Arrival time

Process	Arrival Time	Execution Time	Priority	Service Time

GANTT Chart







Sample Schemas and Tools for solving Scheduling Problems

- **Arrival Time**: Time at which the process arrives in the ready queue.
- **Completion Time**: Time at which process completes its execution.
- **Burst Time**: Time required by a process for CPU execution.
- **Turn Around Time (TAT)** : Time Difference between completion time and arrival time.
  - **Turn Around Time** = **Completion Time** – **Arrival Time**
- **Waiting Time(W.T)**: Time Difference between turn around time and burst time.
  - **Waiting Time** = **Turnaround Time** – **Burst Time**

Preemptive Priority Scheduling

Characteristics of **Preemptive Priority** Scheduling

A CPU algorithm that schedules processes based on priority.

- In **Preemptive** Priority Scheduling, at the time of arrival of a process in the ready queue, its Priority is compared with the priority of the other processes present in the ready queue as well as with the one which is being executed by the CPU at that point of time.
- The one with the **highest** priority among all the available processes will be given the CPU next, whenever the decision is to be made
- The difference between preemptive priority scheduling and non preemptive priority scheduling is that, in the **preemptive** priority scheduling, the **job** which is being executed can be **stopped** at the arrival of a higher priority job
- Once all the jobs get available in the ready queue, the algorithm will behave as non-preemptive priority scheduling, which means the job scheduled will run till the completion and no preemption will be done.

Preemptive Priority Scheduling

Advantages of **Preemptive Priority Scheduling**

- Priority scheduling is simple to understand.
- Priority scheduling is a user-friendly algorithm.
- Based on priority, processes are executed. So, the process which has the highest priority does not need to wait for more time
- Priorities in the Priority scheduling are chosen on the basis of time requirements, memory requirements, and user preferences.

Disadvantages of **Preemptive Priority Scheduling**

- When the multiple processes have the same priorities, then we have to apply another scheduling algorithm.
- In priority scheduling, if the system is crashed, then all low-priority processes that are not yet completed will also get lost.
- Another disadvantage of Priority Scheduling is **starvation**. The problem of starvation is a situation that arises when a process does not get the required resources like CPU because another process is holding the resources, and it waits for a long time for the CPU.
- **'Aging'** is a technique that is used to remove the problem of **starvation**.
- In aging, the system automatically increases the priorities of the processes, waiting for a long time to complete its execution.

Preemptive Priority Scheduling

PES UNIVERSITY ONLINE

Example

Round Robin Scheduling

Characteristics of Round Robin Scheduling

A CPU algorithm that schedules processes based on Time Quantum.

- Round Robin is the preemptive process scheduling algorithm.
- Each process is provided a fix time to execute, it is called a **quantum**.
- Once a process is executed for a given time period, it is preempted and other process executes for a given time period.
- Context switching is used to save states of preempted processes.
- The process that is preempted is added to the end of the queue.
- Round robin is a hybrid model which is clock-driven
- Time slice should be minimum, which is assigned for a specific task that needs to be processed. However, it may differ based on OS.
- It is a real time algorithm which responds to the event within a specific time limit.
- Round robin is one of the oldest, fairest, and easiest algorithm.
- Widely used scheduling method in traditional OS.

Round Robin Scheduling

Characteristics of Round Robin Scheduling

A CPU algorithm that schedules processes based on Time Quantum.

- 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.
- Timer interrupts every quantum to schedule next process
- Performance
  a. large quantum => FIFO
  b. Quantum Small => Quantum must be large with respect to context switch, otherwise overhead is too high

Round Robin Scheduling

Advantages of Round Robin Scheduling

- It doesn't face the issues of starvation or convoy effect.
- All the jobs get a fair allocation of CPU.
- It deals with all process without any priority
- If you know the total number of processes on the run queue, then you can also assume the worst-case response time for the same process.
- This scheduling method does not depend upon burst time. That's why it is easily implementable on the system.
- Once a process is executed for a specific set of the period, the process is preempted, and another process executes for that given time period.
- Allows OS to use the Context switching method to save states of preempted processes.
- It gives the best performance in terms of average response time.

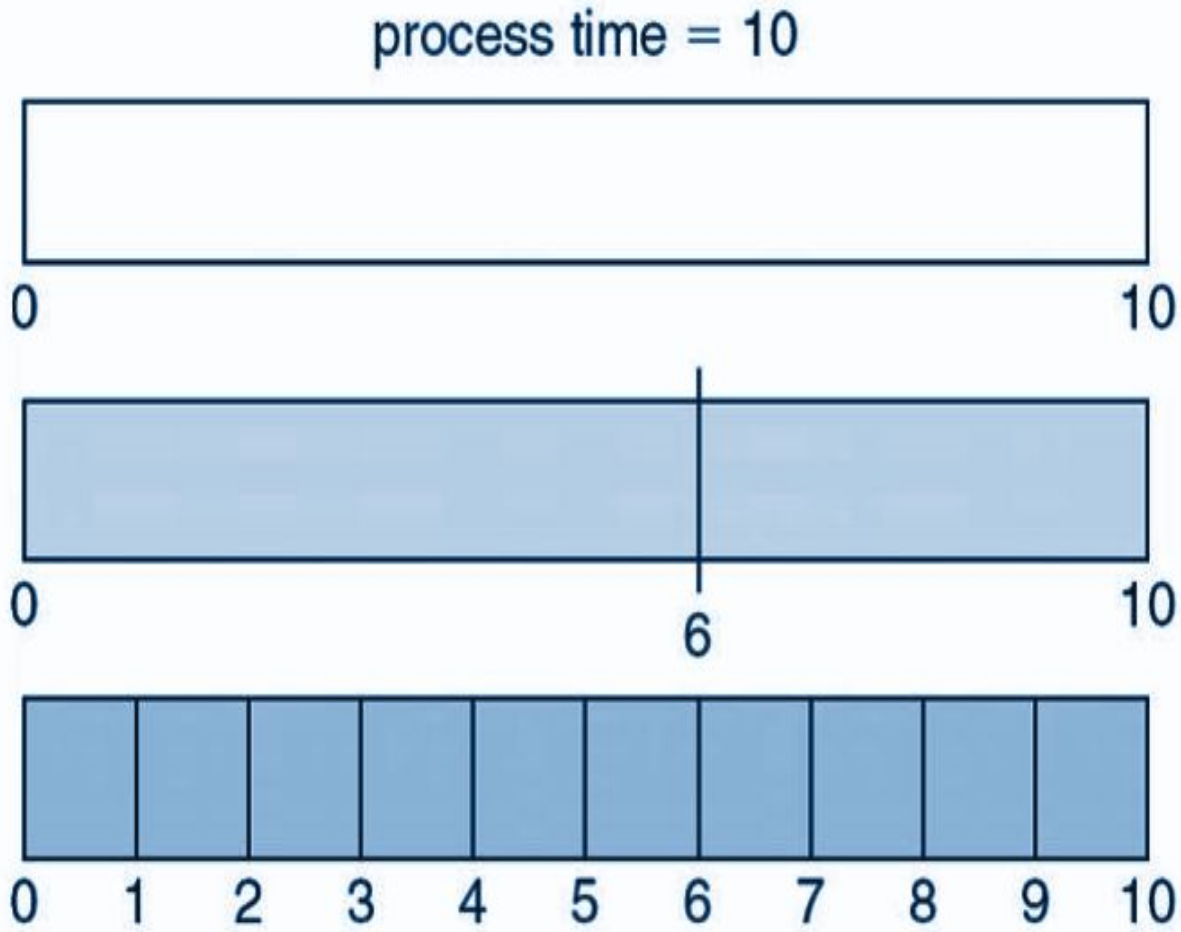
Disadvantages of Round Robin Scheduling

- If slicing time of OS is low, the processor output will be reduced.
- This method spends more time on context switching
- Its performance heavily depends on time quantum.
- Priorities cannot be set for the processes.
- Round-robin scheduling doesn't give special priority to more important tasks.
- Decreases comprehension
- Lower time quantum results in higher the context switching overhead in the system.
- Finding a correct time quantum is a quite difficult task in this system.

Round Robin Scheduling

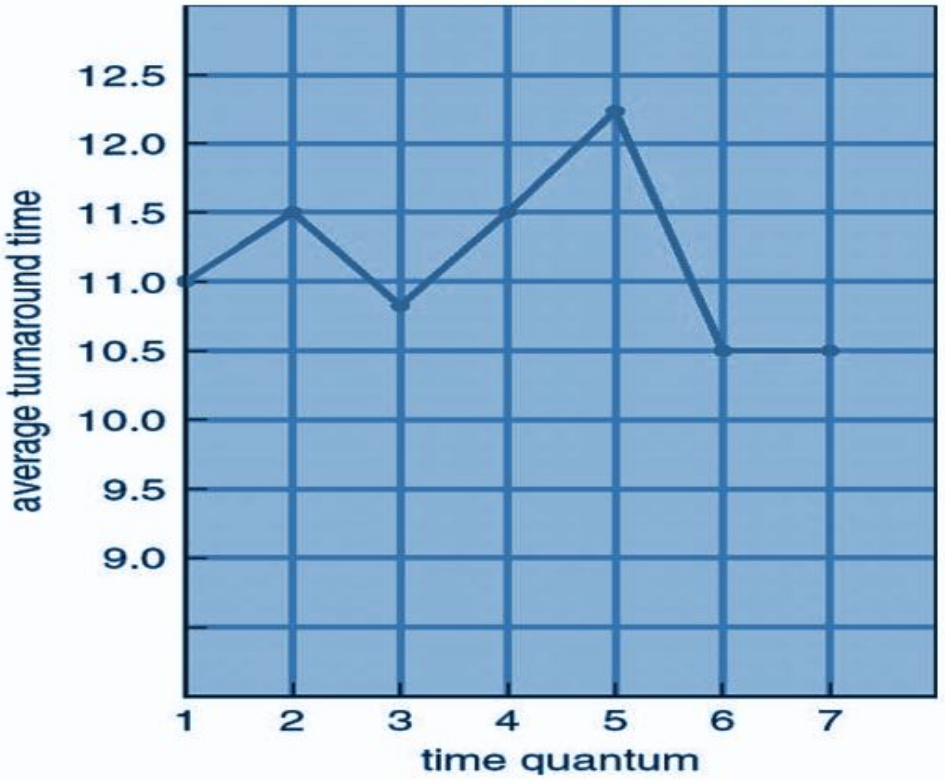
Example

Round Robin Scheduling and Context Switch



quantum	context switches
12	0
6	1
1	9

Round Robin Scheduling and Context Switch



process	time
P_1	6
P_2	3
P_3	1
P_4	7

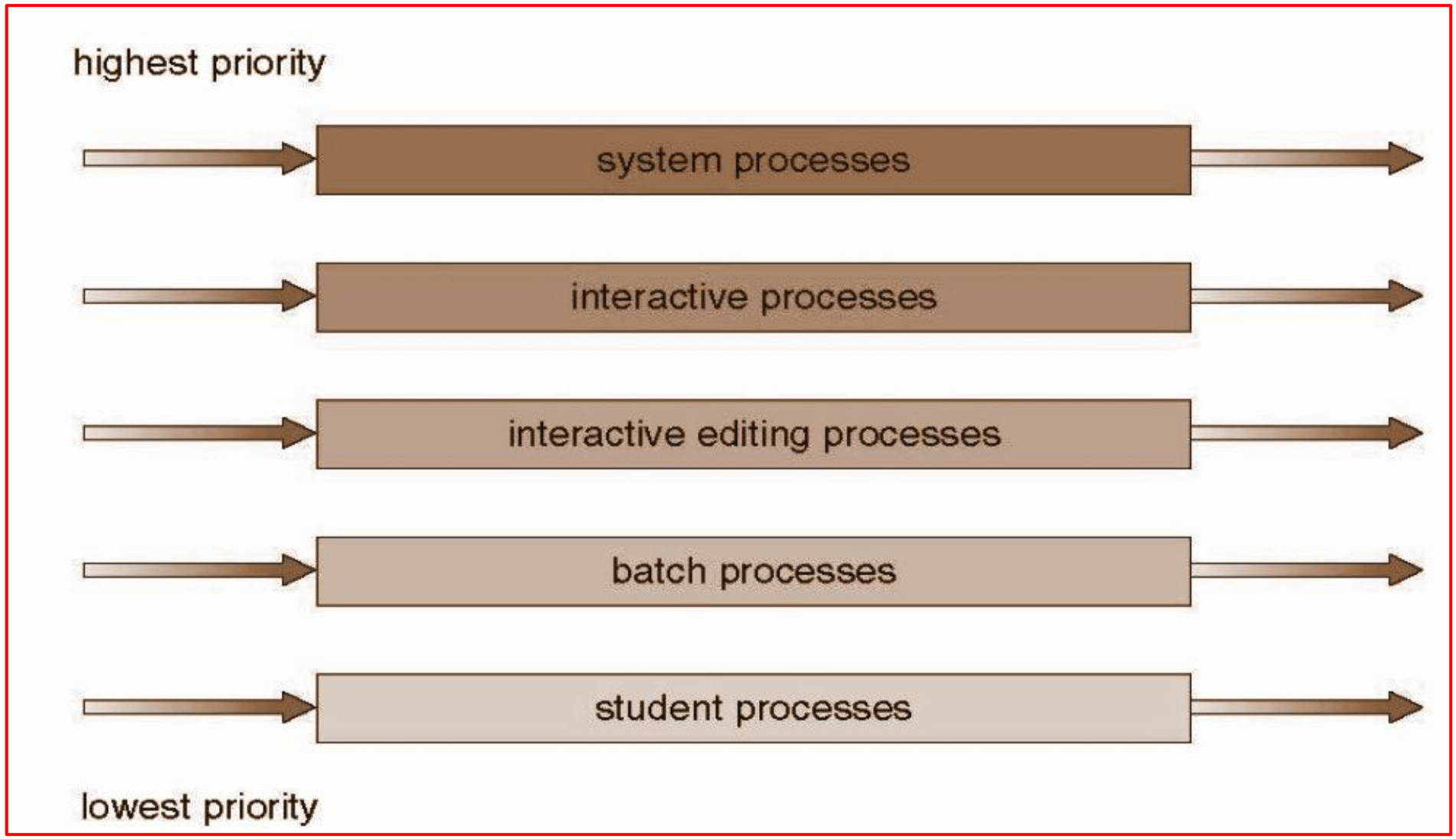
80% of CPU bursts should be shorter than quantum

Slides Adapted from Operating System Concepts 9/e © Authors

Multilevel Queues

- 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 Queues

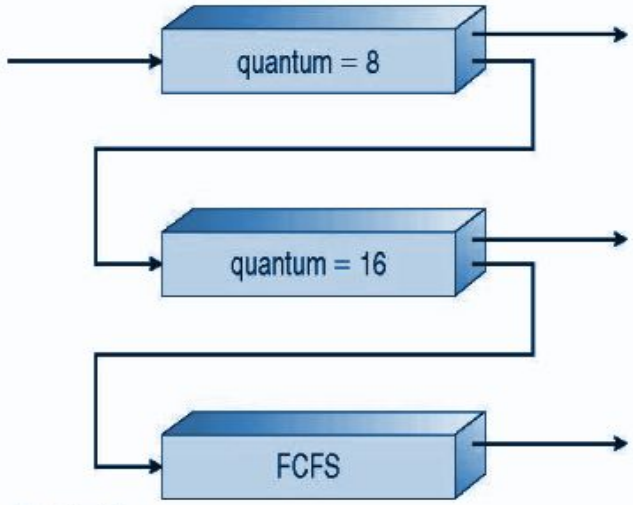


Multilevel Feedback Queues

- 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

Multilevel Feedback Queues

- 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

Topics Outline

- Preemptive Priority Scheduling (PPS)
- Round Robin Scheduling (RR)
- Multilevel Queues
- Multilevel Feedback Queue

THANK YOU

Nitin V Pujari
Faculty, Computer Science
Dean - IQAC, PES University

nitin.pujari@pes.edu

For Course Deliverables by the Anchor Faculty click on www.pesuacademy.com