Multi Level Queue

Nirmala Shinde Baloorkar
Assistant Professor
Department of Computer Engineering

SOMAIYA VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

Outline

- Multilevel Queue Scheduling (MLQ)
- How MLQ Works?
- Multilevel Feedback Queue Scheduling (MLFQ)
- Comparison: MLQ vs. MLFQ
- Advantages & Disadvantages
- Real-World Applications

Multi Level Queue

- MLQ scheduling classifies processes into multiple fixed queues / groups.
- Common division is
  - **foreground** (interactive)
  - **background** (batch)

How MLQ Scheduling Works?

- **Queue Classification**:
  - Foreground (interactive) vs. Background (batch).
- **Queue Assignment**:
  - Processes are assigned based on priority/type.
- **Scheduling**:
  - Each queue follows a distinct scheduling algorithm.
- **Execution Order**:
  - Fixed priority or time-sharing decides execution sequence.

Example 1 - MLQ

- Queue 1 has a higher priority than Queue 2, and Round Robin is used in Queue 1 (Time Quantum=3), whereas first come, first served is used in Queue 2.
- **If Q1 has processes, they execute first before Q2.**

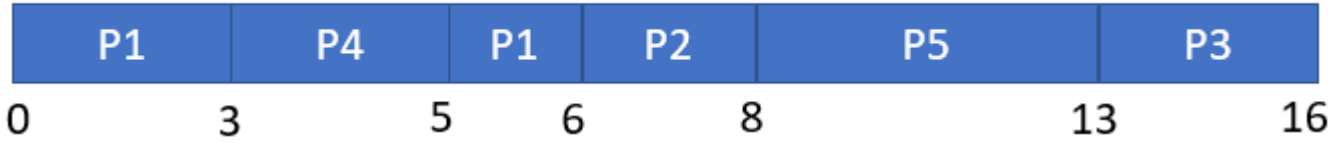
Process	Arrival Time	Burst Time	Queue No
P1	0	4	1
P2	1	2	2
P3	2	3	2
P4	2	2	1
P5	8	5	1

Example 1 - MLQ

- Q1 RR - 3
- Q2 FCFS

Process	Arrival Time	Burst Time	Queue No
P1	0	4	1
P2	1	2	2
P3	2	3	2
P4	2	2	1
P5	8	5	1

- Gantt Chart

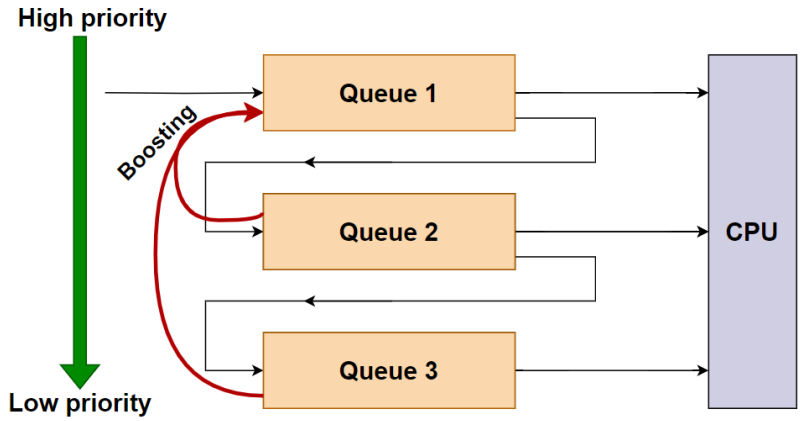


Problems with MLQ

- **Starvation of Lower-Priority Processes**
  - If high-priority queues are always busy, lower-priority processes **may never get CPU time**.
- **Does Not Adapt to Process Behavior**
  - In MLQ, a process with high burst time remains in a lower-priority queue forever, even if its CPU bursts decrease over time.

Multilevel Feedback Queue (MLFQ)

- In a multi-level queue-scheduling algorithm, processes are permanently assigned to a queue.
- Idea: Allow processes to move among various queues.



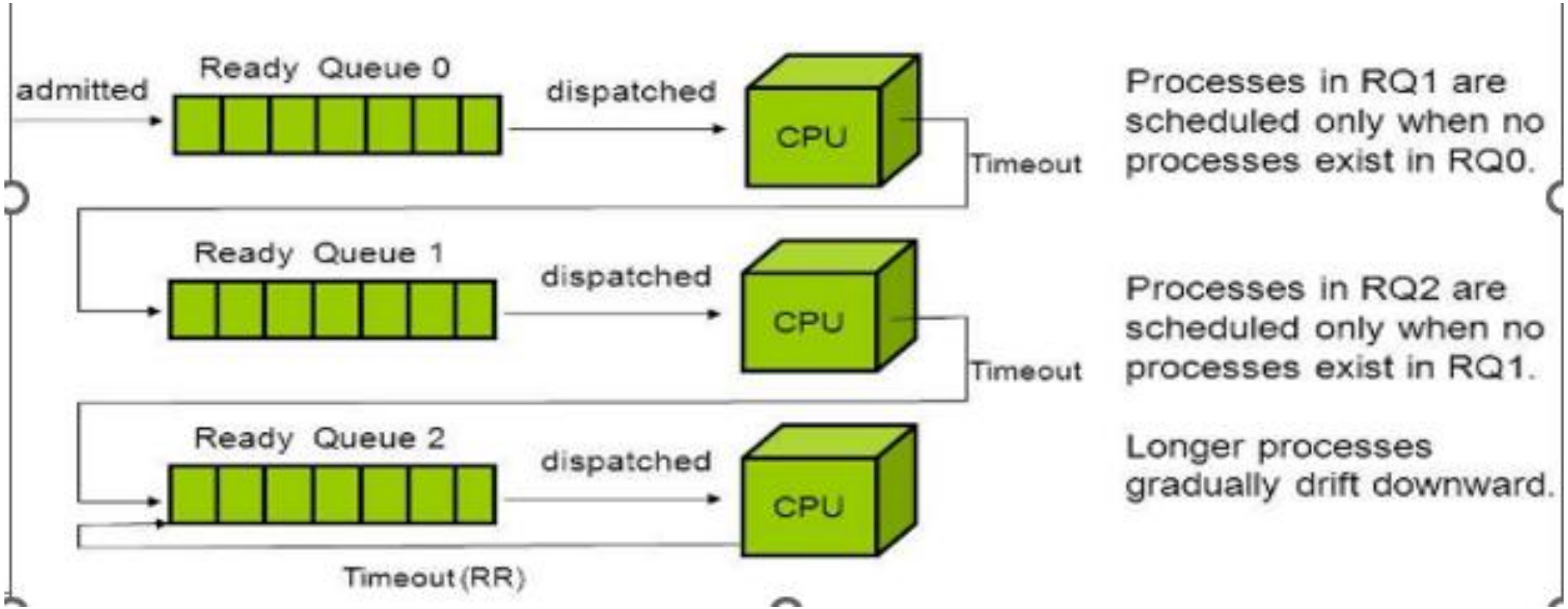
- Uses multiple queues with varying priorities to manage process execution.
- It dynamically adjusts priorities based on process behaviour, promoting or demoting processes between queues.

Multilevel Feedback Queue (continued)

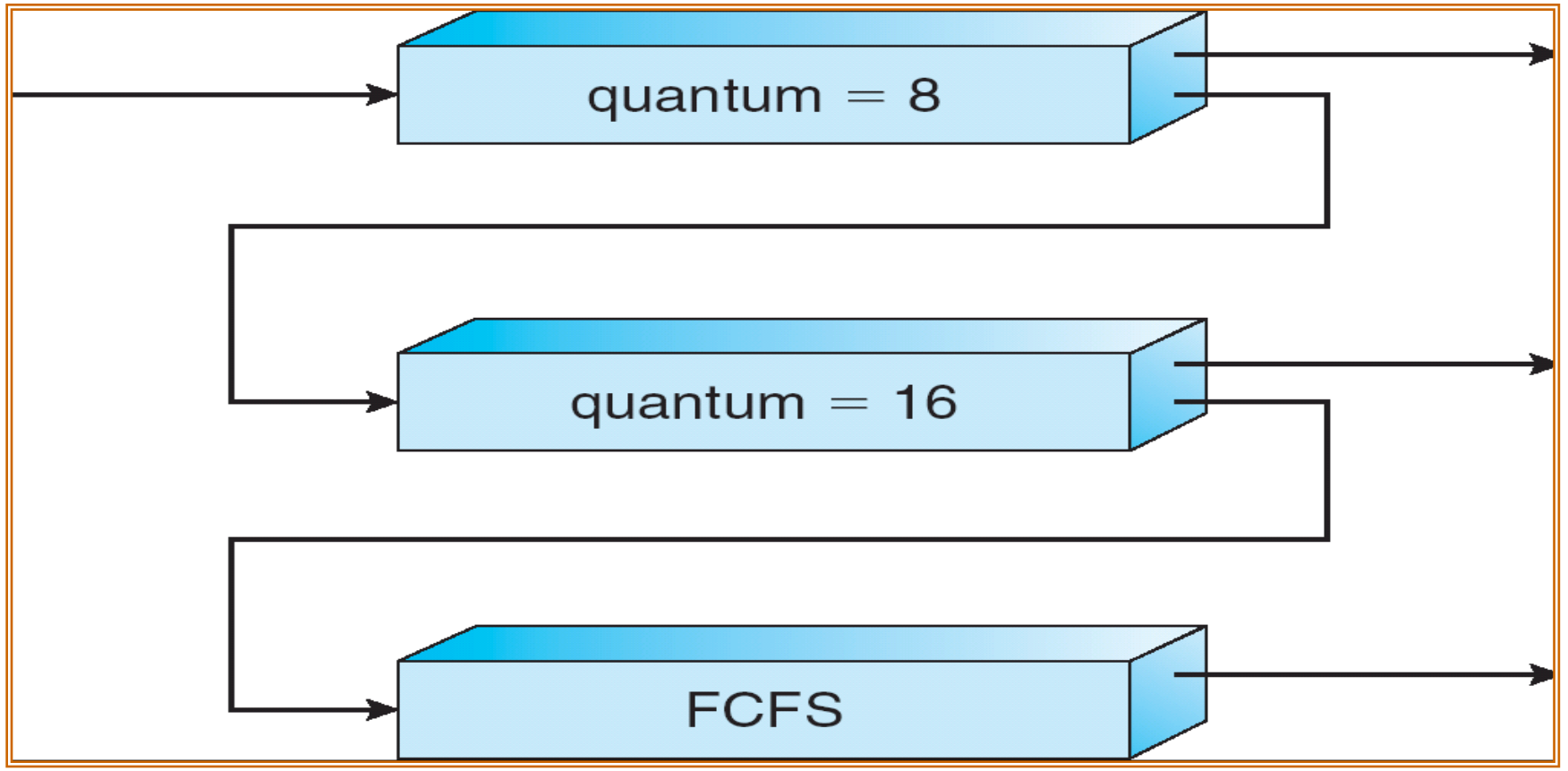
- Multilevel feedback queue scheduler is defined by the following parameters:
  - number of queues
  - scheduling algorithms for each queue
  - method used to determine when to upgrade a process
  - method used to determine when to demote a process
  - method used to determine which queue a process will enter when that process needs service

- The scheduler can be configured to match the requirements of a specific system.

Multilevel Feedback Queue (continued)



Example of Multilevel Feedback Queue



Example 1 - MLFQ

- **Queue Structure:**
  - **Q1 (Priority 1, Round Robin, Time Quantum = 2)**
  - **Q2 (Priority 2, Round Robin, Time Quantum = 4)**
  - **Q3 (Priority 3, FCFS)**

Process	Arrival Time	Burst Time
P1	0	6
P2	1	4
P3	2	8
P4	3	4

Example 1 – MLFQ (continued)

Process	Arrival Time	Burst Time
P1	0	6
P2	1	4
P3	2	8
P4	3	4

- Timer Q1=2, Q2=4,Q3=FCFS
- A process enters **Q1 first**. If it is not completed within its time quantum, it moves to **Q2**. If it still needs more CPU time, it moves to **Q3** (FCFS).
- Gantt Chart

```
| P1 | P2 | P3 | P4 | P1 | P2 | P3 | P4 | P3 |
0    2    4    6    8    12   14   18   20   22
```

- Calculate Turnaround and Waiting Time

Example 2 - MLFQ

- **Queue Structure:**
  - **Q1 (Priority 1, Round Robin, Time Quantum = 1)**
  - **Q2 (Priority 2, Round Robin, Time Quantum = 2)**
  - **Q3 (Priority 3, Round Robin, Time Quantum = 4)**

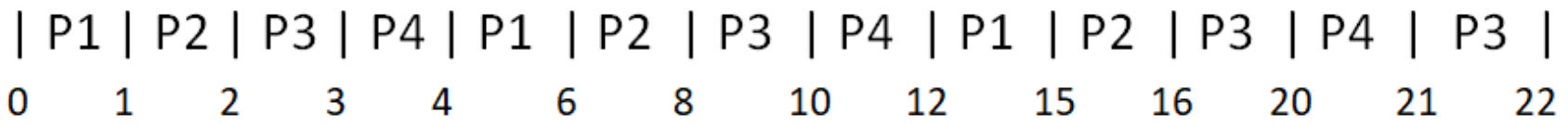
Process	Arrival Time	Burst Time
P1	0	6
P2	1	4
P3	2	8
P4	3	4

Example 2 – MLFQ (continued)

Process	Arrival Time	Burst Time
P1	0	6
P2	1	4
P3	2	8
P4	3	4

- Timer Q1=1, Q2=2, Q3=4

- Gantt Chart



Example 3 Practice– MLFQ

- **Queue Structure:**
  - **Q1 (Priority 1, Round Robin, Time Quantum = 1)**
  - **Q2 (Priority 2, Round Robin, Time Quantum = 1)**
  - **Q3 (Priority 3, Round Robin, Time Quantum = 1)**

Process	Arrival Time	Burst Time
P1	0	6
P2	1	4
P3	2	8
P4	3	4

Example 4 Practice - MLFQ

- **Queue Structure:**
  - **Q1 (Priority 1, Round Robin, Time Quantum = 3)**
  - **Q2 (Priority 2, Round Robin, Time Quantum = 6)**
  - **Q3 (Priority 3, FCFS)**

Process	Arrival Time	Burst Time
P1	0	12
P2	2	4
P3	4	8

Difference b/w MLQ and MLFQ

Multilevel queue scheduling (MLQ)	Multilevel feedback queue scheduling (MLFQ)
It is queue scheduling algorithm in which ready queue is partitioned into several smaller queues and processes are assigned permanently into these queues. The processes are divided on basis of their intrinsic characteristics such as memory size, priority etc.	In this algorithm, ready queue is partitioned into smaller queues on basis of CPU burst characteristics. The processes are not permanently allocated to one queue and are allowed to move between queues.
In this algorithm queue are classified into two groups, first containing background processes and second containing foreground processes. 80% CPU time is given to foreground queue using Round Robin Algorithm and 20% time is given to background processes using First Come First Serve Algorithm.	Here, queues are classified as higher priority queue and lower priority queues. If process takes longer time in execution it is moved to lower priority queue. Thus, this algorithm leaves I/O bound and interactive processes in higher priority queue.
The priority is fixed in this algorithm. When all processes in one queue get executed completely then only processes in other queue are executed. Thus, starvation can occur.	The priority for process is dynamic as process is allowed to move between queue. A process taking longer time in lower priority queue can be shifted to higher priority queue and vice versa. Thus, it prevents starvation.
Since, processes do not move between queues, it has low scheduling overhead and is inflexible.	Since, processes are allowed to move between queues, it has high scheduling overhead and is flexible.

Characteristics of Various Scheduling Policies

Scheduling Policy	Selection Function	Decision Mode	Throughput	Response Time	Overhead	Effect on Processes	Starvation
First-Come, First-Served (FCFS)	Arrival time	Non-preemptive	Low for long processes	High for long processes	Minimal	Simple, long wait times	No, but convoy effect
Shortest Job First (SJF)	Shortest next CPU burst	Non-preemptive	High	Low for short processes	Requires knowledge of process length	Efficient for short processes	Yes, long processes may starve
Shortest Remaining Time First (SRTF)	Shortest remaining CPU burst	Preemptive	High	Low for short processes	High, frequent context switching	Efficient for short processes	Yes, long processes may starve

Characteristics of Various Scheduling Policies

Scheduling Policy	Selection Function	Decision Mode	Throughput	Response Time	Overhead	Effect on Processes	Starvation
Priority Scheduling	Highest priority	Preemptive or Non-preemptive	Varies based on priority	Low for high-priority processes	Requires priority assignment	High-priority processes favored	Yes, low-priority processes may starve
Round Robin (RR)	Time quantum (fixed time slice)	Preemptive	Moderate	Moderate	High, frequent context switching	Fair time-sharing	No, each process gets CPU time

Characteristics of Various Scheduling Policies

Scheduling Policy	Selection Function	Decision Mode	Throughput	Response Time	Overhead	Effect on Processes	Starvation
Multilevel Queue	Based on queue priority	Preemptive or Non-preemptive	Varies based on queue configuration	Varies	High, managing multiple queues	Processes categorized into different queues	Yes, lower-priority queues may starve
Multilevel Feedback Queue	Based on aging and feedback	Preemptive	High	Low for interactive processes	Very high, complex management	Dynamic adjustment of process priority	Reduced, processes can move between queues

Question ?