Great Place To Work®
Certified
FEB 2023 – FEB 2024
INDIA ™

UPES
UNIVERSITY OF TOMORROW

nirf RANKING 2023

#RANKED 52 IN INDIA

QS WORLD UNIVERSITY RANKINGS

NO.1 PVT. UNIVERSITY IN ACADEMIC REPUTATION IN INDIA

ACCREDITED WITH GRADE A NAAC

ACCREDITED **GRADE 'A'** BY NAAC

E-LEAD

PERFECT SCORE OF **150/150** AS A TESTAMENT TO EXCEPTIONAL E-LEARNING METHODS

#University Category

Unit 5 : Process and Thread Management

Lecture 5

Submitted by:

Syed Sajid Hussain

School of Computer Science

UPES, Dehradun

India

Table of Contents

Learning & Course Outcomes

LO1:Understand the principles and challenges of real-time scheduling.

LO2:Differentiate between Rate-Monotonic, Earliest-Deadline-First, and Proportional Share scheduling algorithms.

LO3:Identify and explain the different types of latencies affecting real-time systems.

Course Outcomes

- CO2: Evaluate and analyze process and thread scheduling techniques, discerning their benefits and challenges.

Real Time Scheduling

Real Time systems are categorized into two types: Soft and Hard real time systems.

- **Soft Real Time System:** It provides no guarantee as to when a critical real-time process will be scheduled. They guarantee only that the process will be given preference over noncritical processes.
- **Hard real-time systems:** A task must be serviced by its deadline. Service after the deadline has expired is the same as no service at all-time.

Issues related to Real Time CPU scheduling:

1. Minimizing Latency
2. Priority Based Scheduling
3. Rate-Monotonic Scheduling
4. Earliest-Deadline-First Scheduling
5. Proportional Share Scheduling
6. POSIX Real-Time Scheduling

UPES
UNIVERSITY OF TOMORROW

Real Time Scheduling

Minimizing Latency
In an Event-Driven nature of **Real Time System**, the system is waiting for an event to occur. Event may arise in software or hardware. When an event occurs, the system must respond to that event and service the event as quickly as possible.
Event Latency is the amount of time that elapses from when an event occurs to when it is serviced.
Two types of latencies that affect the performance of the real time systems:

1. Interrupt latency

- Interrupt latency refers to the period of time from the arrival of an interrupt at the CPU to the start of the routine that services the interrupt.
- Interrupt latency must be minimized to ensure that real time tasks receive immediate action.

2. Dispatch latency

- The amount of time required for the scheduling dispatcher to stop one process and start another is known as dispatch latency.
- Real-time operating systems minimize dispatch latency to provide real-time tasks with immediate access to the CPU.

Real Time Scheduling

Priority Based Scheduling

In real time operating system each process is having a deadline. Hence the scheduler for a real-time operating system must support a priority-based algorithm with preemption.

- The process which is having lowest deadline time will be given highest priority.
- Preemptive-priority-based scheduler only guarantees soft real-time functionality.
- Hard real-time systems must further guarantee that real-time tasks will be serviced in accord with their deadline requirements. Hence it uses Admission control technique. In this type of scheduling a process may have to announce its deadline requirements to the scheduler.
- The Admission-control algorithm scheduler either admits the process or rejects it. If process is admitted then there is a guarantee that the process will complete on time. If a process is rejected then the task will be not serviced by its deadline.

Rate Monotonic Scheduling

The rate-monotonic scheduling algorithm schedules periodic tasks using a static priority policy with pre-emption.

- Upon entering the system, each periodic task is assigned a priority inversely based on itsperiod.
- Shorter period will be given higher priority. Longer period will be given lower priority.
- Rate-monotonic scheduling assumes that the processing time of a periodic process is the same for each CPU burst (i.e.) every time a process acquires the CPU, the duration of its CPU burst is the same.

UPES
UNIVERSITY OF TOMORROW

Real Time Scheduling

Scheduling Problem 1:

P1 and P2 are two processes and Periodic Times (i.e. **Deadlines**) and processing times as given in the table. The deadline for each process requires that it complete its CPU burst by the start of its next period.

Process	Periodic Time(p)	Processing Time(t)
P1	p1=50	t1=20
P2	p2=100	T2=35

Condition for Scheduling possibility:

We must check whether it is possible to schedule the tasks so that each process meets its deadlines.

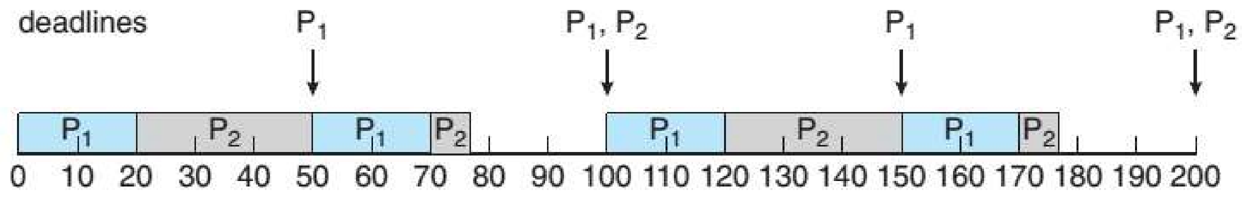
If we measure the CPU utilization of a process *Pi* as the ratio of its burst to its period:

- CPU utilization of *P*1 is 20/50 = 0.40 = **40%**
- CPU utilization of P2 is 35/100 = 0.35 = **35%**
- Total CPU utilization = 40 + 35= **75%**.

We can schedule these tasks in such a way that both meet their deadlines and still leave the CPU with available cycles.

Real Time Scheduling

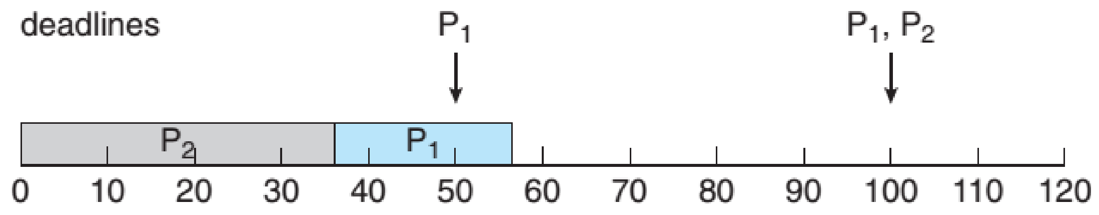
Case 1: By using Rate monotonic Scheduling, we assign P1 a higher priority than P2 because the period of P1 is shorter than that of P2.



- P1 starts first and completes its CPU burst at time 20, thereby meeting its first deadline.
- P2 starts running at this point and runs until time 50.
- At this time, P2 is preempted by P1, although it still has 5 milliseconds remaining in its CPU burst.
- P1 completes its CPU burst at time 70, at which point the scheduler resumes P2.
- P2 completes its CPU burst at time 75, also meeting its first deadline.
- The system is idle until time 100, when P1 is scheduled again.

Real Time Scheduling

Case 2: We assign P2 a higher priority than P1 **(Not using Rate Monotonic Scheduling)**



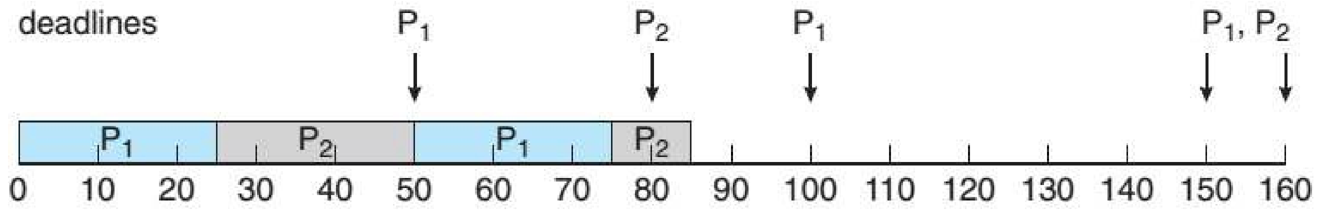
- P2 starts execution first and completes at time 35.
- At this point the process P1 starts and P1 completes its CPU burst at time 55.
- The first deadline for P1 is at time 50, so the scheduler has caused P1 to miss its deadline.

Real Time Scheduling

Scheduling Problem 2:

Process	Periodic Time(p)	Processing Time(t)
P1	p1=50	t1=25
P2	p2=80	T2=35

- P1 having highest priority
- The total CPU utilization of the two processes is (25/50)+(35/80) = 0.94.
- Hence by the formula, the two processes could be scheduled and still leave the CPU with 6 percent available time.



- P1 runs until it completes its CPU burst at time 25.
- Process P2 then begins running and runs until time 50, when it is preempted by P1.
- At this point, P2 still has 10 milliseconds remaining in its CPU burst.
- Process P1 runs until time 75; consequently, P2 misses the deadline for completion of its CPU burst at time 80.

Limitations of Rate monotonic Scheduling: CPU utilization is bounded and it is not always possible fully to maximize CPU resources. The worst-case CPU utilization for scheduling *N* processes is **N(21/N − 1).**

Earliest-Deadline-First Scheduling(EDF)

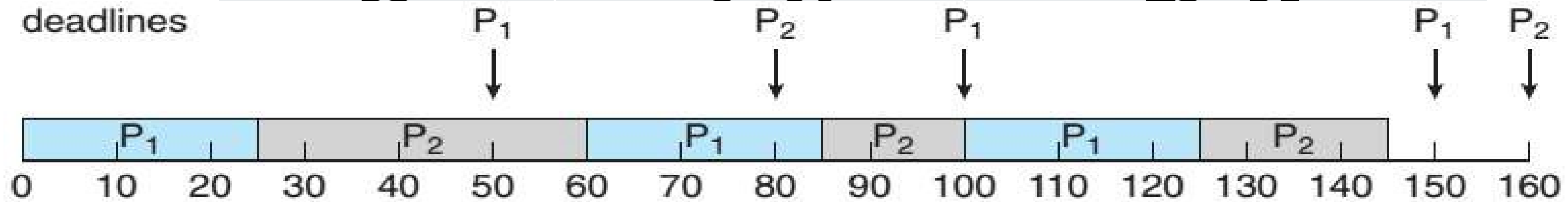
Earliest-Deadline-First Scheduling (EDF)

EDF scheduling dynamically assigns priorities according to deadline.

- Earlier deadline have higher priority. When a process becomes runnable, it must announce its deadline requirements to the system.
- Priorities may have to be adjusted to reflect the deadline of the newly runnable process.

- **Scheduling Problem:**

Process	Periodic Time(p)	Processing Time(t)
P1	p1=50	t1=25
P2	p2=80	t2=35



- Process P1 has the earliest deadline, so its initial priority is higher than that of process P2.
- Process P2 begins running at the end of the CPU burst for P1. However, whereas rate monotonic scheduling allows P1 to preempt P2 at the beginning of its next period at time 50, EDF scheduling allows process P2 to continue running.
- P2 now has a higher priority than P1 because its next deadline (at time 80) is earlier than that of P1 (at time 100). Thus, both P1 and P2 meet their first deadlines.

UPES
UNIVERSITY OF TOMORROW

Earliest-Deadline-First Scheduling(EDF)

- Process P1 again begins running at time 60 and completes its second CPU burst at time 85, also meeting its second deadline at time 100.
- P2 begins running at this point, only to be preempted by P1 at the start of its next period at time 100.
- P2 is preempted because P1 has an earlier deadline (time 150) than P2 (time 160).
- At time 125, P1 completes its CPU burst and P2 resumes execution, finishing at time 145 and meeting its deadline as well.
- The system is idle until time 150, when P1 is scheduled to run once again.

Note: EDF scheduling is theoretically optimal but practically it is impossible.

- Theoretically, it can schedule processes so that each process can meet its deadline requirements and CPU utilization will be 100 percent.
- In practice it is impossible to achieve this level of CPU utilization due to the cost of context switching between processes and interrupt handling.

UPES
UNIVERSITY OF TOMORROW

MCQ’s

1. What is the primary goal of real-time scheduling?
a) To maximize CPU utilization
b) To ensure timely execution of real-time tasks
c) To minimize power consumption
d) To maximize throughput

2. Which of the following scheduling algorithms assigns priorities dynamically based on deadlines?
a) Rate-Monotonic Scheduling
b) First-Come, First-Served (FCFS)
c) Earliest-Deadline-First (EDF)
d) Round Robin (RR)

UPES
UNIVERSITY OF TOMORROW

MCQ’s

3. In Rate-Monotonic Scheduling, the priority of a task is:
a) Based on its deadline
b) Inversely proportional to its period
c) Directly proportional to its burst time
d) Randomly assigned

4. Which latency refers to the time it takes from the arrival of an interrupt to the start of the routine that services the interrupt?
a) Dispatch latency
b) Context-switch latency
c) Event latency
d) Interrupt latency

Answers

1. b) To ensure timely execution of real-time tasks
2. c) Earliest-Deadline-First (EDF)
3. b) Inversely proportional to its period
4. d) Interrupt latency

UPES
UNIVERSITY OF TOMORROW

Summary/Key Points

- Real-Time Scheduling: Soft (no guarantee on scheduling) and Hard (tasks must meet deadlines).
- Scheduling Algorithms: Rate-Monotonic (static priority for periodic tasks), Earliest-Deadline-First (prioritizes closest deadlines), Proportional Share (distributes CPU time proportionally).
- Latency Issues: Event Latency (time from event to service), Interrupt Latency (time from interrupt to service start), Dispatch Latency (time to switch processes).

References

1. D. M. Dhamdhere, Operating Systems - A Concept Based Approach, Third Edition, Tata McGraw-Hill, 2012
2. Avi Silberschatz, Peter Baer Galvin, Greg Gagne, Operating System Concepts, John Wiley & Sons, Ninth edition, 2012
3. M.J.Bach, The design of UNIX operating system, Prentice Hall of India, 1986
4. Uresh Vahalia, UNIX Internals : The New Frontiers, Pearson Education India, 2008
5. A.S.Tanenbaum, Operating System : Design and Implementation,Prentice Hall of India, Third edition, 2009

UPES
UNIVERSITY OF TOMORROW

Coming up-next lecture

UPES
UNIVERSITY OF TOMORROW

UPES
UNIVERSITY OF TOMORROW

Thank You

UPES
UNIVERSITY OF TOMORROW