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

UPES

UNIVERSITY OF TOMORROW

nirf RANKING 2023

#**RANKED 52** IN INDIA

#University Category

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 EXECEPTIONAL E-LEARNING METHODS

Unit 5 : Process and Thread Management

Lecture 3

Submitted by:

Syed Sajid Hussain

School of Computer Science

UPES, Dehradun

India

UPES
UNIVERSITY OF TOMORROW

Table of Contents

UPES
UNIVERSITY OF TOMORROW

Learning & Course Outcomes

- LO1: To understand process concepts
- LO2: To understand process management

Course Outcomes

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

UPES
UNIVERSITY OF TOMORROW

Process Scheduling

Process scheduling is the basis of Multi-programmed operating systems. By switching the CPU among processes, the operating system can make the computer more productive

- In a single-processor system, only one process can run at a time. Others must wait until the CPU is free and can be rescheduled.
- The CPU will sit idle and waiting for a process that needs an I/O operation to complete. If the I/O operation completes then only the CPU will start executing the process. A lot of CPU time has been wasted with this procedure.
- The objective of multiprogramming is to have some process always running to maximize CPU utilization.
- When several processes are in main memory, if one processes is waiting for I/O then the operating system takes the CPU away from that process and gives the CPU to another process. Hence there will be no wastage of CPU time.

Process Scheduling

Concepts of Process Scheduling

1. CPU–I/O Burst Cycle
2. CPU Scheduler
3. Pre-emptive Scheduling
4. Dispatcher

CPU–I/O Burst Cycle

Process execution consists of a **cycle** of CPU execution and I/O wait.

- Process execution begins with a **CPU burst**. That is followed by an **I/O burst.**
- Processes alternate between these two states.
- The final CPU burst ends with a system request to terminate execution.
- Hence the **First cycle** and **Last cycle** of execution must be CPU burst.

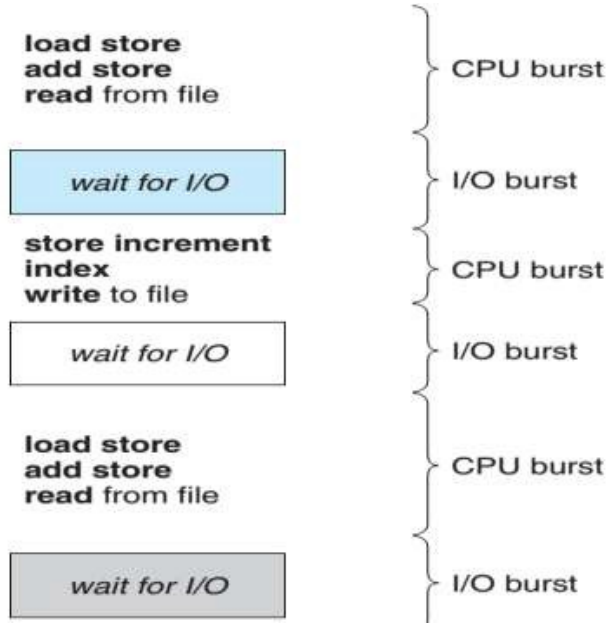


Fig: Process Scheduler

Process Scheduling

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**.

Preemptive Scheduling

CPU-scheduling decisions may take place under the following four cases:

1. When a process switches from the running state to the waiting state. Example: as the result of an I/O request or an invocation of wait( ) for the termination of a child process.
2. When a process switches from the running state to the ready state. Example: when an interrupt occurs
3. When a process switches from the waiting state to the ready state. Example: at completion of I/O.
4. When a process terminates. For situations 2 and 4 are considered as **Pre-emptive scheduling** situations. Mach OS X,WINDOWS 95 and all subsequent versions of WINDOWS are using Preemptive scheduling.

Process Scheduling

Dispatcher

The dispatcher is the module that gives control of the CPU to the process selected by the short-term scheduler. Dispatcher function involves:

1. Switching context
2. Switching to user mode
3. Jumping to the proper location in the user program to restart that program.

The dispatcher should be as fast as possible, since it is invoked during every process switch. The time it takes for the dispatcher to stop one process and start another process running is known as the **Dispatch Latency**.

Process Scheduling

Different CPU-scheduling algorithms have different properties, and the choice of a particular algorithm may favor one class of processes over another.

Many criteria have been suggested for comparing CPU-scheduling algorithms:

- **CPU utilization:** CPU must be kept as busy as possible. CPU utilization can range from 0 to 100 percent. In a real system, it should range from 40 to 90 percent.
- **Throughput:** The number of processes that are completed per time unit.
- **Turn-Around Time:** It is the interval from the time of submission of a process to the time of completion. Turnaround time is the sum of the periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU and doing I/O.
- **Waiting time:** It is the amount of time that a process spends waiting in the ready queue.
- **Response time:** It is the time from the submission of a request until the first response is produced. Interactive systems use response time as its measure.

Note: It is desirable to maximize CPU utilization and Throughput and to minimize Turn- round Time, Waiting time and Response time.

UPES
UNIVERSITY OF TOMORROW

Process Scheduling Algorithms

Process/CPU scheduling deals with the problem of deciding which of the processes in the ready queue is to be allocated the CPU. Different CPU-scheduling algorithms are:

1. First-Come, First-Served Scheduling (FCFS)
2. Shortest-Job-First Scheduling (SJF)
3. Priority Scheduling
4. Round Robin Scheduling
5. Multilevel Queue Scheduling
6. Multilevel Feedback Queue Scheduling

First-Come, First-Served Scheduling (FCFS)

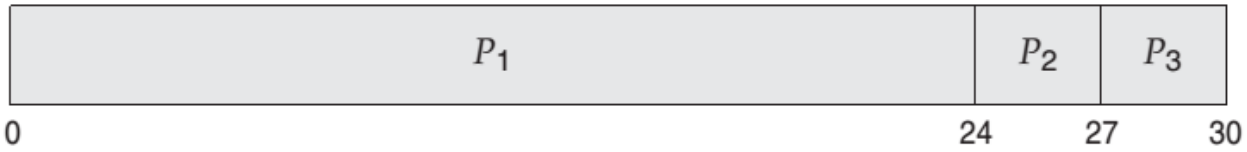
In FCFS, the process that requests the CPU first is allocated the CPU first.

- FCFS scheduling algorithm is Non-preemptive.
- Once the CPU has been allocated to a process, it keeps the CPU until it releases the CPU.
- FCFS can be implemented by using FIFO queues.
- When a process enters the ready queue, its PCB is linked onto the tail of the queue.
- When the CPU is free, it is allocated to the process at the head of the queue.
- The running process is then removed from the queue.

Example:1 Consider the following set of processes that arrive at time 0. The processes arearrived in the order P1, P2, P3, with the length of the CPU burst given in milliseconds.

Process	Burst Time
P1	24
P2	3
P3	3

Gantt Chart for FCFS is:



First-Come, First-Served Scheduling (FCFS)

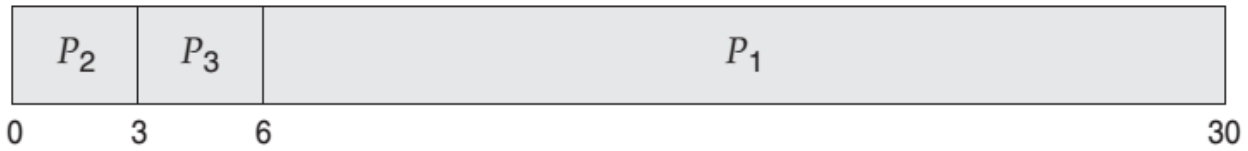
The average waiting time under the FCFS policy is often quite long.

- The waiting time is 0 milliseconds for process *P*1, 24 milliseconds for process *P*2 and 27 milliseconds for process *P*3.
- Thus, the average waiting time is (0 + 24 + 27)/3 = 17 milliseconds.

Convoy Effect in FCFS

Convoy effect means, when a big process is executing in CPU, all the smaller processes must have to wait until the big process execution completes. This will affect the performance of the system.

Example:2 Let us consider same example above but with the processes arrived in the order P2, P3, P1.



The processes coming at P2, P3, P1 the average waiting time (6 + 0 + 3)/3 = 3 milliseconds whereas the processes are coming in the order P1, P2, P3 the average waiting time is 17 milliseconds.

Disadvantage of FCFS:

FCFS scheduling algorithm is Non-preemptive, it allows one process to keep CPU for long time. Hence it is not suitable for time sharing systems.

UPES
UNIVERSITY OF TOMORROW

Shortest-Job-First Scheduling(SJF)

Shortest-Job-First Scheduling (SJF)

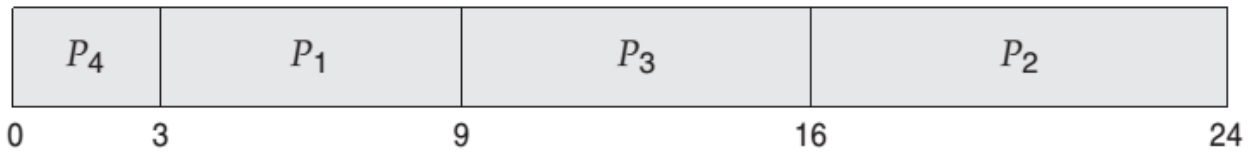
SJF algorithm is defined as **“**when the CPU is available, it is assigned to the process that has the smallest next CPU burst**”.** If the next CPU bursts of two processes are the same, FCFS scheduling is used between two processes.

SJF is also called as **Shortest-Next CPU-Burst** algorithm, because scheduling depends on the length of the next CPU burst of a process, rather than its total length.

Example: Consider the following processes and CPU burst in milliseconds:

Process	Burst Time
P1	6
P2	8
P3	7
P4	3

Gantt Chart of SJF algorithm:



Shortest-Job-First Scheduling(SJF)

- By looking at the table the average waiting time by using SJF algorithm is 7ms.
- SJF gives the minimum average waiting time for a given set of processes. SJF is optimal.
- The average waiting time decreases because moving a short process before long process decrease the waiting time of the short process more than it increases the waiting time ofthe long process.

Difficulty with SJF

The difficulty with the SJF algorithm is —knowing the length of the next CPU request. With Short-Term Scheduling, there is no way to know the length of the next CPU burst. It is not implemented practically.

Waiting Time for Processes:

Process	Burst Time(ms)	Waiting Time
P1	6	3
P2	8	16
P3	7	9
P4	3	0
Average Waiting Time		7ms

UPES
UNIVERSITY OF TOMORROW

Shortest-Job-First Scheduling(SJF)

Solution for the difficulty

One approach to this problem is to try to approximate SJF scheduling.

- We may not know the length of the next CPU burst, but we may be able to predict its value. We expect that the next CPU burst will be similar in length to the previous ones.
- By computing an approximation of the length of the next CPU burst, we can pick the process with the shortest predicted CPU burst.
- The next CPU burst is generally predicted as an **Exponential Average** of the measured lengths of previous CPU bursts.

The following formula defines the Exponential average:

$$\tau_{n+1} = \alpha\, t_n + (1 - \alpha)\tau_n$$

$\mathbf{t_n}$ be the length of the nth CPU burst (i.e. contains the most recent information).

τ_n stores the history.

τ_{n+1} be our predicted value for the next CPU burst.

$\boldsymbol{\alpha}$ controls the relative weight of recent and past history in our prediction ($0 \leq \boldsymbol{\alpha} \leq 1$)

- If $\boldsymbol{\alpha=0}$, then , recent history has no effect
- If $\boldsymbol{\alpha=1}$ then , only the most recent CPU burst matters.
- If $\boldsymbol{\alpha = 1/2}$, so recent history and past history are equally weighted.

UPES
UNIVERSITY OF TOMORROW

Shortest Remaining Time First Scheduling (SRTF)

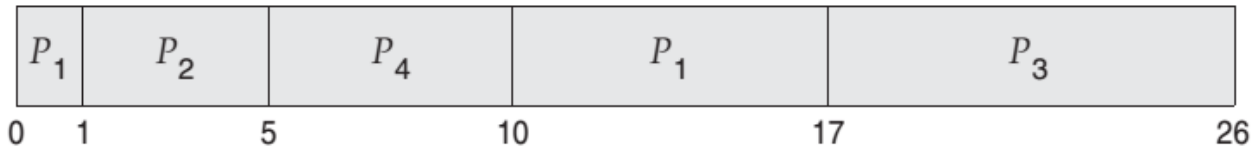
SRTF is the pre-emptive SJF algorithm.

- A new process arrives at the ready queue, while a previous process is still executing
- The next CPU burst of the newly arrived process may be shorter than the currently executing process.
- SRTF will preempt the currently executing process and executes the shortest job.

Consider the four processes with arrival times and burst times in milliseconds:

Process	Arrival Time	Burst Time(ms)
P1	0	8
P2	1	4
P3	2	9
P4	3	5

Gantt Chart for SRTF:



Shortest Remaining Time First Scheduling (SRTF)

Process P1 is started at time 0, since it is the only process in the queue.

- Process P2 arrives at time 1. The remaining time for process P1 (7 milliseconds) is larger than the time required by process P2 (4 milliseconds), so process P1 is preempted and process P2 is scheduled.

- The average waiting time **= 26/4 = 6.5 milliseconds.**

Priority Scheduling

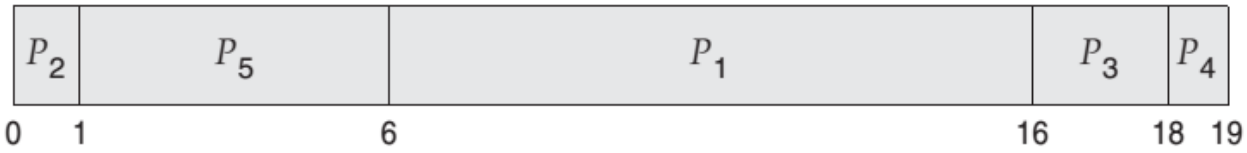
A priority is associated with each process and the CPU is allocated to the process with the highest priority. Equal-priority processes are scheduled in FCFS order.

- An SJF algorithm is special kind of priority scheduling algorithm where small CPU burst will have higher priority.
- Priorities can be defined based on time limits, memory requirements, the number of open files etc.

Example: Consider the following processes with CPU burst and Priorities. All the processes are arrived at time t=0 in the same order. Low numbers are having higher priority.

Process	Burst Time(ms)	Priority
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

Gantt chart for Priority Scheduling:



Priority Scheduling

Priority scheduling can be either **Preemptive or Non-preemptive**.
A **Preemptive Priority** Scheduling algorithm will preempt the CPU if the priority of the newly arrived process is higher than the priority of the currently running process.

Problem: Starvation or Indefinite Blocking

- In priority Scheduling when there is a continuous flow of higher priority processes has come to ready queue then all the lower priority processes must have to wait for the CPU until the all the higher priority processes execution completes.
- This leads to lower priority processes blocked from getting CPU for long period of time. This situation is called Starvation or Indefinite blocking.
- In worst case indefinite blocking may take years to execute the process.

Solution: Aging

Aging involves gradually increasing the priority of processes that wait in the system for a long time.

Process	Burst Time(ms)	Waiting Time
P1	10	6
P2	1	0
P3	2	16
P4	1	18
P5	5	1
Average Waiting Time		8.2 ms

UPES
UNIVERSITY OF TOMORROW

Round Robin(RR) Scheduling

Round-Robin Scheduling (RR)

Round-Robin (RR) scheduling algorithm is designed especially for Timesharing systems.

- RR is similar to FCFS scheduling, but preemption is added to enable the system to switch between processes.
- A small unit of time called a **Time Quantum** or **Time Slice** is defined. A time quantum is generally from 10 to 100 milliseconds in length.
- The ready queue is treated as a **Circular queue**. New processes are added to the tail of the ready queue.
- The CPU scheduler goes around the ready queue by allocating the CPU to each process for a time interval of up to 1 time quantum and dispatches the process.
- If a process CPU burst exceeds 1 time quantum, that process is preempted and is put back in the ready queue.

Round Robin(RR) Scheduling

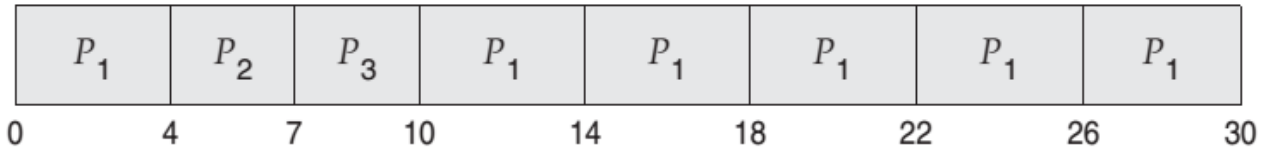
In RR scheduling one of two things will then happen:

1. The process may have a CPU burst of less than 1 time quantum. The process itself will release the CPU voluntarily. The scheduler will then proceed to the next process in the ready queue.
2. If the CPU burst of the currently running process is longer than 1 time quantum, the timer will go off and will cause an interrupt to the operating system. A context switch will be executed and the process will be put at the tail of the ready queue. The CPU scheduler will then select the next process in the ready queue.

Consider the following set of processes that arrive at time 0 and the processes are arrived in the order P1, P2, P3 and Time Quanta=4.

Process	Burst Time (ms)
P1	24
P2	3
P3	3

Gantt chart of Round Robin Scheduling:



Round Robin(RR) Scheduling

- If we use a time quantum of 4 milliseconds, then process *P*1 gets the first 4 milliseconds.
- Since it requires another 20 milliseconds, it is preempted after the first-time quantum and the CPU is given to the next process in the queue, process *P*2.
- CPU burst of Process *P*2 is 3, so it does not need 4 milliseconds then it quits before its time quantum expires. The CPU is then given to the next process *P*3.
- Once each process has received 1 time quantum, the CPU is returned to process *P*1 for an additional time quantum.

The average waiting time under the RR policy is often long.

- *P*1 waits for 6 milliseconds (10 - 4), *P*2 waits for 4 milliseconds and *P*3 waits for 7 milliseconds. Thus, the average waiting time is 17/3 = 5.66 milliseconds. The performance of the RR algorithm depends on the size of the Time Quantum.
- If the time quantum is extremely large, the RR policy is the same as the FCFS policy.
- If the time quantum is extremely small (i.e. 1 millisecond) the RR approach can result in many context switches.
- The time taken for context switch value should be a small fraction of Time quanta then the performance of the RR will be increased.

Note: *A rule of thumb is that 80 percent of the CPU bursts should be shorter than the time quantum.*

UPES
UNIVERSITY OF TOMORROW

MCQ's

1.Which process scheduling algorithm is based on the order of arrival of the processes?
a) Shortest Job First (SJF)
b) Round Robin (RR)
c) First-Come, First-Served (FCFS)
d) Priority Scheduling

2.In the Round Robin (RR) scheduling algorithm, what determines how long each process gets to run?
a) The priority of the process
b) The length of the process's burst time
c) A fixed time quantum
d) The process's arrival time

3.Which scheduling algorithm selects the process with the smallest execution time remaining to execute next?
a) First-Come, First-Served (FCFS)
b) Shortest Remaining Time First (SRTF)
c) Round Robin (RR)
d) Priority Scheduling

UPES
UNIVERSITY OF TOMORROW

MCQ's

4.In Priority Scheduling, how are processes selected for execution?
a) Based on their burst time
b) Based on their arrival time
c) Based on their priority level
d) Based on their memory requirements

5.What is the main disadvantage of the Shortest Job First (SJF) scheduling algorithm?
a) It can lead to starvation of longer processes
b) It requires a fixed time quantum
c) It does not utilize priorities
d) It is complex to implement

6.Which of the following scheduling algorithms can cause the phenomenon known as "starvation"?
a) First-Come, First-Served (FCFS)
b) Round Robin (RR)
c) Shortest Job First (SJF)
d) Multilevel Queue Scheduling

MCQ's

In a Multilevel Queue Scheduling algorithm, how are processes assigned to different queues?
a) Based on their burst time
b) Based on their priority and type
c) Based on their memory requirements
d) Based on their arrival time
Which scheduling algorithm is most suitable for a time-sharing system?
a) First-Come, First-Served (FCFS)
b) Shortest Job First (SJF)
c) Round Robin (RR)
d) Priority Scheduling

What is the main criterion for selecting a process in the Shortest Job First (SJF) scheduling algorithm?
a) Arrival time
b) Priority
c) Shortest burst time
d) Longest burst time

Answers

1. c) Based on their priority level
2. a) It can lead to starvation of longer processes
3. c) Shortest Job First (SJF)
4. b) Based on their priority and type
5. c) Round Robin (RR)
6. c) Shortest burst time

Summary/Key Points

- Process Scheduling: Maximizes CPU utilization (multiprogramming) and interaction (time-sharing).
- CPU Scheduler: Selects processes from the ready queue.
- Preemptive Scheduling: Occurs on state changes and process termination.
- Dispatcher: Switches CPU control to the selected process.
- CPU-Scheduling Algorithms: FCFS, SJF, Priority, RR, Multilevel Queue, Multilevel Feedback Queue.

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

1. Real Time Scheduling
2. Earliest-Deadline-First Scheduling (EDF)

UPES
UNIVERSITY OF TOMORROW

Thank You

UPES
UNIVERSITY OF TOMORROW