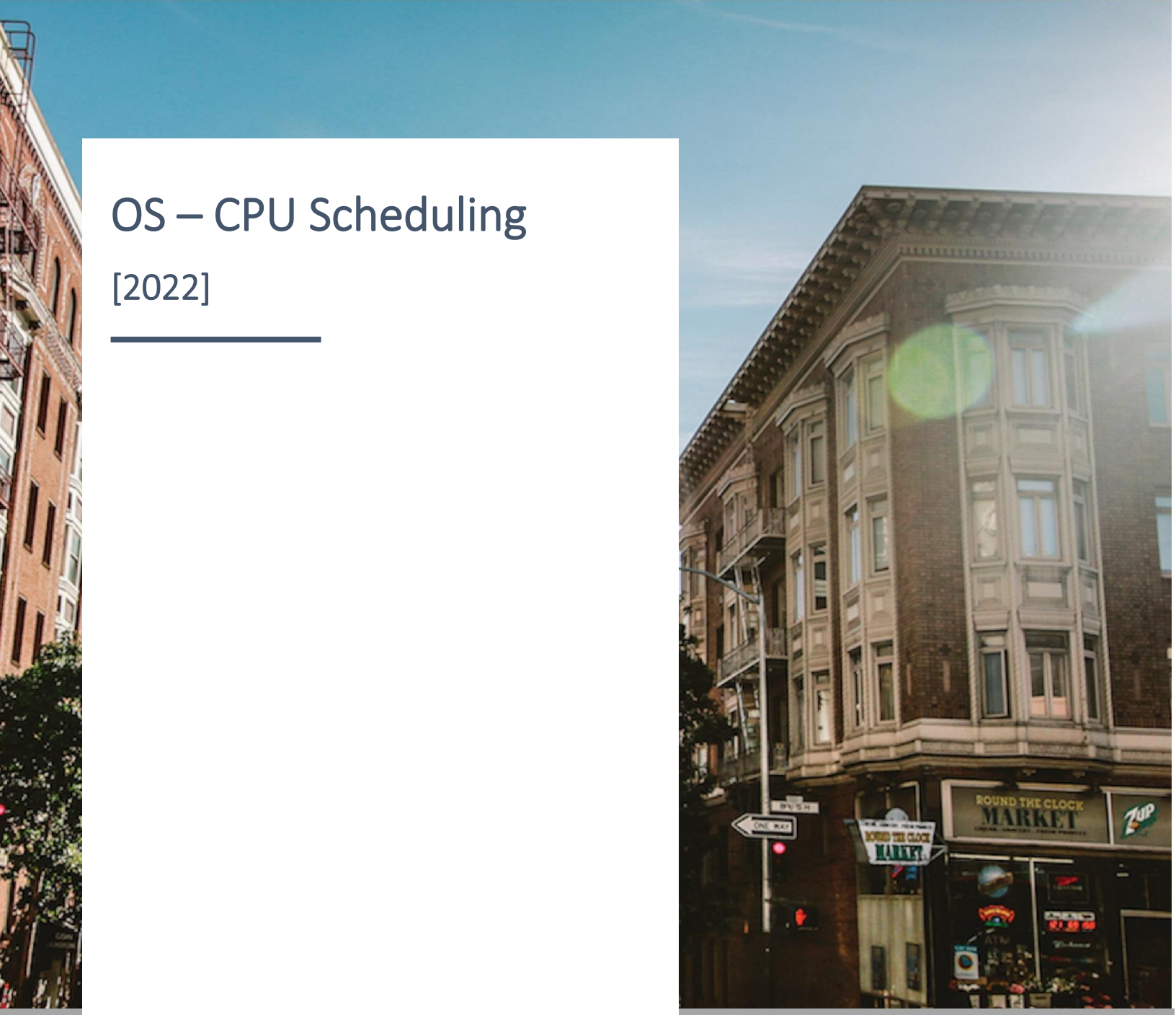


OS – CPU Scheduling

[2022]

[25.08.2022]

Techno India College of Technology
Authored by: Abu Taha Md Farhad
Roll: 31401221052
Reg No: 213141001210016 OF 2021-22
Stream: BCA 2nd year
Subject: Operating System [BCAC302]

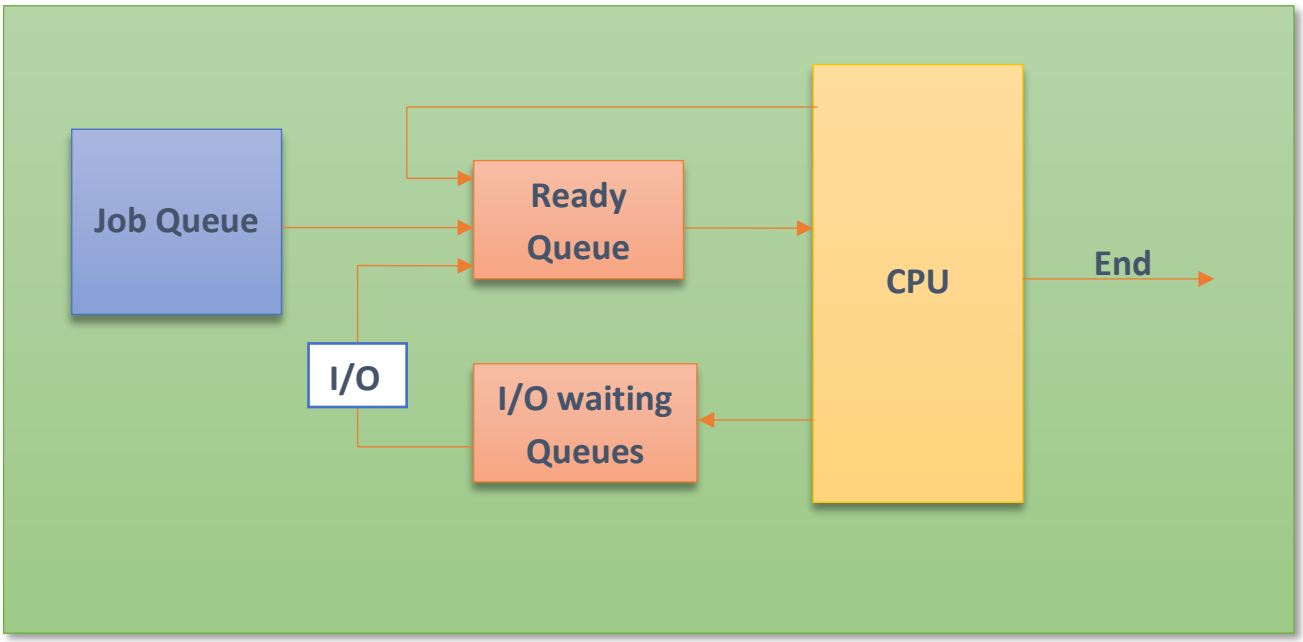
Introduction:

CPU scheduling is the task performed by the CPU that decides the way and order in which processes should be executed. There are two types of CPU scheduling - Preemptive, and non-preemptive. The criteria the CPU takes into consideration while "scheduling" these processes are - CPU utilization, throughput, turnaround time, waiting time, and response time.

Background:

What is CPU Scheduling?

Before we get to CPU scheduling, let's define a process. A process is essentially just a set of instructions or a program in execution.



As we can see in the diagram above, we have processes that come from the job queue to the ready queue (in primary memory) that are one by one, in some manner given resources, and then their execution is completed.
In multiprogramming systems however, the CPU does not remain idle whenever a process currently executing waits for I/O. It starts the execution of other processes, making an attempt to maximize CPU utilization. How does the CPU decide which

process should be executed next from the ready queue for maximum utilization of the CPU? This procedure of "scheduling" the processes, is called CPU scheduling.

Types of CPU Scheduling:

Non-Preemptive Scheduling	Preemptive Scheduling
In the case of non-preemptive scheduling, new processes are executed only after the current process has completed its execution. The process holds the resources of the CPU (CPU time) till its state changes to terminated or is pushed to the process waiting state. If a process is currently being executed by the CPU, it is not interrupted till it is completed. Once the process has completed its execution, the processer picks the next process from the ready queue (the queue in which all processes that are ready for execution are stored).	Preemptive scheduling takes into consideration the fact that some processes could have a higher priority and hence must be executed before the processes that have a lower priority. In preemptive scheduling, the CPU resource are allocated to a process for only a limited period of time and then those resources are taken back and assigned to another process (the next in execution). If the process was yet to complete its execution, it is placed back in the ready state, where it will remain till it gets a chance to execute once again.
Examples of non-preemptive scheduling are **First Come First Serve** and **Shortest Job First**.	Examples of preemptive scheduling are **Round Robin** and **Shortest Remaining Time First**.

Important CPU Scheduling Terminologies:

1. **CPU utilization**: The main purpose of any CPU algorithm is to keep the CPU as busy as possible.
2. **Throughput**: Number of processes that complete their execution per time unit
3. **Turnaround time**: Amount of time to execute a particular process
4. **Waiting time**: Amount of time a process has been waiting in the ready queue

5. **Response time**: Amount of time it takes from when a request was submitted until the first response is produced, not output (for time-sharing environment).

Analysis:

Now we will discuss about one CPU scheduling algorithm.

First Come First Serve:

FCFS considered to be the simplest of all operating system scheduling algorithms. First come first serve scheduling algorithm states that the process that requests the CPU first is allocated the CPU first and is implemented by using FIFO queue.

Characteristics of FCFS:

- FCFS supports non-preemptive and preemptive CPU scheduling algorithms.
- Tasks are always executed on a First-come, First-serve concept.
- FCFS is easy to implement and use.
- This algorithm is not much efficient in performance, and the wait time is quite high.

Advantages of FCFS:

1. Easy to implement
2. First come, first serve method

Disadvantages of FCFS:

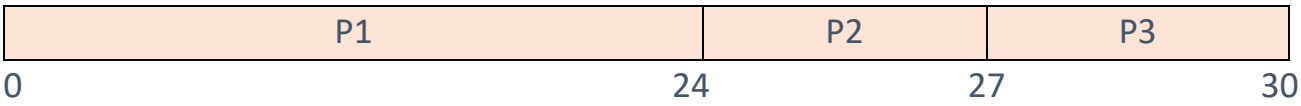
1. FCFS suffers from Convoy effect.
2. The average waiting time is much higher than the other algorithms.
3. FCFS is very simple and easy to implement and hence not much efficient.

Example:

Process	Burst Time
P1	24
P2	3
P3	3

Case1:

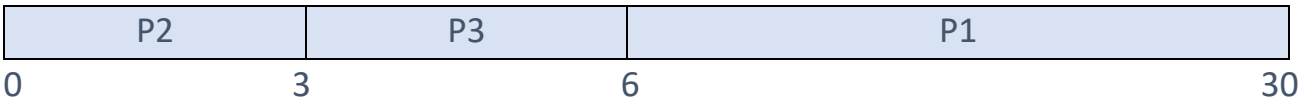
Suppose that the processes arrive in the order: P1, P2, P3
The Gantt Chart for the schedule is:



Waiting time for: P1 = 0, P2 = 24, P3 = 27
Average waiting time: (0 + 24 + 27)/3 = **17**

Case2:

Now suppose that the processes arrive in the order: P2, P3, P1
The Gantt chart for the schedule is:



Waiting time for P1 = 6, P2 = 0, P3 = 3
Average waiting time: (6 + 0 + 3)/3 = **3**
Here we can see in FCFS Scheduling algorithm, by taking the short processes behind the long processes (as Case2) the average waiting time is reduced.

Conclusion:

1. Process Scheduling allows the OS to allocate CPU time for each process. Another important reason to use a process scheduling system is that it keeps the CPU busy at all times. This allows you to get less response time for programs.
2. If most operating systems change their status from performance to waiting then there may always be a chance of failure in the system. So, in order to minimize

this excess, the OS needs to schedule tasks in order to make full use of the CPU and avoid the possibility of deadlock.

Reference:

Reference list:

Operating System Concepts : 8th edition| Abraham Silberschatz, Peter B. Galvin, Greg Gagne

“Scheduling Algorithms in OS”, Scaler Topics, https://www.scaler.com/topics/operating-system/scheduling-algorithms-in-os/, Retrieved: 25-08-2022

“First Come, First Serve – CPU Scheduling | (Non-preemptive)”, GeeksForGeeks, https://www.geeksforgeeks.org/first-come-first-serve-cpu-scheduling-non-preemptive/, Retrieved: 25-08-2022

Thank You