**Unit Four**

**CPU Scheduling**

**4.1 Scheduling Concepts**

Scheduling is a fundamental operating system function. Almost all computer resources are scheduled before use. CPU scheduling is central to operating systems.

**4.2 CPU or I/O Burst Cycle**

The success of CPU scheduling depends on the following observed property of processes: **process execution consists of a cycle of** **CPU execution and I/O wait.** Processes alternate between these two states. Process execution begins with a CPU burst. That is followed by an I/O burst, then another CPU burst, and so on.

**4.3 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 (CPU scheduler). The scheduler selects from among the processes in memory that are ready to execute and allocates the CPU to one of them.

**4.4 Scheduling Schemes**

Two scheduling schemes can be recognized:

• **Preemptive scheduling (يستقطع )**

**• Non-preemptive scheduling) لايستقطع )**

Under non-preemptive scheduling, once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state. On the other hand, Preemptive scheduling occurs when the CPU has been allocated to a process and this process is interrupted by a higher priority process. At this moment the executing process is stopped and returned to the ready queue, the CPU is allocated to the higher priority process.

* **Dispatcher**

It is the module that gives control of the CPU to the process selected by the CPU scheduler. This function involves:

• Switching Context

• Switching to user mode

• Jumping to the proper location in the user program to restart the program.

**4.5 Scheduling Criteria**

Many criteria have been suggested for comparing CPU scheduling algorithms. The criteria include the following:

• **CPU Utilization**: CPU utilization may range from 0 to 100%.

• **Throughput**: it is the number of processes that are completed per unit of time.

• **Turnaround Time**: the interval from the time of submission to the time of completion.

Turnaround Time = waiting time + execution time

• **Waiting time**: is the sum of periods the process spent waiting in the ready queue.

• **Response time**: it is the time from the submission of a request until the first response is produced.

**4.6 Cpu scheduling Algorithms:**

1. **First Come First Served (FCFS)**

With this algorithm, the process that requests the CPU first is allocated to the CPU first. The implementation of the FCFS policy is easily managed with a FIFO queue. The average waiting time under the FCFS policy is often quite long.

**Example 1:** consider the following set of processes that arrive at time 0, using FCFS algorithm to compute average waiting time.

|  |  |
| --- | --- |
| process | Burst Time |
| P1 | 24 |
| P2 | 3 |
| P3 | 3 |

The Gantt Chart is as follows:

|  |  |  |
| --- | --- | --- |
| P1 | P2 | P3 |

0 24 27 30

The average waiting time = (0+24+27) / 3=**17** millisecond

1. **Shortest Job First Scheduling (SJF)**

This algorithm associate with each process the length of the latter's next CPU burst. When the CPU is available, it is assigned to the process that has the smallest next CPU burst. If two processes have the same length, FCFS scheduling is used to break this tie.

**Example 1:** consider the following set of processes that arrives at time 0, using SJF algorithm.

|  |  |
| --- | --- |
| process | Burst time |
| P1 | 6 |
| P2 | 8 |
| P3 | 7 |
| P4 | 3 |

The Gantt Chart is as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| P4 | P1 | P3 | P2 |

0 3 9 16 24

A.W.T = (3+16+9+0) / 4 = 7 milliseconds.

**Example 2:** consider the following set of processes that arrives at different time with the length , using SJF algorithm.

|  |  |  |
| --- | --- | --- |
| process | Arrival Time | Burst Time |
| P1 | 0 | 8 |
| P2 | 1 | 4 |
| P3 | 2 | 9 |
| P4 | 3 | 5 |

The Gantt Chart is as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P1 | P2 | P4 | P1 | P3 |

0 1 5 10 17 26

AWT =((10-1)+(1-1)+(17-2)+(5-3)) / 4=6.5millisecond

1. **Priority Scheduling Algorithm**

In this algorithm, a priority is associated with each process and the CPU is allocated to the process of the highest priority. We use low numbers to represent high priority.

**Example 3:** consider the following set of processes using priority algorithm

|  |  |  |
| --- | --- | --- |
| process | Burst time | priority |
| P1 | 10 | 3 |
| P2 | 1 | 1 |
| P3 | 2 | 3 |
| P4 | 1 | 4 |
| P5 | 5 | 2 |

The Gantt Chart is as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| P2 | P5 | P1 | P3 | P4 |

0 1 6 16 18 19

A.W.T = (6+ 0+16+18+1) /5 = 8.2 milliseconds

A major **problem** with priority algorithm is indefinite **blocking or starvation.** (Process with lower priority must wait very long time)

* The **solution** to this problem is **aging**
* Aging is a technique of **gradually increasing** the priority of processes that wait in the system for a long time.

1. **Round Robin Scheduling Algorithm**

The Round Robin algorithm is designed especially for the time-sharing system. It is similar to FCFS but preemption has added a switch between processes. A small unit of time called time quantum (or time slice) is defined. A time quantum is generally from 10 to 100 milliseconds.

**Example 4** : consider the following set of processes and the time quantum = 4

|  |  |
| --- | --- |
| process | Burst time |
| P1 | 24 |
| P2 | 3 |
| P3 | 3 |

The Gantt Chart is as follows:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P1 | P1 | P1 | P1 | P1 |

0 4 7 10 14 18 22 26 30

The average waiting time = (4+7+(10-4) ) /3 =(17) / 3=5.66 millisecond