***CHAPTER-5***

***PROCESS SCHEDULING***

------------------------------------------------------------------------------------------------------------------------------------------------------------------

**5.1 Basic Concepts**

*Objective of Multiprogramming*: It is to have some process running at all times, to maximize CPU utilization. Several processes are kept in memory at one time. When one process has to wait, the operating system takes the CPU away from that process and gives the CPU to another process.

**5.1.1 CPU I/O Burst Cycle**

*Cycle*: A process execution consists of a cycle of CPU execution and I/O wait. Processes alternate between these two states.

*CPU and I/O Bursts*: Process execution begins with a CPU burst. That is followed by an I/O burst, which is followed by another CPU burst, then I/O burst, and so on. Eventually, the final CPU burst ends with a system request to terminate execution.

**5.1.2 CPU Scheduler**

*Short-Term Scheduler*: Whenever the CPU becomes idle, the operating system must select one of the processes in the ready queue to be executed. This selection process is carried out by the short-term scheduler or CPU scheduler.

**5.1.3 Preemptive Scheduling**

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

1. When a process switches from the running state to the waiting state.
2. When a process switches from the running state to the ready state.
3. When a process switches from the waiting state to the ready state.
4. When a process terminates.

**Non-Preemptive / Cooperative Scheduling**: Once the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating (case4) or by switching to the waiting state (case1).

**Preemptive Scheduling**: Scheduling takes place under conditions 2 and 3.

*Disadvantage of Preemptive Scheduling*: It can result in race conditions when data are shared among several processes. While one process is updating the data, if it is preempted so that the second process can run, then the second process will try to read that data which is in an inconsistent state.

Preemption also affects the design of the operating-system kernel. During the processing of a system call, the kernel may be busy with an activity on behalf of a process. If the process is preempted in the middle of these changes, and the kernel needs to read or modify the same structure, it leads to chaos.

**5.1.4 Dispatcher**

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

* Switching context;
* Switching to user mode; and
* Jumping to the proper location in the user program to restart that program.

**Dispatch Latency**: 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 running another is known as dispatch latency.

------------------------------------------------------------------------------------------------------------------------------------------------------------------

**5.2 Scheduling Criteria**

1. *CPU Utilization*: The CPU must be kept as busy as possible. In a real system, it should range from 40 percent (for a lightly system) to 90 percent (for a heavily loaded system).
2. *Throughput*: One measure of work is the number of processes that are completed per time unit, called throughput. It may range from 10processes/sec (short transactions) to 1process/hour (long transactions).
3. *Turnaround Time*: It 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.
4. *Waiting Time*: Waiting time is the sum of periods spent waiting in the ready queue.
5. *Response Time*: Response time is the time from the submission of a request until the first response is produced.

It is desirable to maximize CPU utilization and throughput and to minimize turnaround time, waiting time and response time.

------------------------------------------------------------------------------------------------------------------------------------------------------------------

**5.6 Real-Time CPU Scheduling**

**Soft Real Time Systems**: They provide no guarantee as to when a critical real-time process will be scheduled.

S

S

S

S