Day 2 – Process Management_Sanket.Shalukar

Monday, August 18, 2025 3:50 PM

Topics that are in the Process management

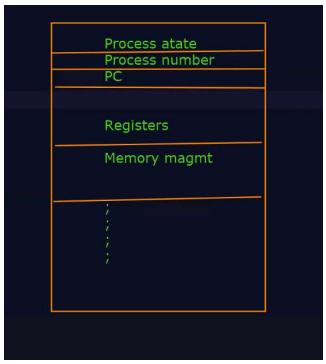
- Process ID
- Process Life cycle
- Schedulers
- Types of Schedulers
- Process creation

Process : The Process is basically an instance of the computer program that is being executes.
Short lifespan
H/W - CPU , I/O devices, memory
Runtime instance
Dynamic instance

Program : Program is a collection of instructions that performed specific task when execution starts.

Thread :

Process control Block :

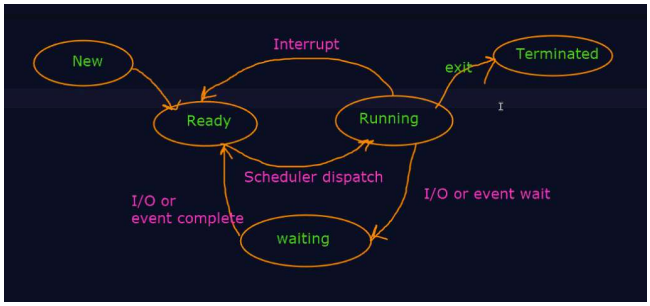


- Process state
- Program counter
- CPu registers
- CPu algorithms
- CPU scheduleing information
- Memory management information
- Accounting information
- I/O status information

Process life cycle :

States

- New state
- Running state
- Waiting state
- Ready state
- Terminate state



Process Scheduling :

It is the act of determining which process is in the ready state. And should be moved to the running state, is called as Schedule state.

Types of Processes :

- Non-Preemptive -
- Preemtative

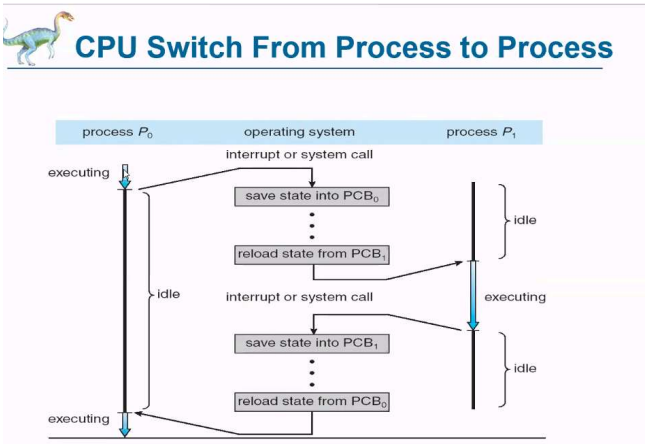
Scheduling algorithms

Non-Preemptive Scheduling

- Once a process starts running, it **cannot be stopped** until it finishes or goes to waiting state.
- CPU is **not taken away** from the process.
- **Response time** may be high (if a long process is running, shorter ones wait).
- **Simpler** to implement.

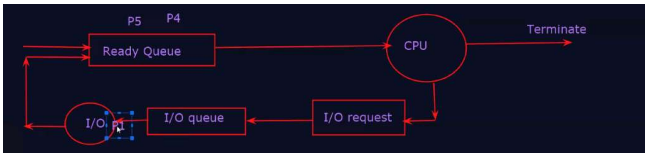
Preemptive Scheduling

- A process can be **interrupted** in the middle and CPU can be given to another process.
- CPU is **taken away** if a higher priority process arrives or time slice expires.
- Provides **better responsiveness** and fairness.
- **More complex** to implement.

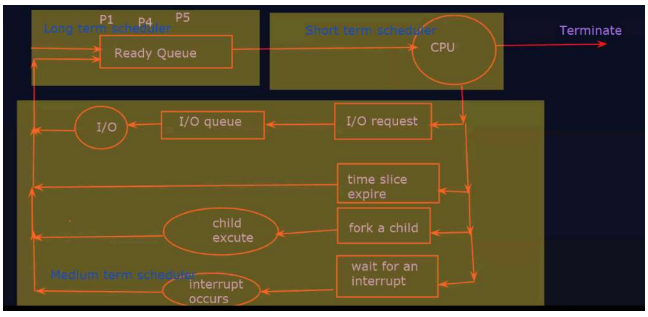


Job queue : Process to be in execution.
Ready queue : Process waiting for execution.

(Above Digram) Normal process of sceduling

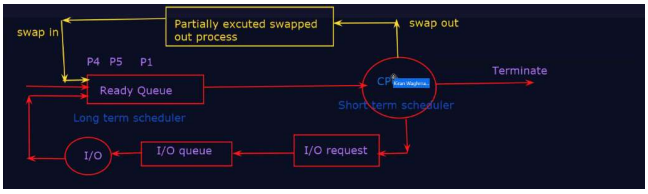


(Above Digram) For I/O Devices



(Above Digram) How Process Scedules into PCB

Uses of Long term and Short term Scheduler.



Long-Term Scheduler (Job Scheduler)

Long term Scheduler increases degree of utilization high.

- **When used**: Decides **which jobs (processes) enter the system** for processing.
- **Main Use**: Controls the **degree of multiprogramming** (how many programs are in memory at once).
- **Example**: Like an **admission gate** → decides who can enter the classroom (system).

Uses:

1. To keep system balanced (not too many processes).
2. Selects jobs from job queue (on disk) and loads them into memory.
3. Helps in **load control** of the system.

Short-Term Scheduler (CPU Scheduler)

Short term scheduler don't have any control over utilization

- **When used**: Decides **which process in ready queue gets CPU next**.
- **Main Use**: Gives quick response and CPU utilization.
- **Example**: Like a **teacher calling one student at a time** to answer (CPU usage).

Uses:

1. Improves **CPU performance**.
2. Switches between processes quickly.
3. Makes system **responsive** for users.

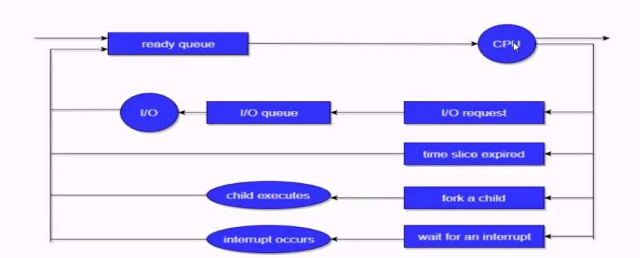
Medium-Term Scheduler

Medium term scheduler decrease the degree of utilization

- **Use**: **Temporarily removes (suspends) a process** from memory and puts it on disk, or brings it back.
- **Main Use**: Improves performance and ensures fair CPU sharing.
- **Example**: Like a **teacher sending a student out for some time** (suspend) and then calling them back later .

Uses:

- Decides **which process gets CPU next**.
- Improves **CPU utilization** (keeps CPU busy).
- Gives **quick response** to users.
- Reduces waiting time and turnaround time for processes.



In the first two cases, the process eventually switches from the waiting state to the ready state, and is then put back in the ready queue. A process continues this cycle until it terminates, at which time it is removed from all queues and

Parent process : PID

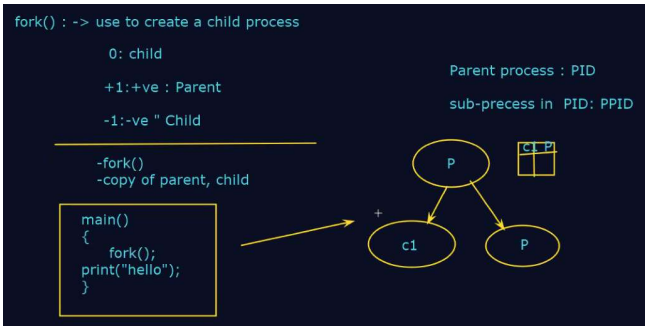
sub-precess in PID: PPID

Context Switch – It saves remain mpleted That saved into context switch.

Operations on process :

- **Creation Process**
  Fork or spawn - [fork ()]

  Relation - **2 to the power of n-1**



Termination Process
When a process finishes its work and ends, it is called process termination.

Zombie Process

A process that has finished execution but its entry is still in the process table (waiting for parent to read its status).

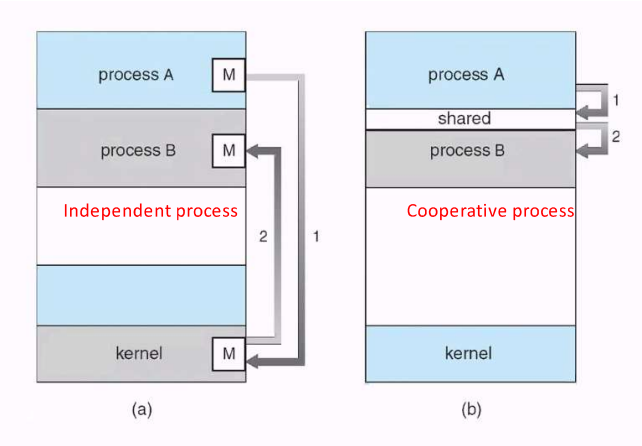
Orphan Process

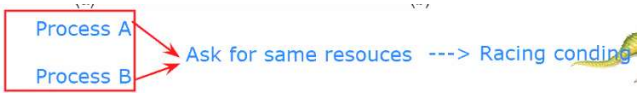
A child process whose parent has already finished/terminated before the child is done.

Interprocess Communication

- Processes within a system may be **independent** or **cooperating**
- Cooperating process can affect or be affected by other processes, including sharing data
- Reasons for cooperating processes:
  - Information sharing
  - Computation speedup
  - Modularity
  - Convenience
- Cooperating processes need **interprocess communication** (**IPC**)
- Two models of IPC
  - Shared memory
  - Message passing

Communication Process.





There is solution to avoid **racing condition** that is **synchronization**.