# Process Life Cycle

## Process States

- New
  - New process PCB is created and added into job queue. PCB is initialized and process get ready for execution.
- Ready
  - The ready process is added into the ready queue. Scheduler pick a process for scheduling from ready queue and dispatch it on CPU.
- Running
  - The process runs on CPU. If process keeps running on CPU, the timer interrupt is used to forcibly put it into ready state and allocate CPU time to other process.
- Waiting
  - If running process request for IO device, the process waits for completion of the IO. The waiting state is also called as sleeping or blocked state.
- Terminated
  - If running process exits, it is terminated.
- Linux: TASK_RUNNING (R), TASK_INTERRUPTIBLE (S), TASK_UNINTERRUPTIBLE (D), TASK_STOPPED(T), TASK_ZOMBIE (Z), TASK_DEAD (X)

# Types of Scheduling

## Non-preemptive

- The current process gives up CPU voluntarily (for IO, terminate or yield).
- Then CPU scheduler picks next process for the execution.
- If each process yields CPU so that other process can get CPU for the execution, it is referred as "Co-operative scheduling".

## Preemptive

- The current process may give up CPU voluntarily or paused forcibly (for high priority process or upon completion of its time quantum)

# Scheduling criteria's

## CPU utilization: Ideal - max

- On server systems, CPU utilization should be more than 90%.
- On desktop systems, CPU utilization should around 70%.

### Throughput: Ideal - max

- The amount of work done in unit time.

### Waiting time: Ideal - min

- Time spent by the process in the ready queue to get scheduled on the CPU.
- If waiting time is more (not getting CPU time for execution) -- Starvation.

### Turn-around time: Ideal - CPU burst + IO burst

- Time from arrival of the process till completion of the process.
- CPU burst + IO burst + (CPU) Waiting time + IO Waiting time

### Response time: Ideal - min

- Time from arrival of process (in ready queue) till allocated CPU for first time.

## Scheduling Algorithms

### FCFS

- Process added first in ready queue should be scheduled first.
- Non-preemptive scheduling
- Scheduler is invoked when process is terminated, blocked or gives up CPU is ready for execution.
- Convoy Effect: Larger processes slow down execution of other processes.

### SJF

- Process with lowest burst time is scheduled first.
- Non-preemptive scheduling
- Minimum waiting time

### SRTF - Shortest Remaining Time First

- Similar to SJF - but Preemptive scheduling
- Minimum waiting time

### Priority

- Each process is associated with some priority level. Usually lower the number, higher is the priority.
- Preemptive scheduling or Non Preemptive scheduling
- Starvation
    - Problem may arise in priority scheduling.
    - Process not getting CPU time due to other high priority processes.
    - Process is in ready state (ready queue).
    - May be handled with aging -- dynamically increasing priority of the process.

### Round-Robin

- Preemptive scheduling

- Process is assigned a time quantum/slice.
- Once time slice is completed/expired, then process is forcibly preempted and other process is scheduled.
- Min response time.

# Thread concept

- Threads are used to execute multiple tasks concurrently in the same program/process.
- Thread is a light-weight process.
    - For each thread new control block and stack is created. Other sections (text, data, heap, ...) are shared with the parent process.
    - Inter-thread communication is much faster than inter-process communication.
    - Context switch between two threads in the same process is faster.
- Thread stack is used to create function activation records of the functions called/executed by the thread.

## Process vs Thread

- In modern OS, process is a container holding resources required for execution, while thread is unit of execution/scheduling.
- Process holds resources like memory, open files, IPC (e.g. signal table, shared memory, pipe, etc.).
- PCB contains resources information like pid, exit status, open files, signals/ipc, memory info, etc.
- CPU time is allocated to the threads. Thread is unit of execution.
- TCB contains execution information like tid, scheduling info (priority, sched algo, time left, ...), Execution context, Kernel stack, etc.
- terminal> ps -e -o pid,nlwp,cmd
- terminal> ps -e -m -o pid,tid,nlwp

## main thread

- For each process one thread is created by default called as main thread.
- The main thread executes entry-point function of the process.
- The main thread use the process stack.
- When main thread is terminated, the process is terminated.
- When a process is terminated, all threads in the process are terminated.