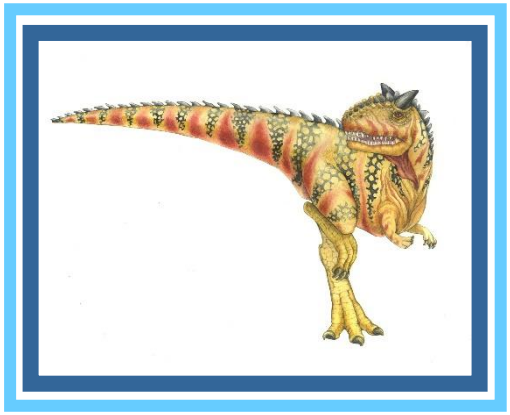
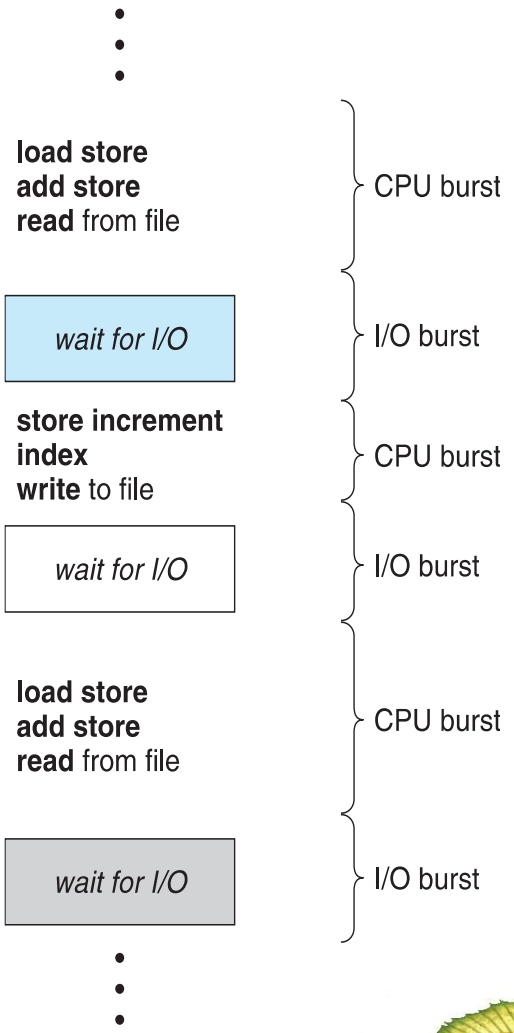
Chapter 6: CPU Scheduling



Basic Concepts

- Maximum CPU utilization obtained with multiprogramming (multiple program is running)
- CPU–I/O Burst Cycle – Process execution consists of a **cycle** of **CPU execution and I/O wait**
- **CPU burst** followed by **I/O burst**
- CPU burst distribution is of main concern



CPU Scheduler

- **Short-term scheduler** selects from among the processes in ready queue, and allocates the CPU to one of them
  - Queue may be ordered in various ways
- CPU scheduling decisions may take place when a process:
  1. Switches from running to waiting state
  2. Switches from running to ready state(when interrupt happened)
  3. Switches from waiting to ready
  4. Terminates
- Scheduling under 1 and 4 is **nonpreemptive(never taken the cpu only if completed or go to waiting)**
- All other scheduling is **preemptive (can taken the CPU from the process)**
  - Consider access to shared data that use by others who stopped
  - Consider interrupts occurring during crucial OS activities

Dispatcher

- Dispatcher module gives control of the CPU to the process selected by the short-term scheduler; this involves:
  - switching context
  - switching to user mode
  - jumping to the proper location in the user program to restart that program
- **Dispatch latency** – time it takes for the dispatcher to stop one process and start another running

Scheduling Criteria

- **CPU utilization** – keep the CPU as busy as possible
- **Throughput –** number of processes that complete their execution per time unit
- **Turnaround time** – amount of time to execute a particular process from begging until the termination
- **Waiting time** – amount of time a process has been waiting in the ready queue
- **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)

Scheduling Algorithm Optimization Criteria

- Max CPU utilization
- Max throughput
- Min turnaround time
- Min waiting time
- Min response time

First- Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The Gantt Chart for the schedule is:

- Waiting time for P_1 = 0; P_2 = 24; P_3 = 27
- Average waiting time: (0 + 24 + 27)/3 = 17

FCFS Scheduling (Cont.)

Suppose that the processes arrive in the order:

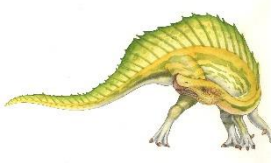
P_2 , P_3 , P_1

- The Gantt chart for the schedule is:

- Waiting time for $P_1 = 6; P_2 = 0; P_3 = 3$
- Average waiting time: (6 + 0 + 3)/3 = 3
- Much better than previous case
- **Convoy effect** - short process behind long process
- **nonpreemptive(never taken the cpu only if completed or go to waiting)**

Shortest-Job-First (SJF) Scheduling

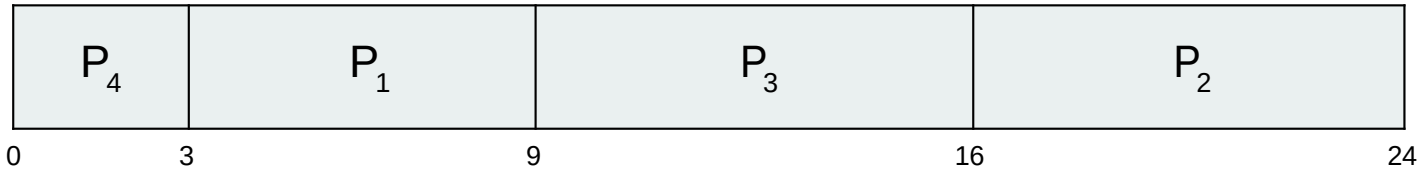
- Associate with each process the **length of its next CPU burst**
  - Use these lengths to schedule the process with the shortest time
- SJF is optimal – gives minimum average waiting time for a given set of processes
  - The difficulty is knowing the length of the next CPU request
  - Could ask the user
- Can only estimate the length – should be similar to the previous one
  - Then pick process with shortest predicted next CPU burst
- Preemptive version called **shortest-remaining-time-first**
- **nonpreemptive**



Example of SJF

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3

- SJF scheduling chart



- Average waiting time = (3 + 16 + 9 + 0) / 4 = 7

Example of SJF Scheduling (Premptive)

Consider the following four processes, with the length of the CPU burst given in milliseconds and the processes arrive at the ready queue at the times shown:

Process ID	Arrival Time	Burst Time
P1	0	~~8~~ 7
P2	1	4
P3	2	9
P4	3	5

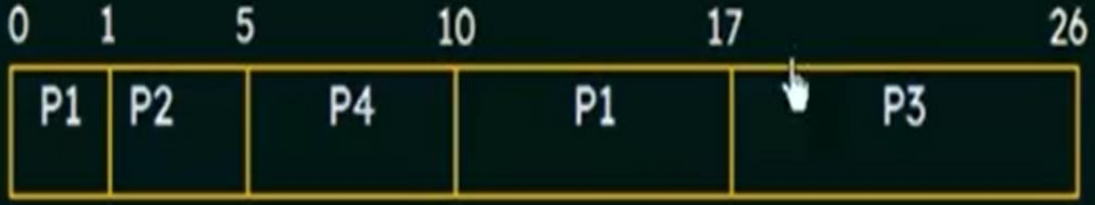
Waiting Time for P1 = (10 - 1 - 0) = 9 ms

Waiting Time for P2 = (1 - 0 - 1) = 0 ms

Waiting Time for P3 = (17 - 0 - 2) = 15 ms

Waiting Time for P4 = (5 - 0 - 3) = 2 ms

Gantt Chart:



Average Waiting Time

= (9 + 0 + 15 + 2)/4 = 6.5 ms

Waiting Time = Total waiting Time - No. of milliseconds Process executed - Arrival Time

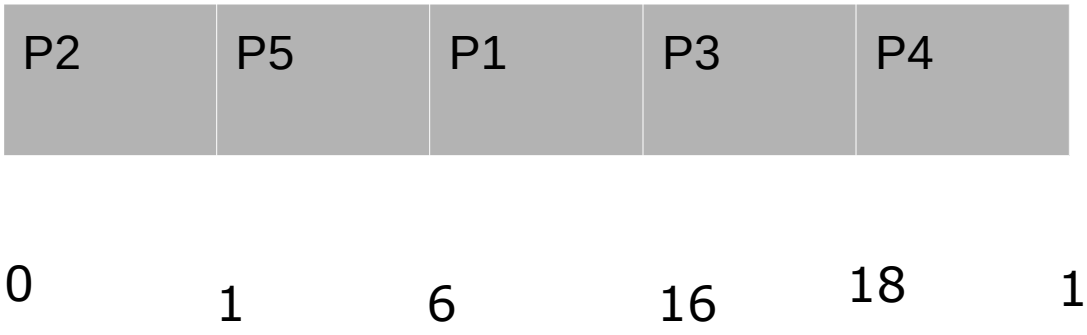
Priority Scheduling

- A priority number (integer) is associated with each process
- The CPU is allocated to the process with the highest priority (smallest integer ≡ highest priority)
  - Preemptive
  - Nonpreemptive
- SJF is priority scheduling where priority is the inverse of predicted next CPU burst time
- Problem ≡ **Starvation** – low priority processes may never execute
- Solution ≡ **Aging** – as time progresses increase the priority of the process

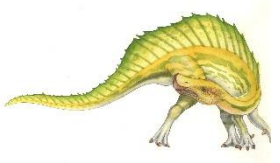
Example of Priority Scheduling

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- Priority scheduling Gantt Chart



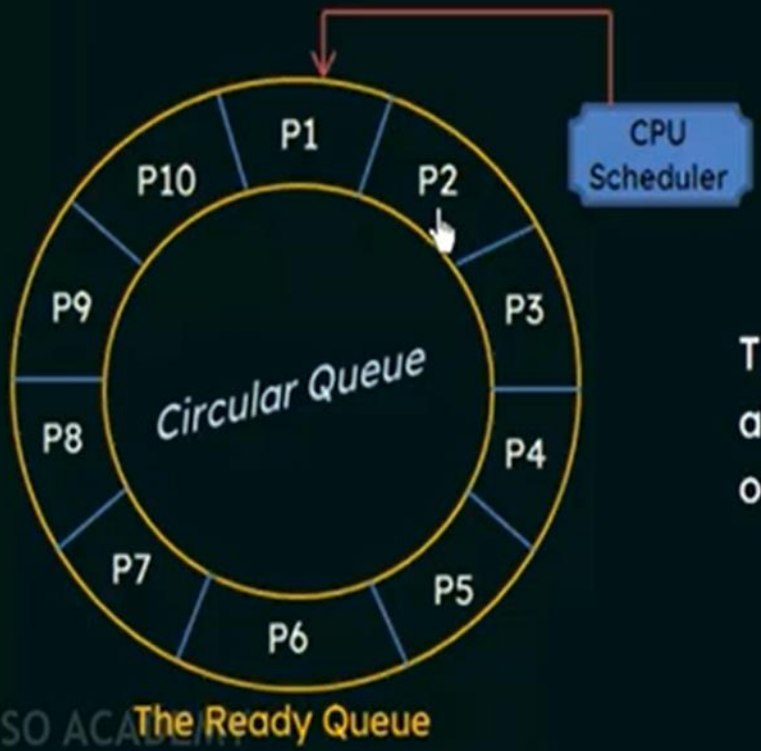
- Average waiting time = (6+0+16+18+1)/5=8.2 msec



Round Robin (RR)

Scheduling Algorithms
(Round-Robin Scheduling)

- The round-robin (RR) scheduling algorithm is designed especially for timesharing systems.
- It is similar to FCFS scheduling, but preemption is added to switch between processes.
- A small unit of time, called a time quantum or time slice, is defined (generally from 10 to 100 milliseconds)



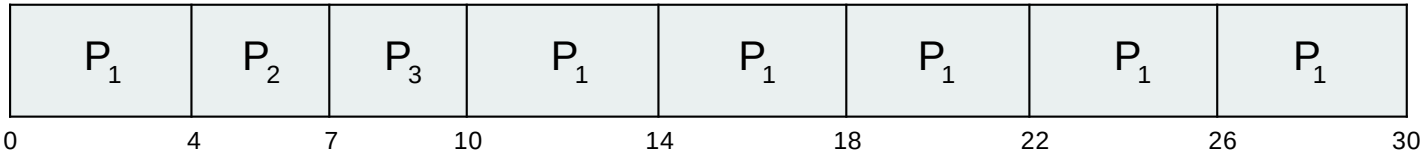
- The ready queue is treated as a circular queue.

The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum.

Example of RR with Time Quantum = 4

Process	Burst Time
P_1	24
P_2	3
P_3	3

- The Gantt chart is:



- Typically, higher average turnaround than SJF, but better ***response***
- q should be large compared to context switch time
- q usually 10ms to 100ms, context switch < 10 usec

P_1	P_2	P_3	P_1	P_1	P_1	P_1	P_1

0 4 7 10 14 18 22 26 30

Method 1

Turn Around time = Completion time – Arrival time

Waiting time = Turn Around time – Burst time

Process ID	Completion Time	Turnaround Time	Waiting Time
P1	30	30 – 0 = 30	30 – 24 = 6
P2	7	7 – 0 = 7	7 – 3 = 4
P3	10	10 – 0 = 10	10 – 3 = 7

Average Turn Around time

= (30 + 7 + 10) / 3

= 47 / 3 = 15.66 ms

Average waiting time

= (6 + 4 + 7) / 3

= 17 / 3 = 5.66 ms

Method 2

Waiting time = Last Start Time - Arrival Time – (Preemption x Time Quantum)

Process ID	Waiting Time
P1	26 – 0 – (5x4) = 6
P2	4 – 0 – (0x4) = 4
P3	7 – 0 – (0x4) = 7

Average waiting time

= (6 + 4 + 7) / 3

= 17 / 3 = 5.66 ms

Multilevel Queue

- Ready queue is partitioned into separate queues, eg:
  - **foreground** (interactive)
  - **background** (batch)
- Process permanently in a given queue
- Each queue has its own scheduling algorithm:
  - foreground – RR
  - background – FCFS
- Scheduling must be done between the queues:
  - Fixed priority scheduling; (i.e., serve all from foreground then from background). Possibility of starvation.
  - Time slice – each queue gets a certain amount of CPU time which it can schedule amongst its processes; i.e., 80% to foreground in RR
  - 20% to background in FCFS

Multilevel Queue Scheduling