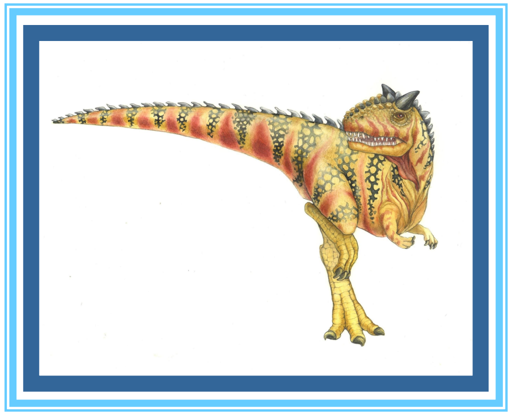
Chapter 6: CPU Scheduling



Previous Lecture

Process Synchronization

- Background
- The Critical-Section Problem
- Peterson's Solution
- Synchronization Hardware
- Mutex Locks
- Semaphores
- Classic Problems of Synchronization
- Monitors
- Synchronization Examples
- Alternative Approaches

Chapter 6: CPU Scheduling

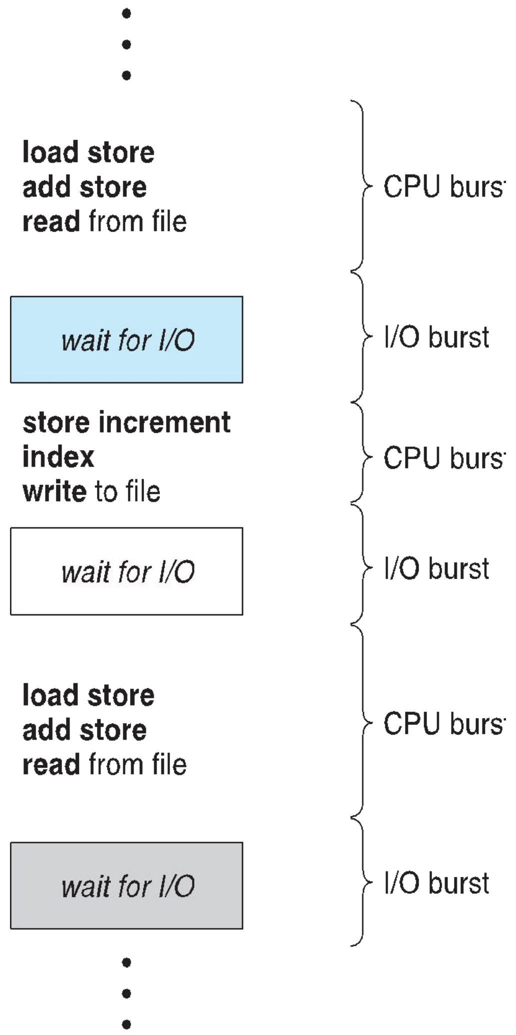
- Basic Concepts
- Scheduling Criteria
- Scheduling Algorithms
- Multiple-Processor Scheduling

Objectives

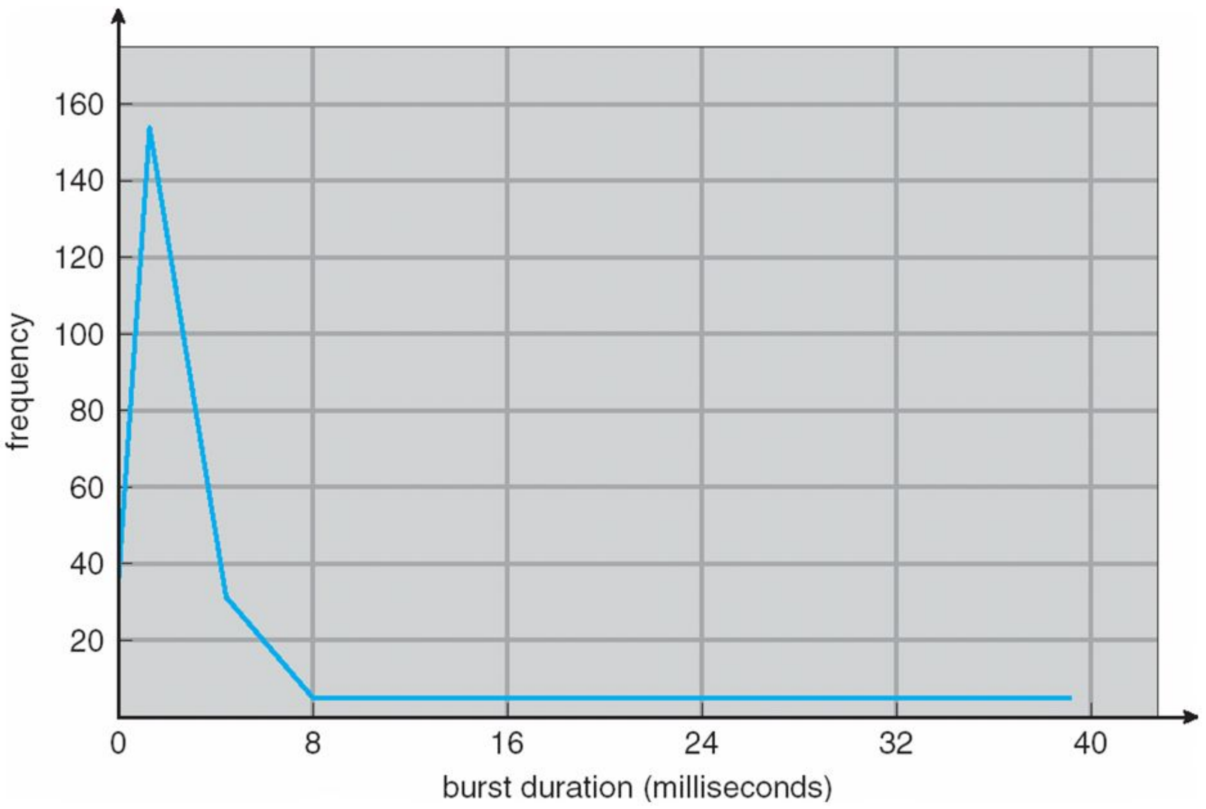
- To introduce CPU scheduling, which is the basis for multiprogrammed operating systems
- To describe various CPU-scheduling algorithms
- To discuss evaluation criteria for selecting a CPU-scheduling algorithm for a particular system
- To examine the scheduling algorithms of several operating systems

Basic Concepts

- Maximum CPU utilization obtained with multiprogramming
- 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



Histogram of CPU-burst Times

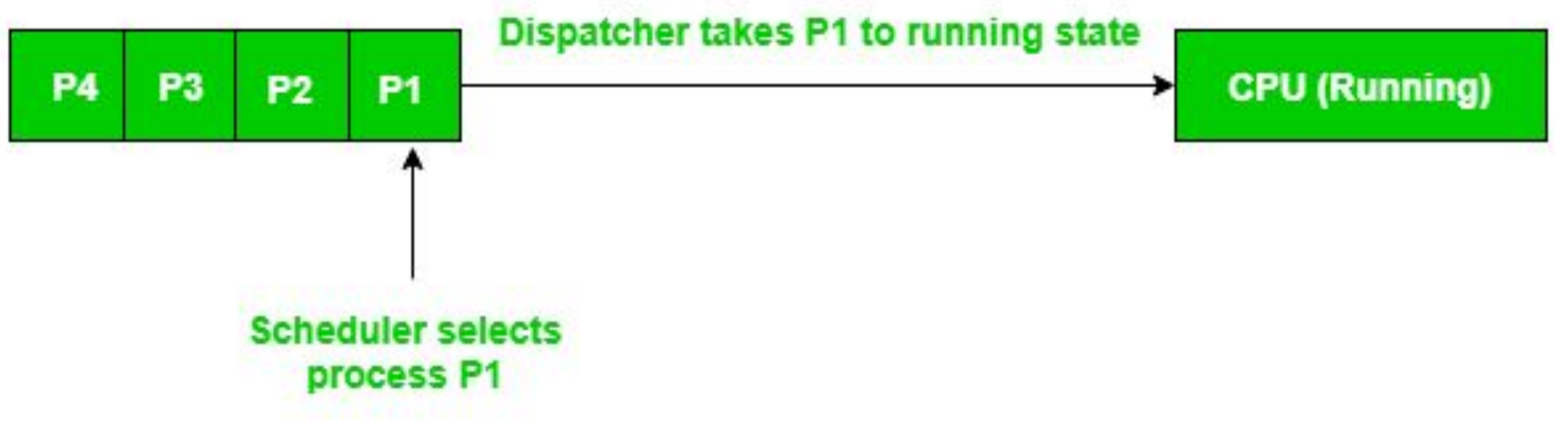


CPU Scheduler

- **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
  3. Switches from waiting to ready
  4. Terminates
- Scheduling under 1 and 4 is **non-preemptive**
- All other scheduling is **preemptive**
  - Consider access to shared data
  - Consider preemption while in kernel mode
  - 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

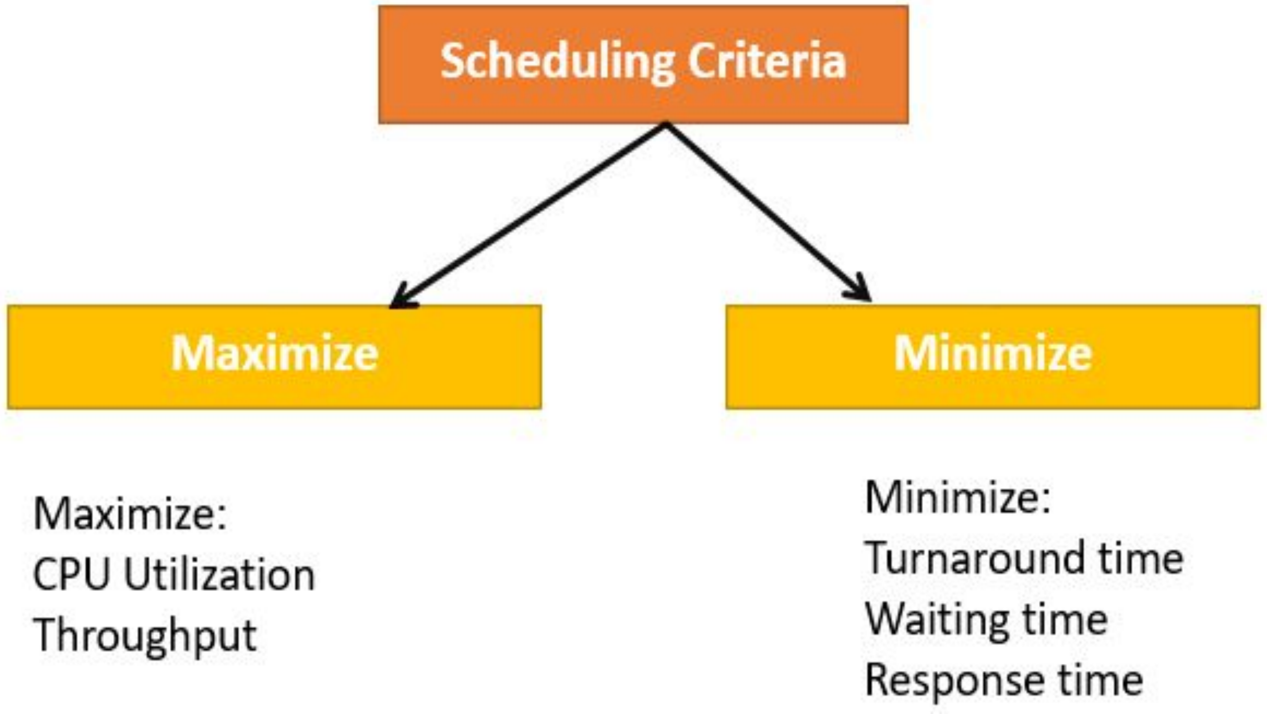

P4
P3
P2
P1
Dispatcher takes P1 to running state
CPU (Running)
Scheduler selects process P1

Scheduling Criteria

- **CPU utilization** – keep the CPU as busy as possible
- **Throughput** – # of processes that complete their execution per time unit
- **Turnaround time** – amount of time to execute a process
- **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

- A CPU scheduling algorithm tries to maximize and minimize the following:



Maximize:
CPU Utilization
Throughput

Minimize:
Turnaround time
Waiting time
Response time

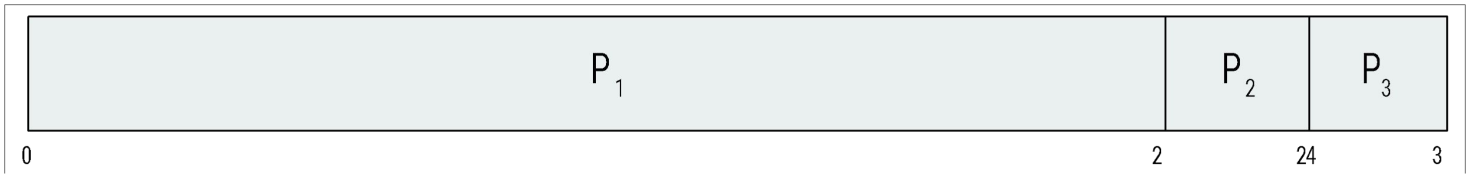
First- Come, First-Served (FCFS) Scheduling

- **First come first serve** (FCFS) scheduling algorithm simply schedules the jobs according to their arrival time.
- The job which comes first in the ready queue will get the CPU first. The lesser the arrival time of the job, the sooner will the job get the CPU.
- FCFS scheduling may cause the problem of starvation if the burst time of the first process is the longest among all the jobs.

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:



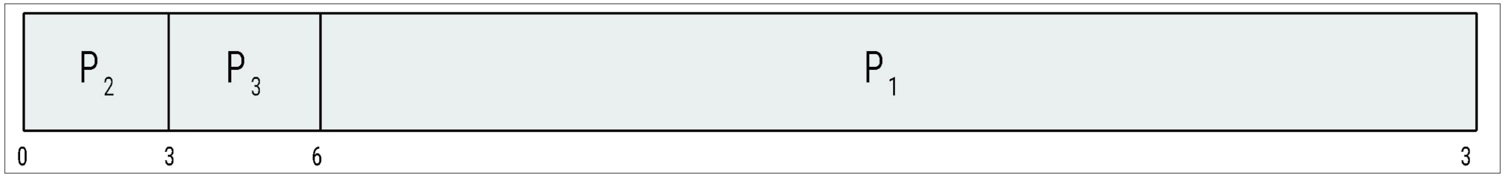
- Waiting time for P_1 = 0; P_2 = 24; P_3 = 27
- Average waiting time: (0 + 24 + 27)/3 = 17

FCFS Scheduling (Cont.)

Suppose that the processes arrive in the order:

P_2 , P_3 , P_1

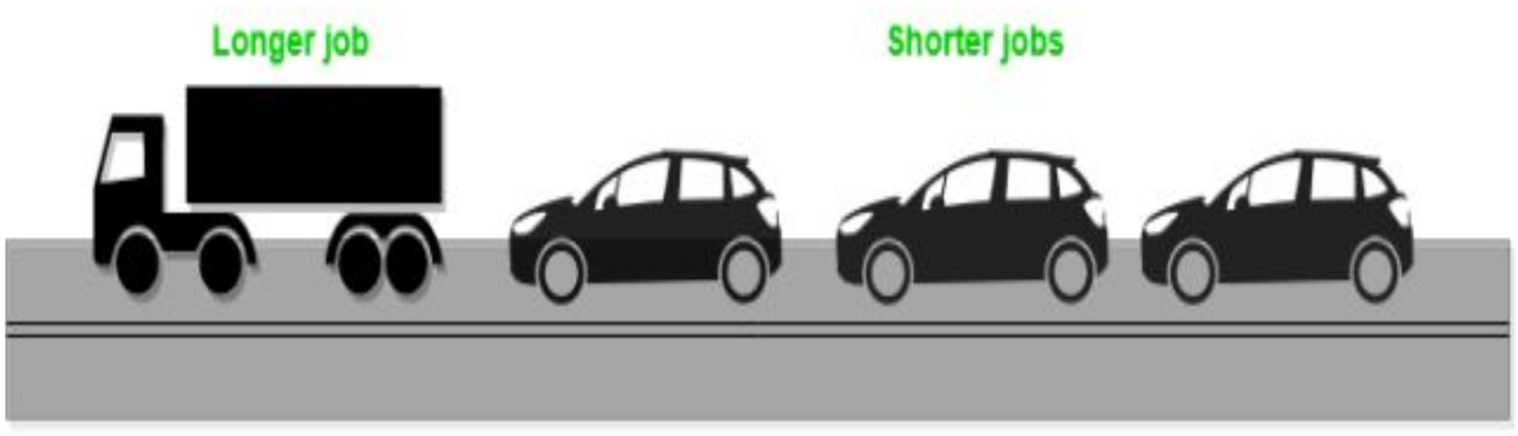
- The Gantt chart for the schedule is:



- 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
  - Consider one CPU-bound and many I/O-bound processes

Convey Effect

- Convoy Effect is phenomenon associated with the First Come First Serve (FCFS) algorithm, in which the whole Operating System slows down due to few slow processes.



- FCFS algorithm is non-preemptive in nature, that is, once CPU time has been allocated to a process, other processes can get CPU time only after the current process has finished. This property of FCFS scheduling leads to the situation called Convoy Effect.

FCFS Advantages and Disadvantages

- **Advantages**
  - It is simple and easy to understand.
- **Disadvantages**
  - The process with less execution time suffer i.e. waiting time is often quite long.
  - Favors CPU Bound process then I/O bound process.
  - Here, first process will get the CPU first, other processes can get CPU only after the current process has finished its execution. Now, suppose the first process has large burst time, and other processes have less burst time, then the processes will have to wait more unnecessarily, this will result in *more average waiting time*, i.e., Convey effect.
  - This effect results in lower CPU and device utilization.
  - FCFS algorithm is particularly troublesome for time-sharing systems, where it is important that each user get a share of the CPU at regular intervals.