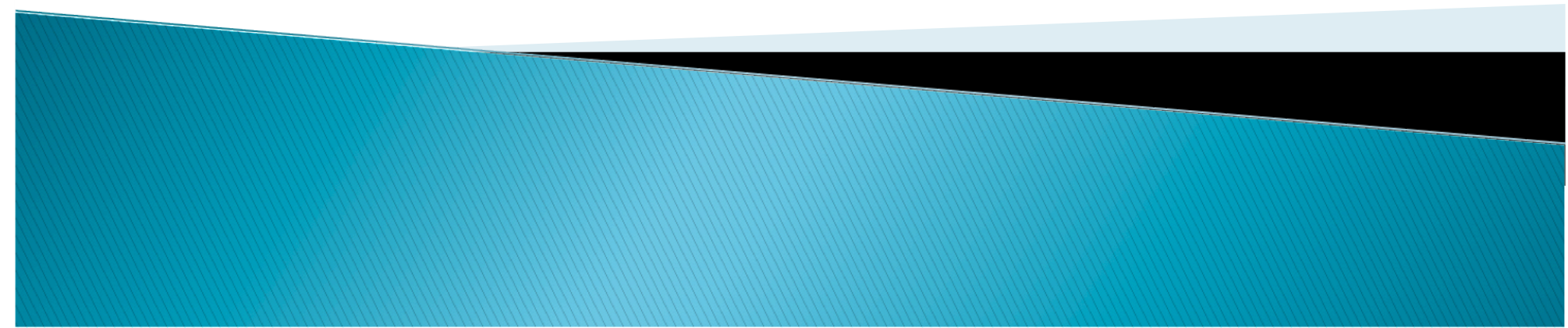
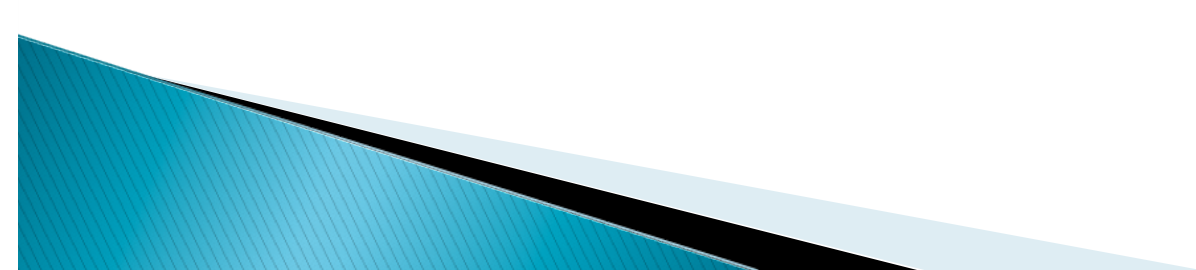


Chapter 5: CPU Scheduling



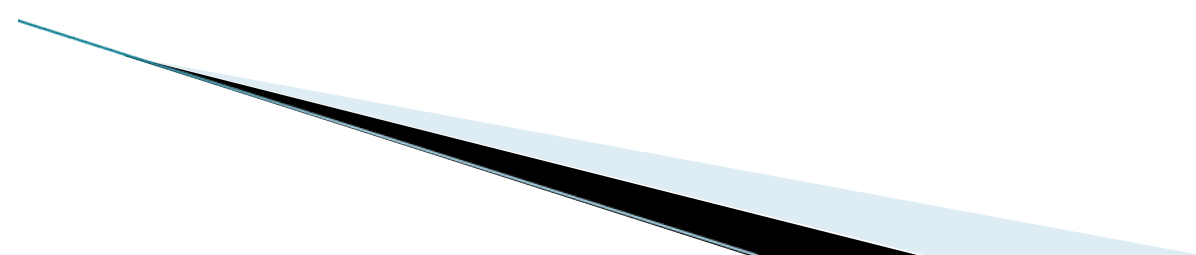
Basic Concepts

- CPU scheduling is the basis of multiprogrammed operating systems
- On operating systems that support threads, it is threads—not processes—that are in fact being scheduled by the operating system.
- However, the terms process scheduling and thread scheduling are often used interchangeably.
- In a single-processor system, only one process can run at a time; any others must wait until the CPU is free and can be rescheduled.

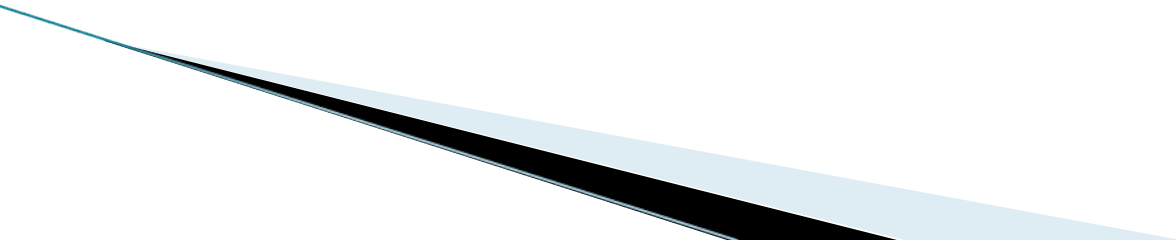


Basic Concepts

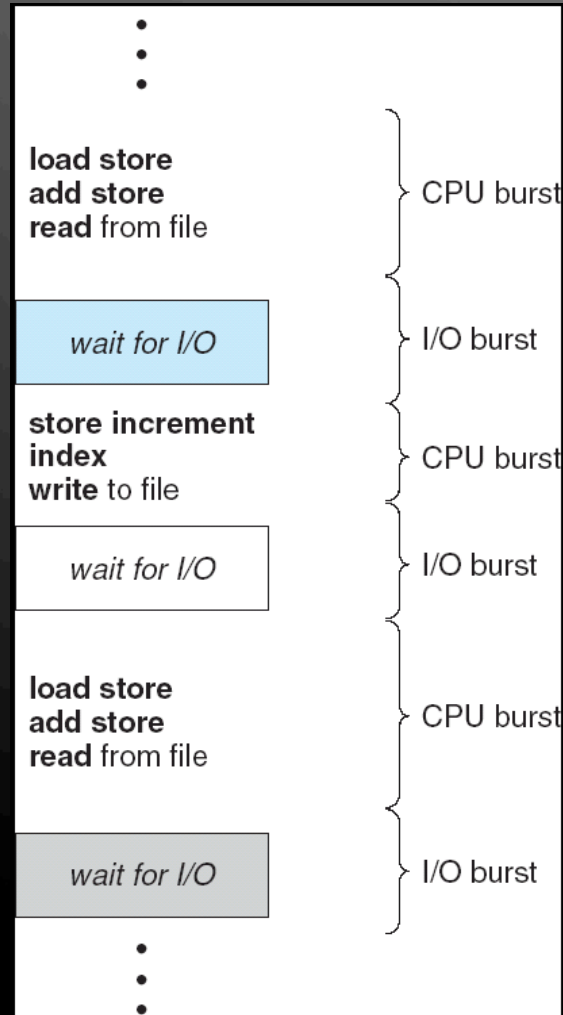
- With multiprogramming, we try to use this time productively. Several processes are kept in memory at one time.
- When one process has to wait, the operating system takes the CPU away from that process and gives the CPU to another process. This pattern continues.



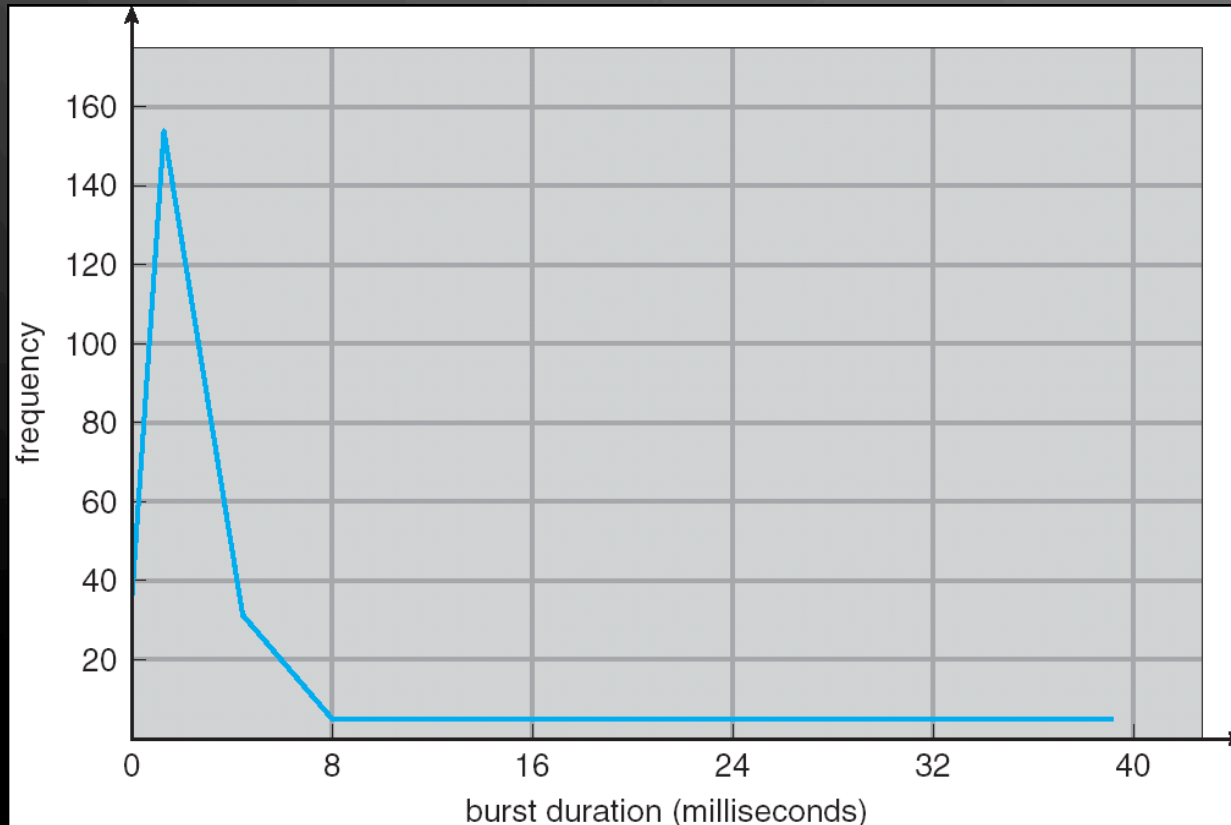
CPU-I/O Burst Cycle

- Process execution consists of a cycle of CPU execution and I/O wait. Processes alternate between these two states. Process execution begins with a CPU burst. That is followed by an I/O burst, which is followed by another CPU burst, then another I/O burst, and so on.
 - The durations of CPU bursts have been measured extensively.
 - Although they vary greatly from process to process and from computer to computer, generally we have a large number of short CPU bursts and a small number of long CPU bursts.
- 

Alternating Sequence of CPU And I/O Bursts

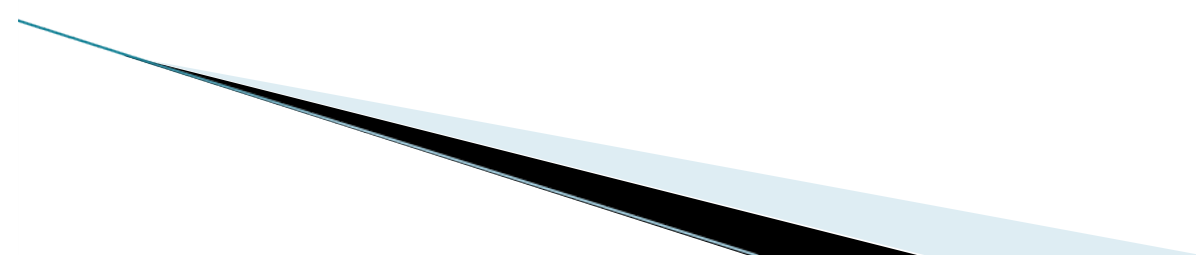


Histogram of CPU-burst Times

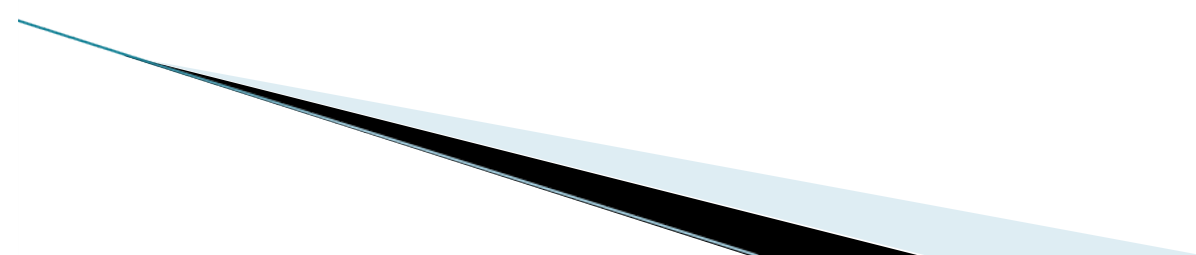


CPU-I/O Burst Cycle

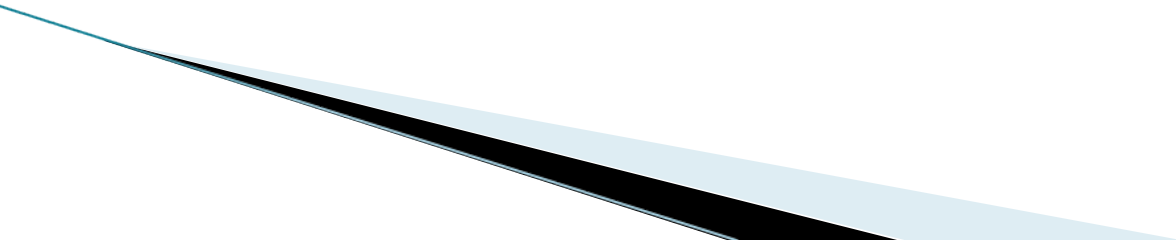
- An I/O-bound program typically has many short CPU bursts. A CPU-bound program might have a few long CPU bursts. This distribution can be important in the selection of an appropriate CPU-scheduling algorithm.



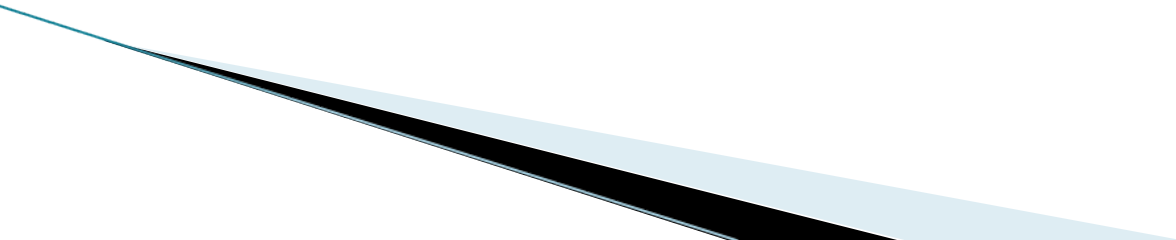
CPU Scheduler

- Whenever the CPU becomes idle, the operating system must select one of the processes in the ready queue to be executed. The selection process is carried out by the short-term scheduler (or CPU scheduler).
 - The scheduler selects a process from the processes in memory that are ready to execute and allocates the CPU to that process.
- 

CPU Scheduler

- 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
 - For situations 1 and 4, there is no choice in terms of scheduling. A new process (if one exists in the ready queue) must be selected for execution. There is a choice, however, for situations 2 and 3.
 - When scheduling takes place only under circumstances 1 and 4, we say that the scheduling scheme is **nonpreemptive** or cooperative; otherwise, it is **preemptive**.
- 

Pre-emptive and non-preemptive scheduling?

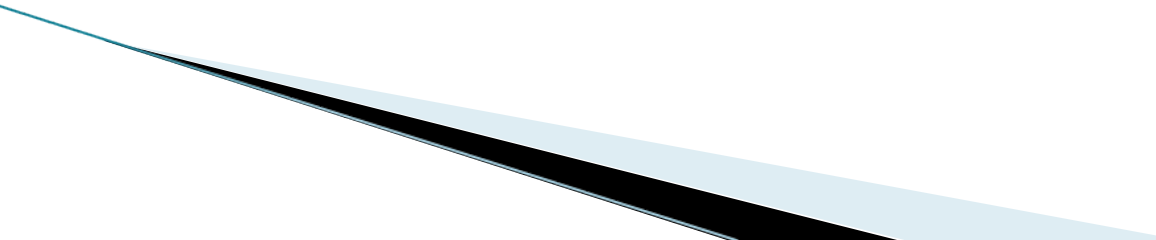
- Processes have priorities and at times it is necessary to run a certain process that has a higher priority before another process although it is running. Therefore, the running process is interrupted for some time and resumed later when the high priority process has finished its execution. This is called preemptive scheduling. Precisely
 - ***Preemptive scheduling***: The preemptive scheduling is prioritized. The highest priority process should always be the process that is currently utilized.
 - ***Non-Preemptive scheduling***: When a process enters the state of running, the state of that process is not deleted from the scheduler until it finishes its service time.
- 

Dispatcher

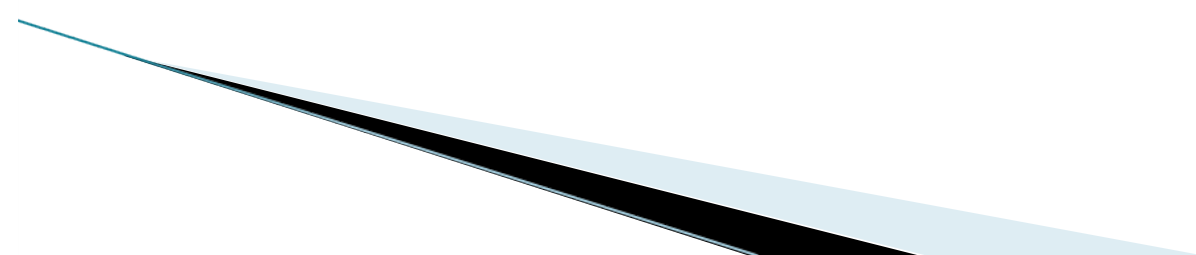
- Dispatcher module gives control of the CPU to the process selected by the short-term scheduler; this involves:
 - switching context
 - switching to user mode
 - jumping to the proper location in the user program to restart that program
- ***Dispatch latency*** – time it takes for the dispatcher to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

Scheduling Criteria

Many criteria have been suggested for comparing CPU scheduling algorithms. The criteria include the following:

- ❑ **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 particular process
 - ❑ **Waiting time** – amount of time a process has been waiting in the ready queue. It is the sum of the periods spent waiting in the ready queue.
 - ❑ **Response time** – amount of time it takes from when a request was submitted until the first response is produced
- 

Optimization Criteria

- ❑ Max CPU utilization
 - ❑ Max throughput
 - ❑ Min turnaround time
 - ❑ Min waiting time
 - ❑ Min response time
- 
- A decorative graphic at the bottom left of the slide, consisting of a light blue triangle pointing upwards and to the right, with a black triangle pointing downwards and to the right, partially overlapping the blue one.

First-Come, First-Served (FCFS) Scheduling

<u>Process</u>	<u>Burst Time</u>
----------------	-------------------

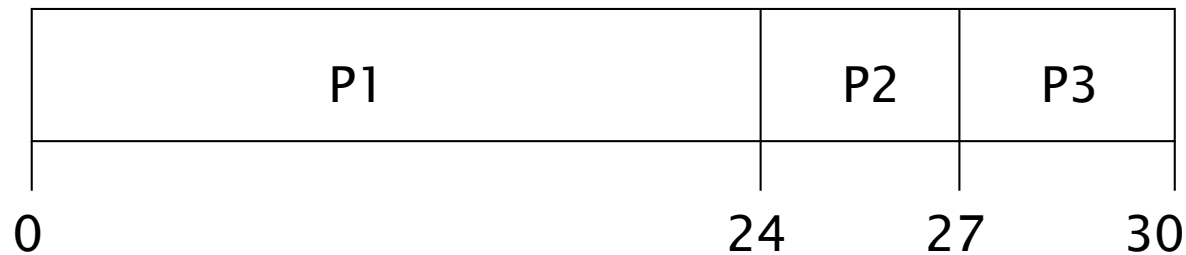
<i>P1</i>	24
-----------	----

<i>P2</i>	3
-----------	---

<i>P3</i>	3
-----------	---

- Suppose that the processes arrive in the order: *P1* , *P2* , *P3*

The Gantt Chart for the schedule is:



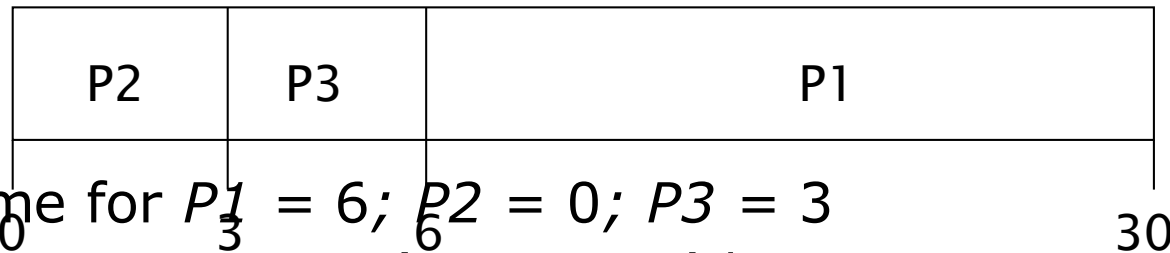
- Waiting time for *P1* = 0; *P2* = 24; *P3* = 27
- Average waiting time: $(0 + 24 + 27)/3 = 17$

FCFS Scheduling (Cont.)

Suppose that the processes arrive in the order

$P2, P3, P1$

- The Gantt chart for the schedule is:



- Waiting time for $P1 = 6$; $P2 = 0$; $P3 = 3$
- Average waiting time: $(6 + 0 + 3)/3 = 3$
- Much better than previous case
- *Convoy effect* short process behind long process