



Operating Systems

Introduction to CPU Scheduling

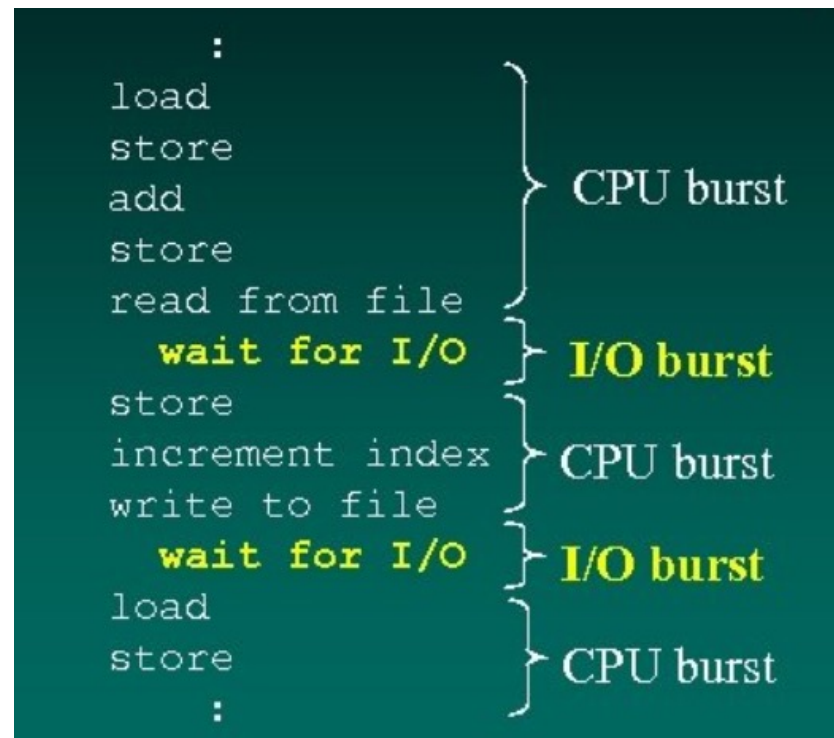
Seyyed Ahmad Javadi

sajavadi@aut.ac.ir

Fall 2022

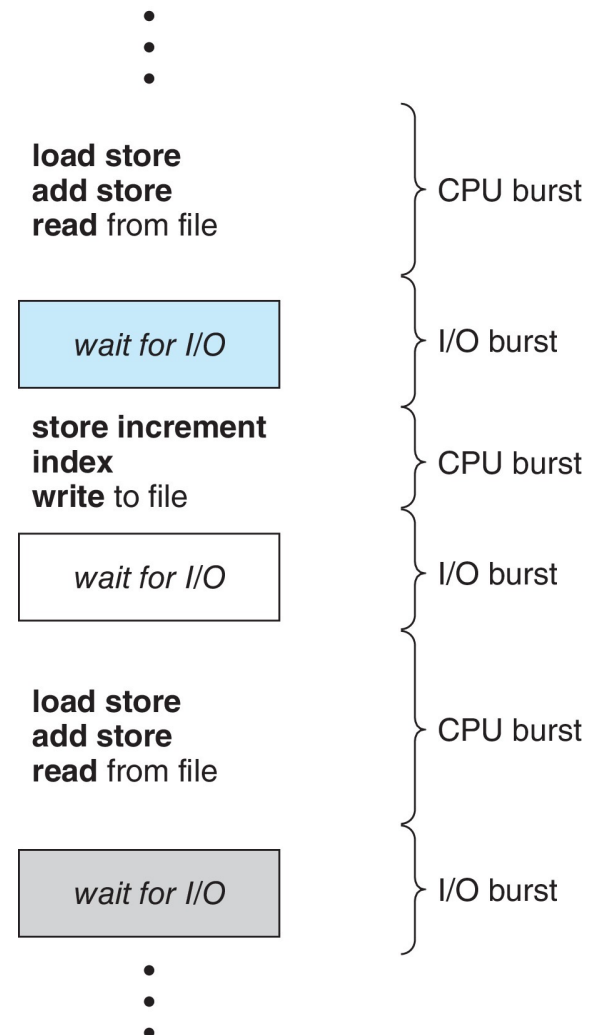
Basic Concepts

- Maximum CPU utilization obtained with multiprogramming
- CPU-I/O Burst Cycle
 - Process execution consists of a **cycle** of CPU execution and I/O wait

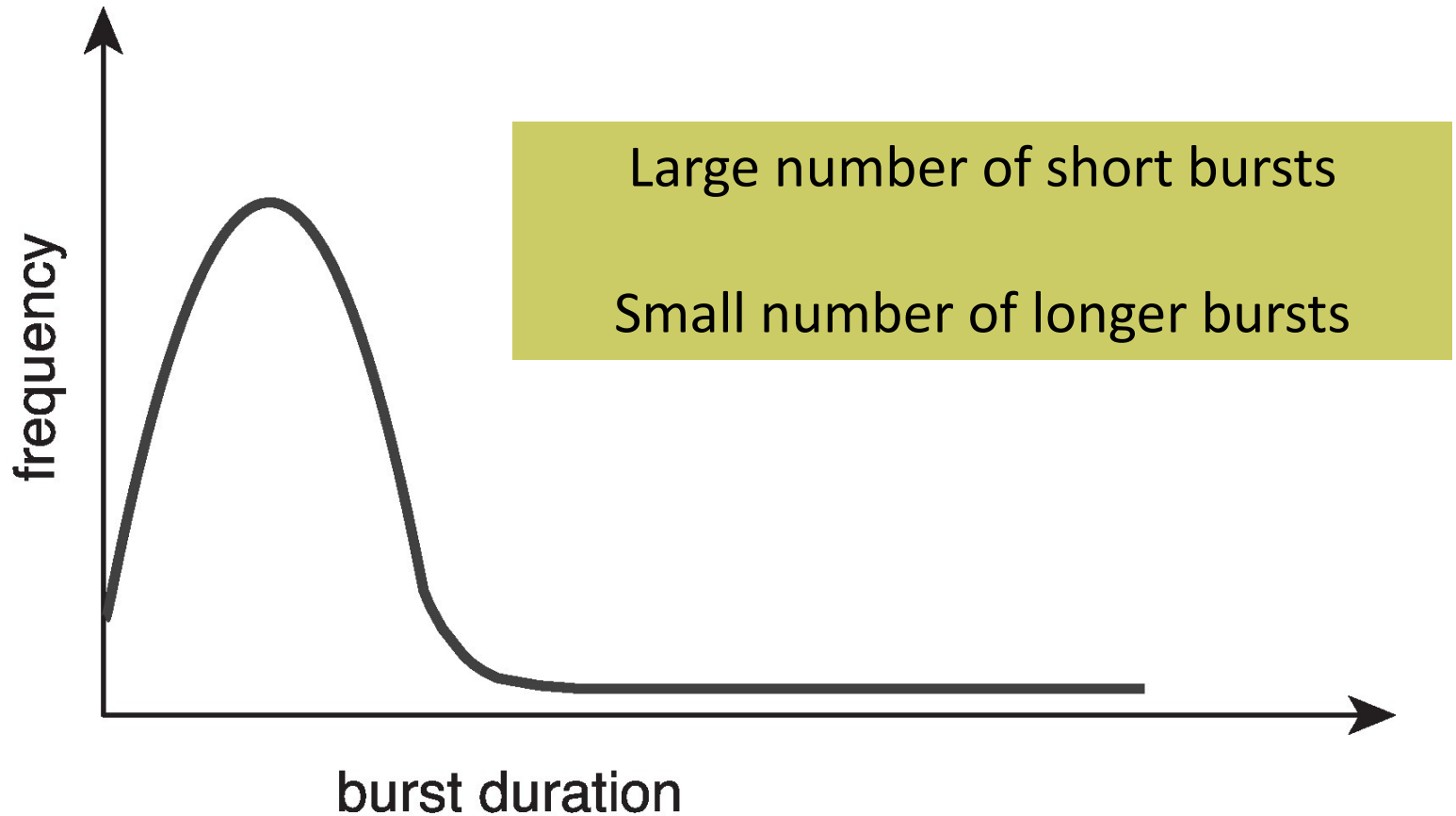


Basic Concepts

- **CPU burst** followed by **I/O burst**
- CPU burst distribution is of main concern

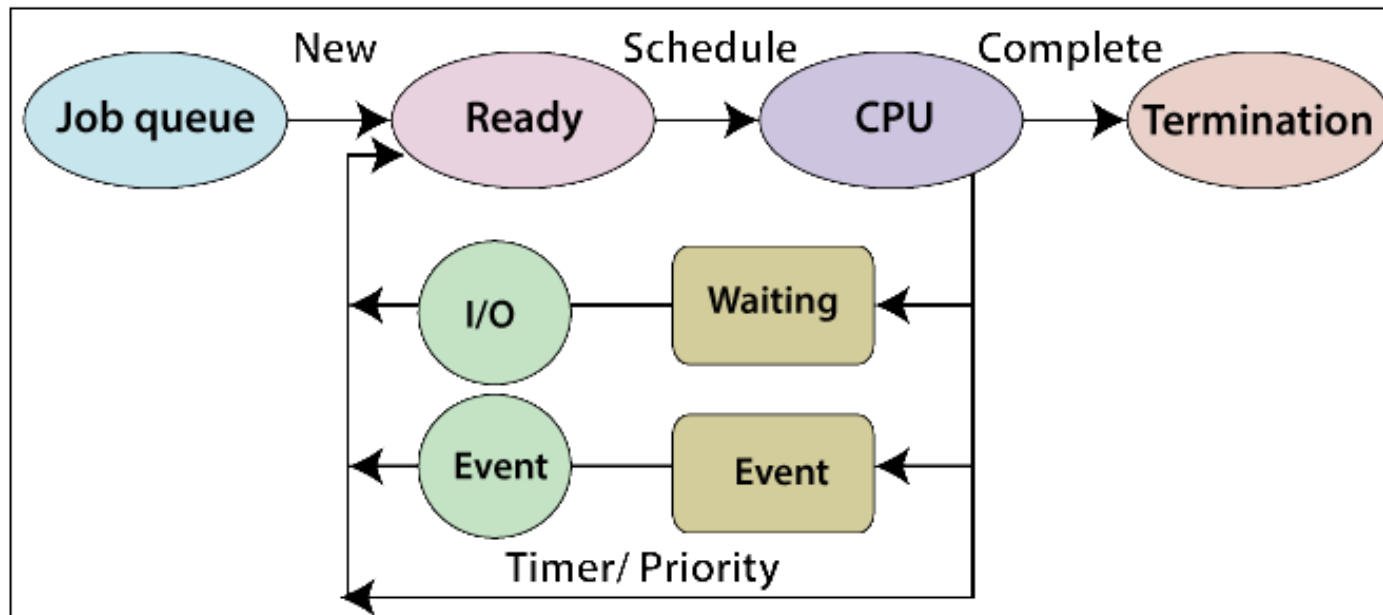


Histogram of CPU-burst Times



CPU Scheduler

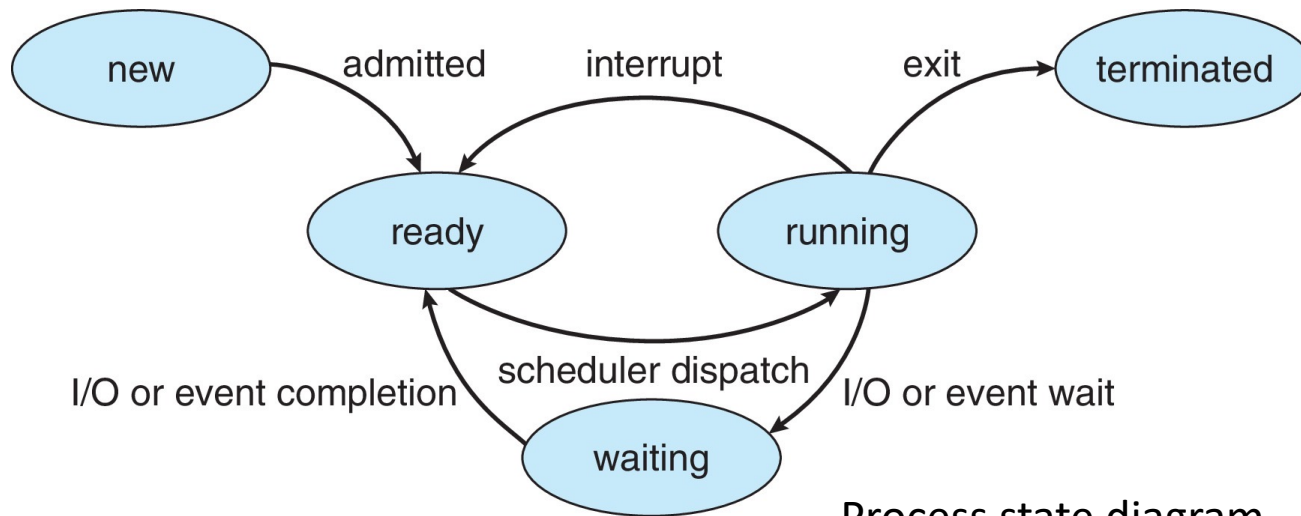
- The **CPU scheduler** selects from among the processes in ready queue and allocates a CPU core to one of them.
 - Queue may be ordered in various ways.



<https://www.tutorialandexample.com/process-schedulers-and-process-queue/>

CPU Scheduler (cont.)

- 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**



Process state diagram

CPU Scheduler (cont.)

- **Four possible scheduling situations**
 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 must be selected for execution.
 - If at least one process exists in the ready queue

- For situations 2 and 3, however, there is a choice.



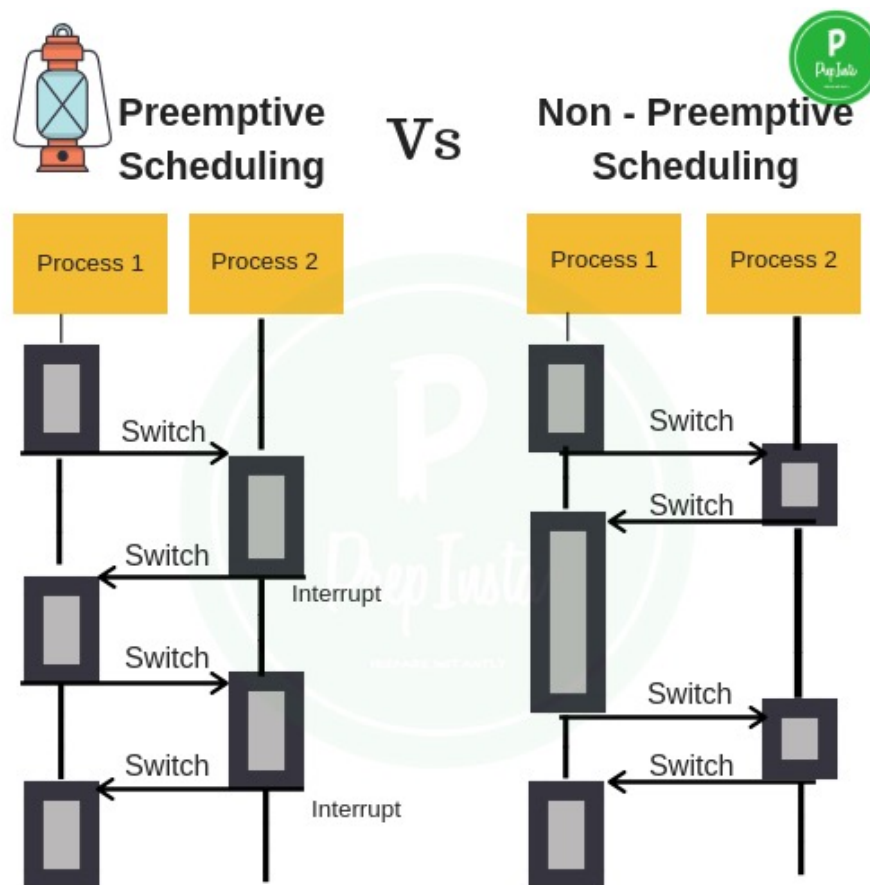
Preemptive and Nonpreemptive Scheduling

■ Non-preemptive (or cooperative)

- Circumstances 1 and 4

■ Preemptive

- Circumstances 2 and 3



Preemptive and Non-preemptive Scheduling (cont.)

■ Non-preemptive scheduling

- Once the CPU has been allocated to a process, the process keeps the CPU until it releases it either by terminating or by switching to the waiting state.

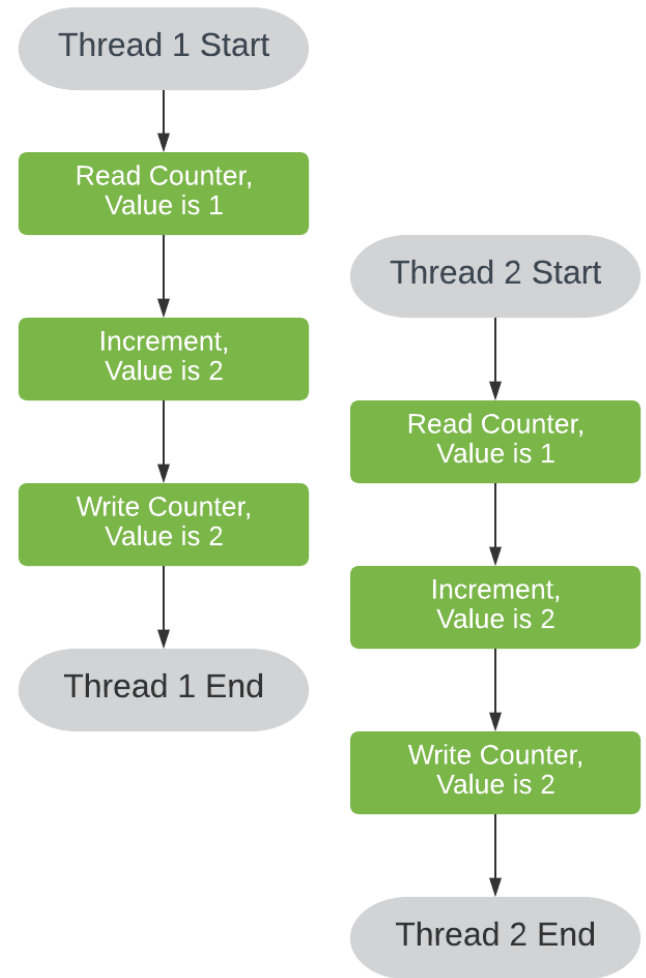
■ Virtually all modern operating systems use preemptive scheduling algorithms.

- Including Windows, MacOS, Linux, and UNIX



Preemptive Scheduling and Race Conditions

- **Preemptive scheduling** can result in **race conditions** when *data are shared* among several processes.



Preemptive Scheduling and Race Conditions (cont.)

- Consider the case of two processes that share data.
 - While one process is **updating the data**, it is preempted so that the second process can run.
 - The second process then tries to read the data, which are in an **inconsistent state**.

- This issue will be explored in detail in Chapter 6.



Dispatcher

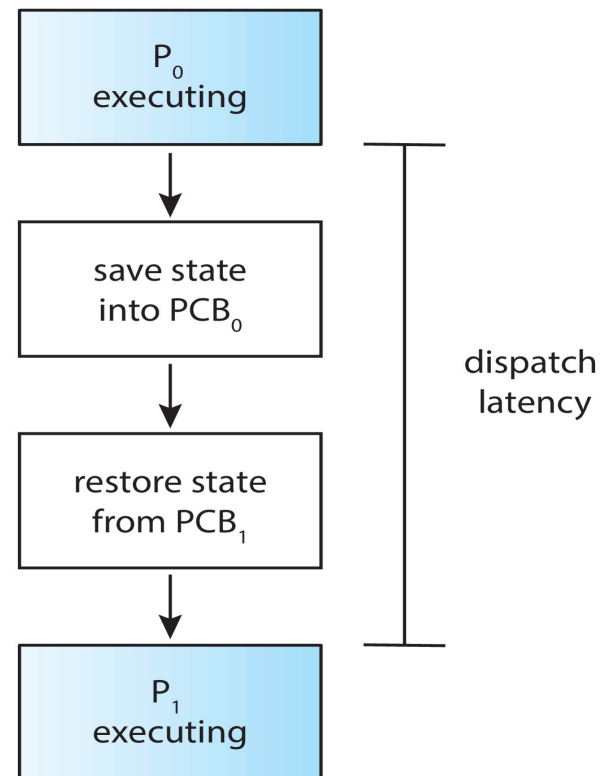
- Gives control of the CPU to the process selected by the CPU scheduler

- 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.



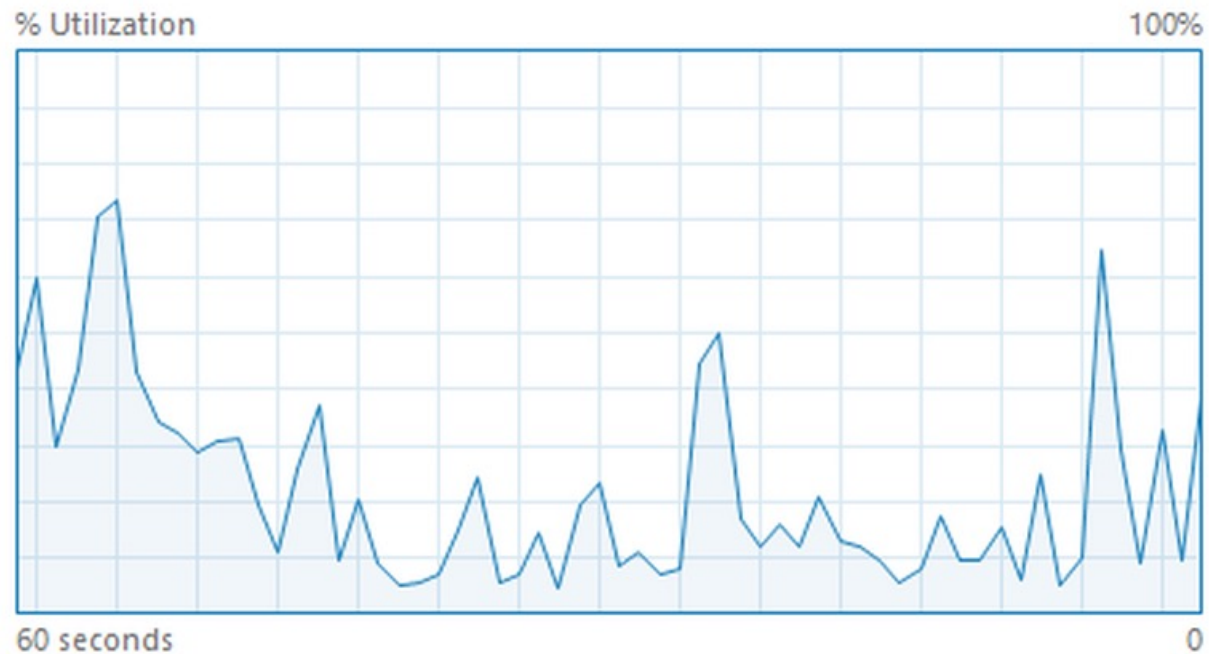
Scheduling Criteria

- CPU utilization
- Throughput
- Turnaround time
- Waiting time
- Response time



CPU Utilization

- Keep the CPU as busy as possible.



Throughput

- Number of processes that complete their execution per time unit.



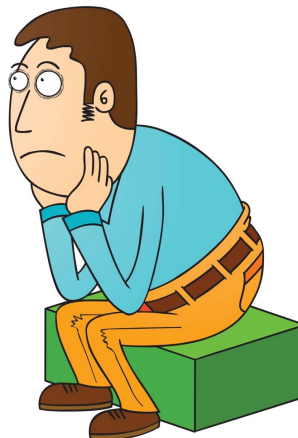
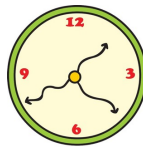
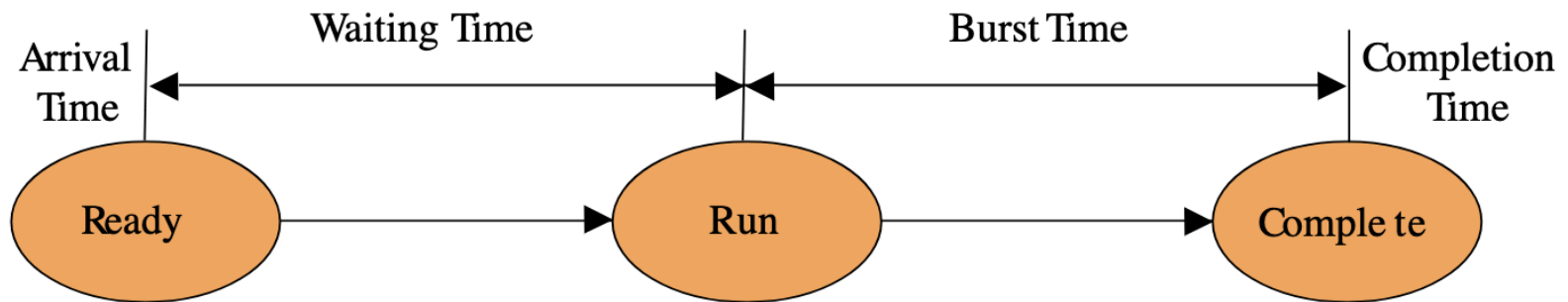
Turnaround Time

- Amount of time to execute a particular process.
- Sum of the periods spent waiting, in the ready queue, executing on the CPU, and doing I/O.



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.



Scheduling Algorithm Optimization Criteria

Criteria	Min or Max?
CPU utilization	
Throughput	
Turnaround time	
Waiting time	
Response time	



Scheduling Algorithm Optimization Criteria

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

