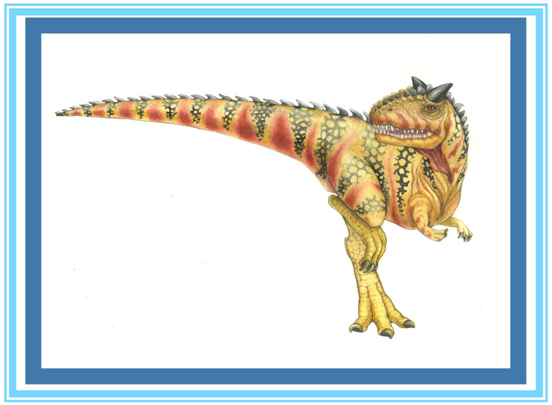
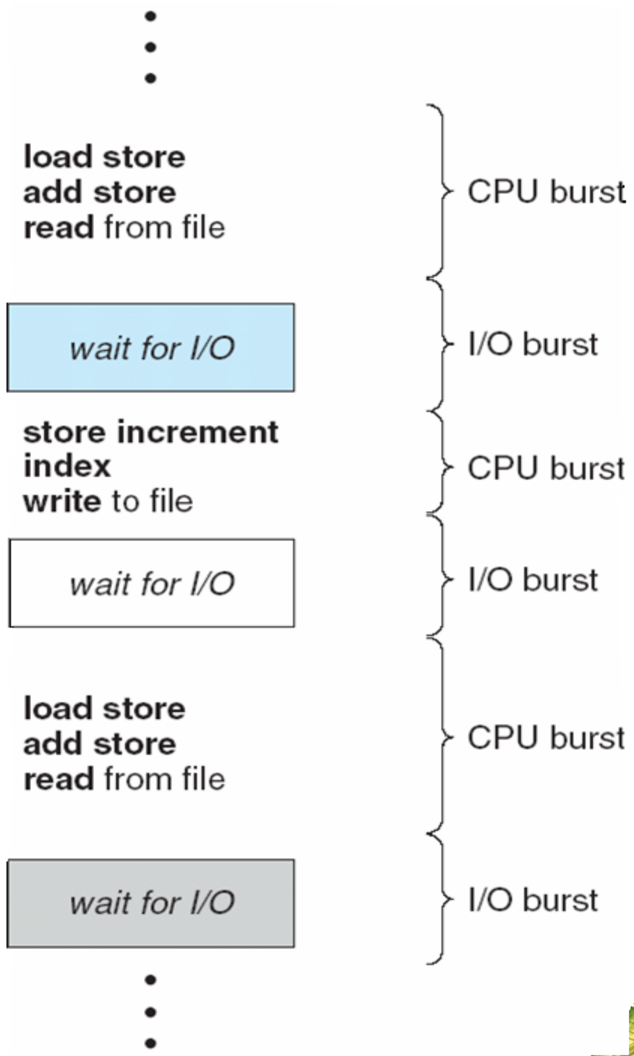
Chapter 5: CPU Scheduling

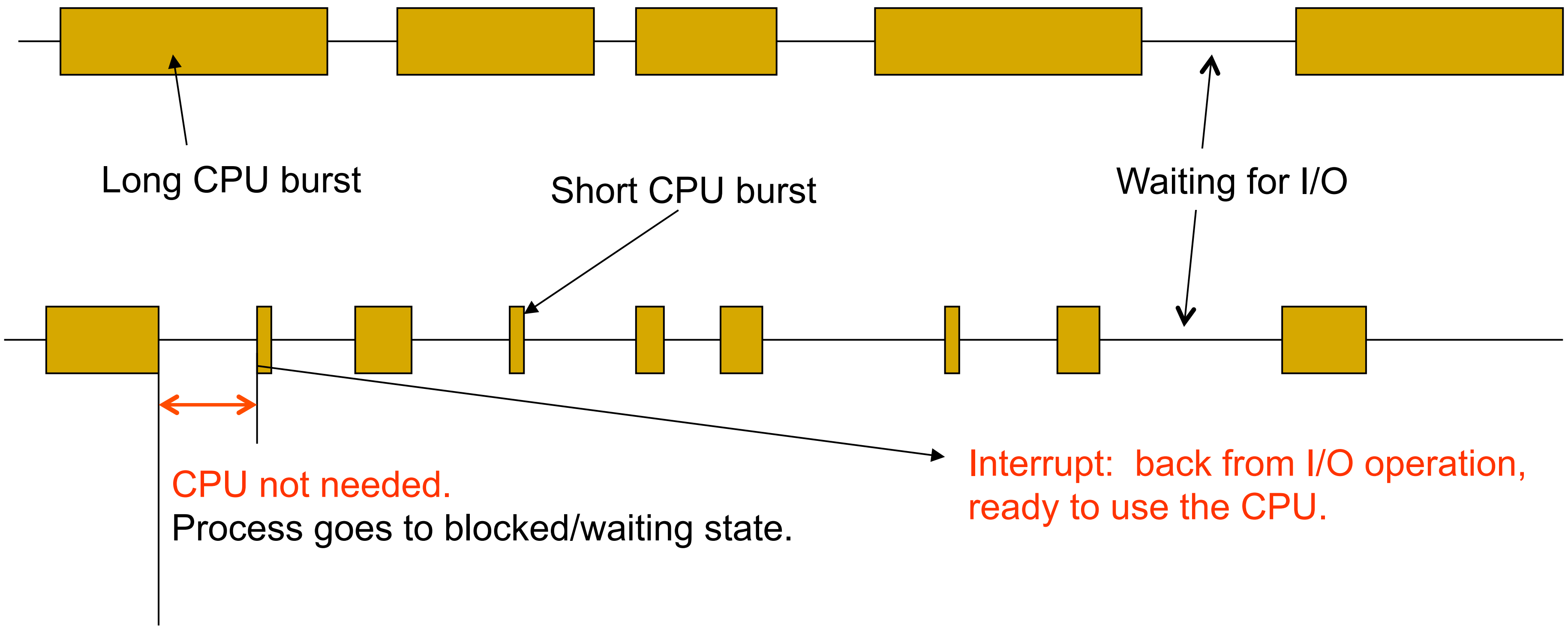


Cycle of CPU/IO Burst

- CPU–I/O Burst Cycle – Process execution consists of a *cycle* of CPU execution and I/O wait

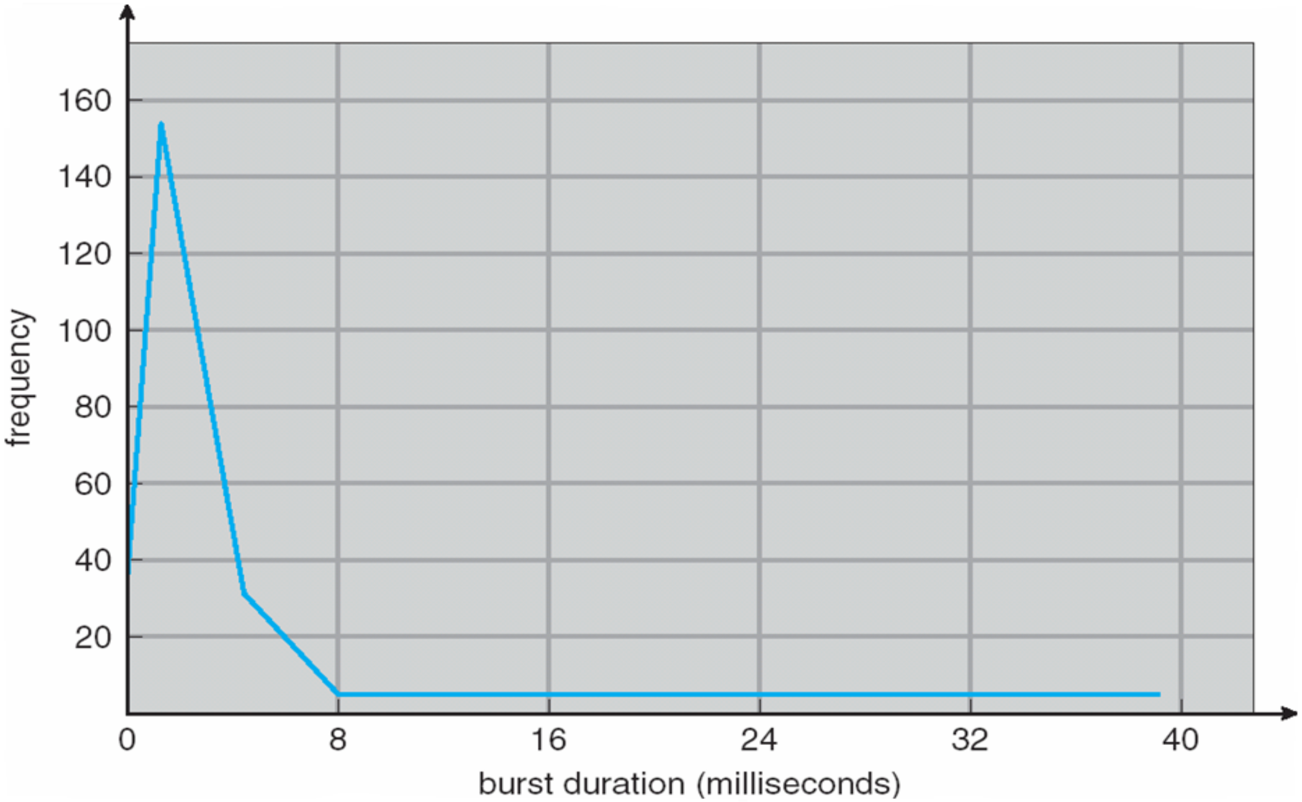


CPU and I/O Bursts



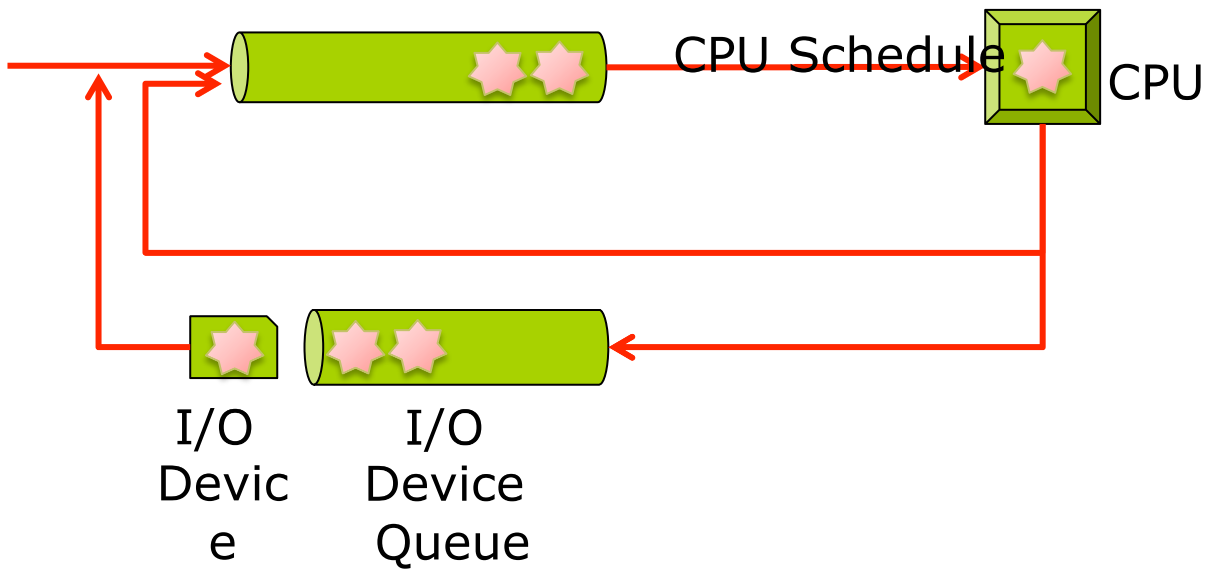
Histogram of CPU-burst Times

- **CPU burst** distribution

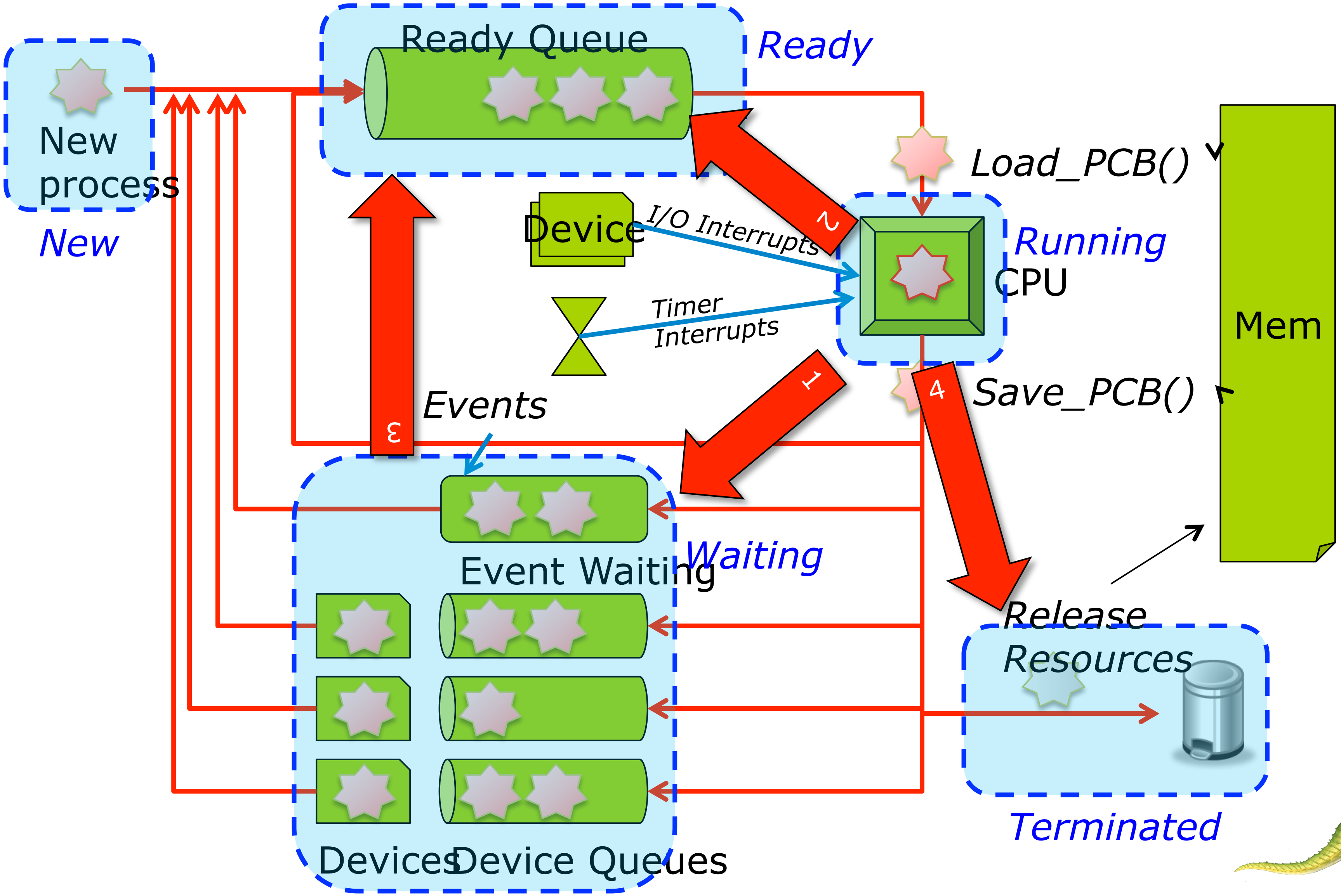


CPU Scheduler

- Short-term scheduler selects a process in ready queue
- Ready queue is not necessarily FIFO
  - Queue may be ordered in various ways



When CPU-scheduling occurs



Preemptive vs. non-Preemptive

- Preemptive: A Process can be suspended and resumed
- Non-preemptive: A process runs until it voluntarily gives up the CPU (waiting on I/O or terminate).
- Most modern OSs use preemptive CPU scheduling, implemented via timer interrupts.

- Non-preemptive is used when suspending a process is impossible or very expensive: e.g., can't "replace" a flight crew in middle of flight.

Silberschatz, Galvin and Gagne ©2009

CPU Scheduler

- CPU scheduling occurs when a process:
  1. running → waiting
  2. running → ready
  3. waiting → ready
  4. running → Terminated
- Scheduling under 1 and 4 is **nonpreemptive**
- All other scheduling is **preemptive**
  - Consider access to shared data
  - Consider preemption while in kernel mode
  - Consider interrupts occurring during crucial OS activities

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 particular 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

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

Scheduling Policies

- Non-preemptive
  - First Come First Served
  - Shortest Job First (aka Shortest Process Next)

First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive at time 0 in the order: P_1 , P_2 , P_3
  The Gantt Chart for the schedule is:

- Waiting time for P_1 = ; P_2 = ; P_3 =
- Average waiting time:

First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive at time 0 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

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

Example of SJF

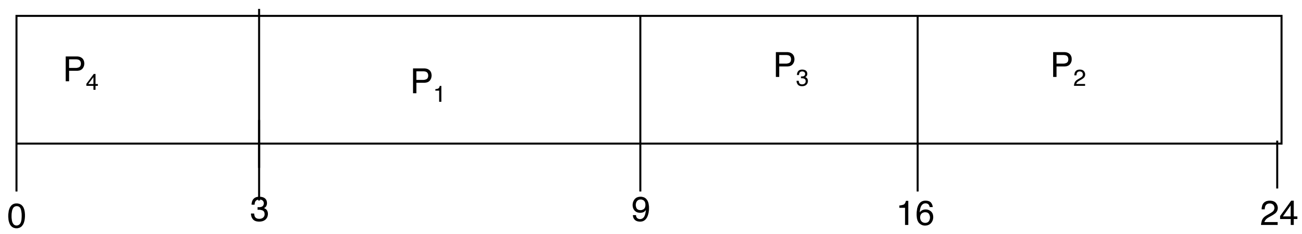
Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3

- SJF scheduling chart
- Average waiting time =

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

Determining Length of Next CPU Burst

- Can only estimate the length – should be similar to the previous one
  - Then pick process with shortest predicted next CPU burst

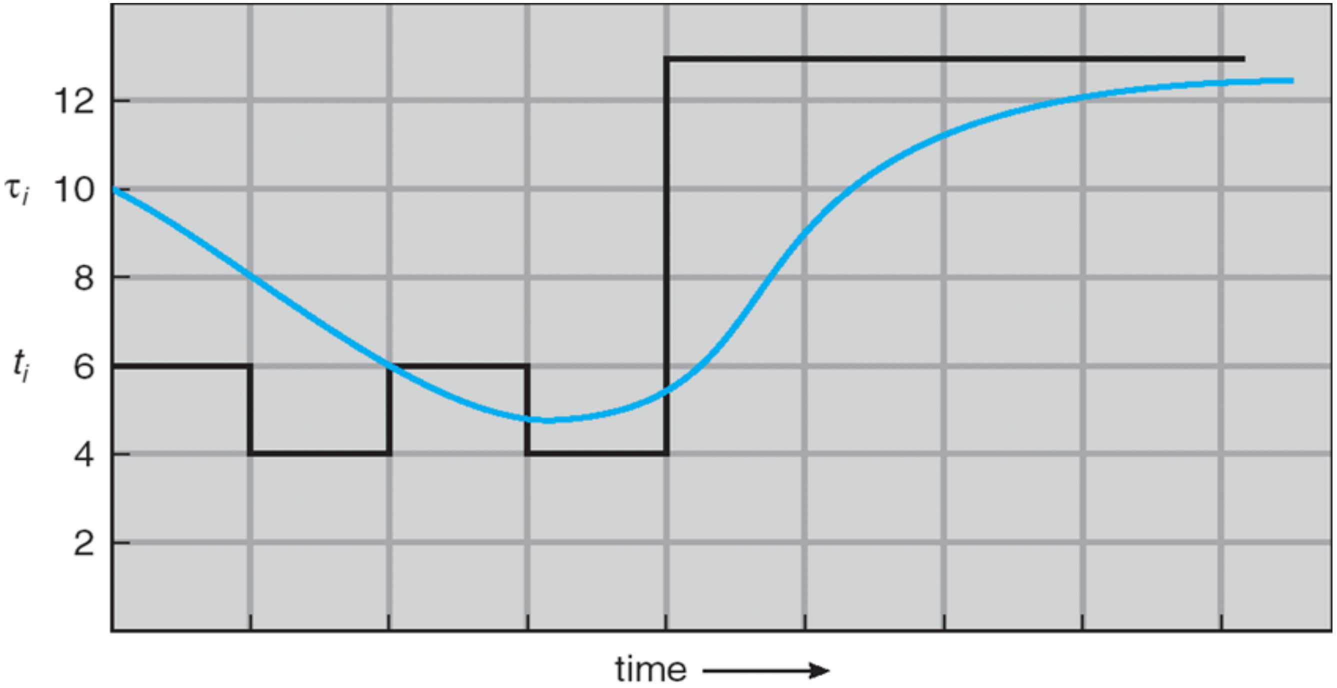
- Can be done by using the length of previous CPU bursts, using exponential averaging

1. $t_n =$ actual length of n^{th} CPU burst
2. $\tau_{n+1} =$ predicted value for the next CPU burst
3. $\alpha, 0 \leq \alpha \leq 1$
4. Define :

$$\tau_{n=1} = \alpha\, t_n + (1 - \alpha)\tau_n .$$

- Commonly, α set to ½
- Preemptive version called **shortest-remaining-time-first**

Prediction of the Length of the Next CPU Burst



CPU burst (t_i)		6	4	6	4	13	13	13 ...
"guess" (τ_i)	10	8	6	6	5	9	11	12 ...

Example of Shortest-remaining-time-first

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- *Preemptive* SJF Gantt Chart

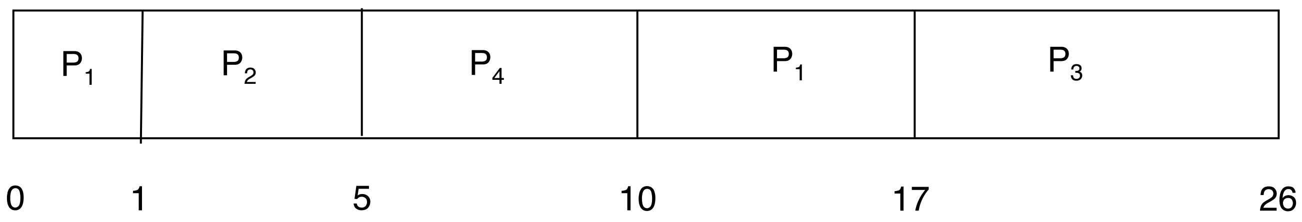
- Average waiting time =

Example of Shortest-remaining-time-first

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- *Preemptive* SJF Gantt Chart



- Average waiting time = [(10-1)+(1-1)+(17-2)+5-3)]/4 = 26/4 = 6.5 msec