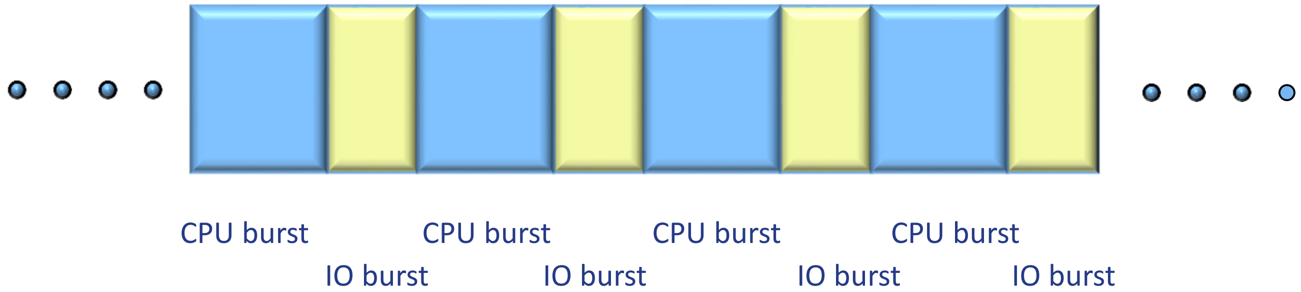
CPU Scheduling

Scheduling criteria, Scheduling Algorithms: Preemptive and non-preemptive, FCFS, SJF, RR

CPU-I/O Burst Cycle

- Process execution consists of a cycle of CPU execution and I/O wait.
- When one process does I/O, a scheduler will typically switch the CPU to another process.



CPU-Bound and I/O-Bound Processes

- A typical process execution has
  - a large number of short CPU bursts
  - a small number of long CPU bursts
- A **CPU-bound** process – mostly long CPU bursts
- A **I/O-bound** process – mostly short CPU bursts

CPU Scheduling

- Selecting a waiting process from the ready queue and allocating the CPU to it.
  - It executes frequently
  - It must be fast
    - If it executes every 100 ms and it takes 10 ms, then 10% of the CPU time has been used for scheduling.
- Many ways to do CPU scheduling.
  How do we compare them?

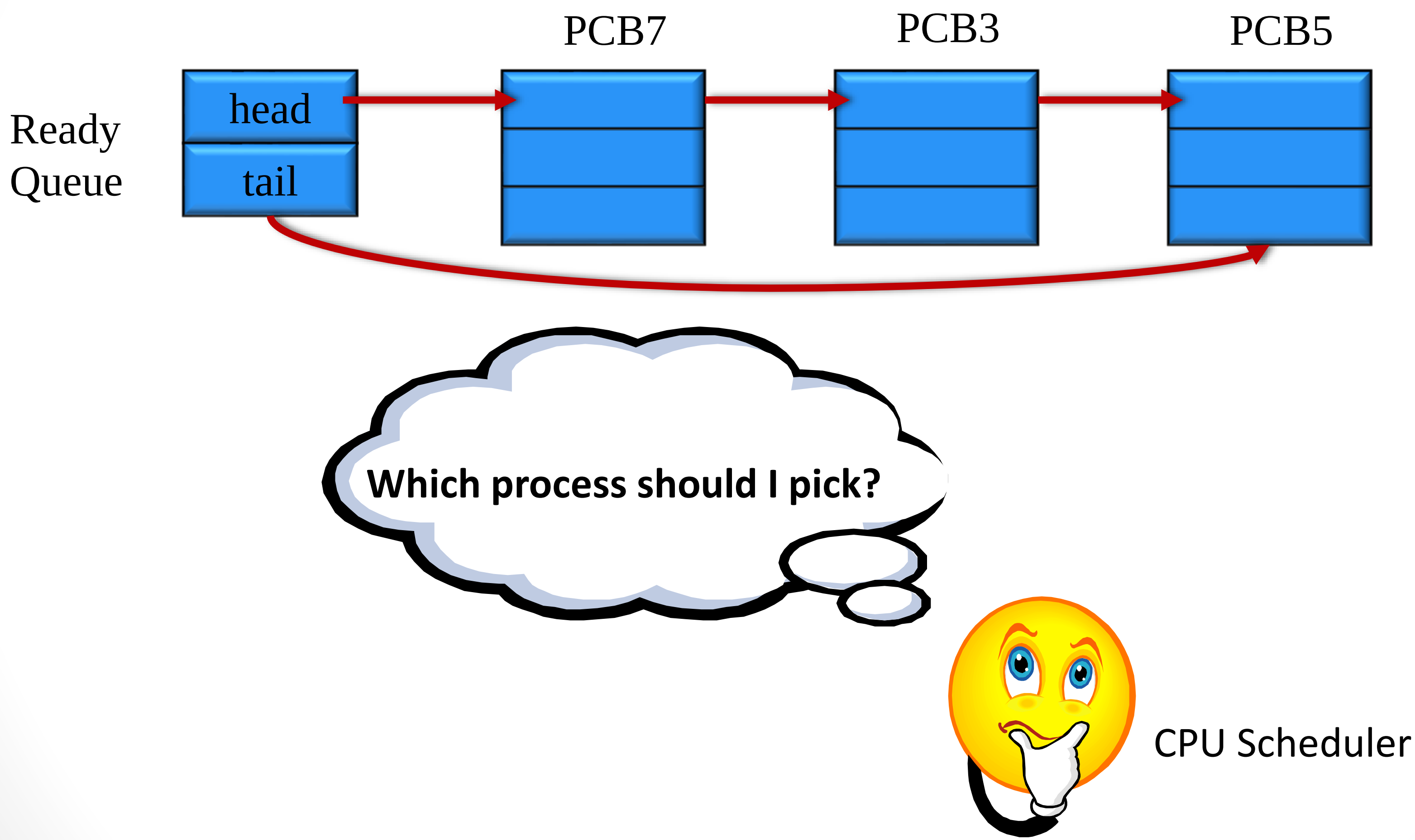
Scheduling Criteria

- Maximize
  - **CPU utilization** – percentage of CPU time spent doing work
  - **Throughput** – number of processes completed per time unit.
- Minimize
  - **Turnaround time** – interval from submission to completion of a process.
  - **Waiting time** – time spent waiting in the ready queue
  - **Response time** – time from submission of a request until the first response is produced (relevant for interactive systems)

Preemptive vs. Non-preemptive Scheduling

- **Non-preemptive Scheduling**: Once the system has assigned a CPU to a process, the system cannot remove that CPU from that process
  - Simpler
  - Up to the process to release the CPU
- **Preemptive Scheduling**: The system can remove the CPU from the running process.
  - Need extra hardware (timer)
  - What if the process is in the middle of updating some data?

Scheduling Algorithms



First-Come, First-Served (FCFS)

- Use FIFO queue.
  - FIFO = First-In First-Out
- Non-preemptive – A process doesn't give up CPU until it either terminates or performs I/O.

FCFS Example

- The following set of processes arrive at time 0

Process	CPU Burst Time (ms)
P1	24
P2	3
P3	3

Assume the processes arrive in the order P1, P2, P3, and are served in FCFS order

FCFS Example

Process	CPU Burst Time
P1	24
P2	3
P3	3

Gantt Chart

Average Waiting time =(0ms + 24ms + 27ms) / 3 = 17ms

Average Turnaround time =(24ms + 27ms + 30ms) / 3 = 27ms

Waiting – time spent waiting in the ready queue

Turnaround – interval from submission to completion of a process.

What if the processes arrive in the order P2, P3, P1?

Process	CPU Burst
P2	3
P3	3
P1	24

Gantt Chart

Average Waiting time =(0ms + 3ms + 6ms) / 3 = 3ms

Average Turnaround time =(30ms + 3ms + 6ms) / 3 = 13ms

Arrival order	Average waiting time	Turnaround time
P1, P2, P3	17	27
P2, P3, P1	3	13

Performance of FCFS Scheduling

- The average waiting time may vary if the process CPU-burst times vary greatly
  - Convoy effect—all the other processes wait for the one big process to release the CPU.
  - Low CPU and device utilization

Shortest-Job-First(SJF)

- Assign CPU to the process that has the smallest next CPU-burst time
- May be either preemptive or non-preemptive

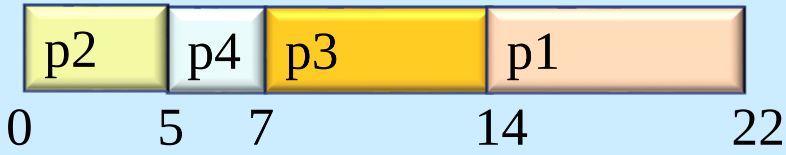
SJF Example

- Consider the following set of processes and their arrival times

Process	Arrival Time	CPU Burst (ms)
P1	0	8
P2	0	5
P3	2	7
P4	2	2

The arrival order of the processes is P1, P2, P3, P4

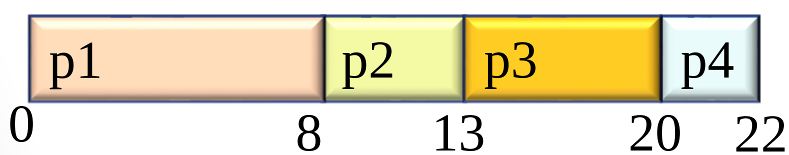
Use non-preemptive SJF



Average Waiting time = (14+0+5+7)/4 = 6.5ms

Process	Arrival Time	CPU Burst (ms)
P1	0	8
P2	0	5
P3	2	7
P4	2	2

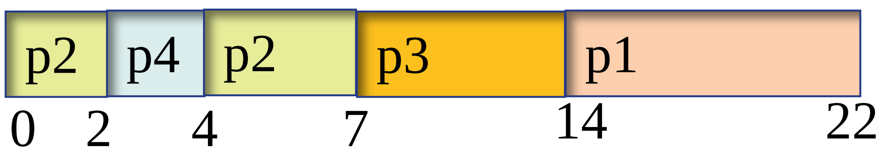
Use FCFS



Average Waiting time = (0+8+13+20)/4 = 10.25ms

Process	Arrival Time	CPU burst
P1	0	8
P2	0	5
P3	2	7
P4	2	2

Use preemptive SJF



Average Waiting time = (14+2+5+7)/4 = 7ms

P1 14-0 (0 is arrival time) = 14

P2 7-2 = 5

P3 7 = 7

P4 2 =2

Shortest-Job-First(SJF)

- Optimal with respect to average waiting time
- How does scheduler know the length of the next CPU-burst time?

Round-Robin(RR) Scheduling

- Similar to FCFS but with preemption.
- Have a **time quantum** (time slice). Let the first process in the queue run until it exceeds the time quantum, then run the next process.

RR - Example

- Consider the following set of processes with their arrival times

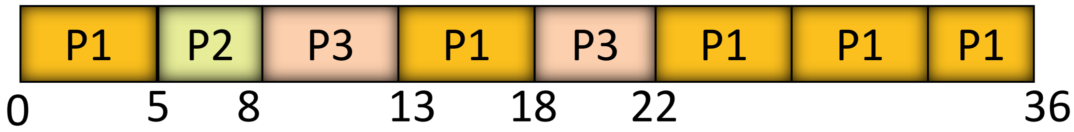
Process	Arrival Time	CPU Burst (ms)
P1	0	24
P2	0	3
P3	0	9

The arrival order of the processes is P1, P2, P3

Use RR Scheduling

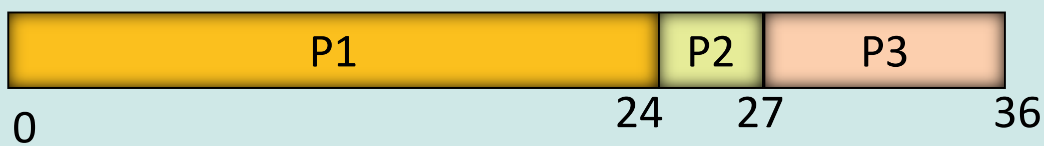
Process	Arrival Time	CPU Burst (ms)
P1	0	24
P2	0	3
P3	0	9

Time quantum = 5ms



Average Waiting Time = (12+5+13)/3 = 10ms

Time quantum = 24ms



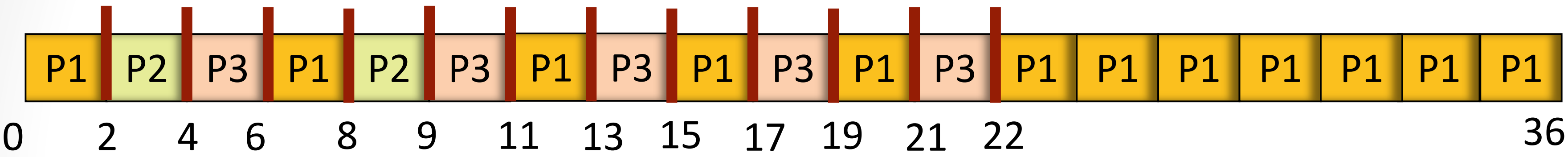
Average Waiting Time = (0+24+27)/3 = 17ms

Turns into FCFS scheduling

Using RR Scheduling

Process	Arrival Time	CPU Burst (ms)
P1	0	24
P2	0	3
P3	0	9

Time quantum = 2ms



Average waiting Time = (12+6+13)/3 = 10.33ms

If context-switch time = 1ms

Total context-switch time = 12ms

Round-Robin (RR) Scheduling

- Very small quantum
  - large context switch overhead.
- Very big quantum
  - turns into FCFS.

Round-Robin (RR) Scheduling

- The time quantum should be large with respect to the context-switch time.
  - In most modern OSs
    - Quantum time range: 10-100 milliseconds
    - Context-switch time: <10 microseconds
- The time quantum should not be too large.
  - If the quantum time is too large, RR scheduling degenerates into FCFS.

Summary

- CPU Scheduling
- Criteria
  - MAX: Utilization, Throughput
  - MIN: Waiting time, Turnaround time, Response time
- Algorithms
  - First-Come First-Served
  - Shortest-Job-First
  - Round-Robin