CS 5323 – OS II

Lecture 7 – Process Scheduling

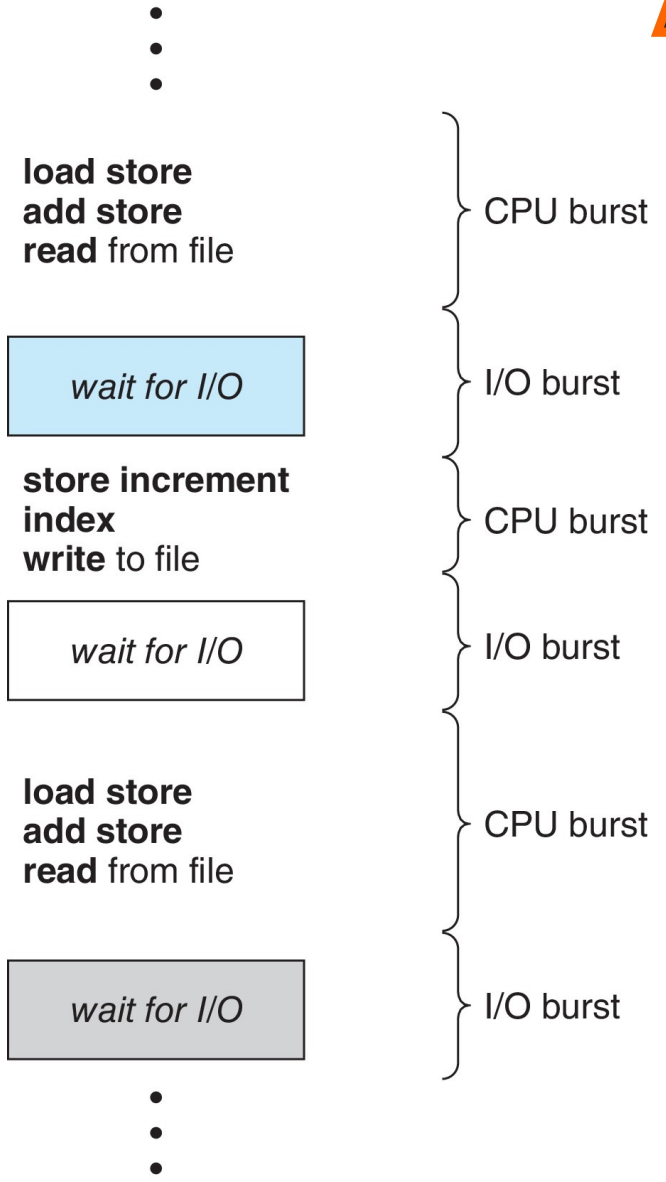
Logistics

- Quiz 3 will be posted on Wednesday.
  - Due 02/14/2022 11:59 pm
- Assignment 2 will be posted on Wednesday.
  - Due 02/25/2022 11:59 pm
- Midterm 02/21/2022
  - Covers everything up to and including scheduling and deadlocks

Process Scheduling

Basic Concepts

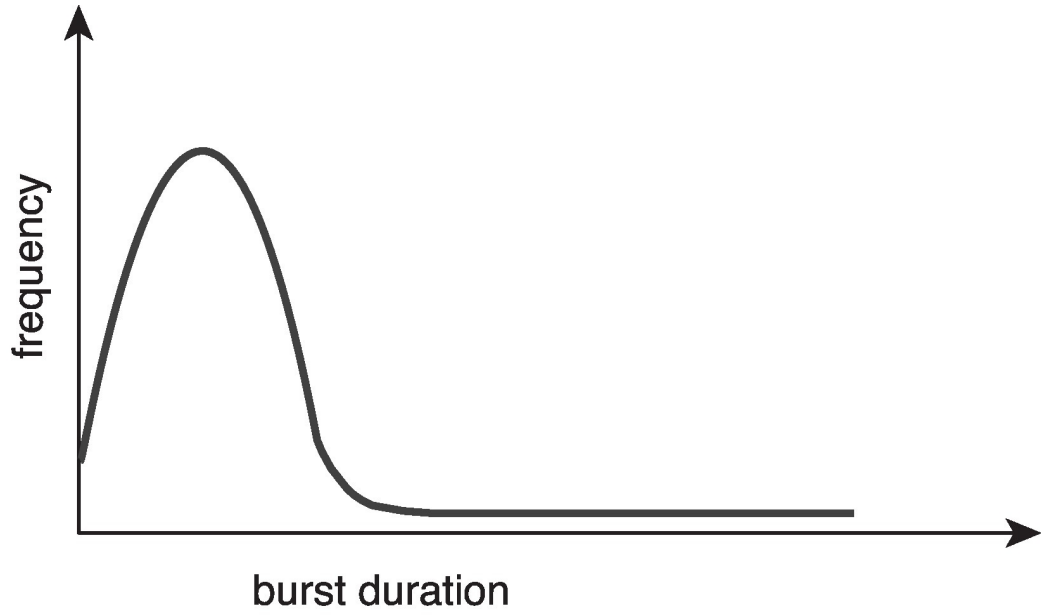
- Maximum CPU utilization obtained with multiprogramming
- CPU–I/O Burst Cycle – Process execution consists of a **cycle** of CPU execution and I/O wait
- **CPU burst** followed by **I/O burst**
- CPU burst distribution is of main concern



Histogram of CPU-burst Times

Large number of short bursts

Small number of longer bursts

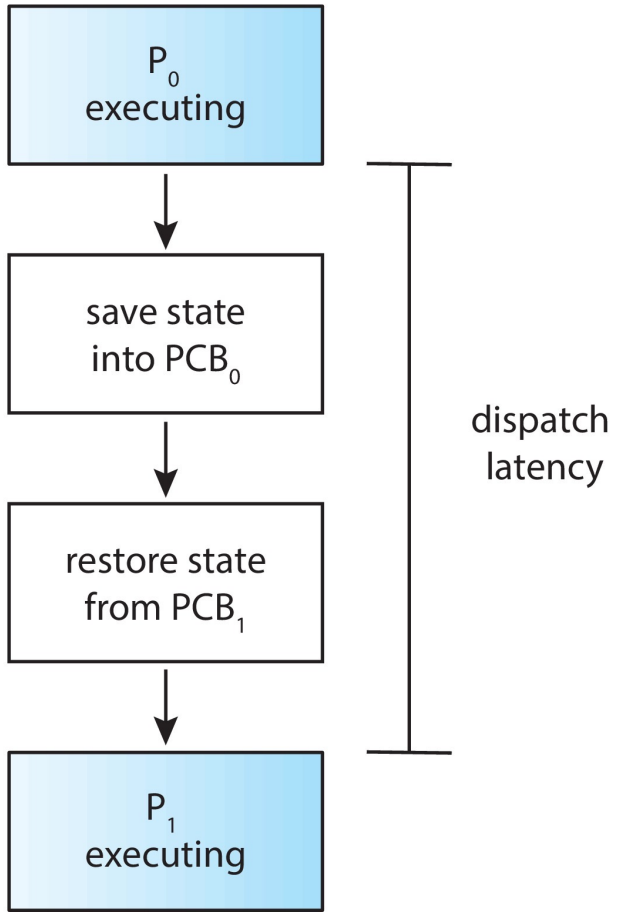


CPU Scheduler

- The **CPU scheduler** selects from among the processes in ready queue, and allocates the a CPU core to one of them
  - Queue may be ordered in various ways
  - Done by short-term scheduler
  - Queue is not necessarily FIFO!!
- 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



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 Metrics

- **Completion Time**: Time taken for the execution to complete, starting from arrival time.
- **Turn Around Time**: Time taken to complete after arrival. In simple words, it is the difference between the Completion time and the Arrival time.
- **Waiting Time**: Total time the process has to wait before it's execution begins. It is the difference between the Turn Around time and the Burst time of the process.

Non-preemptive Scheduling

First- Come, First-Served (FCFS) Scheduling

- Simplest scheduling scheme
- Any process that requests the CPU time is given access through a queue
- When process requests CPU time, its PCB added to a linked-list, at the tail.
- Process at the head of the queue is removed and executed.

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

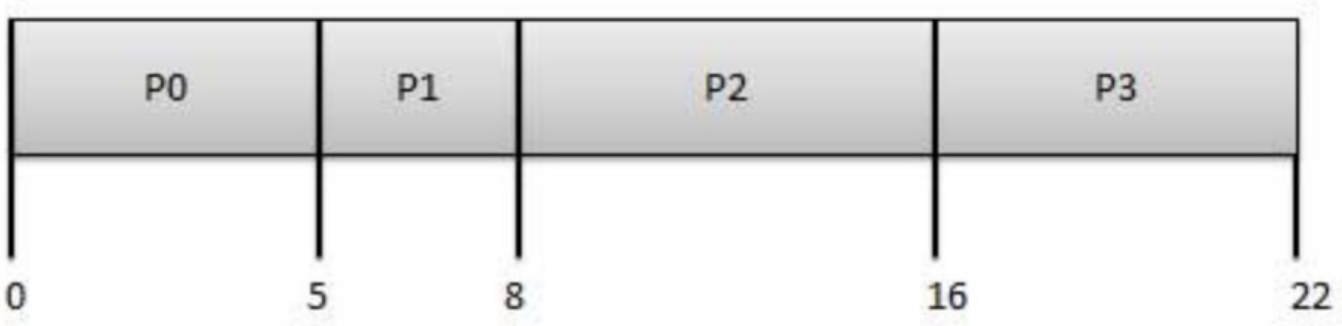
FCFS (contd.)

- Non-preemptive
- Once CPU has been allocated to a process, it runs till completion
  - Or I/O-bound waiting
- Troublesome for time-sharing systems.
  - Why?
  - Each user should get a share of the CPU at regular intervals.
    - Disastrous to allow one process to use CPU for extended periods

FSCS Example -- Review

- Consider the processes P0, P1, P2, P3 given in the below table, arrives for execution in the **given order**, **arrival times**, and given **Burst Time.** What is the **average waiting time and average turn around time** using the FCFS scheduling algorithm?

Process ID	Arrival Time	Burst Time
0	0	5
1	1	3
2	2	8
3	3	6



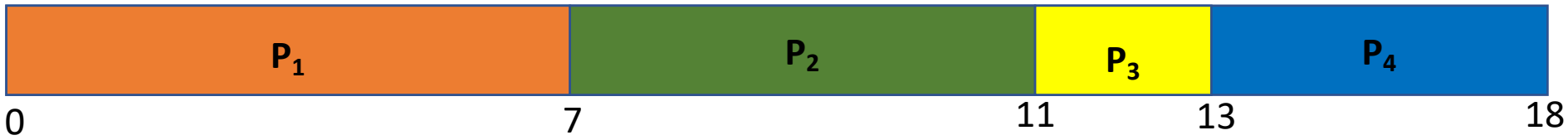
Completion Time <-- From Gantt chart
Turnaround time = Completion time – arrival time
Wait time = Turnaround time – burst time

Process ID	Arrival Time	Burst Time	Completion Time	Turnaround Time	Wait time
0	0	5	5	5	0
1	1	3	8	7	4
2	2	8	16	14	6
3	3	6	22	19	13

FSCS Example 1

- Consider the processes P0, P1, P2, P3 given in the below table, arrives for execution in the **given order**, **arrival times**, and given **Burst Time.** What is the **average waiting time and average turn around time** using the FCFS scheduling algorithm?

Process ID	Arrival Time	Burst Time
1	0	7
2	0	4
3	3	2
4	4	5



Process ID	Arrival Time	Burst Time	Completion Time	Turnaround time	Wait time
1	0	7	7	7	0
2	0	4	11	11	7
3	3	2	13	10	8
4	4	5	18	14	9

Av. TAT = 10.5ms Av. WT = 6 ms

FCFS – Example 2

Process ID	Arrival Time	Burst Time
1	0	8
2	0	6
3	0	1
4	0	9
5	0	3

Process ID	Arrival Time	Burst Time	Completion Time	Turnaround time	Wait time
1	0	8	8	8	0
2	0	6	14	14	8
3	0	1	15	15	14
4	0	9	24	24	15
5	0	3	27	27	24

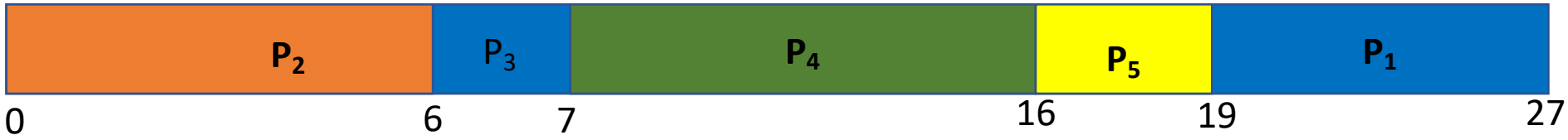
Av. TAT = 17.6 ms Av. WT = 12.2 ms

Priority Scheduling

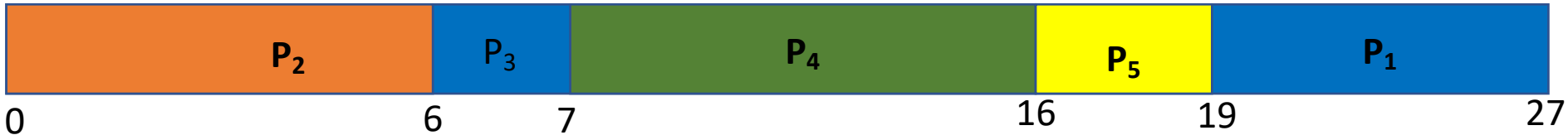
- A priority number (integer) is associated with each process
- The CPU is allocated to the process with the highest priority (smallest integer ≡ highest priority)
- **Higher priority executes first!**

FCFS with Priority

Process ID	Arrival Time	Burst Time	Priority
1	0	8	4
2	0	6	1
3	0	1	2
4	0	9	2
5	0	3	3



Process ID	Arrival Time	Burst Time	Priority
1	0	8	4
2	0	6	1
3	0	1	2
4	0	9	2
5	0	3	3



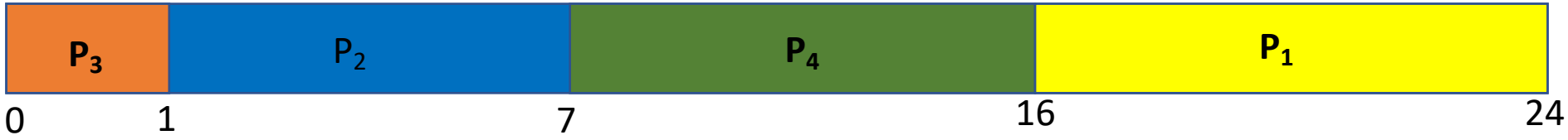
Process ID	Arrival Time	Priority	Burst Time	Completion Time	Turnaround time	Wait time
1	0	4	8	27	27	19
2	0	1	6	6	6	0
3	0	2	1	7	7	6
4	0	2	9	16	16	7
5	0	3	3	19	19	16

Av. TAT = 15 ms Av. WT = 9.6 ms

FCFS with Priority – Example 2

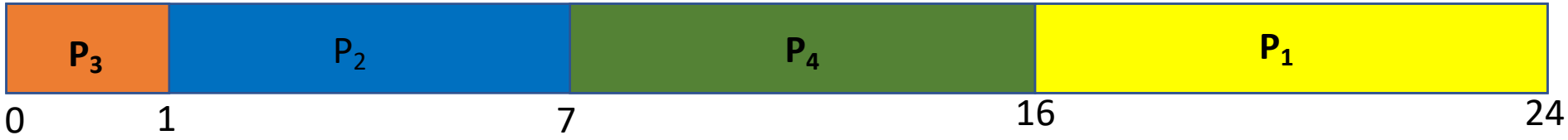
Process ID	Arrival Time	Burst Time	Priority
1	2	8	4
2	1	6	1
3	0	1	3
4	3	9	2

FCFS with Priority – Example 2



Process ID	Arrival Time	Burst Time	Priority
1	2	8	4
2	1	6	1
3	0	1	3
4	3	9	2

FCFS with Priority – Example 2



Process ID	Arrival Time	Burst Time	Priority	Completion Time	Turnaround Time	Wait time
1	2	8	4	24	22	14
2	1	6	1	7	6	0
3	0	1	3	1	1	0
4	3	9	2	16	13	4

Av. TAT = 10.5 ms Av. WT = 4.5ms

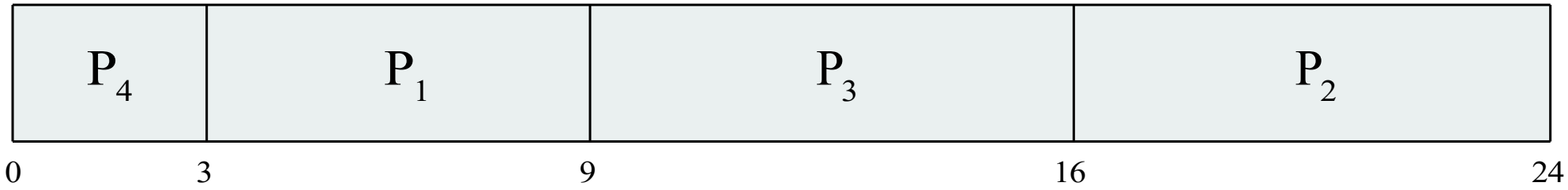
Shortest-Job-First (SJF) Scheduling

- 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
- SJF is priority scheduling where priority is the inverse of predicted next CPU burst 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
- FCFS wait time: 10.25

SJF – Example 2

Process ID	Arrival Time	Burst Time
1	0	8
2	0	6
3	0	1
4	0	9
5	0	3

Process ID	Arrival Time	Burst Time	Completion Time	Turnaround time	Wait time
1	0	8	18	18	10
2	0	6	10	10	4
3	0	1	1	1	0
4	0	9	27	27	18
5	0	3	4	4	1

Av. TAT = 12 ms Av. WT = 6.6 ms

Priority Scheduling

- A priority number (integer) is associated with each process
- The CPU is allocated to the process with the highest priority (smallest integer ≡ highest priority)
  - Preemptive
  - Nonpreemptive
- Problem ≡ **Starvation** – low priority processes may never execute
  - Famous case of a process on an IBM 7094 computer at MIT
    - Low priority process initiated in 1967 had been blocked until the system was shut down in 1973!
- Solution ≡ **Aging** – as time progresses increase the priority of the process

Preemptive Scheduling

Preemptive Scheduling

- **Preemptive Scheduling** is defined as the scheduling which is done when the process changes from running state to ready state or from waiting for the state to ready state.
- Resources are allocated to execute the process for a certain period. After this, the process is taken away in the middle and is placed in the ready queue its bursts time is left and this process will stay in ready line until it gets its turn to execute.

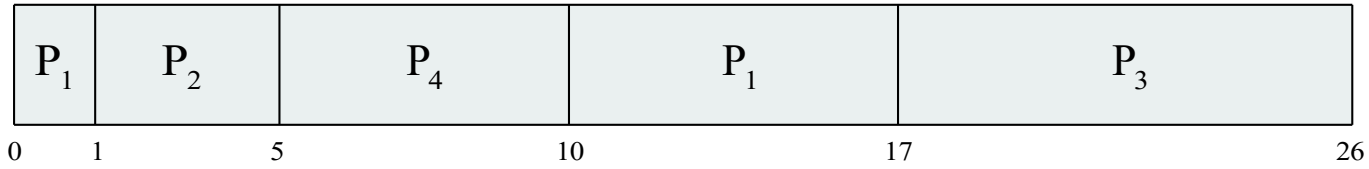
Shortest Remaining Time First

- Preemptive version of Shortest Job First scheduling
- Will reduce average wait time

Shortest Remaining Time First

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

Shortest Remaining Time First



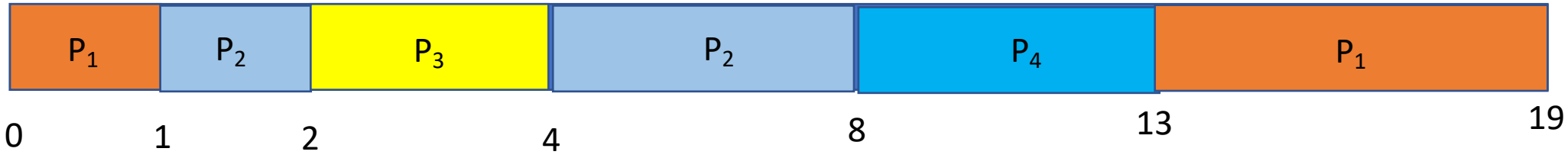
Process ID	Arrival Time	Burst Time	Completion Time	Turnaround Time	Wait time
1	0	8	17	17	9
2	1	4	5	4	0
3	2	9	26	24	15
4	3	5	10	7	2

Av. TAT = 13 ms Av. WT = 6.5 ms

SRT – Example 2

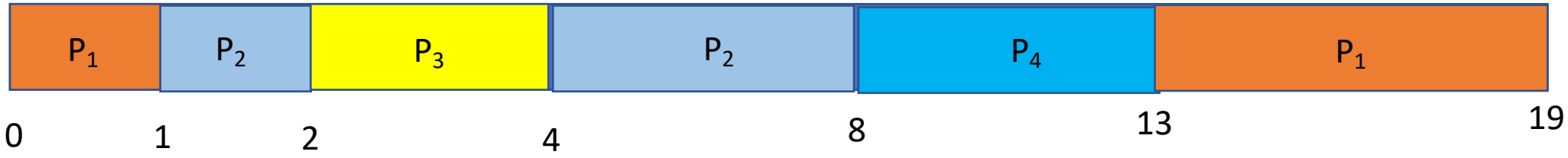
Process ID	Arrival Time	Burst Time
1	0	7
2	1	5
3	2	2
4	3	5

SRT – Example 2



Process ID	Arrival Time	Burst Time
1	0	7
2	1	5
3	2	2
4	3	5

SRT – Example 2



Process ID	Arrival Time	Burst Time	Completion Time	Turnaround Time	Wait time
1	0	7	19	19	12
2	1	5	8	7	2
3	2	2	4	2	0
4	3	5	13	10	5

Av. TAT = 9.5 ms

Av. WT = 5.75 ms

Round Robin (RR)

- Each process gets a small unit of CPU time (**time quantum** q), usually 10-100 milliseconds. After this time has elapsed, the process is preempted and added to the end of the ready queue.
- If there are n processes in the ready queue and the time quantum is q, then each process gets $1/n$ of the CPU time in chunks of at most q time units at once. No process waits more than $(n-1)q$ time units.
- Timer interrupts every quantum to schedule next process
- Performance
  - q large $\Rightarrow$ Same as FCFS
  - q small $\Rightarrow$ q must be large with respect to context switch, otherwise overhead is too high

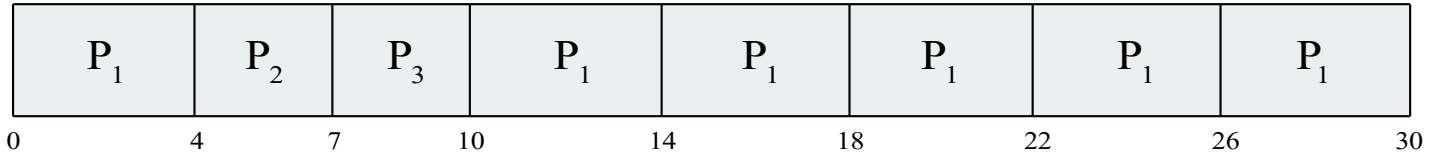
Example of RR with Time Quantum = 4

Process	Burst Time
P_1	24
P_2	3
P_3	3

Example of RR with Time Quantum = 4

Process	Burst Time
P_1	24
P_2	3
P_3	3

- The Gantt chart is:



Process ID	Arrival Time	Burst Time	Completion Time	Turnaround time	Wait time
1	0	24	30	30	6
2	0	3	7	7	4
3	0	3	10	10	7

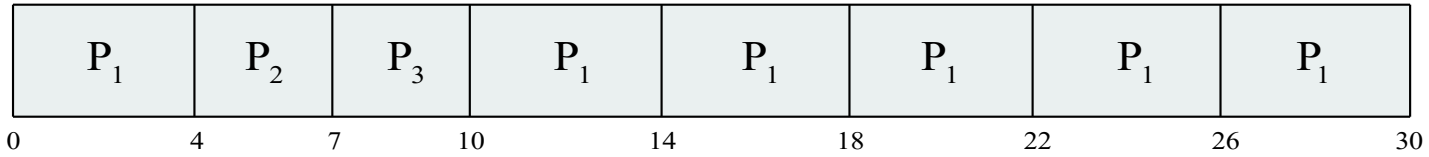
Av. TAT = 15.67 ms

Av. Wait Time = 5.67 ms

Example of RR with Time Quantum = 4

Process	Burst Time
P_1	24
P_2	3
P_3	3

- The Gantt chart is:



- Typically, higher average turnaround than SJF, but better ***response***
- q should be large compared to context switch time
- q usually 10ms to 100ms, context switch < 10 usec

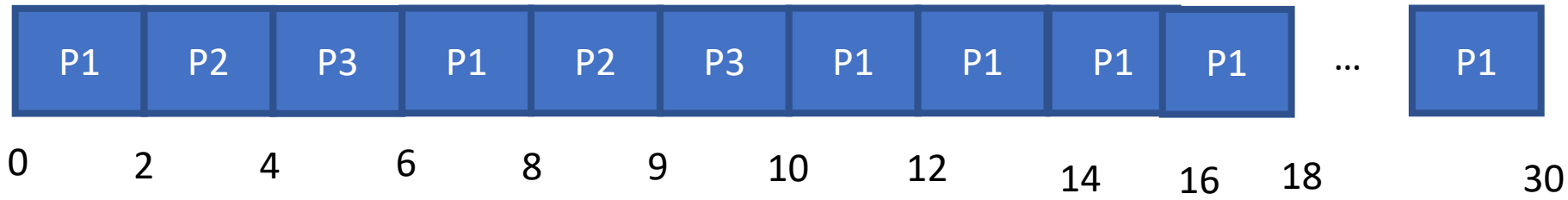
Example of RR with Time Quantum = 2

Process	Burst Time
P_1	24
P_2	3
P_3	3

Example of RR with Time Quantum = 2

Process	Burst Time
P_1	24
P_2	3
P_3	3

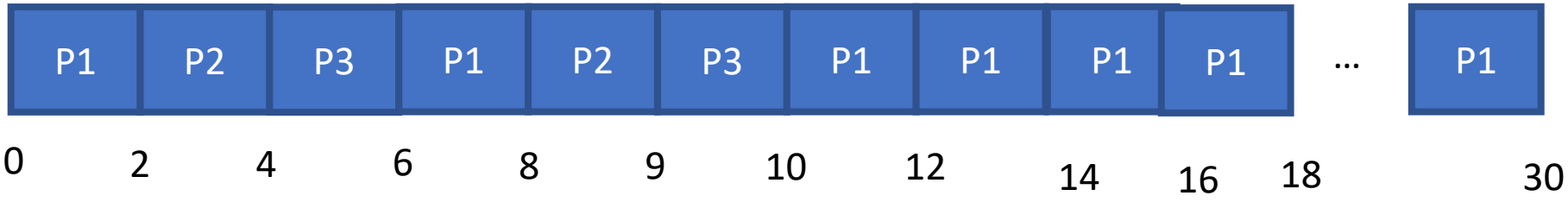
- The Gantt chart is:



Example of RR with Time Quantum = 2

Process	Burst Time
P_1	24
P_2	3
P_3	3

- The Gantt chart is:



Process ID	Arrival Time	Burst Time	Completion Time	Turnaround time	Wait time
1	0	24	30	30	6
2	0	3	9	9	6
3	0	3	10	10	7

Av. WT = 6.33 ms

Av. TAT = 16.33 ms