CSC 112: Computer Operating Systems
Lecture 5

Scheduling
Exercises Solution

Department of Computer Science,

Hofstra University

Acknowledgement: Lecture slides based on UC Berkeley CS 162: Operating Systems and System Programming

Predicting Burst Time

- Use exponential averaging $\tau_n = \alpha t_{n-1} + (1-\alpha)\tau_{n-1}$ to predict the next burst time. Assume initial estimate $\tau_0 = 10$, and the actual burst times of the first four processes t_0, t_1, t_2, t_3 are 4, 8, 6 and 7, respectively. Given $\alpha = 0.5$. Compute the predicted burst times τ_1, τ_2, τ_3, τ_4.

Predicting Burst Time ANS

- Use exponential averaging $\tau_n = \alpha t_{n-1} + (1-\alpha)\tau_{n-1}$ to predict the next burst time. Assume initial estimate $\tau_0 = 10$, and the actual burst times of the first four processes t_0, t_1, t_2, t_3 are 4, 8, 6 and 7, respectively. Given $\alpha = 0.5$. Compute the predicted burst times τ_1, τ_2, τ_3, τ_4. (It is best if you can bring a calculator. But if you do not have one, just writing out the following formulas is OK.)

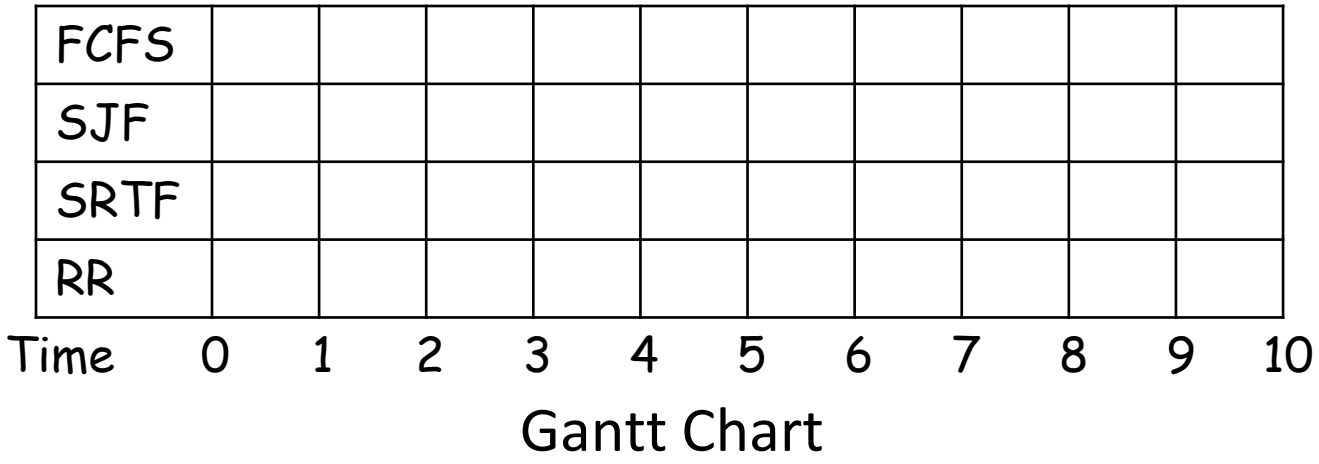
- $\tau_1 = 0.5 \times 4 + 0.5 \times 10 = 7$
- $\tau_2 = 0.5 \times 8 + 0.5 \times 7 = 7.5$
- $\tau_3 = 0.5 \times 6 + 0.5 \times 7.5 = 6.75$
- $\tau_4 = 0.5 \times 7 + 0.5 \times 6.75 = 6.875$

Scheduling

- Here is a table of processes and their arrival and execution times.
- 1) Draw the Gantt chart under 4 policies: First Come First Serve (FCFS), Shortest Job First (SJF), Shortest-Remaining-Time-First (SRTF), Round-Robin (RR) with time quantum = 1. Assume that context switch overhead is 0. For RR, assume that an arriving process is scheduled to run at the beginning of its arrival time, i.e., it is added to the head of the queue upon arrival.
- 2) Compute the finish times and response times for all 5 processes, and the average response time. (If the division is hard, write a fraction like 28/5 instead of 5.6)

Scheduling I

PID	Arriv. time	Exec Time	FCFS Finish Time	FCFS Response Time	SJF Finish Time	SJF Response Time	SRTF Finish Time	SRTF Response Time	RR Finish Time	RR Response Time
1	0	2								
2	1	6								
3	4	2								
				Avg RT		Avg RT		Avg RT		Avg RT

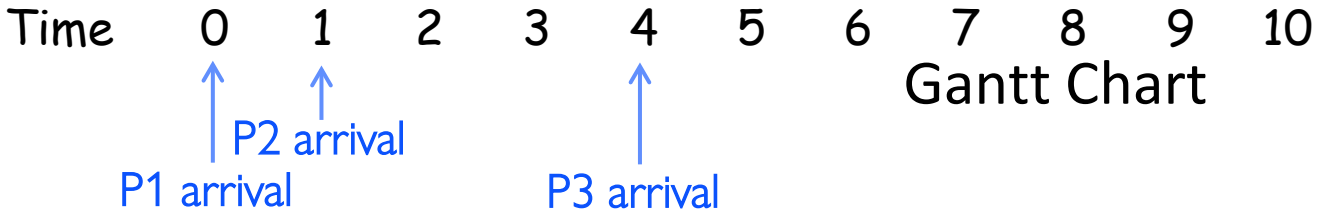


Gantt Chart

Scheduling I ANS

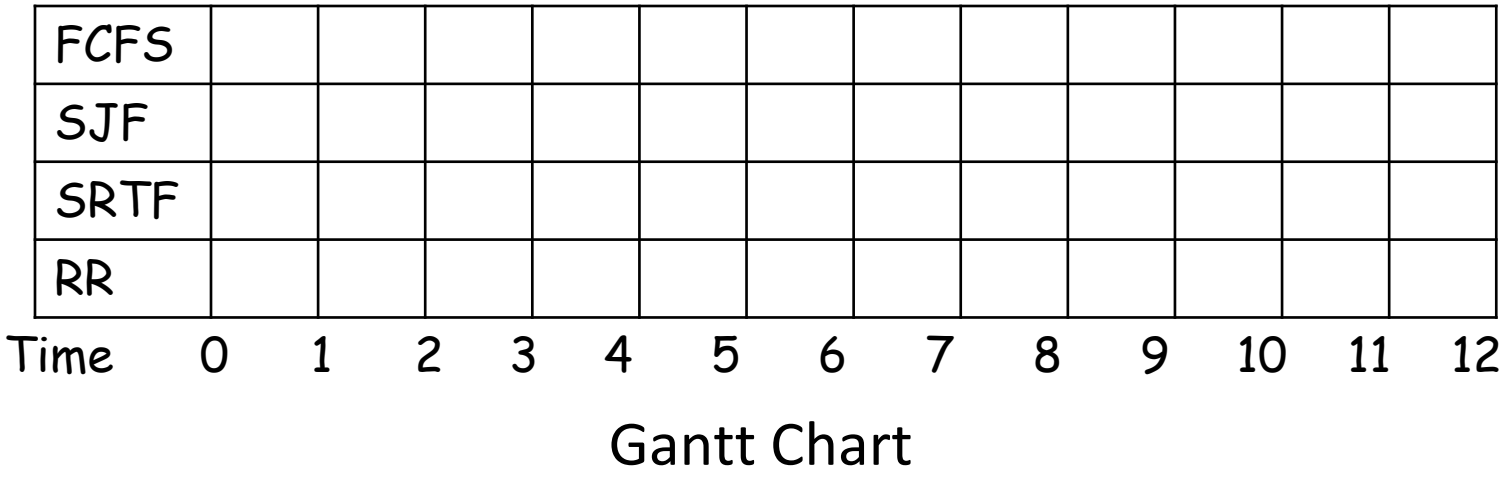
PID	Arriv. time	Exec Time	FCFS Finish Time	FCFS Response Time	SJF Finish Time	SJF Response Time	SRTF Finish Time	SRTF Response Time	RR Finish Time	RR Response Time
1	0	2	2	2	2	2	2	2	3	3
2	1	6	8	7	8	7	10	9	10	9
3	4	2	10	6	10	6	6	2	7	3
				Avg RT 5		Avg RT 5		Avg RT 4.3		Avg RT 5

FCFS	1	1	2	2	2	2	2	2	3	3
SJF	1	1	2	2	2	2	2	2	3	3
SRTF	1	1	2	2	3	3	2	2	2	2
RR	1	2	1	2	3	2	3	2	2	2



Scheduling II

PID	Arriv. time	Exec Time	FCFS Finish Time	FCFS Response Time	SJF Finish Time	SJF Response Time	SRTF Finish Time	SRTF Response Time	RR Finish Time	RR Response Time
1	0	3								
2	1	5								
3	3	2								
4	9	2								
				Avg RT		Avg RT		Avg RT		Avg RT



Gantt Chart

Scheduling II ANS

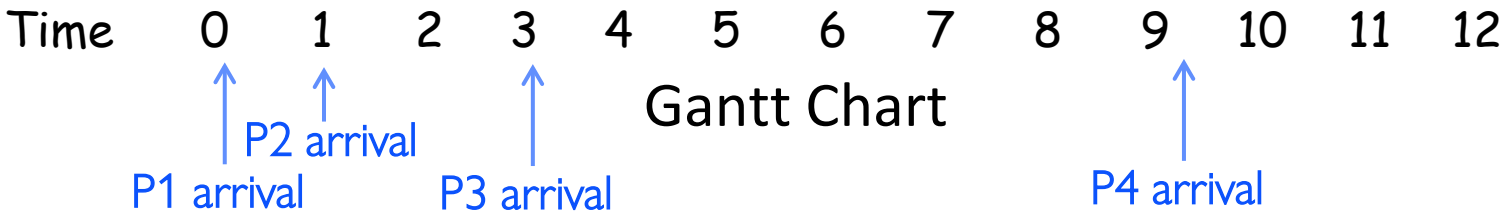
PID	Arriv. time	Exec Time	FCFS Finish Time	FCFS Response Time	SJF Finish Time	SJF Response Time	SRTF Finish Time	SRTF Response Time	RR Finish Time	RR Response Time
1	0	3	3	3	3	3	3	3	6	6
2	1	5	8	7	10	9	10	9	11	10
3	3	2	10	7	5	2	5	2	7	4
4	9	2	12	3	12	3	12	3	12	3
				Avg RT 5		Avg RT 4.25		Avg RT 4.25		Avg RT 5.75

FCFS	1	1	1	2	2	2	2	2	3	3	4	4
SJF	1	1	1	3	3	2	2	2	2	2	4	4
SRTF	1	1	1	3	3	2	2	2	2	2	4	4
RR	1	2	1	3	2	1	3	2	2	4	2	4

Scheduling II ANS

- RR scheduling explanations:
- Time 1: P2 arrives, and it runs immediately based on our assumption that an arriving process is added to the head of the queue upon arrival.
- Time 3: P3 arrives, and it runs immediately.
- Time 4: P2 runs since it is next in ready queue based on the cyclic pattern of "12, 12".
- Time 6: P3 runs based on the cyclic pattern of "321, 321".
- Time 9: P4 arrives, and it runs immediately.

FCFS	1	1	1	2	2	2	2	2	3	3	4	4
SJF	1	1	1	3	3	2	2	2	2	2	4	4
SRTF	1	1	1	3	3	2	2	2	2	2	4	4
RR	1	2	1	3	2	1	3	2	2	4	2	4



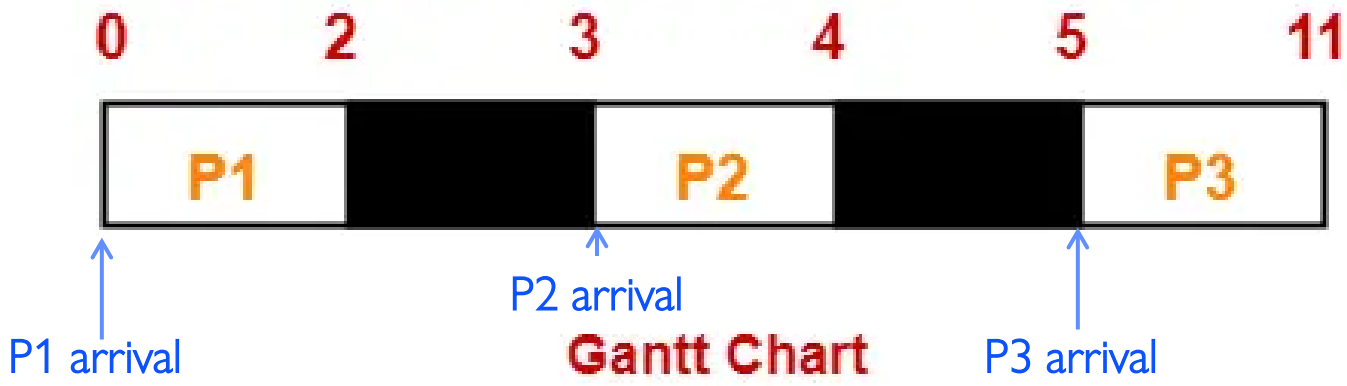
Scheduling III

- Consider the set of 3 processes whose arrival time and CPU burst times are given below. If the CPU scheduling policy is **FCFS**, draw the Gantt chart and calculate the average response time.

PID	Arriv. time	Exec Time	FCFS Finish Time	FCFS Response Time
1	0	2		
2	3	1		
3	5	6		
				Avg RT

Scheduling III ANS

PID	Arriv. time	Exec Time	FCFS Finish Time	FCFS Response Time
1	0	2	2	2
2	3	1	4	1
3	5	6	11	6
				Avg RT 3



Gantt Chart

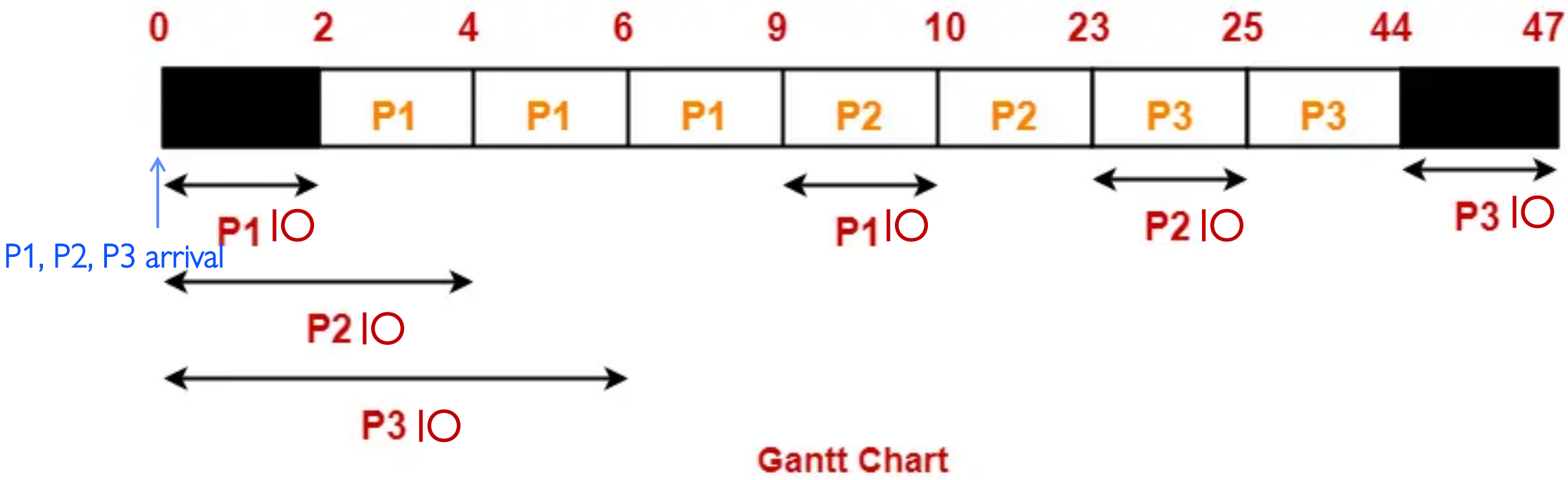
Scheduling with Bursts I

- Consider the set of 3 processes whose arrival time and CPU/IO burst times are given below. If the CPU scheduling policy is **Shortest Remaining Time First (SRTF)**, draw the Gantt chart and calculate the average response time. (Note: consider the overlap of computation and IO busts of different processes)

PID	Arriv. time	IO Burst	CPU Burst	IO Burst
1	0	2	7	1
2	0	4	14	2
3	0	6	21	3
				Avg RT

Scheduling with Bursts I ANS

PID	Arriv. time	IO Burst	CPU Burst	IO Burst	Finish Time	Resp. Time
1	0	2	7	1	10	10
2	0	4	14	2	25	25
3	0	6	21	3	47	47
					Avg RT	27.3



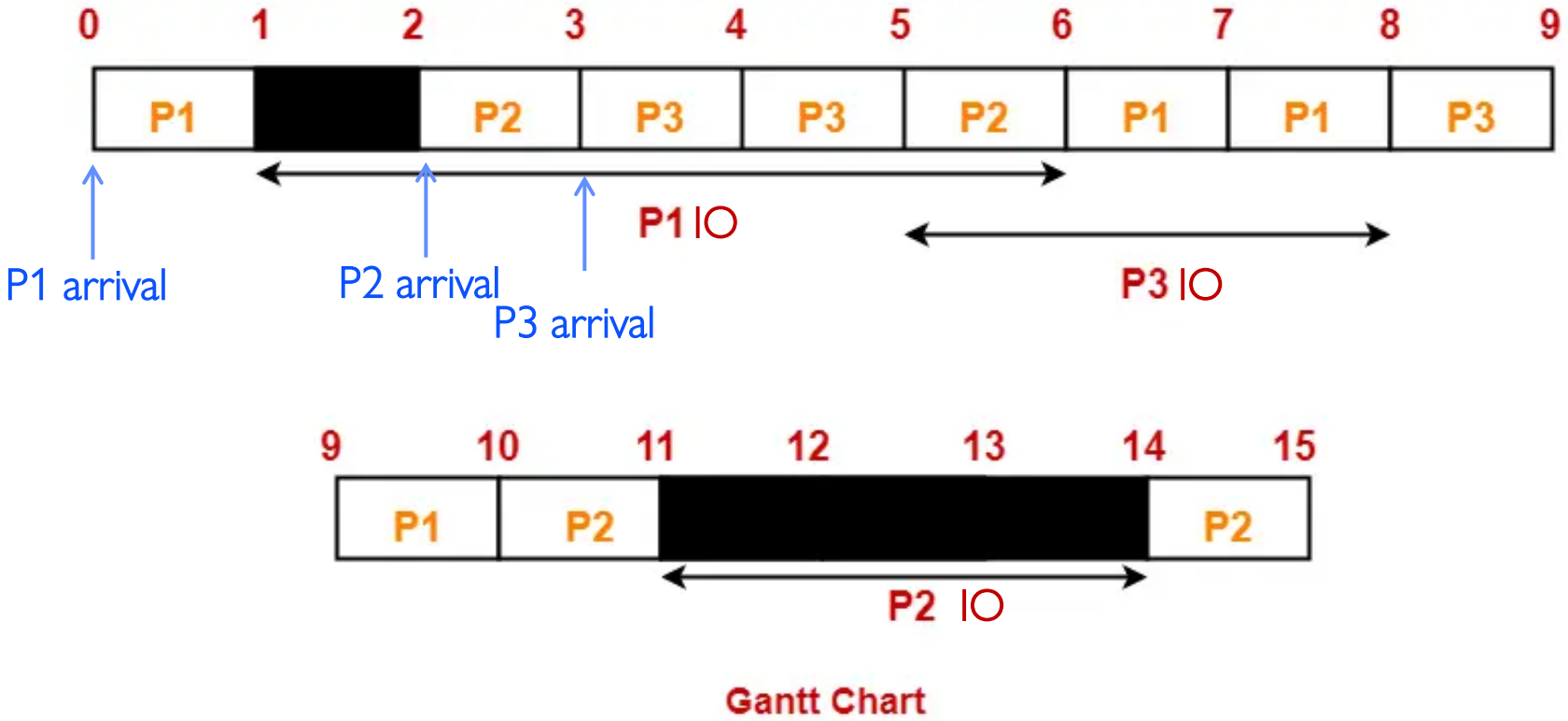
Scheduling with Bursts II

- Consider the set of 3 processes whose arrival time and CPU/IO burst times are given below. If the CPU scheduling policy is **Fixed-Priority Scheduling** (larger number denotes higher priority), draw the Gantt chart and calculate the average response time.

PID	Arriv. time	Priority	CPU Burst	IO Burst	CPU Burst
1	0	2	1	5	3
2	2	1	3	3	1
3	3	3	2	3	1
			Avg RT		Avg RT

Scheduling with Bursts ANS

PID	Arriv. time	Priority	CPU Burst	IO Burst	CPU Burst	Finish Time	Resp. Time
1	0	2	1	5	3	10	10
2	2	1	3	3	1	15	13
3	3	3	2	3	1	9	6
						Avg RT	9.67



For fixed-priority scheduling, draw the Gantt chart starting from highest priority process and go down to the lowest priority (first P3, then P1, then P2). Schedule CPU bursts of lower priority processes in the remaining time slots that are left over by higher priority processes.