CPU SCHEDULING.

Chapter 6

RECAP QUESTIONS (CHAPTER 3)

1. When we discuss about Interprocess communication, we say the communication is under the control of the users not the operating systems, is this true?

2. Symmetry in addressing is observed when only the recipient is named in direct communication approach of IPC, is this correct?

OBJECTIVES.

- To introduce CPU scheduling, which is the basis for multiprogrammed operating systems
- To describe various CPU-scheduling algorithms
- To discuss evaluation criteria for selecting a CPU-scheduling algorithm for a particular system
- To examine the scheduling algorithms of several operating systems

CPU SCHEDULING.

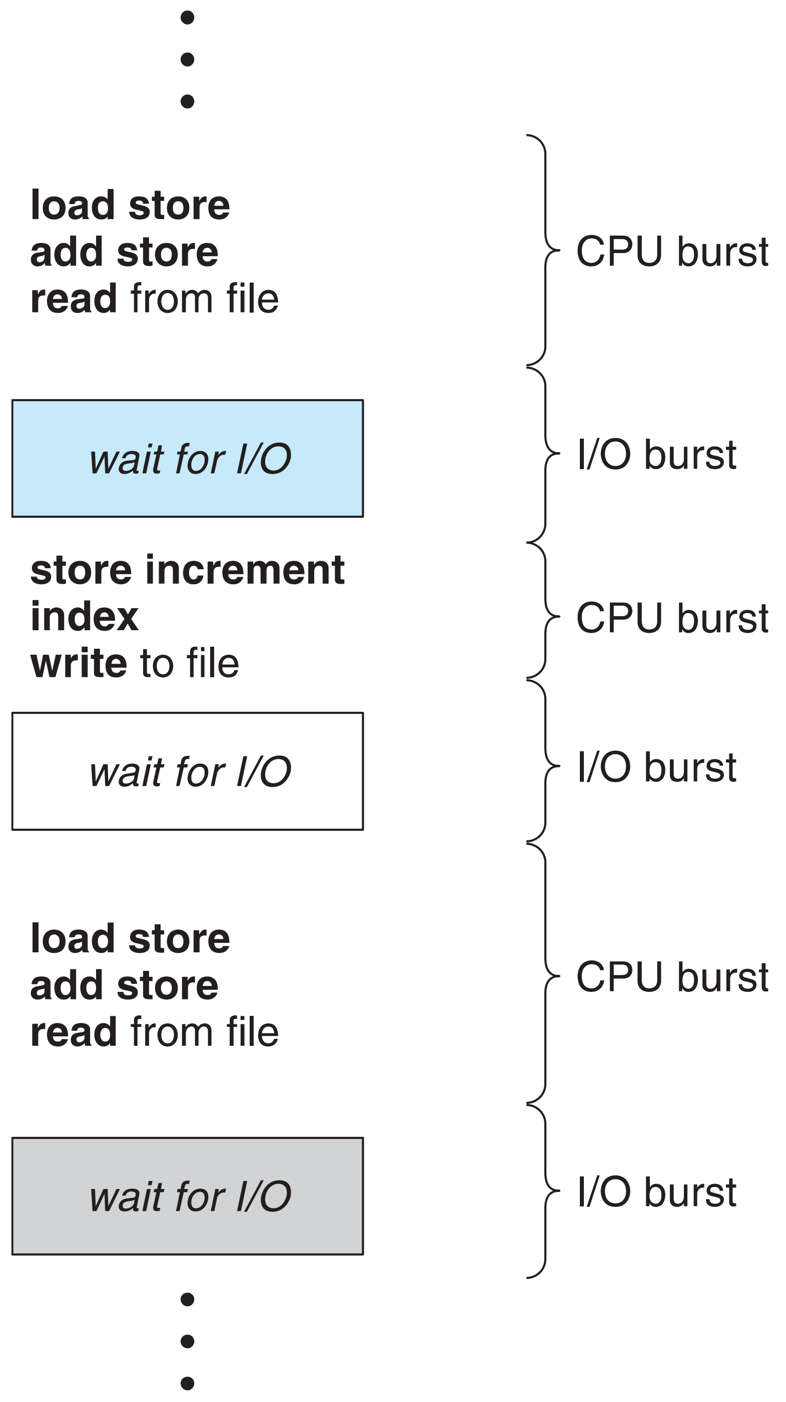
- The operating system can make the computer more productive, how?
  - by switching the CPU among processes.
- **Maximum CPU utilization** is obtained with multiprogramming

MULTIPROGRAMMING.

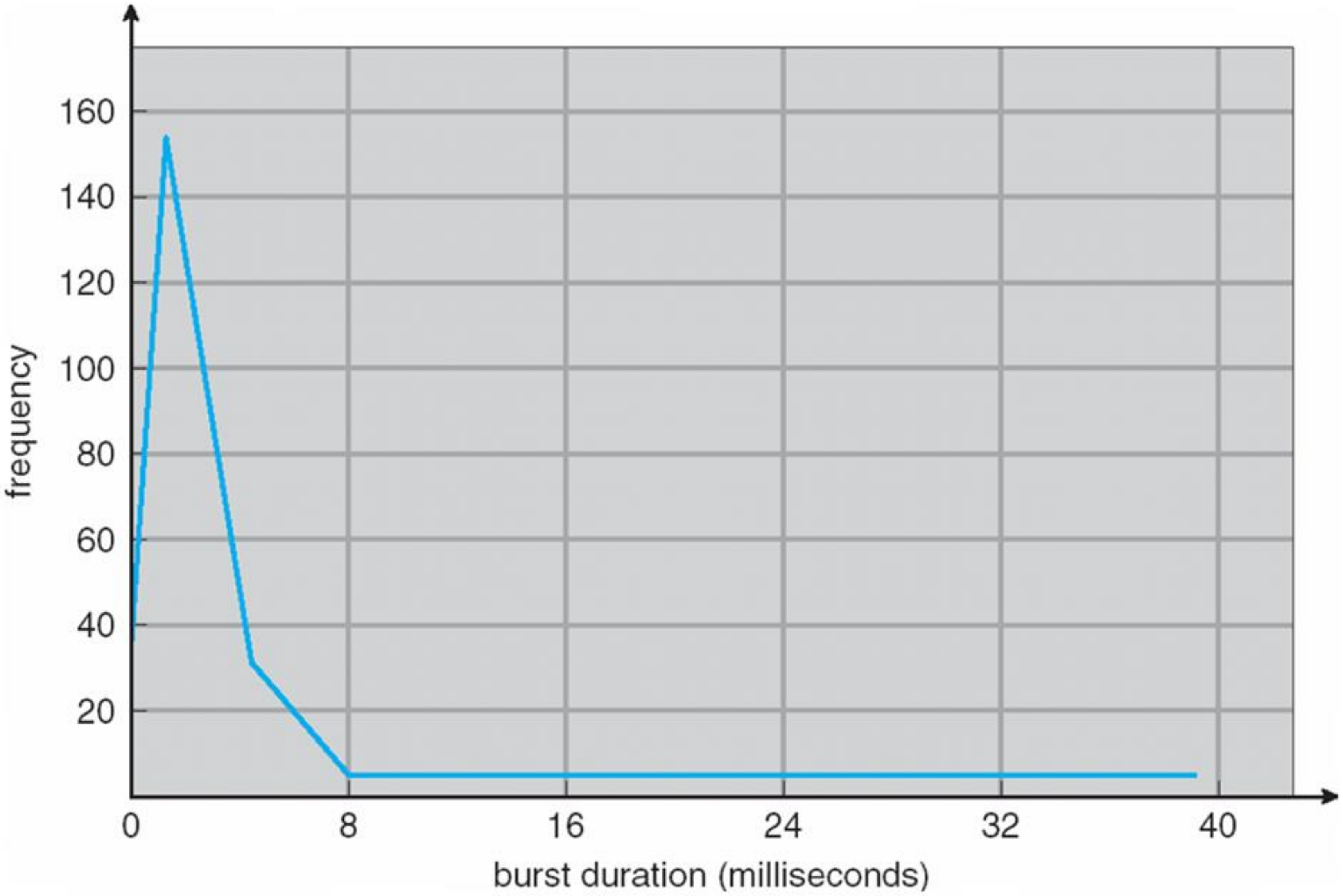
- **Objective:** Have some process running at all times to maximize CPU utilization.
- **Idea:** A process is executed until it must wait (typically for the completion of some I/O request) ----> CPU sits idle.
  - *Waiting time is wasted; no useful work is accomplished.*
  - **Multiprogramming:** This time used productively. Several processes are kept in memory at one time. When one process has to wait, the operating system takes the CPU away from that process and gives the CPU to another process. This pattern continues. Every time one process has to wait, another process can take over the use of CPU.
- **Scheduling of this kind** is a ***fundamental operating-system function***

BASIC CONCEPTS: CPU CYCLE

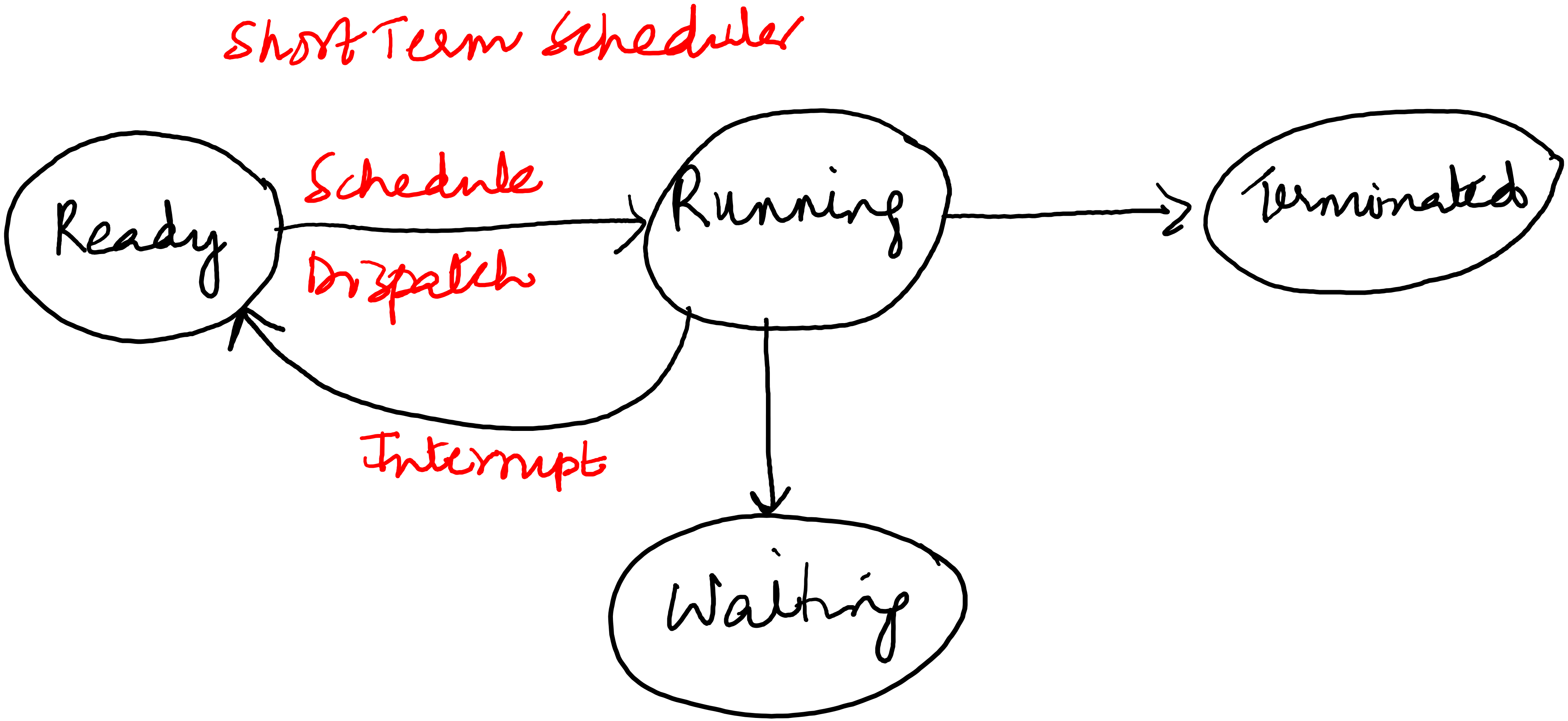
- **CPU–I/O Burst Cycle:** Process execution consists of a cycle of ***CPU execution*** and ***I/O wait***
- CPU burst followed by I/O burst
- CPU burst distribution is of main concern



HISTOGRAM OF CPU-BURST TIMES



REVISITING PROCESS STATE DIAGRAM

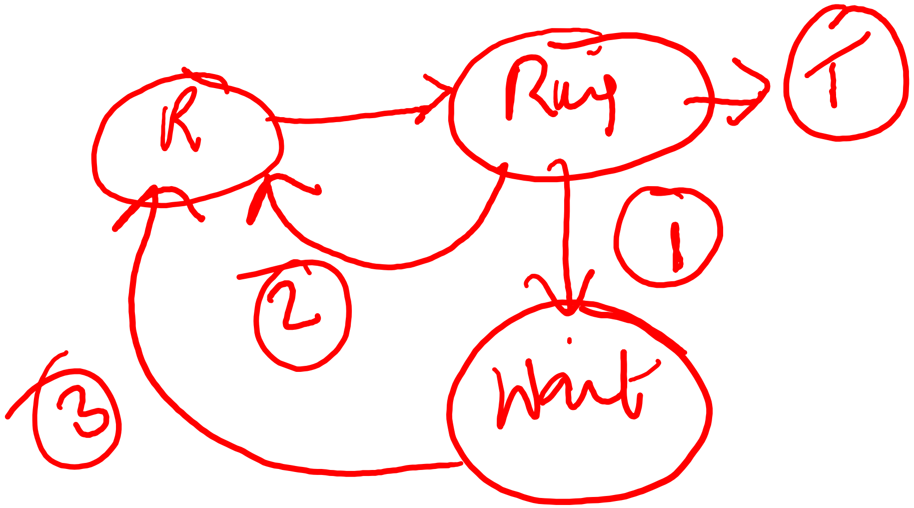


CPU SCHEDULER

CPU/Short-term scheduler selects from among the processes in ready queue, and allocates the CPU to one of them

Queue may be ordered in various ways. **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



CPU SCHEDULER

- 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

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

1. **CPU utilization:** keep the CPU as busy as possible
2. **Throughput:** # of processes that complete their execution per time unit
3. **Turnaround time:** amount of time to execute a particular process
4. **Waiting time:** amount of time a process has been waiting in the ready queue
5. **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

1. Max CPU utilization
2. Max throughput
3. Min turnaround time
4. Min waiting time
5. Min response time

Q: If the degree of multiprogramming is high, how do you think these optimization criteria will be affected?

SCHEDULING ALGORITHMS

1. FIRST- COME, FIRST-SERVED (FCFS)

Suppose that the processes **arrive in the order: P_1 , P_2 , P_3**
The Gantt Chart for the schedule is:

Process	Burst Time
P_1	24
P_2	3
P_3	3

Waiting time for $P_1 = 0$; $P_2 = 24$; $P_3 = 27$
Average waiting time: $(0 + 24 + 27)/3 = 17$

All processes **arrive at the same time.**

1. FIRST- COME, FIRST-SERVED (FCFS)

Turn Around Time (TAT) for any process is:

(1) Wait Time + Burst Time

(2) Exit Time – Arrival Time

Process	Burst Time
P_1	24
P_2	3
P_3	3

Using (1)

$TAT(P_1) = (0+24) = 24$

$TAT(P_2) = (24+3) = 27;$

$TAT(P_3) = (27+3) = 30$

Using (2)

$TAT(P_1) = (24\text{-}0) = 24$

$TAT(P_2) = (27\text{-}0) = 27;$

$TAT(P_3) = (30\text{-}0) = 30$

FIRST- COME, FIRST-SERVED (FCFS) EX.2

Suppose that the processes **arrive in the order:** P_2 , P_3 , P_1

The Gantt Chart for the schedule is:

Process	Burst Time
P_1	24
P_2	3
P_3	3

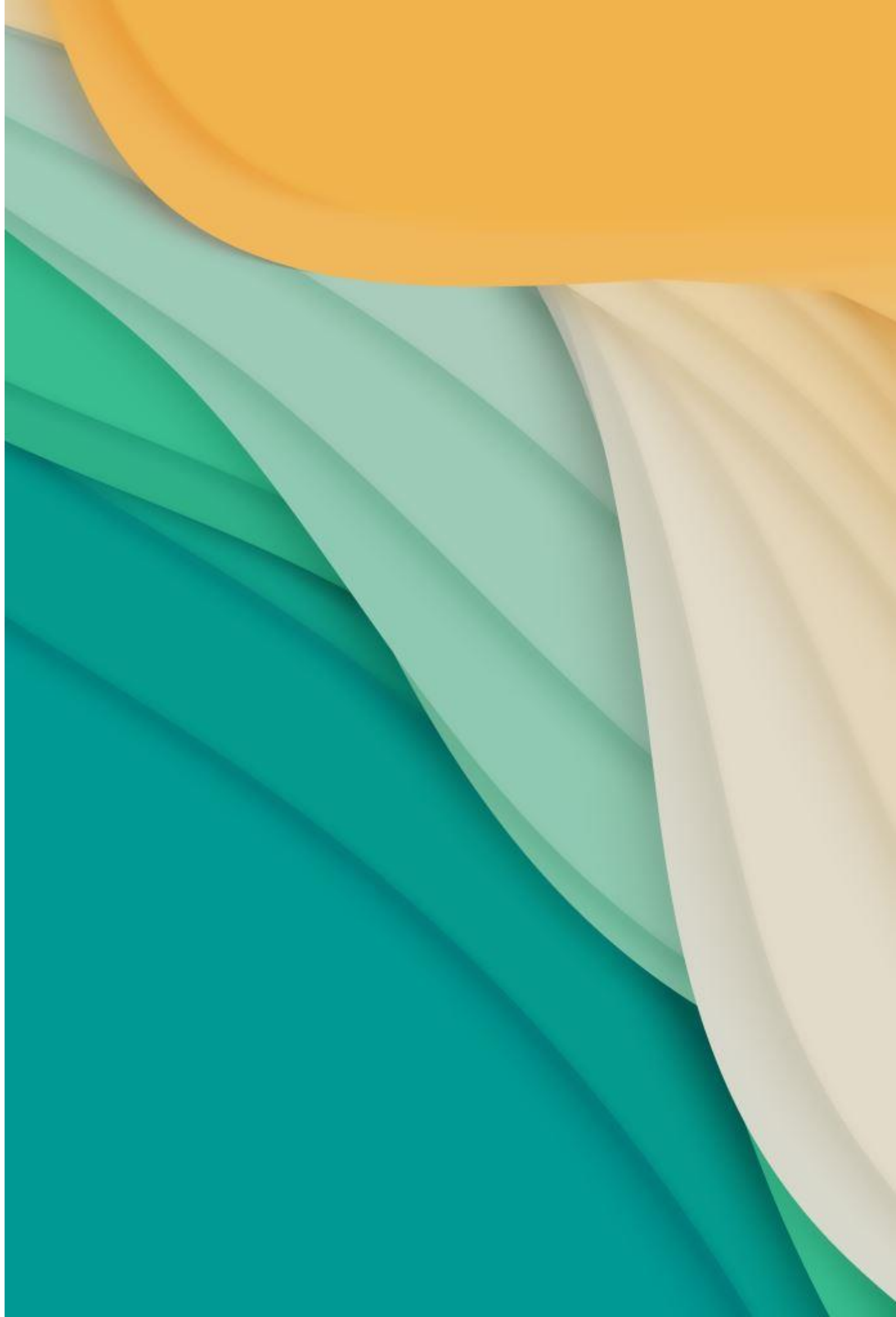
Much better than previous case

Waiting time for $P_1 = 6; P_2 = 0; P_3 = 3$

Average waiting time: (6 + 0 + 3)/3 = 3

Convoy effect: Short process behind long process. Consider one CPU-bound and many I/O-bound processes

BACK FROM BREAK.



FIRST- COME, FIRST-SERVED (FCFS) Homework

Process	Burst Time	Arrival Time
P_1	7	0
P_2	4	0
P_3	1	0
P_4	4	0

The Gantt Chart for the schedule is:

Waiting time

$P_1 =$

$P_2 =$

$P_3 =$

$P_4 =$

Average waiting time:

Turn Around Time (1)

$P_1 =$

$P_2 =$

$P_3 =$

$P_4 =$

Turn Around Time (2)

$P_1 =$

$P_2 =$

$P_3 =$

$P_4 =$

FIRST- COME, FIRST-SERVED (FCFS) Homework

Process	Burst Time	Arrival Time
P_1	7	4
P_2	4	2
P_3	1	1
P_4	4	3

The Gantt Chart for the schedule is:

Waiting time

$P_1 =$

$P_2 =$

$P_3 =$

$P_4 =$

Average waiting time:

Turn Around Time (1)

$P_1 =$

$P_2 =$

$P_3 =$

$P_4 =$

Turn Around Time (2)

$P_1 =$

$P_2 =$

$P_3 =$

$P_4 =$

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
- ***Preemptive version*** called **shortest-remaining-time-first**

FCFS (SJF) | NON PREEMPTIVE

Process	Burst Time	Arrival Time
P_1	6	0
P_2	8	0
P_3	7	0
P_4	3	0

The Gantt Chart for the schedule is:

Waiting time

Turn Around Time (1)

Turn Around Time (2)

SHORTEST-JOB-FIRST (SJF) | NON PREEMPTIVE

Process	Burst Time	Arrival Time
P_1	6	0
P_2	8	0
P_3	7	0
P_4	3	0

Turn Around Time (TAT) for any process is:

(1) Wait Time + Burst Time

(2) Exit Time – Arrival Time

The Gantt Chart for the schedule is:

P_4	P_1	P_3	P_2
0 – 3	3 – 9	9 – 16	16 – 24

Waiting time

$P_1 = 3$

$P_2 = 16$

$P_3 = 9$

$P_4 = 0$

Average waiting time: (3+16+9)/4= 7

Turn Around Time (1)

$P_1 = (3 + 6)=9$

$P_2 = (16+8)=24$

$P_3 = (9+7)=16$

$P_4 = (0+3)=3$

Turn Around Time (2)

$P_1 = (9-0)=6$

$P_2 = (24-0)=24$

$P_3 = (16-0)=24$

$P_4 = (3-0)=0$

SHORTEST-JOB-FIRST (SJF) | PREEMPTIVE

Process	Burst Time	Arrival Time
P_1	8 -1 = 7	0
P_2	4 2 ✓	1
P_3	9	→ 2
P_4	5 ✓	3

WT

Turn Around Time (TAT) for any process is:

(1) Wait Time + Burst Time | EXIT − AT

(2) Final Allocation Time– Arrival Time – Previous Execution

TAT (P1) = 17 (9+8)

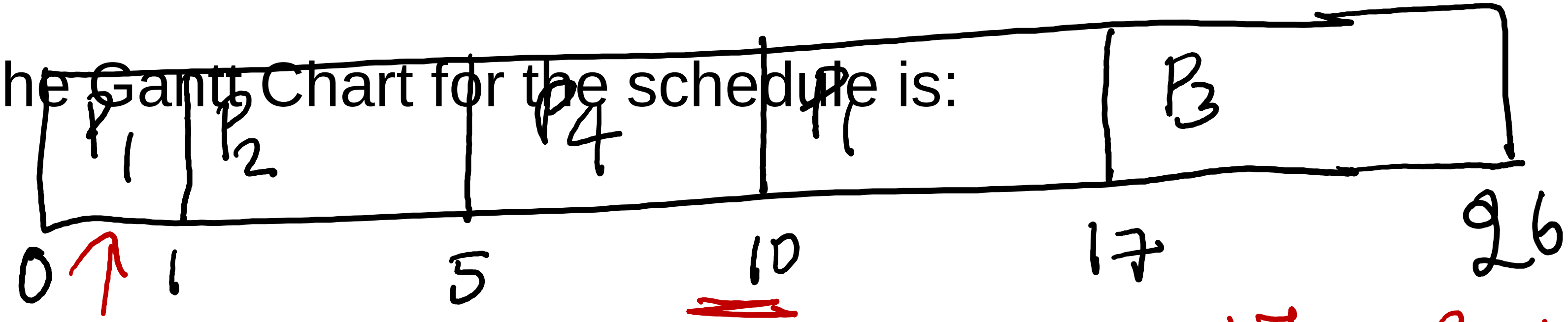
(P2) = 4 (0+4)

(P3) = 24 (15+9)

(P4) = 7 (2+5)

A(TAT) = 13

The Gantt Chart for the schedule is:



WT (P1) = 10 − 0 − 1 = 9

(P2) = 1 − 1 − 0 = 0

P3 = 17 − 2 − 0 = 15

P4 = 5 − 3 − 0 = 2

AWT = 6.5

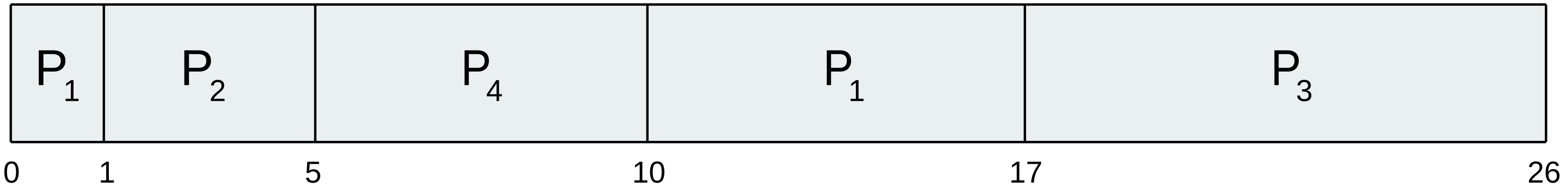
SHORTEST-JOB-FIRST (SJF)

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

Waiting time with Arrival Time for process is:
Final Allocation Time – Arrival Time – Previous Execution

Turn Around Time (TAT) for any process is:
(1) Wait Time + Burst Time
(2) Exit Time – Arrival Time

The Gantt Chart for the schedule is:



Waiting time

$P_1 = 10 – 1 - 0 = 9$
$P_2 = 1 – 1 - 0 = 0$
$P_3 = 17 – 2 - 0 = 15$
$P_4 = 5 – 3 - 0 = 2$

Average waiting time:

Turn Around Time (1)

$P_1 = 9 + 8 = 17$
$P_2 = 0 + 4 = 4$
$P_3 = 15 + 2 = 17$
$P_4 = 2 + 5 = 7$

Turn Around Time (2)

$P_1 = 17 – 0 = 17$
$P_2 = 5 – 1 = 4$
$P_3 = 26 – 9 = 17$
$P_4 = 10 – 3 = 7$

WITH PRE-EMPTION (SJF)

WT

TAT

P1	0 (AT)	7 (BT)
P2	2	4
P3	4	1
P4	5	4

$WT(P_1) = 11 - 0 - 2 = 9$

$(P_2) = 5 - 2 - 2 = 1$

$(P_3) = 4 - 4 - 0 = 0$

$(P_4) = 7 - 5 - 0 = 2$

TAT

$P_1 = 16$

$P_2 = 5$

$P_3 = 1$

$P_4 = 6$

P_1	P_2	P_3	P_2	P_4	P_1

0 2 $\underbrace{\quad}_{2}$ 4 5 7 11 16

$AWT(P) = 3$

Without (NP)

P_1	P_3	P_2	P_4

0 7 8 12 16

Without Preemption (Non-preemptive)

	AT	BT
P_1	0	7
P_2	2	4
P_3	4	1
P_4	5	4

P_1	P_3	P_2	P_4

0   7   8   12   16

↑ (at 7) all process arrive

AWT = ? 4

ATAT = ? 8

Waiting Time

$P_1 = 0$

$P_2 = 8 - 2 = 6$

$P_3 = 7 - 4 = 3$

$P_4 = 12 - 5 = 7$

Turnaround Time

(1) WT + BT

$P_1 = 0 + 7 = 7$

$P_2 = 6 + 4 = 10$

$P_3 = 3 + 1 = 4$

$P_4 = 7 + 4 = 11$

(2) Exit Time − AT

$P_1 = 7 - 0 = 7$

$P_2 = 12 - 2 = 10$

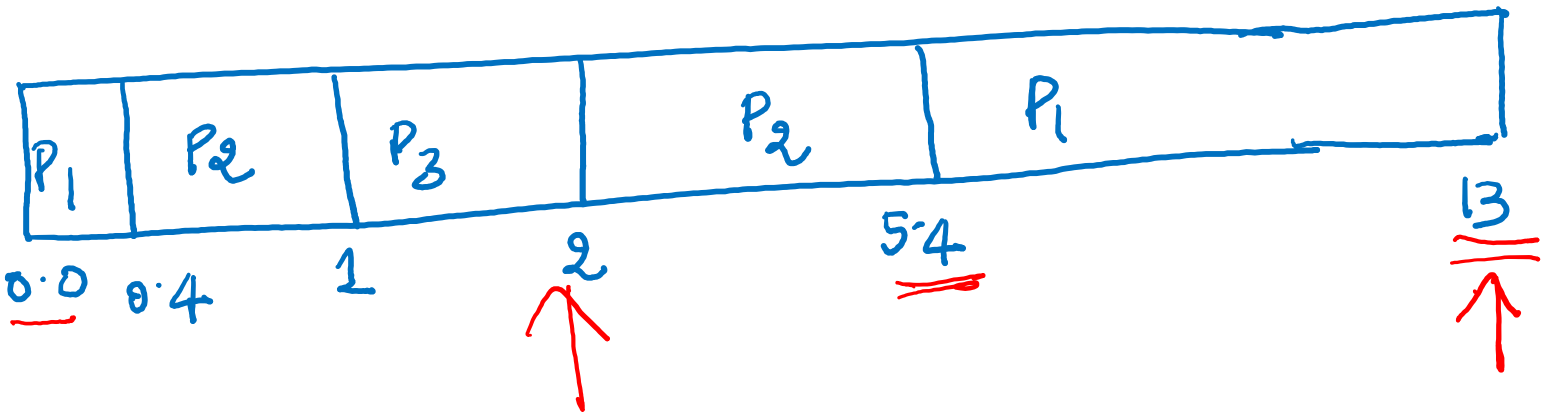
$P_3 = 8 - 4 = 4$

$P_4 = 16 - 5 = 11$

SJF (Preemptive & Non-preemptive)

HW

PID	AT	BT
P_1	0.0	8 ~~7~~ 6
P_2	0.4	4 (3.4)
P_3	1.0	1



WT, AWT, TAT, ATAT

$P1 = 5.4 - 0 - 0.4 = 5$

$P2 = 2 - 0.4 - 0.6 = 1$

$P3 = 1 - 1 = 0$

$AWT = \underline{2}$

TAT (i)

$P1 = 5 + 8 = \underline{13}$

$P2 = 1 + 4 = 5$

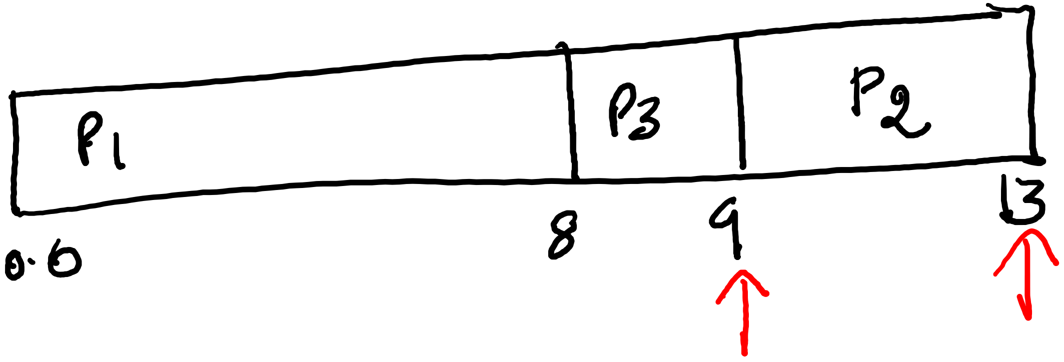
$P3 = 0 + \cancel{8}\,1 = \cancel{8}\,1$

$A(TAT) = 6.33$

HW

SJF (Non-preemptive)

PID	AT	BT
P_1	0.0	8
P_2	0.4	4
P_3	1.0	1



WT
$P1 = 0$
$P2 = 9 - 0.4 = 8.6$
$P3 = 8 - 1 = 7.0$

AWT = ? 5.2

TAT
$P1 = 0 + 8 = 8$
$P2 = 8.6 + 4 = 12.6$
$P3 = 7 + 1 = 8$

ATAT ? $= 9.53$

SHORTEST JOB FIRST – LAST ONE

PID	BT	AT
P_1	40	0
P_2	30	25
P_3	30	30
P_4	35	60
P_5	5	100
P_6	10	105

PRIOITY SCHEDULING

- A ***priority* number** ***(integer)*** is associated with each process
- The CPU is allocated to the process with the highest priority
  - Preemptive
  - Nonpreemptive
- SJF is priority scheduling where priority is the inverse of predicted next CPU burst time

PRIORITY SCHEDULING

PID	PRIORITY	BT	AT
P_1	40	20	0
P_2	30	25	25
P_3	30	25	30
P_4	35	15	60
~~P_5~~	5	(5) ~~10~~	100
P_6	10	10	105

$115 - 100 = 5 = 10$

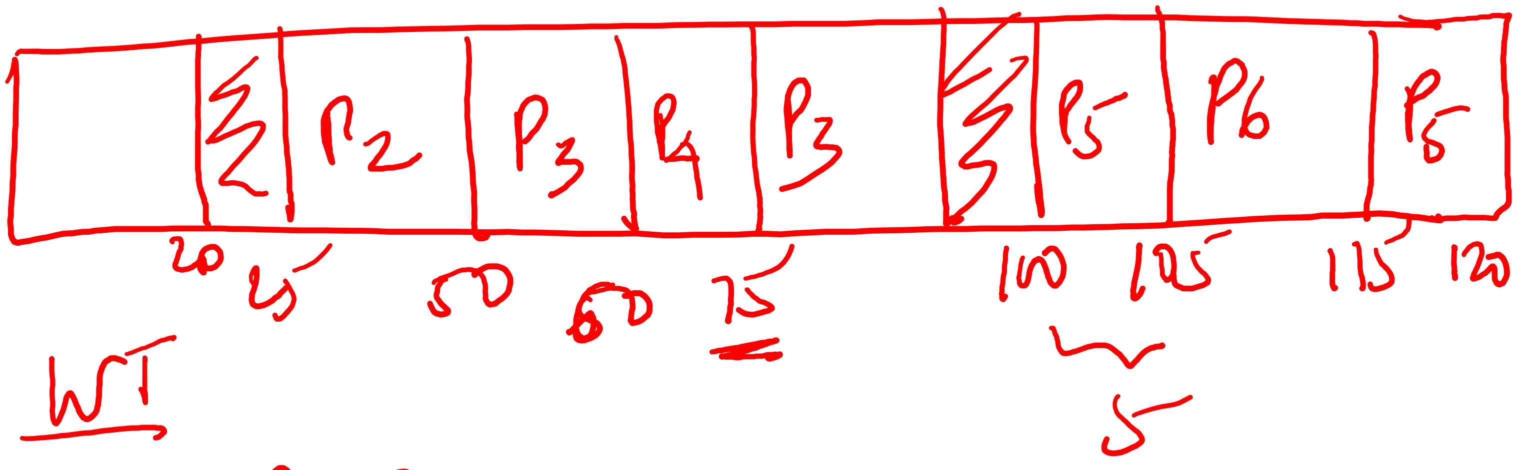
$P_5 = \cancel{100} -$

$P_6 = 105 - 105 - 0 = 0$

$AWT = 7.5$

$TAT\ (P_1) = 20$

$P_2 = 25$



WT

$P_1 = 0 - 0 = 0$

$P_2 = 25 - 25 = 0$

$P_3 = 75 - 30 - 10 = 35$

$P_4 = 60 - 60 - 0 = 0$

$A(TAT) = 25$

$P_3 = 60$

$P_4 = 15$

$P_5 = 20$

$P_6 = 10$

PRIORITY SCHEDULING (NON - PREEMPTIVE)

PID	BT	Priority
~~P1~~	10	3
~~P2~~	1	1
~~P3~~	2	4
P4	1	5
~~P5~~	5	2

AWT = 8.2
ATAT = 12

P2	P5	P1	P3	P4

0 1 6 16 18 19

WT (P1) = 6 P4 = 18
P2 = 0 P5 = 1
P3 = 16

TAT P1 = 16 P4 = 19
P2 = 1 P5 = 6.
P3 = 18

STARVATION OR INDEFINITE BLOCKING

Major problem: *A process that is ready to run but waiting for the CPU can be considered blocked*.

- A priority scheduling algorithm can leave some low priority processes waiting indefinitely.
- In a heavily loaded computer system, a steady stream of higher-priority processes can prevent a low-priority process from ever getting the CPU.

What's the solution?

Aging: Involves gradually increasing the priority of processes that wait in the system for a long time *(127 low – 0 high)*

ROUND ROBIN SCHEDULING.

- Designed especially **for timesharing systems** and is *similar to FCFS* scheduling, but **preemption is added** to enable the system to switch between processes.
- A small unit of time, called a **time quantum or time slice**, is defined *(Generally from 10 to 100 milliseconds in length)*
- The ready queue is treated as a **circular queue**.
  - The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum.

ROUND ROBIN SCHEDULING.

- To implement RR scheduling, treat the ready queue as a FIFO queue of processes.
- New processes are added to the tail of the ready queue.
- The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after 1 time quantum, and dispatches the process.
- Two things can happen:

1. Voluntary release of CPU on completion of time quantum
2. Timer goes off, causes an interrupt to OS, context switch happens

(Pre-emptive)

AT = 0   **RR**

q = 4ms

PID	BT
P_1	24 (20)
~~P_2~~	3
~~P_3~~	3

P_1	P_2	P_3	P_1		→
0–4	4–7	7–10	10–14	14–16	16–30

WT ⇒ $P_1 = 10 - 0 - 4 = 6$

$P_2 = 4 - 0 - 0 = 4$

$P_3 = 7 - 0 - 0 = 7$

AWT = 5.6

TAT(P_1) = 30

$P_2 = 7$

$P_3 = 10$

RECAP QUESTION.

What is the ***problem faced*** when implementing Priority Scheduling and what is the ***solution***?

Problem --> ***Starvation:*** Low priority processes may never execute

Solution --> ***Aging:*** As time progresses increase the priority of the process

PID	BT	Priority
P_1	2	2
P_2	1	1
P_3	8	4
P_4	4	2
P_5	5	3

$q = 2$

NP

Round Robin

FCFS
SJS
Priority
RR

PID	BT		Priority
P_1	2		2
P_2	1		1
P_3	8	~~6~~ ~~4~~ 2	4
P_4	4	~~2~~	2
P_5	5	~~3~~ ~~1~~	3

$q = 2$

$AT = 0$

P_1	P_2	P_3	P_4	P_5	P_3	P_4	P_3	P_3	P_5	P_3

0 2 3 5 7 9 11 13 15 17 18 20

$P_1 = 0 - 0 - 0 = 0$

$P_2 = 2 - 0 - 0 = 2$

$P_3 = 18 - 0 - 6 = 12$

$P_4 = 11 - 0 - 2 = 9$

$P_5 = 17 - 0 - 4 = 13$

$A\,WT = 7.2$

TAT

(WT + BT)

$P_1 = 0 + 2 = 2$

$P_2 = 2 + 1 = 3$

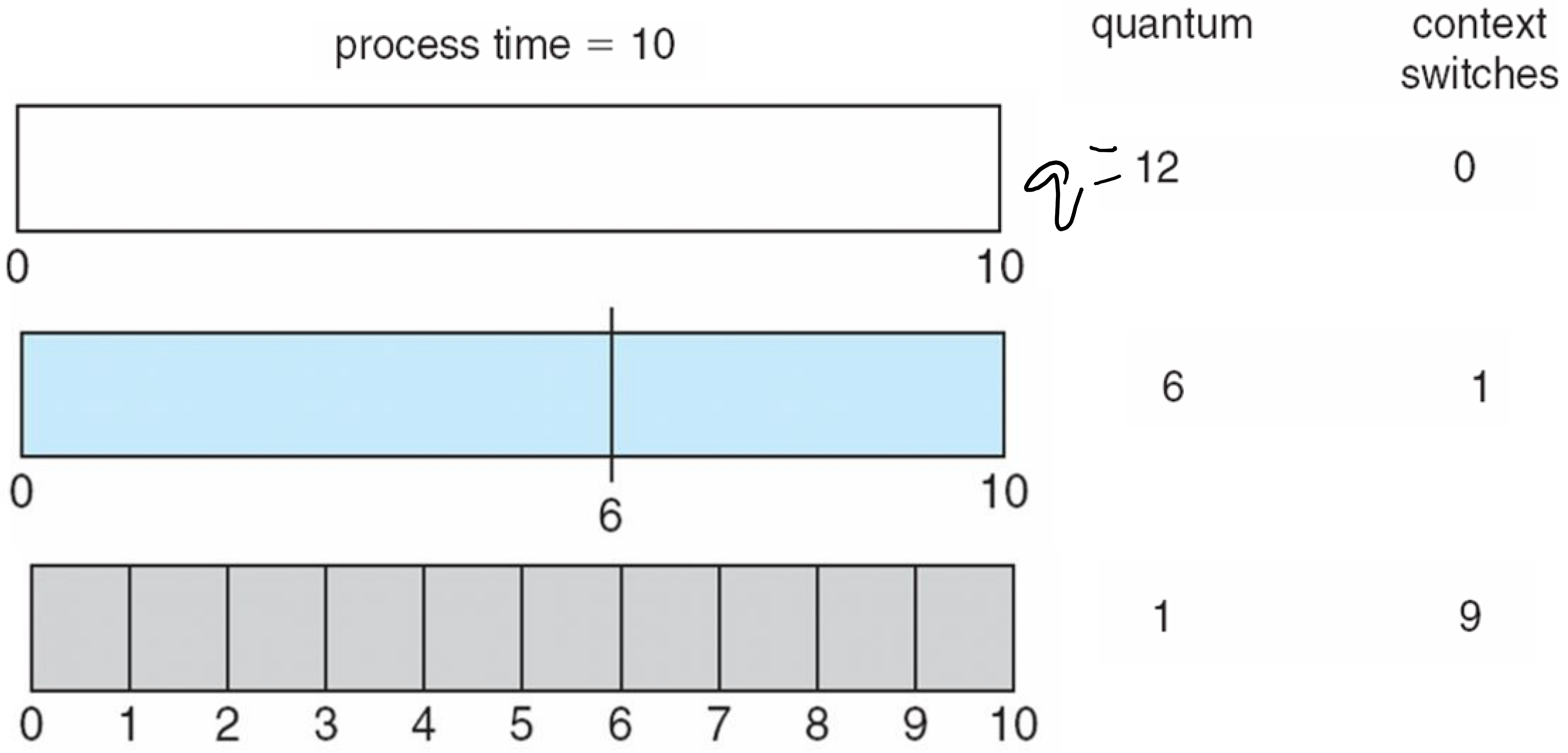
$P_3 = 12 + 8 = 20$

$P_4 = 9 + 4 = 13$

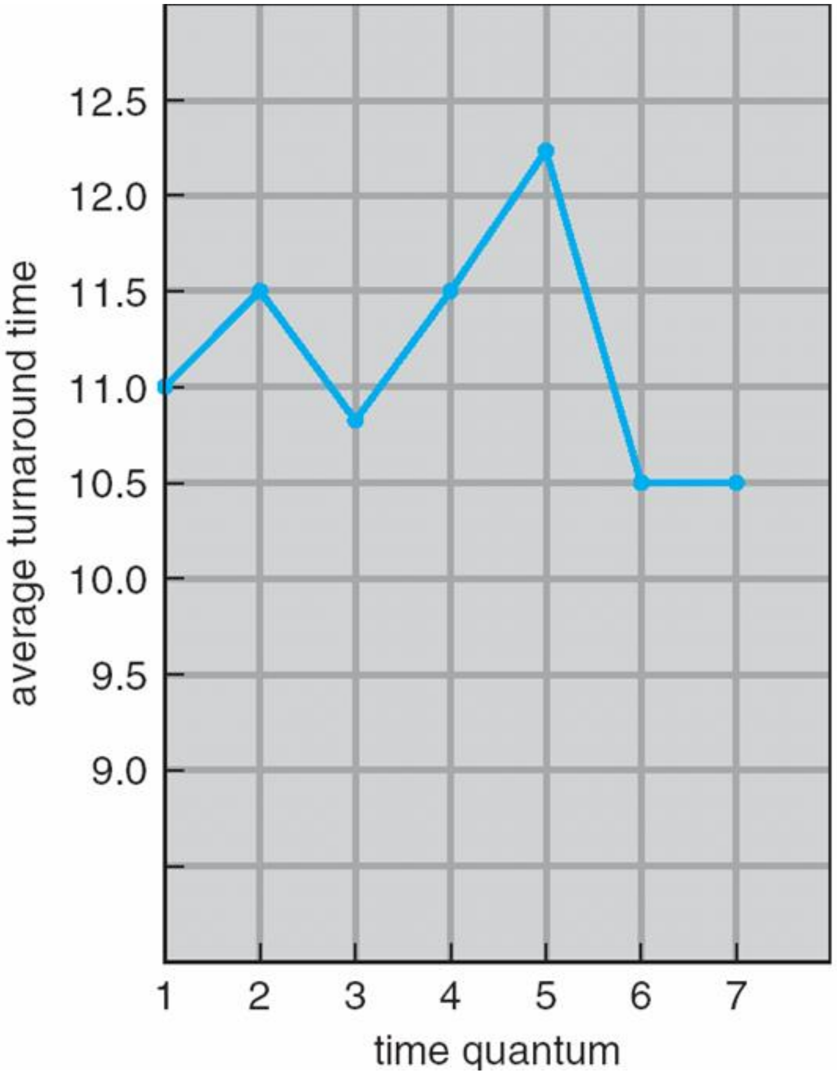
$P_5 = 13 + 5 = 18$

$ATAT = 11.2$

ROUND ROBIN SCHEDULING: *TIME QUANTUM AND CONTEXT SWITCH TIME*



TURNAROUND TIME VARIES WITH THE TIME QUANTUM



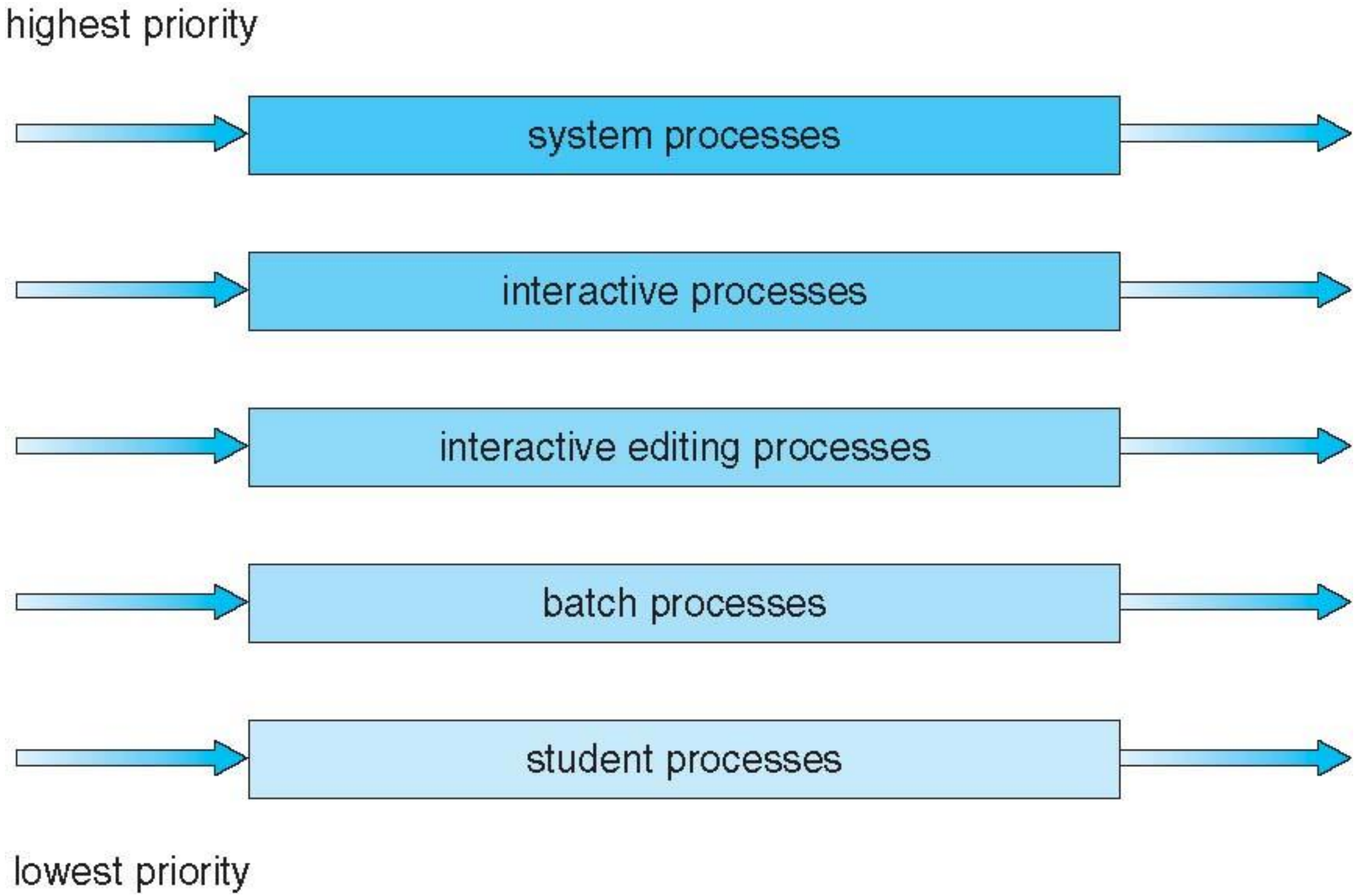
process	time
P_1	6
P_2	3
P_3	1
P_4	7

80% of CPU bursts should be shorter than q

MULTI LEVEL QUEUE

- Ready queue is partitioned into separate queues, eg:
  - **foreground** (interactive)
  - **background** (batch)
- Process permanently in a given queue
- Each queue has its own scheduling algorithm:
  - foreground – RR
  - background – FCFS
- Scheduling must be done between the queues:
  - Fixed priority scheduling; (i.e., serve all from foreground then from background). Possibility of starvation.
  - Time slice – each queue gets a certain amount of CPU time which it can schedule amongst its processes; i.e., 80% to foreground in RR
  - 20% to background in FCFS

MULTILEVEL QUEUE SCHEDULING



MULTILEVEL QUEUE SCHEDULING

Consider below data of four processes under multilevel queue scheduling, Q No. denotes the queue of the process

PID	Arrival Time	Burst Time	Q No.
P1	0	4	1
P2	0	3	1
P3	0	8	2
P4	10	5	1

Priority of queue 1 is greater than queue 2. Queue 1 uses RR (TQ=2) and queue 2 uses FCFS.

Draw the Gantt chart for above data and Calculate AWT, TAT

P1	P2	P1	P2	P3	P4	P3
0	2	4	6	7	10	15 20

PID	BT	AT	Q
P_1	~~4~~ ~~2~~ 0	0	1
P_2	~~3~~ ~~1~~	0	1
P_3	~~8~~ 5	0	2
P_4	5	10	1

Q1 → HP

TQ = 2

(RR)

P_1

P_2

P_4

Q2 → LP

(FCFS)

P_3

AWT = 4.5

TAT

$P_1 = 6$

$P_2 = 7$

$P_3 = 12 + 8$

$= 20$

$P_4 = 0 + 5 = 5$

P_1	P_2	P_1	P_2	P_3	P_4	P_4	P_4	P_3
0–2	2–4	4–6	6–7	7–10	10–12	12–14	14–15	15–20

0 2 4 6 7 10 12 14 15 20

WT $P_1 : 4 - 0 - 2 = 2$

$P_2 : 6 - 0 - 2 = 4$

$P_3 = 15 - 0 - 3 = 12$

$P_4 = 10 - 10 - 0 = 0$

ATAT = 9.5

MULTILEVEL FEEDBACK QUEUE SA

- A process can move between the various queues; ***aging can be implemented this way***
- Multilevel-feedback-queue scheduler defined by the following parameters:
  - number of queues
  - scheduling algorithms for each queue
  - method used to determine when to upgrade a process
  - method used to determine when to demote a process
  - method used to determine which queue a process will enter when that process needs service

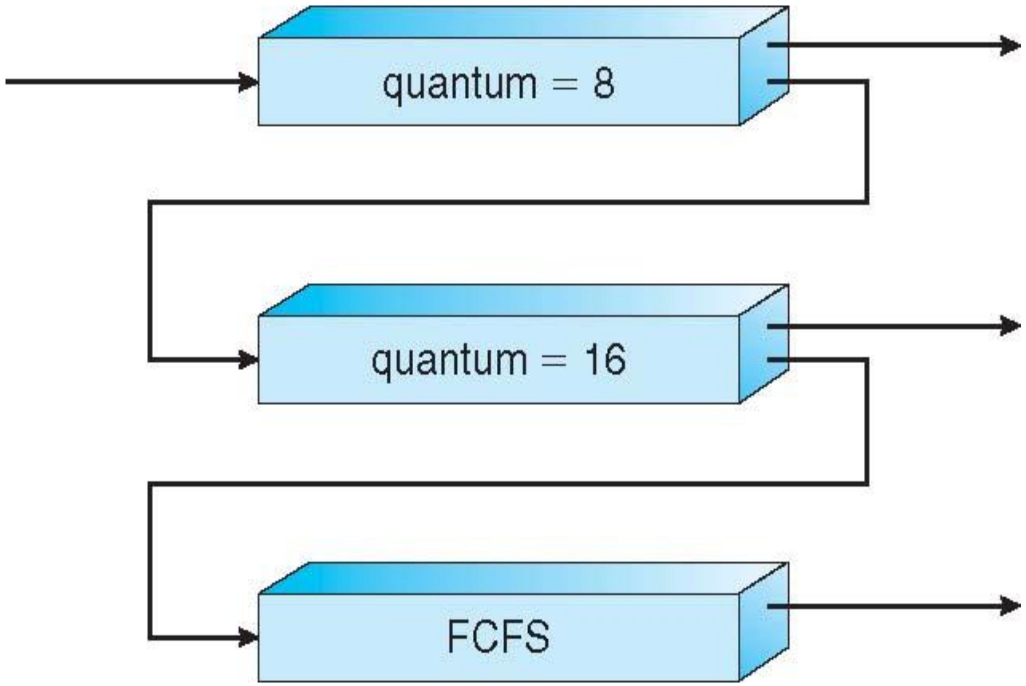
EXAMPLE OF MULTILEVEL FEEDBACK QUEUE

Three queues:

- Q_0 – RR with time quantum 8 milliseconds
- Q_1 – RR time quantum 16 milliseconds
- Q_2 – FCFS

Scheduling

- A new job enters queue Q_0 which is served FCFS
  - When it gains CPU, job receives 8 milliseconds
  - If it does not finish in 8 milliseconds, job is moved to queue Q_1
- At Q_1 job is again served FCFS and receives 16 additional milliseconds
  - If it still does not complete, it is preempted and moved to queue Q_2



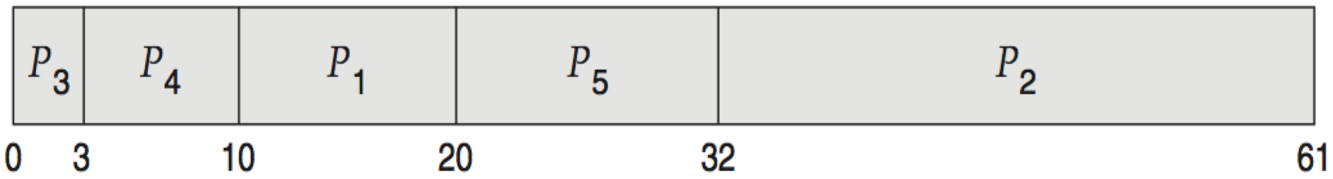
ALGORITHM EVALUATION

- Question: How to select CPU-scheduling algorithm for an OS?
- Answer: Determine criteria, then evaluate algorithms
- **Deterministic modeling**
  - Type of **analytic evaluation**
  - Takes a particular predetermined workload and defines the performance of each algorithm for that workload
- Consider 5 processes arriving at time 0:

Process	Burst Time
P_1	10
P_2	29
P_3	3
P_4	7
P_5	12

DETERMINISTIC EVALUATION

- For each algorithm, calculate minimum average waiting time
- Simple and fast, but requires exact numbers for input, applies only to those inputs
  - FCFS:

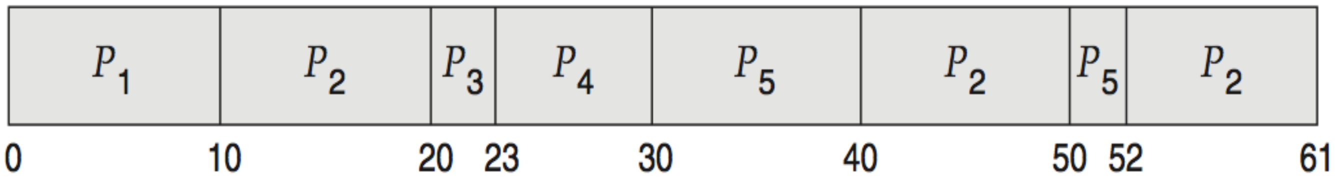


28 ms

  - Non-preemptive:

13 ms

  - RR:

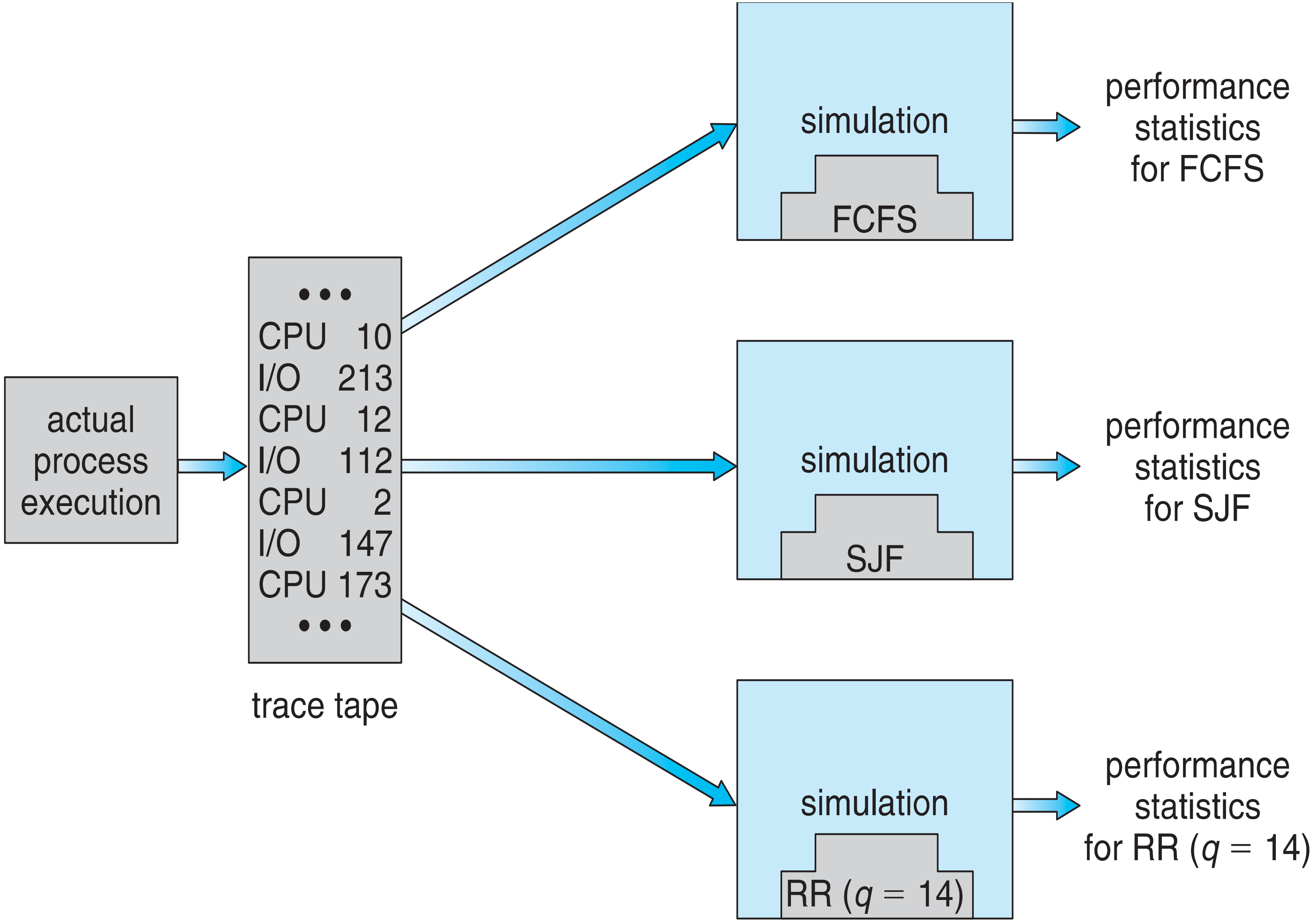


23 ms

SIMULATIONS

- Simulations more accurate
- Programmed model of computer system
- Clock is a variable
- Gather statistics indicating algorithm performance
- Data to drive simulation gathered via
- Random number generator according to probabilities
- Distributions defined mathematically or empirically
- Trace tapes record sequences of real events in real systems

EVALUATION OF CPU SCHEDULERS BY SIMULATION



THAT IS ALL FOR TODAY.

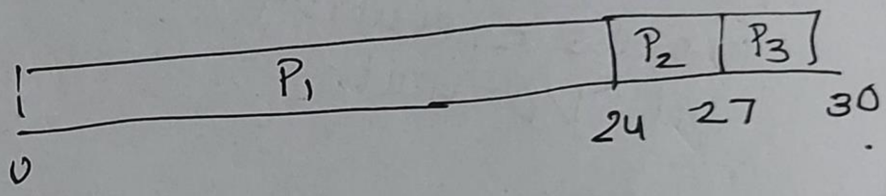
JUST IN CASE: MY HANDWRITING IS NOT READABLE

SCHEDULING ALGORITHMS

1. First Come First Served (FCFS)

Process	Burst Time
P_1	24
P_2	3
P_3	3

Gantt chart



· waiting time for $P_1 = 0$; $P_2 = 24$; $P_3 = 27$ (calculated from starting point)

$$\text{Average} = \frac{0+24+27}{3} = 17$$

Turnaround time

TAT = ~~Wag~~ Wait time + Burst Time

TAT = completion / Exit time − Arrival time.

TAT (P1) = 24 (0 + 24) (24 − 0)

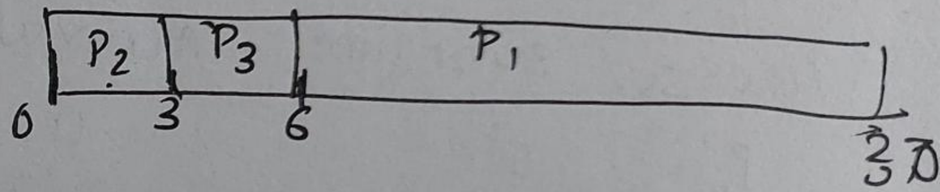
TAT (P2) = ~~3~~ 24 + 3 = 27 (27 − 0)

TAT (P3) = 27 + 3 = 30 (30 − 0)

PC: Pravallika for copy of her notes

Process	Burst Time
P_1	24
P_2	3
P_3	3

P_2 P_3 P_1



Waiting Time: $P_1 \rightarrow 6$

$P_2 \rightarrow 0$

$P_3 \rightarrow 3$

Avg waiting time = 3

TAT(P_1) = 30

TAT(P_2) = 3

TAT(P_3) = 6

convoy effect

arrived at sam time

PC: Pravallika for copy of her notes

convoy effect

		arrived at sam time
P_1	7	① P_1 P_2 P_3 P_4
P_2	4	② P_3 P_2 P_4 P_1
P_3	1	
P_4	4	

P_1	P_2	P_3	P_4

0 7 11 12 16

P_3	P_2	P_4	P_1

0 1 5 9 16

P_1 - 0
P_2 - 7
P_3 - 11
P_4 - 12

$\frac{30}{\cancel{3}4} = \cancel{10}\ 7.5$

TAT P_1 - 7
TAT P_2 - 11
TAT P_3 - 12
TAT P_4 - 16

P_1 = 1
P_2 = 9
P_3 = 0
P_4 = 5

$\frac{15}{4} = 3.75$

TAT P_1 - 16
TAT P_2 - 5
TAT P_3 - 1
TAT P_4 - 9

PC: Pravallika for copy of her notes

P_4 = 12 ... = 11

$\frac{32}{4} = 8$

SJF (Pre emptive & non pre emptive

PID	AT	Burst
P_1	0.0	8
P_2	0.4	4
P_3	1.0	1

pre emptive

P_1	P_2	P_3	P_2	P_1
0.4	1.0	2.0	5.4	13.0

Non PE

P_1	P_3	P_2
0.0 – 8.0	8.0 – 9.0	9.0 – 13.0

Waiting, Avg WT, TAT avg TAT

PE

$P_1 = 5$

$P_2 = 2 - 0.4 - \cancel{0.6} =$ ~~avg~~

$P3 = 0$

$\frac{6}{3} = 2$

TAT

$P_1 = 13$

$P_2 = 5$

$P_3 = 1$

$\frac{19}{3} = 6.33$

NPE

$P_1 = 0$

$P_2 = 9 - 0.4 = 8.6$

$P_3 = 8 - 1 = 7$

$\frac{15.6}{3}$

5.2

TAT

$P_1 = 8$

$P_2 = 12.6$

$P_3 = 8$

$\frac{28.6}{3} = 9.53$

PC: Pravallika for copy of her notes

$\frac{32}{4} = 8$

SJF (Pre emptive & non pre emptive

PID	AT	Burst
P_1	0.0	8
P_2	0.4	4
P_3	1.0	1

pre emptive

P_1	P_2	P_3	P_2	P_1
0.4	1.0	2.0	5.4	13.0

Non PE

P_1	P_3	P_2
0.0 – 8.0	8.0 – 9.0	9.0 – 13.0

Waiting, Avg WT, TAT avg TAT

PE

$P_1 = 5$

$P_2 = 2 - 0.4 - \cancel{0.6} = \cancel{avg}$

$P3 = 0$

$\frac{6}{3} = 2$

TAT

$P_1 = 13$

$P_2 = 5$

$P_3 = 1$

$\frac{19}{3} = 6.33$

NPE

$P_1 = 0$

$P_2 = 9 - 0.4 = 8.6$

$P_3 = 8 - 1 = 7$

$\frac{15.6}{3}$

5.2

TAT

$P_1 = 8$

$P_2 = 12.6$

$P_3 = 8$

$\frac{28.6}{3} = 9.53$

PC: Pravallika for copy of her notes

	Burst	Arrival time
P_1	8	0
P_2	4	1
P_3	9	2
P_4	5	3

P_1	P_2	P_4	P_3
0	8	12	17

(end: 26)

W.T.

$P_1 = 0$

$P_2 = 8 - 1 = 7$

$P_3 = 17 - 2 = 15$

$P_4 = 12 - 3 = 9$

TAT

$P_1 = 8$

$P_2 = 11$

$P_3 = 24$

$P_4 = 14$

PC: Pravallika for copy of her notes

P4 4 5 2

P1	P2	P3	P2	P4	P1

0 2 4 5 7 11 16

non pre emptive

P1	P3	P ~~4~~2	P4

0 7 8 12 16

WT

$P_1 = 0$

$P_2 = 8 - 2 = 6$

$P_3 = 7 - 4 = 3$

$P_4 = 12 - 5 = 7$

TAT

$P_1 = 7$

$P_2 = 10$ 12

$P_3 = 4$ 8

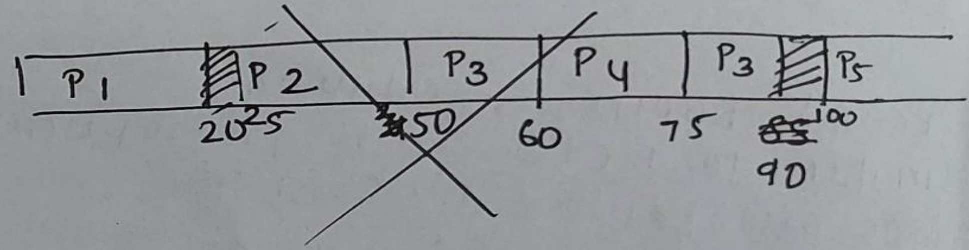
$P_4 = 11$ 16

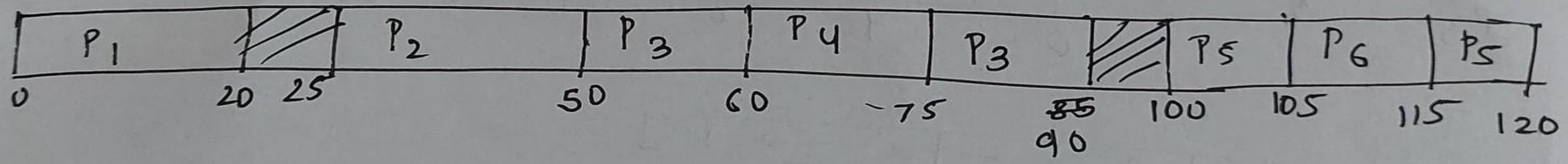
$\frac{32}{4} = 8$

PC: Pravallika for copy of her notes

PRIORITY SCHEDULING

PID	PRIORITY	BT	AT
P_1	40	20	0
P_2	30	25	25
P_3	30	25	30
P_4	35	15	60
P_5	5	10	100
P_6	10	10	105





WT

$P_1 = 0 - 0 = 0$

$P_2 = 25 - 25 = 0$

$P_3 = 75 - 30 - 10 = 35$

$P_4 = 60 - 60 = 0$

$P_5 = 115 - 100 - 5 = 10$

$P_6 = 105 - 105 = 0$

TAT

$P_1 = 20$

$P_2 = 25$

$P_3 = 60$

$P_4 = 15$

$P_5 = 20$

$P_6 = 10$

$TAT_{AVG} = 25$

PC: Pravallika for copy of her notes