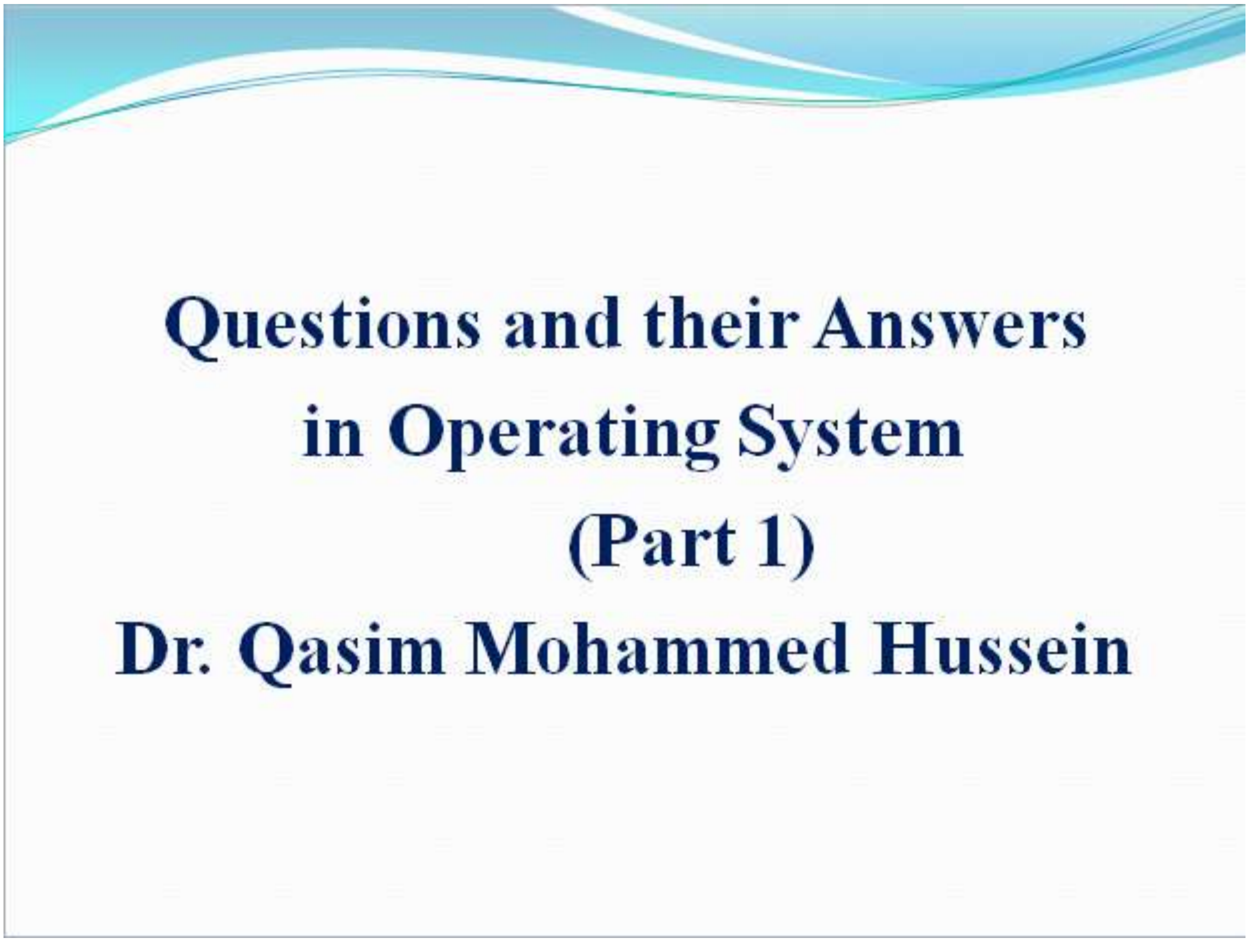
Operating System: Questions and their answers : Processes and Deadlock (Part 1)

Data · July 2015

DOI: 10.13140/RG.2.1.2024.1765

1 author:

Qasim Mohammed Hussein
Tikrit University



Q1: Define: Operating system, Process, Process state, PCB, Dispatcher, Dispatch latency, Interrupt, protection, Deadlock state, Safe state, Resource allocation state?

<u>**Answer**</u>

Operating system is a set of programs that control the execution of application programs and act as an intermediary between a user of a computer and the computer hardware.

Process is a program in execution. It is the unit of work in most system.

Process state is the current activity of that process.

The **PCB** is a data structure. It use as the repository for any information that may vary from process to process. Each process has its PCB.

The **dispatcher** is the module that gives control of the CPU to the process selected by the short-term scheduler.

Dispatch latency is the time takes for the dispatcher to stop one process and start another running.

Interrupt is an event that alters the sequence in which the processor executes instructions.

Protection refers to mechanism that control the access of programs or users to the both system and resources

Deadlock state: A set of processes is in a deadlock state when every process in the set is waiting for an event that can be caused only by another process in the set.

Safe state: A state is *safe* if the system can allocate resources to each process (up to its maximum) in some order and still avoid a deadlock.

Resource allocation state: It is defined as the number of available and allocated resources and the maximum demands of the processes.

Q2: Why are the programs and data not resided in main memory permanently?

Answer

There are two reasons:

1) Main memory is usually too small to store all needed programs and. Data permanently.

Main memory is a *volatile* storage device that loses its contents when power is turned off or otherwise lost

Q3: What are the activities of the operating system in regard to process management?

Answer

The operating system is responsible for the following activities in regard to process management:

1) Creating and deleting both user and system processes
2) Suspending and resuming processes
3) Providing mechanisms for process synchronization
4) Providing mechanisms for process communication
5) Providing mechanisms for deadlock handling

Q4: What are the activities of the operating system in regard to file management?

Answer

The operating system is responsible for the following activities in regard to file management:

1) Creating and deleting files
2) Creating and deleting directories to organize files
3) Supporting primitives for manipulating files and directories
4) Mapping files onto secondary storage
5) Backing up files on stable (nonvolatile) storage media

Q5: Explain with diagram the process states?

<u>Answer</u>

Each process may be in one of the following states:
1. New. The process is being created.
2. Running. Instructions are being executed.
3. Waiting. The process is waiting for some event to occur (such as an I/O completion or reception of a signal).
4. Ready. The process is waiting to be assigned to a processor.
5. Terminated. The process has finished execution.

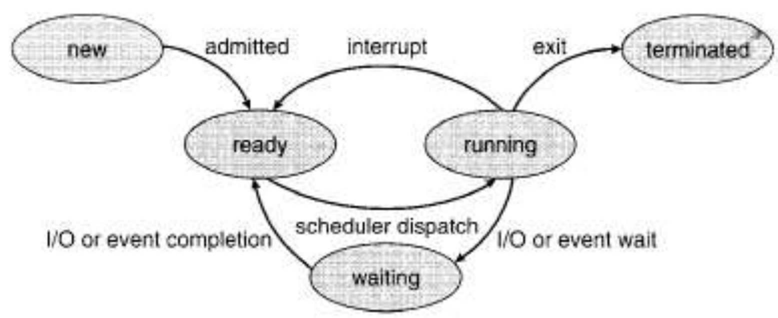


Diagram of process state.

Q6: What is information in the PCB?
<u>Answer</u>
PCB contains the following information

- Process state. The state may be new, ready, running, waiting, halted, and so on.
- Program counter. The counter indicates the address of the next instruction to be executed for this process.
- CPU registers. They include accumulators, index registers, stack pointers, and general-purpose registers, plus any condition-code information.
- CPU-scheduling information. This information includes a process priority, pointers to scheduling queues, and any other scheduling parameters.
- Memory-management information. This information may include such information as the value of the base and limit registers, the page tables, or the segment tables, depending on the memory system used by the operating system.
- Accounting information. This information includes the amount of CPU and real time used, time limits, account numbers, job or process numbers, and so on.
- I/O status information. This information includes the list of I/O devices allocated to the process, a list of open files, and so on.

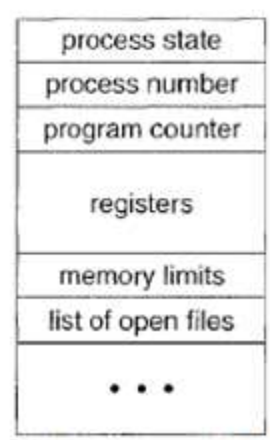


Fig. 2: Process control block

Q7: What are the Operations on Processes?

<u>Answer</u>

The operations on processes are process **creation** and **termination.**

1. **Process Creation**: A process may create several new processes, via a create-process system call, during the course of execution. The creating process is called a ***parent process***, and the new processes are called children of that process.
2. **Process Termination:** A process terminates when it finishes executing its final statement and asks the operating system to delete it or by another process via an appropriate system call.

Q8: In process creation, what are the possibilities in concerned (1) Parent execution (2) Address space of the new process (child)?

<u>Answer</u>

(1) There are two possibilities of execution:

a) The parent continues to execute concurrently with its children.
b) The parent waits until some or all of its children have terminated.

(2) There are also two possibilities in terms of the address space of the new process:

a) The child process is a duplicate of the parent process .
b) The child process has a new program loaded into it.

Q9: A parent may terminate the execution of one of its children. What are the reasons?

<u>Answer</u>

a) The child has exceeded its usage of some of the resources that it has been allocated.
b) The task assigned to the child is no longer required.
c) The parent is exiting, and the operating system does not allow a child to continue if its parent terminates.

Q10: What are the events occur when the process allocate to the CPU and is executing?

<u>Answer</u>

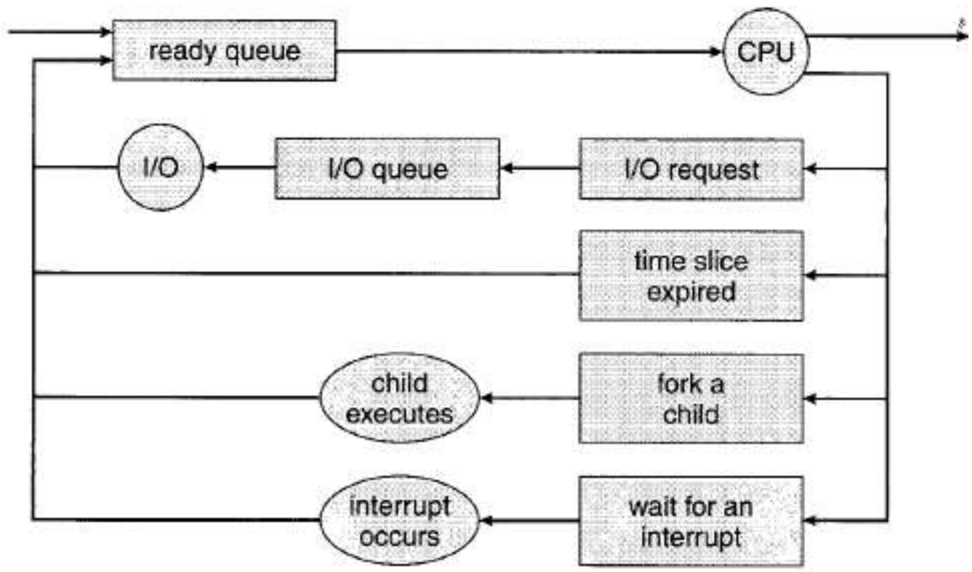
One of the following events may occur:

a) The process could issue an I/O request and places in an I/O queue.

b) The process could create a new sub process and wait for its termination.

c) The process could remove forcibly from the CPU as a result of an interrupt and be put back in the ready queue.

Exit



Q11: What are the differences between:

(1) Long-term scheduler and short-term scheduler?
(2) Preemptive and non-preemptive scheduling?
(3) CPU-bound process and I/O-bound process?
(4) Time sharing system and batch system?

<u>Answer</u>

(1) The differences between long term scheduler and short term scheduler

	Long term scheduler	short term scheduler
1	It called job scheduler	It called CPU scheduler
2	It selects processes from the pool and loads them into memory for execution.	It selects from among the processes that are ready to execute and allocates the CPU to one of them.
3	It takes many minutes for loading the new process.	It takes (10-100)ms for taking the new process.

2) Preemptive and non-preemptive scheduling?

	Non-preemptive scheduling	Preemptive scheduling
	When the CPU has been allocated to a process, the process keeps the CPU until it releases the CPU either by terminating or by switching to the waiting state	Scheduling allows a process to be interrupted in the midst of its execution, taking the CPU away and allocating it to another process.

3) CPU-bound process and I/O-bound process

		I/O bound
	Process uses more of its time doing computations.	Process is one that spends more of its time doing I/O than it spends doing computations.

4) Time sharing system and batch system?

Batch system	Time sharing system
1. The user submits jobs in regular schedule to operator. 2. No interaction between users and systems. 3. It uses by single user at time. 4. The design and implementation is simple	1. On line communication. 2. There is interaction between user and system. 3. Many users simultaneously use the system. 4. Design and implementation are complex

Q12: What are the CPU scheduling algorithm criteria?

Answer Q

The criteria include the following:

1. **CPU utilization**. Keep the CPU as busy as possible. In a real system, it should range from 40 percent (for a lightly loaded system) to 90 percent (for a heavily used system).
2. **Throughput.** The number of processes that are completed per time unit.
3. **Turnaround time (TAT).** The interval from the time of submission of a process to the time of completion. **TAT=execution time + waiting time.**
4. **Waiting time**. It is the sum of the periods spent waiting in the ready queue.
5. **Response time**. It is the time it takes to start responding, not the time it takes to output the response.

Q13: What are the advantages of: (A) Aging techniques (B) RAG (C) PCB (D) cooperating processes (E) System call in OS.

Answer Q(A) Advantages of aging techniques

Aging is a technique to solve the starvation problem, it increases the priority of processes that wait in the system for a long time.

Answer (B) Advantages of RAG

It shows the possibility of deadlock occurs. If RAG does not have a cycle, then the system is *not* in a deadlocked state. If there is a cycle, then the system may or may not be in a deadlocked state.

Answer (C) Advantages of PCB

PCB uses as the repository for any process information.

Answer (D) **Advantages of cooperating processes:**

1) Information sharing: Allow concurrent access to same information.
2) Computation speedup: Break task into subtasks, each of which will be executing in parallel with the others that speedup the execution.
3) Modularity: can dividing the system functions into separate processes or threads.
4) Convenience: Many tasks work at the same time.

Answer (E) **Advantages of system call.**

System call provides the interface between a running program and the operating system.

Q14: What are the deadlock conditions?

Answer

1) Mutual exclusion. At least one resource must be held in a non-sharable mode; that is, only one process at a time can use the resource. If another process requests that resource, the requesting process must be delayed until the resource has been released.
2) Hold **and** wait. A process must be holding at least one resource and waiting to acquire additional resources that are currently being held by other processes.
3) No **preemption.** Resources cannot be preempted; that is, a resource can be released only voluntarily by the process holding it, after that process has completed its task.
1) Circular wait. A set { P1 , •••*r Pn}* of waiting processes must exist such that Po is waiting for a resource held by *PI*, PI is waiting for a resource held by *P2* , ... , Pn-1is waiting for a resource held by *Pn* , and *Pn* is waiting for a resource held by Po.

Q15: What are the methods for Handling Deadlocks states?

Answer Q

we can deal with the deadlock problem in one of three ways:

1) We can use a protocol to prevent or avoid deadlocks. ensuring that the system will *never* enter a deadlock state.
2) We can allow the system to enter a deadlock state, detect it, and recover.
3) We can ignore the problem altogether and pretend that deadlocks never occur in the system.

Q16: How can we prevent the occurrence of a deadlock occurs?

Answer Q

By ensuring that at least one of deadlock conditions cannot hold. As follow:

1) **Mutual Exclusion:** The mutual-exclusion condition must hold for non-sharable resources.
2) **Hold and wait**: we must guarantee that, whenever a process requests a resource, it does not hold any other resources.
3) **No Preemption:** we can use the following protocol. If a process is holding some resources and requests another resource that cannot be immediately allocated to it (the process must wait), then all resources currently being held are preempted.
4) **Circular Wait:** By impose a total ordering of all resource types and to require that each process requests resources in an increasing order of enumeration.

Q17: How can we avoiding deadlocks occur?

<u>Answer</u>

For avoiding deadlocks is to require additional information about how resources are to be requested.

Each request requires the following:

1) The resource currently available.
2) The resource currently allocated to each process.
3) The future requests and releases of each process.

The above information is used to decide whether the current request can be satisfied or must wait to avoid a possible future deadlock.

A **deadlock-avoidance algorithm** dynamically examines the resource-allocation state to ensure that a circular wait condition can never exist. The resource-allocation *state* is defined by the number of available and allocated resources and the maximum demands of the processes.

Q18: Explain the deadlock detection?

<u>Answer Q</u>

If a system does not employ either a deadlock-prevention or a deadlock avoidance algorithm then a deadlock situation may occur. The system must provide:

1. An algorithm that examines the state of the system to determine whether a deadlock has occurred.
2. An algorithm to recover from the deadlock.

A detection and recovery will incur a considerable overhead in computation time that includes:

1) Run time cost of maintaining the necessary information.
2) Executing the detection algorithm.
3) The potential losses inherent in recovery from a deadlock.

Q19: How can recovery from deadlock state?

<u>Answer</u>

When a deadlock exists, several alternatives are available.

1. One possibility is to inform the operator that a deadlock has occurred and to let the operator deal with the deadlock manually.
2. The system *recovers* from the deadlock automatically. There are two options for breaking a deadlock.
   - A) **Process Termination**: Aborting processes to eliminate the deadlock. There are two methods:
     1) Abort **all** deadlocked processes.
     2) Abort one process at a time **until** the deadlock cycle is eliminated.
   - B) **Resource Preemption**: preempt: some resources from processes and give these resources to other processes until the deadlock cycle is broken. three issues need to be addressed:
     1) **Selecting a victim**. Which resources and which processes are to be preempted?
     2) **Rollback**. If we preempt a resource from a process, what should be done with that process?
     3) **Starvation**. How do we ensure that starvation will not occur, guarantee that resources will not always be preempted from the same process?

Q20: Chose the suitable word(s) to fill the following blanks?

a) A program in execution is called ----------- .
(Process, Instruction, Procedure, Function)..
b) Interval between the time of submission and completion of the job is called
(Waiting time , Turnaround time , Throughput , Response time.)
c) A scheduler which selects processes from secondary storage device is called
(Short term scheduler, Long term scheduler, Medium term scheduler, Process scheduler).
d) In ------------ several programs are kept in main memory at the same time.
(multiprocessor, multiprogramming, buffering, on- line operation).
e) The ---------- is used as the repository for any information of the process. **(process state, deadlock, CPU, PCB).**
f) A state is ------------ if the system can allocate resources for each process in some order and still avoid a deadlock.
(Starvation, mutual exclusion, RAG, safe state)
g) Each process in a system has a segment of code, called ------------- , in which the process may be changing common variables, updating a table, writing a file.
(Critical section, semaphore, race condition, segment table).
h) In overlay allocation method, the program size can be ------------- than the amount of memory allocated to it. **(less , equal, greater, equal or less) .**
i) ------------ is the module that gives control of the CPU to the process selected by the short term scheduler. **(Dispatcher, LTS, processor, aging).**
j) -------- system is used when there are rigid time requirements on the operation of processor. **(Batch, Time sharing, Real time, None).**
k) In ------ system, the processors communicate with one and other through various communications lines. **(Parallel, distributed, overlay, spooling)**
l) In ------- scheduling algorithm, the process that requests the CPU first is allocated the CPU first. **(RR, FCFS, SJF, Priority).**
m) In a computer, to start running-for instance, when it is powered up or rebooted, it needs to have an ---------- program to run **(Bootstrap, Compiler, Editor, Word).**
n) The ------------ memory is a technique that allows the execution of processes that may not be completely in memory. **(Cache, Register, Secondary, Virtual)**
o) In ------- storage management, The user has control over the entire memory space. **(Bare machine, Overlay, Swapping, Fixed partition).**
p) Deadlock --------- provides a set of methods for ensuring that at least one of the necessary conditions cannot hold.**(Prevention, Avoidance, Handling, Recovery).**
q) ----------- refers to mechanism that control the access of programs or users to the both system and resources.**(Security, CPU, Protection, I/O).**
r) In ------- scheduling algorithm, a small unit of time, called time quantum, for each process. **(RR, FCFS, SJF, Priority).**
s) ------------ is the sum of the periods spent waiting in the ready queue. **(response time, turnaround time, waiting time , execution time**)
t) In paging, the logical address space is breaking into fixed –sized blocks called ----------------. **(frame, page, segment, register**)

u) ----------------- Scheduler select process the process that is ready queue to execute and allocate the CPU to it. **(LTS, STS, MTS, dispatcher).**

v) In ------------ fit placement strategy, Allocate the *smallest* hole that is big enough. **(first fit, best fit, worst fit, demand)**

w) Aging is a technique to handle -------------------- problem. **(starvation, synchronization, fragmentation, segmentation)**

x) Each process in a system has a segment of code, called -------------, in which the process may be changing common variables, updating a table, writing a file. (**Critical section, semaphore, race condition, segment table).**

y) The bootstrap program is stored in **(ROM, RAM, register, cache)**

z) We protect the -------------------- to prevent a user program from an infinite loop. (**I/O device, memory, CPU, demand).**

aa) A state is ------------ if the system can allocate resources for each process in some order and still avoid a deadlock. **(starvation, mutual exclusion, RAG, safe state)**

bb) Possible side effects of**---------------** deadlocks are low device utilization and reduced system throughput. **(Preventing, avoiding, RAG, recovering).**

cc)**- - - - -** is a segment of code in system process **in** which the process may be changing common variables, updating a table, writing a file, and so on. (semaphore, Critical section, Race condition, PCB)

dd) - - - - - *is* the sum of the periods spent waiting in the ready queue. (response time, turnaround time, waiting time , execution time

Answer Q20

(a) Process	(b) turnaround time	(c) long term scheduler
(d) Multiprogramming	(e) PCB	(f) safe state
(g) Critical section	(h) greater than	(i) dispatch
(j) Real time	(k) distributed	(l) FIFO
(m) Bootstrap	(n) virtual	(o) bare machine
(p) Prevention	(q) protection.	(r) RR
(s) Waiting time	(t) page	(u) STS
(v) best - fit	(w) starvation	(x) critical section
(y) ROM	(z) CPU	(aa) safe state.
(bb) preventing	(cc) critical section	(dd) waiting time.

Process scheduling examples

Q21: Find the average waiting time and average turnaround time for executing the following processes using FCFS (first-come first-service) scheduling?

Process	Burst time
P0	7
P1	5
P2	2
P3	9

<u>Answer Q21</u>

The first step of the solution is founding the Gantt chart.

Gantt chart:

P0	P1	P2	P3

0 7 12 14 23

The waiting time of P0 = 0
The waiting time of P1 = 7
The waiting time of P2 = 12
The waiting time of P3 = 14
The average waiting time = (0 + 7 + 12 + 14) /4 = 33/4 = 8.25

The turnaround time of p0 = 7
The turnaround time of p1 = 12
The turnaround time of p2 = 14
The turnaround time of p3 = 23
The average turnaround time = (7+12+14+23)/4 = 56/4 = 14

Q22: Find the average waiting time (A.W.T) and average turnaround time (A.T.A.T) for executing the following process using (1) Preemptive short-job first (2) Non-preemptive short-job first?

Process	P1	P2	P3	P4	P5
Burst time	5	13	8	4	10
Arrival time	2	3	0	5	1

<u>Answer</u>

(1) <u>Using preemptive short-job first</u>

Gantt chart

P3	P1	P4	P3	P5	P2

0 2 7 11 17 27 40

W.T. of p1 = 2 -2 = 0 T.A.T. of P1=7 – 2 = 5

W.T. of p2 = 27 – 3 = 24 T.A.T. of P2= 40 – 3 = 37

W.T. of p3 = 0+ (11-2) = 9 T.A.T. of P3= 17 – 0 = 17

W.T. of p4 = 7 – 5 = 2 T.A.T. of P4= 11 – 5 = 6

W.T. of p5 = 17 – 1 = 16 T.A.T. of P5= 27 -1 = 26

A.W.T. = (0+24+9+2+16)/5 = 51/5 = 10.2

A.T.A.T = (5+37+17+6+26)/5 = 91/5 =18.2

2) **Using non- preemptive short-job first**

Gantt chart

P3	P4	P1	P5	P2

0 8 12 17 27 40

W.T. of p1 = 12 -2 = 10 T.A.T. of P1= 17 – 2 = 15

W.T. of p2 = 27 – 3 = 24 T.A.T. of P2= 40 – 3 = 37

W.T. of p3 = 0 T.A.T. of P3= 8 – 0 = 8

W.T. of p4 = 8 – 5 = 3 T.A.T. of P4= 12 – 5 = 7

W.T. of p5 = 17 – 1 = 16 T.A.T. of P5= 27 -1 = 26

A.W.T. = (10+24+0+3+16)/5 = 53/5 = 10.6

A.T.A.T = (15+37+8+7+26)/5 = 93/5 = 18.6

Q23: Find the average waiting time and turnaround time for executing the following process using priority scheduling algorithm?

Process	P1	P2	P3	P4	P5
Burst time	5	13	8	6	12
Priority	1	3	0	4	2

Answer

Gantt chart

P3	P1	P5	P2	P4

0 8 13 25 38 44

W.T. of p1 = 8 T.A.T. of P1=13

W.T. of p2 = 25 T.A.T. of P2= 38

W.T. of p3 = 0 T.A.T. of P3= 8

W.T. of p4 = 38 T.A.T. of P4= 44

W.T. of p5 = 13 T.A.T. of P5= 25

A.W.T. = (8+25+0+38+13)/5 = 84/5 = 16.8

A.T.A.T = (13+38+8+44+25)/5 = 128/5 =25.6

Q24: Find the average waiting time (A.W.T.) and the average turnaround time (A.T.A.T.) for executing the following processes using round-robin algorithm, where time quantum is 5?

Process	P1	P2	P3	P4	P5
Burst time	11	4	14	9	21
Arrival time	5	0	0	1	2

<u>Answer</u>

P2= ~~4~~ 0
P3= ~~14~~ ~~9~~ ~~4~~ 0
P4= ~~9~~ ~~4~~ 0
P5= ~~21~~ ~~16~~ ~~11~~ ~~6~~ ~~1~~ 0
P1= ~~11~~ ~~6~~ ~~1~~ 0

Gantt chart

P2	P3	P4	P5	P1	P3	P4	P5	P1	P3	P5	P1	P5	P5

0 4 9 14 19 24 29 33 38 43 47 52 53 58 59

W.T. of P1 =(19-5)+(38-24)+(52-43) = 14 + 14+9 = **37**
W.T. of P2 = **0**
W.T. of P3 = (4-0) + (24-9) + (43-29) = 4 + 15 + 14 = **33**
W.T. of P4 = (9-1) + (29-14) = 8 + 15 = **23**
W.T. of P5 = (14-2) + (33-19) + (47-38) + (53-52) = 12+14+9+1=**36**
A.W.T. = (37 + 0+ 33 + 23 + 36) /5 = 129 /5 = 25.8

T.A.T of P1 = (53-5)= **48**
T.A.T of P2 = (4-0) = **4**
T.A.T of P 3= (47 – 0) = **47**
T.A.T of P 4= (33-1) = **32**
T.A.T of P 5= (59-2) = **57**
A.T.A.T. = (48 + 4+ 47 + 32 + 57) /5 = 188/5 = 37.6

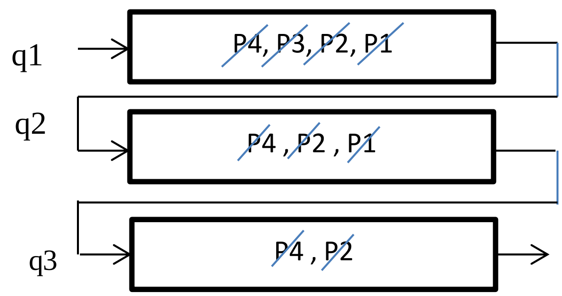
Q25: Consider a multilevel feedback queue scheduling (MLFBQ) with three queues q1, q2, and q3. q1 and q2 use round-robin algorithm with time quantum (TQ) = 5, and 4 respectively. q3 use first-come first-service algorithm. Find the average waiting time (A.W.T) and average turnaround time (A.T.A.T) for executing the following process?

Processes	P1	P2	P3	P4
Burst time	8	22	4	12

Answer

In MLFBQ scheduling algorithm, the process move between queues. If a process uses too much CPU time, it will be moved to a lower-priority queue.

	q1	q2	q3
P1:	~~8~~	3	-
P2:	22	17	13
P3:	4	-	-
P4:	12	7	3



Gantt chart

P1	P2	P3	P4	P1	P2	P4	P2	P4

0 5 10 14 19 22 26 30 43 46

W.T. of p1 = 0 + (19-5) = 14 T.A.T. of P1=22

W.T. of p2 = 5+ (22-10) + (30-26) = 5+12+4 = 21 T.A.T. of P2= 43

W.T. of p3 = 10 T.A.T. of P3= 14

W.T. of p4 = 14 + (26-19) + (43-30) = 14+7+13= 34 T.A.T. of P4= 46

A.W.T. = (14 + 21 +10 + 34)/4 = 79/4 = 19.75

A.T.A.T = (22 + 43 + 14 + 46) = 125/4 = 31.25

Deadlock examples

Q26: Suppose we have two resources, A, and B. A has 6 instances and B has 3 instances. Can the system execute the following processes without deadlock occurring?

Process	Allocate		Maximum need	
	A	B	A	B
P1	1	1	2	2
P2	1	0	4	2
P3	1	0	3	2
P4	0	1	1	1
P5	2	1	6	3

<u>Answer</u>

Process	Need	
	A	B
P1	1	1
P2	3	2
P3	2	2
P4	1	0
P5	4	2

Available: A= 1; B= 0

```
     1  0       1  1      2  2      3  2
      0  1      1  1      1  0      1  0
P4 ======> P1 ======> P3 ======> P2 ======> P5
     1  1        2  2      3  2      4  2
```

We can execute the processes in the sequence <P4, P1, P3, P2, P5> without deadlock.

Q27: Consider we have five processes P0, P1, . . . P5 and three resources A, B, and C. Is the executing the following processes in the safe state?

Process	Allocation			Maximum need			Available		
	A	B	C	A	B	C	A	B	C
P0	1	2	0	2	2	2	0	1	0
P1	1	0	0	1	1	0			
P2	1	1	1	1	4	3			
P3	0	1	1	1	1	1			
P4	0	0	1	1	2	2			
P5	1	0	0	1	5	1			

<u>Answer</u>

We find the resources that need for each process

Process	Need		
P0	1	0	2
P1	0	1	0
P2	0	3	2
P3	1	0	0
P4	1	2	1
P5	0	5	1

P1		p3		p4		p0		p2		p5
	0 1 0		1 1 0		1 2 1		1 2 2		2 4 2	
	1 0 0		0 1 1		0 0 1		1 2 0		1 1 1	
	1 1 0		1 2 1		1 2 2		2 4 2		3 5 3	

The process in safe state if they are executed in the sequence<P1, P3, P4, P0, P2, P5>

Q28: Suppose we have five processes and three resources, A, B, and C. A has 2 instances, B has 5 instances and C has 4 instances. Can the system execute the following processes without deadlock occurring, where we have the following?

Process	Maximum need			Allocation		
	A	B	C	A	B	C
P1	1	2	3	0	1	1
P2	2	2	0	0	1	0
P3	0	1	1	0	0	1
P4	3	5	3	1	2	1
P5	1	1	2	1	0	1

<u>Answer</u>

The available is A=0, B=1, C=0.

The current need is

Process	Current need		
	A	B	C
P1	1	1	2
P2	2	1	0
P3	0	1	0
P4	2	3	2
P5	0	1	1

P3		P5		P1		?
	0 1 0		0 1 1		1 1 2	
	0 0 1		1 0 1		0 1 1	
	0 1 1		1 1 2		1 2 3	

The deadlock is occurred since the available resources is less than the needs of P2 and P4.

Reference

Operating system concepts

By: Abraham Silberschatz, Peter Baer Galvin, Greg Gang