Name: Pauravi Nagarkar
SCU ID: W1650209
OS ASSIGNMENT 3

Ques 1: Local Area Networks utilize a media access method called CSMA/CD, in which stations sharing a bus can sense the medium and detect transmissions as well as collisions. In the Ethernet protocol, stations requesting the shared channel do not transmit frames if they sense the medium is busy. When such transmission has terminated, waiting stations each transmit their frames. Two frames that are transmitted at the same time will collide. If stations immediately and repeatedly retransmit after collision detection, they will continue to collide indefinitely.

(a) Is this a resource deadlock or a livelock?

(b) Can you suggest a solution to this anomaly?

(c) Can starvation occur with this scenario?

Ans 1.

a) This resource is a livelock for sure.

- Livelock occurs when two or more processes continuously repeat same sequence of interaction in response to other process without doing any useful work.
- Here for livelock processes are not in waiting state and are running concurrently.
- Here when medium is busy, waiting stations transmits frame, i.e there is no waiting queue.
- The non-waiting stations transmits the frame continuously but will even stop. When medium will two requesting stations are free to transfer.
- The resource can be in deadlock when both the stations are waiting for transfer and non can transfer.

b) Yes the anomaly is caused by livelock can be solved by:

- The livelock occur when two process are concurrently repeats same sequence o steps without doing useful work.
- the only solution to this Problem is setting timer before sending next signal.
- The station sending signal should wait some random time before transmitting the signal to avoid collusion of the signal.
- Each station should set different timers. If each station wait before transmitting the next frame. The anomaly can be solved easily.

c) Yes, there is a chance of starvation, as some station won't get chance to transmit as well as won't any signal. So, Starvation is bound to happen in livelock.

- As access to the channel is probabilistic, and because newly arriving stations can compete and be allocated the channel before stations that have retransmitted some number of times, starvation is possible.
- But in deadlock the starvation is bound to happen. Here starvation can be avoided by setting the timer to every station before every collusion retransmission.

Ques 2 : Consider the following state of a system with four processes, *P1*, *P2*, *P3*, and *P4*, and five types of resources, *RS1*, *RS2*, *RS3*, *RS4*, and *RS5:*

Show that there is a deadlock in the system. Identify the processes that are deadlocked.

C =

0	1	1	1	2
0	1	0	1	0
0	0	0	0	1
2	1	0	0	0

R =

1	1	0	2	1
0	1	0	2	1
0	2	0	3	1
0	2	1	1	0

E = (24144)

A = (01021)

Ans 2 :

A) Identifying the processes that are in deadlock.

- Given data:

- Current allocation Matrix (C) =

	RS1	RS2	RS3	RS4	RS5
P1	O	1	1	1	2
P2	0	1	0	1	0
P3	0	0	0	0	1
P4	2	1	0	0	0

- Requested Matrix (R) =

	RS1	RS2	RS3	RS4	RS5
PS1	1	1	0	2	1
PS2	0	1	0	2	1
PS3	0	2	0	3	1
PS4	0	2	1	1	0

- Existing Resource (E) =

2	4	1	4	4

- Available Resource(A) =

0	1	0	2	1

B) Identifying processes in deadlock using Deadlock detection algorithm.

- P1 cannot be satisfied as RS1 is not available. P1 needs 1 available resource it has none. Also available resource for RS1 IS 0.
- P2 can be satisfied, P2 has satisfied all requested resources except RS4, it has one it will take one more from available resource vector eventually P2 will run and then it will returns all its resources back.

New A = C[P2] + current A

A =

0	2	0	3	1

- P3 will also run, as requested matrix of P3 is (0 2 0 3 1) its current allocated matrix is (0 0 0 0 1)
- It will check for available resource to get RS2, RS4. P3 needs 2 unit of RS2 which is available and 3 unit of RS4 which is available. Hence it borrows the resources satisfies its condition. Hence P3 will run and return the resources.

New A = C[P3] + current A = [0 0 0 0 1] + [0 2 0 3 1]

A =

0	2	0	3	2

- Checking for P4 , it requested matrix is (0 2 1 1 0), Its current allocation is (2 1 0 0 0) that is it needs RS2,RS3,RS4. Our Available matrix is (0 2 0 3 2). Hence, we won't be able to satisfy its RS3 resource. Hence our P4 won't run as it won't get enough resources.

C) Now the system will be in deadlock as

- P4 needs 1 resource of RS3 and we have 0 available as one existing resource of RS3 is acquired by P1.
- Similarly, P1 needs 1 resource of RS1 and 0 available as all the existing RS1 resources are acquired by P4 .
- As P1 & P4 both will wait for each other to run and release the resource, but as both are holding same common resource the system will be in deadlock.

D) Final current allocation and request Matric will be

C =

	RS1	RS2	RS3	RS4	RS5
P1	0	1	1	1	2
P2	0	0	0	0	0
P3	0	0	0	0	0
P4	2	1	0	0	0

R =

	RS1	RS2	RS3	TS4	RS5
P1	1	1	0	2	1
P2	0	0	0	0	0
P3	0	0	0	0	0
P4	0	2	1	1	0

Ques 3: In the below figure, If D asks for one more unit, does this lead to a safe state or an unsafe one? What if the request came from C instead of D? (**Note :** Maximum units available is 10)

	Has	Max
A	1	6
B	1	5
C	2	4
D	4	7

Free: 2

Ans 3:

A. When D asks for one more unit

- We need to find a sequence the is guaranteed to work and finish all processes.
- When D request and gets another unit, it give us:

	Has	Max
A	1	6
B	1	5
C	2	4
D	5	7
	Free : 1	

- Now we have only 1 free unit left and each of the other active processes needs more units.
- There is no sequence that guarantee that all process will finish running all. As A need 5 more to complete, B needs 4 more, C needs 2 more and D needs 2 more units.
- Hence process won't complete the execution hence we are in unsafe state.

B. When C request for one unit:

- Initial state:

	Has	Max
A	1	6
B	1	5
C	2	4
D	4	7
	Free : 2	

- Number of free units are 2. When C requests for a unit then it becomes:

	Has	Max
A	1	6
B	1	5
C	3	4
D	4	7
	Free :1	

- And we still have 1 free unit available which can be used to complete C when it asks for max unit it needs.

	Has	Max
A	1	6
B	1	5
C	4	4
D	4	7
	Free : 0	

- So C executes successfully, and free its resources hence we get following result:

	Has	Max
A	1	6
B	1	5
C	0	-
D	4	7
	Free: 4	

- Now we have 4 units free. Either B or D can request for maximum units, and it can satisfy both.
- Let consider B requests maximum required units.

It already has 1 unit it request 4 and max units needed for execution is 5. Hence process B

will execute and release all its resources/units.

	Has	Max
A	1	6
B	5	5
C	0	-
D	4	7
	Free: 0	

- After B completes, it will release all its units:

	Has	Max
A	1	6
B	0	-
C	0	-
D	4	7
	Free :5	

- Now remaining both D or A can request maximum of units. Both will complete execution.
- Let consider A requests max required units:
- A already has 1 it need max 6 for execution , we have 5 units available for execution. A use them so A will finish its execution.

	Has	Max
A	6	6
B	0	-
C	0	-
D	4	7
	Free:0	

- Hence A will finish execution and release the unit then available units will be .

	Has	Max
A	0	-
B	0	-
C	0	-
D	4	7
	Free: 6	

- After completion of A we have 6 free units left which can be used for D.
- D has 4 unit he will need more 3 for execution. He will use available 3 units units complete its execution and return back all the available resources.

	Has	Max
A	0	-
B	0	-
C	0	-
D	7	7
	Free : 3	

- Now the final available units after completion of all the processes will be.

	Has	Max
A	0	-
B	0	-
C	0	-
D	0	-
	Free : 10	

- At end we have Available units: 10
- Thus, by starting with C asking for one unit, all the states are safe, execution is completed, and we don't encounter deadlock.

Ques 4: Solved and attached the zip file.

----------X----------