Name: Sumaiya Man-Area
ID: 221-15-4671
Section: 61-J

Answer to the question no-1

(a)

(i)

Process	Allocation A	B	C	D	Maximum Need A	B	C	D	Connected A	B	C	D	Available A	B	C	D
P_0	0	2	3	0	1	4	5	0	1	2	2	0	7	5	8	10
P_1	3	1	0	1	5	2	0	2	2	1	0	1	7	7	11	10
P_2	1	0	1	0	1	2	1	2	0	2	0	2	10	8	12	11
P_3	0	0	0	1	0	1	0	1	0	1	0	0	11	8	12	12
P_4	2	1	1	0	3	2	3	1	1	1	2	1	11	8	12	12
P_5	1	1	0	0	2	2	1	1	1	1	1	1	13	9	13	12
	7	5	5	2									14	10	13	12

Given that,

$A = 14$
$B = 10$
$C = 13$ and $D = 12$

Now, the safe sequence will be —

$P_0 \rightarrow P_1 \rightarrow P_2 \rightarrow P_3 \rightarrow P_4 \rightarrow P_5$

(ii)

Given that, $P_0 (13, 5, 3, 1)$ and $P_1 (0, 1, 0, 0)$

Now,

if P_0 request $\le$ need (b)

$(13, 5, 31) \nleq (1, 2, 2, 0)$

Not safe, request exceeds Need (P_0)

For P_1: check if request $\leq$ Need (P_1)

$(0,1,0,0) \leq (2,1,0,1)$ safe

Granting the request:

1) available update: $(7, 4, 8, 10)$
2) Updated allocation: $(3, 2, 0, 1)$
3) Needed update: $(2, 0, 0, 1)$

So, P_0 can't be granted but P_1 can be granted.

(b)

here, available resources:

R_1 (Laptop): 3, R_2 (Network Routers): 2, R_3 (Hard Drives): 2

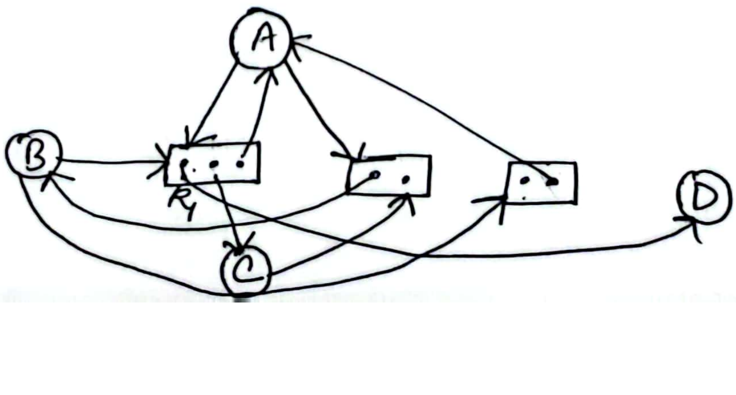
Now,

Team A: Allocation : 1 Laptop, 1 hard drive
Request : 1 laptop, 1 router

Team B: Allocation : 1 router
Request : 1 laptop, 1 hard drive

Team C : Allocation : 1 laptop
Request : None

Resource Allocation Graph:



(c)

The resource allocation graph represents safe state:

$P_2 \rightarrow P_3 \rightarrow P_1 \rightarrow P_4 \rightarrow P_5 \rightarrow P_0$

As, there's no cycle so each resource has enough instance to satisfy the requests. here, P_2 can finish because it has no request where the P_3 need R_4, which can be finish as well. After that P_1 can finish and then P_4 as well. P_5 can finish as R_2 is available. Then P_0 can finish at last, all process executed.

Answer to the question no-2

(a)

Dynamic Partitioning Scheme:

Memory is allocated dynamically which is based on process requests and after that creating partitions of required sizes.

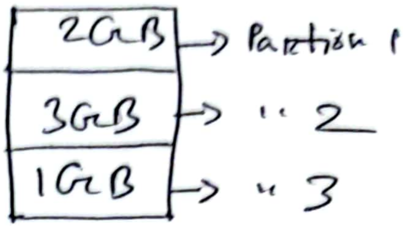
Advantages:

i) Flexible for processes of different sizes

ii) Efficient memory utilization.

Disadvantages:

1. Overhead in managing memory allocation
2. External fragmentation



(b)

The reasons behind the laggy issue in performance is described below:

1. Memory Leaks: the processes fails to release memory.

2. Insufficient Memory: (i) Excessive disk shows the system
(ii) System spends too much time in swapping.

3. Memory fragmentation: Non-contiguous blocks prevent longer allocation

Solutions for perfomance improvement:

1. Performs memory compaction to reduce fragmentation
2. Detect, fix memory leaks
3. Uses effective memory allocations algorithms

Answer to the question no-3

(a)

Given,

Physical memory: 3 frames

reference sequence: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5, 6, 2, 1, 3, 7, 7, 0, 9

LRU:

Q	1	2	3	4	1	2	5	1	2	3	4	5	6	2	1	3	7	7	0	9
F_1	1	1	1	4	4	4	5	5	5	3	3	3	6	6	6	3	3	3	3	9
F_2		2	2	2	1	1	1	1	1	1	4	4	4	2	2	2	7	7	7	7
F_3			3	3	3	2	2	2	2	2	2	5	5	5	1	1	1	1	0	0
	F	F	F	F	F	F	F	H	H	F	F	F	F	F	F	F	F	H	F	F

Hit ratio $= \frac{3}{10} \times 100\% = 15\%$

Fault ratio $= \frac{17}{20} \times 100\% = 85\%$

Optimal:

	1	2	3	4	1	2	5	1	2	3	4	5	6	2	1	3	7	7	0	9
F_1	1	1	1	1	1	1	1	1	1	3	4	4	4	4	4	3	3	3	3	9
F_2		2	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	0	0
F_3			3	4	4	4	5	5	5	5	5	5	6	6	6	6	7	7	7	7
	F	F	F	F	H	H	F	H	H	F	F	H	F	H	F	F	F	H	F	F

Hit ratio $= \frac{7}{20} \times 100 = 35\%$, fault ratio $= \frac{13}{20} \times 100\%$ $= 65\%$

so, optimal is more effcient.

(b)

Stratigies to improve LRU efficiency:

(i) second chance algorithm;

(ii) Clock algorithm

(iii) Adaptive Algorithm.

Stratigies to improve optimal Algorithm efficiency:

It motivates to forecast future page usage despite the fact that they are impracticable.

Practical constraints and Trade off:

a. Hardware support

b. complexity vs performance

c. Prediction challenges

In conclusion page replacement effciency can be increased by utilizing optimals principles and improving LRU.