Page No.
Date

Operating System and Concepts.
(BCSC0055)

Name: Luvkush Sharma
Rollno.: 2115800015 (14)
Course: B.Tech CS (Hons.)

1) A deadlock is a situation where a set of processes are blocked because each process is holding a resource and waiting for another resource acquired by some other process.

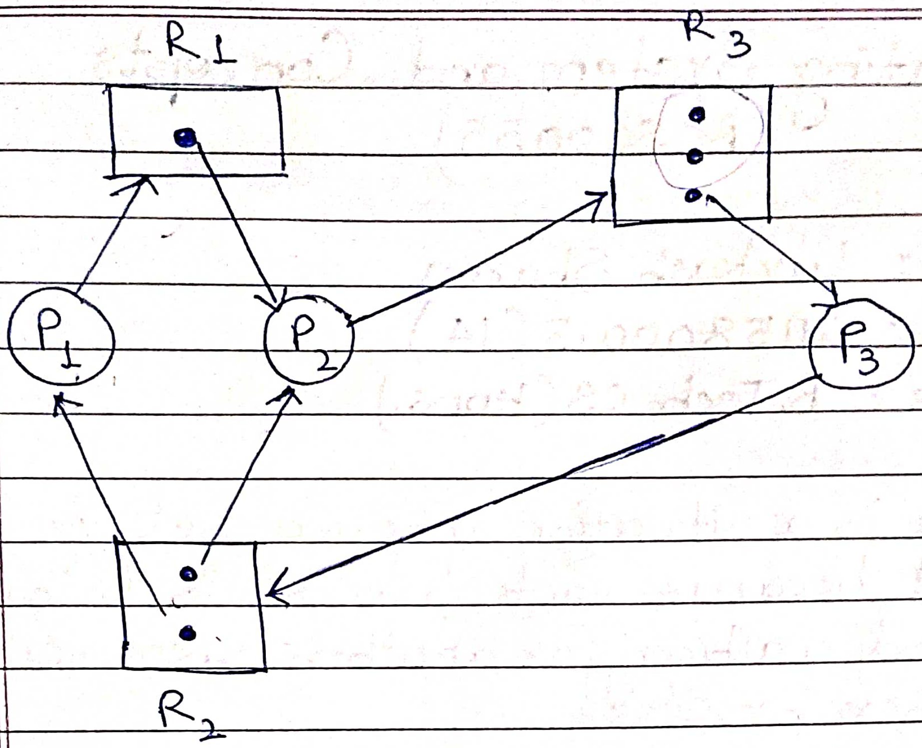
To prevent system from hold and wait we can implement a protocol where a process requests all the resources it needs at once before starting execution. Release resources that are not needed immediately by a process, so they can be used by other processes.

To prevent system from circular wait we can implement a resource allocation strategy that prevents deadlock. for example, we can use banker's algorithm to ensure that resources are allocated in a safe manner.

2)

$P1 \rightarrow R1 \rightarrow P2 \rightarrow R3 \rightarrow P3 \rightarrow R2 \rightarrow P1$

$P2 \rightarrow R3 \rightarrow P3 \rightarrow R2 \rightarrow P2$



~~We can see that there is a cycle in the graph that includes all three processes and all three resources.~~

~~The cycle can be traced as follows:~~

~~R1 → P2 → R3 → P3 → R2 → P1~~

Therefore, there is not a deadlock in this system

3) a) Content of the need matrix

Need (max − allocation)

	A	B	C
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

b) Find the safe sequence ? if exist

Step-1: $\boxed{Need_i \leq Available}$

eck P_0 7 4 3 ≤ 3 3 2 No.

heck P_1 1 2 2 ≤ 3 3 2 Yes.

So, first P_1 executes and after execution allocated resource will be free.

$\langle P_1 \rangle$

Available

A	B	C
5	3	2

$$\left[\because \begin{array}{r} 3\ 3\ 2 \\ \underline{2\ 0\ 0} \\ 5\ 3\ 2 \end{array} \right]$$

heck P_2

6 0 0 ≤ 5 3 2 No.

heck P_3 0 1 1 ≤ 5 3 2 Yes

$\langle P_1, P_3 \rangle$

Available Now

A	B	C
7	4	3

heck P_4 4 3 1 ≤ 7 4 3 Yes

$\langle P_1, P_3, P_4 \rangle$

Available Now.

7 4 5

Check P_0 7 4 3 $\leq$ 7 4 5 Yes

$\langle P_1, P_3, P_4, P_0 \rangle$

Available Now

7 5 5

Check P_2 6 0 0 $\leq$ 7 5 5 Yes

$\langle P_1, P_3, P_4, P_0, P_2 \rangle$

Available Now.

10 5 7

Safe sequence : $\langle P_1, P_3, P_4, P_0, P_2 \rangle$

c)

i) Request $\leq$ Need

$\langle 3, 3, 0 \rangle \leq \langle 4, 3, 1 \rangle$

ii) Request $\leq$ Available

$\langle 3, 3, 0 \rangle \leq 3\ 3\ 2$

iii) Allocate $\langle 3, 3, 0 \rangle$ from available

Page No.
Date

i.e Allocation of P_4 = Allocation (current) + Request

	A	B	C
P_4	3	3	2

Also, update available:

A	B	C
0	0	2

Now, find safe sequence, if it exists then system not in deadlock and request is granted.

Need matrix of P_4 [∵ $Need_i$ = $Need_i$ − $Request_i$]

A	B	C
1	0	1

Now, find safe sequence.

Check. Need ≤ Available

Check P_0: 7 4 3 ≤ 0 0 2 No.

Check P_1: 1 1 2 ≤ 0 0 2 No.

Check P_2: 6 0 0 ≤ 0 0 2 No.

Check P_3: 0 1 1 ≤ 0 0 2 No.

Check P_4: 1 0 1 ≤ 0 0 2 No.

After assigning extra resource to P_1, we are unable to find safe sequence. Hence system in deadlock.

So, Request not granted.

4) i) Need

	A	B	C	D
P_0	0	0	0	0
P_1	0	7	5	0
P_2	1	0	0	2
P_3	0	0	2	0
P_4	0	6	4	2

ii) $Need_i \leq Available$

P_0 → 0 0 0 0 ≤ 1 5 2 0 Yes

<P_0>

Available Now.
1 5 3 2

P_1 → 0 7 5 0 ≤ 1 5 3 2 No.

P_2 → 1 0 0 2 ≤ 1 5 3 2 Yes

Page No.
Date

$\langle P_0, P_2 \rangle$

Available Now.

2 8 8 6

$P_3 \rightarrow$ 0 0 2 0 $\leq$ 2 8 8 6 Yes.

$\langle P_0, P_2, P_3 \rangle$

Available Now.

2 14 11 8

$P_4 \rightarrow$ 0 6 4 2 $\leq$ 2 14 11 8 Yes

$\langle P_0, P_2, P_3, P_4 \rangle$

Available Now.

2 14 12 12

$P_1 \rightarrow$ 0 7 5 0 $\leq$ 2 14 12 12 Yes.

Safe sequence. $\langle P_0, P_2, P_3, P_4, P_1 \rangle$

Hence, system in a safe state.

Page No.
Date

iii) a) Request ≤ Need

$\langle 0,4,2,0 \rangle \leq \langle 0,7,5,0 \rangle$

b) Request ≤ Available

$\langle 0,4,2,0 \rangle \leq \langle 1,5,2,0 \rangle$

c) Allocate the demand to P_1 from available.

Allocation of P_1 New $= 1\ 0\ 0\ 0 + 0\ 4\ 2\ 0$

$\Rightarrow \langle 1,4,2,0 \rangle$

Update Available.

A	B	C	D
1	1	0	0

$Need_i = Need_i - Request_i$

A	B	C	D
0	3	3	0

Now, find safe sequence.

$P_0 \rightarrow$ Need ≤ Available

$0\ 0\ 0\ 0 \leq 1\ 1\ 0\ 0$ Yes

$\langle P_0 \rangle$

Available New.

1 1 1 2

Page No.
Date

$P_1 \rightarrow$ 0 3 3 0 $\leq$ 1 1 1 2 No.

$P_2 \rightarrow$ 1 0 0 2 $\leq$ 1 1 1 2 Yes

$< P_0, P_2 >$

Available Now.

A	B	C	D
2	4	6	6

$P_3 \rightarrow$ 0 0 2 0 $\leq$ 2 4 6 6 Yes

$< P_0, P_2, P_3 >$

Available Now.

2 10 9 8

$P_4 \rightarrow$ 0 6 4 2 $\leq$ 2 10 9 8 Yes

$< P_0, P_2, P_3, P_4 >$

Available Now.

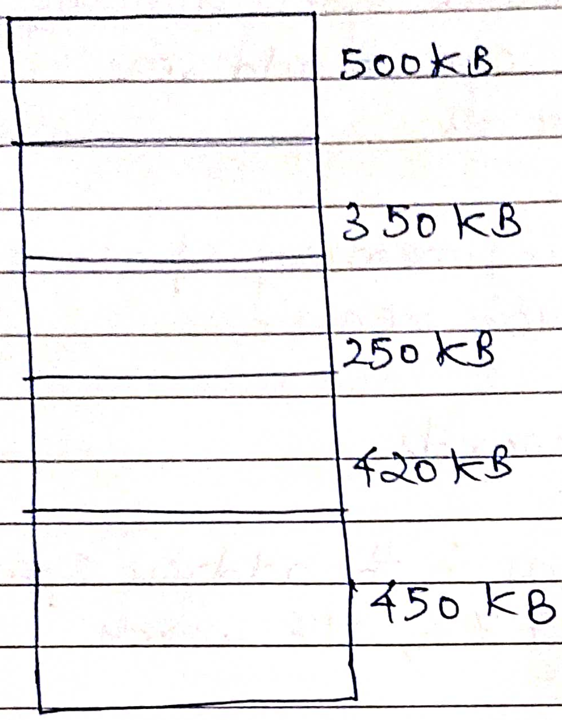
2 10 10 12

$P_1 \rightarrow$ 0 3 3 0 $\leq$ 2 10 10 12 Yes

$\boxed{< P_0, P_2, P_3, P_4, P_1 >}$ Safe sequence.

As, safe sequence exists hence system still not in deadlock i.e request granted.

13)



Process : 325 KB, 150 KB, 400 KB & 375 KB.

Best fit :- The best fit algorithm will allocate the smallest possible partition that can accomomodate a process, out of all available partitions.

325 KB process allocated to the 350 KB partition.

150 KB process allocated to the 250 KB partition.

400 KB process allocated to the 420 KB partition.

375 KB process allocated to the 450 KB partition.

Page No.
Date

Worst fit :- The worst-fit algorithm will allocate the largest possible partition available.

325 KB process allocated to the 500 KB partition.

150 KB process allocated to the 450 KB partition.

400 KB process allocated to the 420 KB partition.

375 KB would not be allowed to allocate as no partition is large enough!

14. Logical memory size : 32K words. $= 2^{15}$
Physical memory size ! 512 words.

Page size : 64 words.

No. of pages in logical memory =
Logical memory size / Page size

$$= \frac{32K}{64} \Rightarrow \frac{32 \times 2^{10}}{64} \Rightarrow \frac{2^{15}}{2^{6}} \Rightarrow 2^{9}$$

$\Rightarrow$ 512 pages.

No. of frames in physical memory =
Physical memory size / Page size

$= 512/64 \Rightarrow 8$ frames

No. of pages = No. of entries in the table

No. of entries in the page table = 512

$\longleftarrow 15 \longrightarrow$

p	d
9	6

$\longleftarrow 9 \longrightarrow$

f	d
3	6

Logical memory space $= 2^{15} = 32768$ words

Physical memory space $= 2^{9} = 512$ words.

Page No.
Date

12. No. of pages = 64

No. of frames = 32

No. of words each page contains = 1024.

Therefore, the logical address space has a total of $64 \times 1024 \Rightarrow 65536$ words.

a) Logical memory = 64 pages $= 2^6$ pages.

Size of each word = $1024 = 2^{10}$.

Hence total logical memory $= 2^6 \times 2^{10} = 2^{16}$

Hence the logical address will have 16 bits.

b) Physical memory = 32 frames $= 2^5$ frames

Size of each word = $1024 = 2^{10}$

Hence total physical size $= 2^5 \times 2^{10} = 2^{15}$

Hence there will be 15 bits in the physical address

Page No.
Date

9) page table contains 64 entries i.e 64 pages. of 11 bits (including valid/invalid bit) each.

page size of 512 bytes.

a) No. of pages = 64 = 2^6 pages

6 bits of logical address space are required to specify the page no.

b) page size = 512 = 2^9 bytes i.e 9 bits are required to specify the offset within the page.

c) 15.

The 6 MSB of the logical address would specify the page no., the 9 LSB would specify the page offset, making the total no. of bits 15.

d) 15 bit address creates an address space of 2^{15} = 32,768.

e) A page table entry contains the frame no. and a valid/invalid bit. If the total no. of bits is 11, the no. of bits in a page frame no. are 10.

Page No.
Date

f) The page frame offset is the same as the page offset, a 9-bit value

g) 19
The 10 MSB of the physical address would specify the page frame no., the 9 LSB would specify the page offset, making the total no. of bits 19.

h) A 19-bit address creates an address space of 2^{19}.

11) Logical memory space $= 2^{36}$

page size $= 8K \Rightarrow 2^3 \times 2^{10} \Rightarrow 2^{13}$

a) No. of pages $= \dfrac{2^{36}}{2^{13}} \Rightarrow 2^{23}$

b) page table entry = 4 bytes i.e 8×4 bits

i.e $\underbrace{32 \text{ bits}}_{f}$ or 2^5

& d is 13

So, physical address $= 2^{45}$

Page No.
Date

8) a) Physical memory = 2^{24} bytes

256 pages of logical address space.

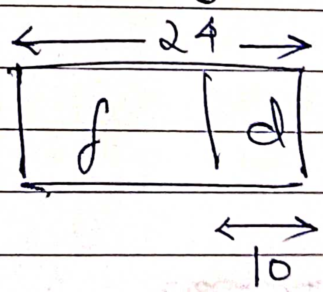
page size = 2^{10} bytes.

So, Logical address space = $2^{10} \cdot 2^{8} = 2^{18}$ byte

b) The page frame size is the same as the page size.

i.e the page frame size is 2^{10} bytes.

c)



Page frame no. Bits = 14

d) The page table must contain an entry for each page. Since there are 256 pages in the logical address space, the page table must be 256 entries long.

e) Each entry contains 1 bit to indicate if the page is valid and 14 bits to specify the page frame number.