Page No.
Date

v Imp

Bankers Algorithm (Safety Algorithm)

- It is deadlock avoidance algorithm.
- It handles multiple instances of same resources.

Available :- no. of available resources of each type.

Max :- maximum need of any process for any resource.

Allocation :- no. of resources allocated to each process.

Need :- is calculated based on the formula
(Max - Allocation)

Max = Allocation + Need

Allocation = Max - Need

Total Resources = Total Allocation + Available

Page No.
Date

Numerical

1)

	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0
P_1	1	0	0	0	1	7	5	0				
P_2	1	3	5	4	2	3	5	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				
	2	9	10	12								

1) Need Matrix ?
2) Is system in ~~safe syst~~ safe state? If yes, find the safe sequence

→ **Solution** Total Resources = A → 3, B → 14, C → 12, D → 12

Need = Max − Allocation

	Allocation				Max				Available				Need			
	A	B	C	D	A	B	C	D	A	B	C	D	A	B	C	D
✓ P_0	0	0	1	2	0	0	1	2	1	5	2	0	0	0	0	0
✓ P_1	1	0	0	0	1	7	5	0	1	5	3	2	0	7	5	0
✓ P_2	1	3	5	4	2	3	5	6	2	8	8	6	1	0	0	2
✓ P_3	0	6	3	2	0	6	5	2	2	14	11	8	0	0	2	0
✓ P_4	0	0	1	4	0	6	5	6	2	14	12	12	0	6	4	2
									3	14	12	12				

Safe sequence

P_0, P_2, P_3, P_4, P_1

∴ The system is in safe state & the safe sequence is $P_0\ P_2\ P_3\ P_4\ P_1$

2) Resources: A - 10 instances
B - 5 instances
C - 7 instances

Process	Allocation			Max		
	A	B	C	A	B	C
P_0	0	1	0	7	5	3
P_1	2	0	0	3	2	2
P_2	3	0	2	9	0	2
P_3	2	1	1	2	2	2
P_4	0	0	2	4	3	3
	7	2	5			

Available = Total Resources - Total Allocation

A = 10 - 7 = 3
B = 5 - 2 = 3
C = 7 - 5 = 2

Need = Max - Allocation

Process	Allocation			Max			Available			Need		
	A	B	C	A	B	C	A	B	C	A	B	C
✓ P_0	0	1	0	7	5	3	3	3	2	7	4	3
✓ P_1	2	0	0	3	2	2	5	3	2	1	2	2
✓ P_2	3	0	2	9	0	2	7	4	3	6	0	0
✓ P_3	2	1	1	2	2	2	7	4	5	0	1	1
✓ P_4	0	0	2	4	3	3	7	5	5	4	3	1
							10	5	7			

Safe Sequence (P_1 P_3 P_4 P_0 P_2)

Page No.
Date

Algorithm

n = no. of processes
m = no. of resources

Input - Processes (~~n~~)
- any 2 out of 3 (Max, Need, Allocation)
- available or total no. of resources

Step 1: flag[i] = 0 for i = 0 to (n-1) & find Need[n][m] = Max[n][m] - allocation[n][m]

Step 2: find a process P_i such that :-
flag[i] = 0 & $Need_i$ <= Available

Step 3: If such i exists then flag[i] = 1, available = available + allocation;
goto step 2

otherwise goto step 4

Step 4: if flag[i] = 1 for all i then system is in safe state otherwise unsafe state