Paper Name : Operating System

Prepared by :Debanjali Jana

Deadlock avoidance –

Banker's algorithm :

Deadlock avoidance can be done by Banker's algorithm.

Here a resource request is allowed only if it results in a safe state.

The following data structures are used to implement the algorithm.

Available :

- It is a 1-d array that tells the number of each resource type (instance of resource type) which is currently available.
- Available[R1] = 5 means there are **5** instances of resource type R1 which arecurrently available

Max :

- It is a 2-d array that defines the maximum number of each resource type required by a process for successful execution.

Allocation :

- It is a 2-d array that defines the number of resources of each type currently allocated to each process.

Need :

- It is a 2-d array that indicates the number of remaining instances of each resource type required for execution.

Consider a system with five processes P_1, P_2, P_3, P_4 and P_5 and three resources of type A, B, C. Resource type A has 10 instances, B has 5 instances and type C has 7 instances

Process	Allocation	Max	Available
	A B C	A B C	A B C
P_1	0 1 0	7 5 3	3 3 2
P_2	2 0 0	3 2 2	
P_3	3 0 2	9 0 2	
P_4	2 1 1	4 2 2	
P_5	0 0 2	5 3 3	

I. What is the content of the need matrix?
II. Is the system in safe state? If so, find the safe sequence

The content of the matrix Need is defined to be Max - Allocation and is as follows:

Process	Allocation	Max	Available	Need
	A B C	A B C	A B C	A B C
P_1	0 1 0	7 5 3	3 3 2	7 4 3
P_2	2 0 0	3 2 2		1 2 2
P_3	3 0 2	9 0 2		6 0 0
P_4	2 1 1	4 2 2		2 1 1
P_5	0 0 2	5 3 3		5 3 1
	7 2 5			

Now P_2 is able to fulfil it's needs so it gets terminated after execution
So it releases its resources also.

Process	Allocation	Max Need	Available	Need
	A B C	A B C	A B C	A B C
P_1	0 1 0	7 5 3	5 3 2	7 4 3
P_3	3 0 2	9 0 2		6 0 0
P_4	2 1 1	4 2 2		2 1 1
P_5	0 0 2	5 3 3		5 3 1

Now P_4 is able to fulfil it's needs so it gets terminated after execution
So it releases its resources also.

Process	Allocation	Max Need	Available	Need
	A B C	A B C	A B C	A B C
P_1	0 1 0	7 5 3	7 4 3	7 4 3
P_3	3 0 2	9 0 2		6 0 0
P_5	0 0 2	5 3 3		5 3 1

Now P_5 is able to fulfil it's needs so it gets terminated after execution
So it releases its resources also.

Process	Allocation	Max Need	Available	Need
	A B C	A B C	A B C	A B C
P_1	0 1 0	7 5 3	7 4 5	7 4 3
P_3	3 0 2	9 0 2		6 0 0

Now P_1 is able to fulfil it's needs so it gets terminated after execution
So it releases its resources also.

Process	Allocation	Max Need	Available	Need
	A B C	A B C	A B C	A B C
P_3	3 0 2	9 0 2	7 5 5	6 0 0

Now P_3 is able to fulfil it's needs so it gets terminated after execution
So it releases its resources also.

Now available resources become A B C
10 5 7 which is equal to the total resources

Therefore, safe sequence :

< P_2, P_4 , P_5, P_1, P_3>

The current state is safe because a safe sequence exists.

_____