Exp 7 C2 16010121110

TITLE: Simulate Bankers Algorithm for Deadlock Avoidance

AIM: Implementation of Banker's Algorithm for Deadlock Avoidance

---

Expected Outcome of Experiment:

CO 3. To understand the concepts of process synchronization and deadlock.

Books/ Journals/ Websites referred:

1. **Silberschatz A., Galvin P., Gagne G. "Operating Systems Principles", Willey Eight edition.**
2. **Achyut S. Godbole , Atul Kahate "Operating Systems" McGraw Hill Third Edition.**
3. **William Stallings, "Operating System Internal & Design Principles", Pearson.**
4. **Andrew S. Tanenbaum, "Modern Operating System", Prentice Hall.**

Pre Lab/ Prior Concepts:

Knowledge of deadlocks and all deadlock avoidance methods.

Description of the application to be implemented:

The Banker's algorithm is a resource allocation and deadlock avoidance algorithm developed by Edsger Dijkstra.

DATA STRUCTURES

(where n is the number of processes in the system and m is the number of resource types)

Available

It is a 1-d array of size ‘m’ indicating the number of available resources of each type.

Available[j] = k means there are ‘k’ instances of resource type Rj

Max

It is a 2-d array of size ‘n*m’ that defines the maximum demand of each process in a system.

Max[i, j] = k means process Pi may request at most ‘k’ instances of resource type Rj.

Allocation

It is a 2-d array of size ‘n*m’ that defines the number of resources of each type currently allocated to each process.

Allocation[i, j] = k means process Pi is currently allocated ‘k’ instances of resource type Rj

Need

It is a 2-d array of size ‘n*m’ that indicates the remaining resource need of each process.

Need [i, j] = k means process Pi currently needs ‘k’ instances of resource type Rj

Need [i, j] = Max [i, j] – Allocation [i, j]

Source - https://www.geeksforgeeks.org/bankers-algorithm-in-operating-system-2/

Implementation details:

```
C = [

[3,2,2],

[6,1,3],

[3,1,4],
```

```
[4,2,2]

]



A = [

[1,0,0],

[6,1,2],

[2,1,1],

[0,0,2]

]

R = [3,3,2] #total

V = [0,1,1] #avaiable

process = range(5)

import numpy as np

A = np.array(A)
```

```
C = np.array(C)

R = np.array(R)

V = np.array(V)

process = np.array(process)

D = C - A



while(len(D)!=0):

index = -1

terminate = True

for i in D:

index +=1

print(V)

if((i <= V).all()):

print("process ", process[index], " can continue")
```

```
V = V + A[index]

C=np.delete(C, index, axis=0)

A=np.delete(A, index, axis=0)

D=np.delete(D, index, axis=0)

process = np.delete(process,index,axis = 0)

terminate = False

if(len(process)==0):

exit

break

if(terminate ==True):

print("Non safe state")

break

continue
```

```
process 1 can continue
process 0 can continue
process 2 can continue
process 3 can continue
```

Conclusion: Thus we have implemented bankers algorithm. The bankers algorithm is an algorithm to prevent deadlock. It is a deadlock avoidance algorithm used to avoid deadlocks and to ensure safe execution of processes. The algorithm maintains a matrix of maximum and allocated resources for each process and checks if the system is in a safe state before allowing a process to request additional resources.

Post Lab Objective Questions

1) The wait-for graph is a deadlock detection algorithm that is applicable when:
   a) All resources have a single instance
   b) All resources have multiple instances
   c) Both a and b
   d) None of the above

Ans: c)

2) Resources are allocated to the process on non-sharable basis is _
   a) Hold and Wait
   b) Mutual Exclusion
   c) No pre-emption
   d) Circular Wait

Ans:b)

3) Which of the following approaches require knowledge of the system state?
   a) Deadlock Detection
   b) Deadlock Prevention
   c) Deadlock Avoidance
   d) All of the above

Ans: d)

4) Consider a system having 'm' resources of the same type. These resources are shared by 3 processes A, B, C which have peak time

demands of 3, 4, 6 respectively. The minimum value of 'm' that ensures that deadlock will never occur is

a) 11
b) 12
c) 13
d) 14

Post Lab Descriptive Questions

1. Consider a system with total of 150 units of memory allocated to three processes as shown:

Process	Max	Hold
P_1	70	45
P_2	60	40
P_3	60	15

Apply Banker's algorithm to determine whether it would be safe to grant each of the following request. If yes, indicate sequence of termination that could be possible.

SAFE STATE

1) The P_4 process arrives with max need of 60 and initial need of 25 units.
   process 0 can continue
   process 1 can continue
   process 2 can continue
   process 3 can continue

2) The P_4 process arrives with max need of 60 and initial need of 35 units.
   process 0 can continue
   process 1 can continue
   process 2 can continue
   process 3 can continue

Date: 30 oct 23 **Signature of faculty in-charge**