Q0)

Given the following state for the Banker’s Algorithm. Suppose we have 3 resources (R1-3) and 3 processes (P1-3).

R1 = Scanner

R2 = Printer

R3 = CD

Given available vector R1=2, R2 = 1, R3 = 2 Then we have the following situation:

• Snapshot at time T0:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Current Allocation | | | Max Allocation | | |
|  | R1 | R2 | R3 | R1 | R2 | R3 |
| P1  P2  P3 | 0  1  3 | 2  2  0 | 0  1  3 | 6  3  5 | 5  2  2 | 1  2  3 |

a) Calculate the original resources

original resources : P1 P2 P3

14 9 6

**b)** Calculate the Need matrix

Need [i] = Max [i] - Allocation [i]  
Need for P1: (6, 5, 1) - (0, 2, 0) = 6, 3, 1  
Need for P2: (3, 2, 2) - (1, 2, 1) = 2, 0, 1  
Need for P3: (5, 2, 3) - (3, 0, 3) = 2, 2, 0

**c)** Is the system in a safe state? If so, show one sequence of processes which allows the system to complete. If not, explain why. **Show your computation step-by-step.**

**Step 1:** For Process P1:

Need <= Available

6, 3, 1 <= 3, 3, 2 condition is **false**.

**So, we examine another process, P2.**

**Step 2:** For Process P2:

Need <= Available

1, 2, 2 <= 3, 3, 2 condition **true**

New available = available + Allocation

(3, 3, 2) + (2, 0, 0) => 5, 3, 2

**Similarly, we examine another process P3.**

**Step 3:** For Process P3:

P3 Need <= Available

2, 0, 2 < = 5, 3, 2 condition is **false**.

New available resource = Available + Allocation

1, 2, 3 + 5, 3, 2 => 6, 5, 5

Now, we again examine each type of resource request for processes P1 and P3.

**Hence, we execute the banker's algorithm to find the safe state and the safe sequence like P2, P4, P5, P1 and P3.**

d) Process P1 request (1,1,3). Should this request be granted? Why or why not? Show your computation step by step

**Q1**) Given the following state for the Banker’s Algorithm. Suppose we have 3 resources (R1-3) and 3 processes (P1-3).

Maximum resources R1=4, R2=5, R3=6. Then we have the following situation:

• Snapshot at time T0:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Current Allocation | | | Request | | |
|  | R1 | R2 | R3 | R1 | R2 | R3 |
| P1  P2  P3 | 1  2  0 | 1  1  1 | 3  0  1 | 0  1  0 | 4  1  1 | 3  1  0 |

**a)** Calculate the available vector.

Available A B C

1 2 0

**b)** Calculate the Need matrix.

Need A B C

P0 5 6 2

P1 1 1 0

P2 2 3 3

P3 0 1 1

P4 3 5 2

**c)** Is the system in a safe state? If so, show one sequence of processes which allows the system to complete. If not, explain why. **Show your computation step-by-step.**

1. Initialize the Work and Finish vectors.

Work = Available = (1, 2, 0)

Finish = (false, false, false, false, false)

2. Find index i such that Finish[i] = false and Needi <= Work

i Work = Work + Allocationi Finish

1 (1, 2, 0) + (1, 1, 2) = (2, 3, 2) (false, true, false, false, false)

3 (2, 3, 2) + (2, 1, 1) = (4, 4, 3) (false, true, false, true, false)

2 (4, 4, 3) + (0, 3, 0) = (4, 7, 3) (false, true, true, true, false)

4 (4, 7, 3) + (1, 1, 1) = (5, 8, 4) (false, true, true, true, true)

0 (5, 8, 4) + (1, 1, 1) = (6, 9, 5) (true, true, true, true, true)

3. Since Finish[i] = true for all i, hence the system is in a safe state. The sequence of

processes which allows the system to complete is P1, P3, P2, P4, P0.

**d)** P1 need 2 more R2. Should this request be granted? Why or why not? Show your computation step by step

edit.

1. Check that Request2 <= Need2.

Since (1, 2, 0) <= (2, 3, 3), hence, this condition is satisfied.

2. Check that Request2 <= Available.

Since (1, 2, 0) <= (1, 2, 0), hence, this condition is satisfied.

3. Modify the system’s state as follows:

Available = Available – Request2 = (1, 2, 0) – (1, 2, 0) = (0, 0, 0)

Allocation2 = Allocation2 + Request2 = (0, 3, 0) + (1, 2, 0) = (1, 5, 0)

Need2 = Need2 – Request2 = (2, 3, 3) – (1, 2, 0) = (1, 1, 3)

4. Apply the safety algorithm to check if granting this request leaves the system in a

safe state.

1. Initialize the Work and Finish vectors.

Work = Available = (0, 0, 0)

Finish = (false, false, false, false, false)

2. At this point, there does not exist an index i such that Finish[i] = false and

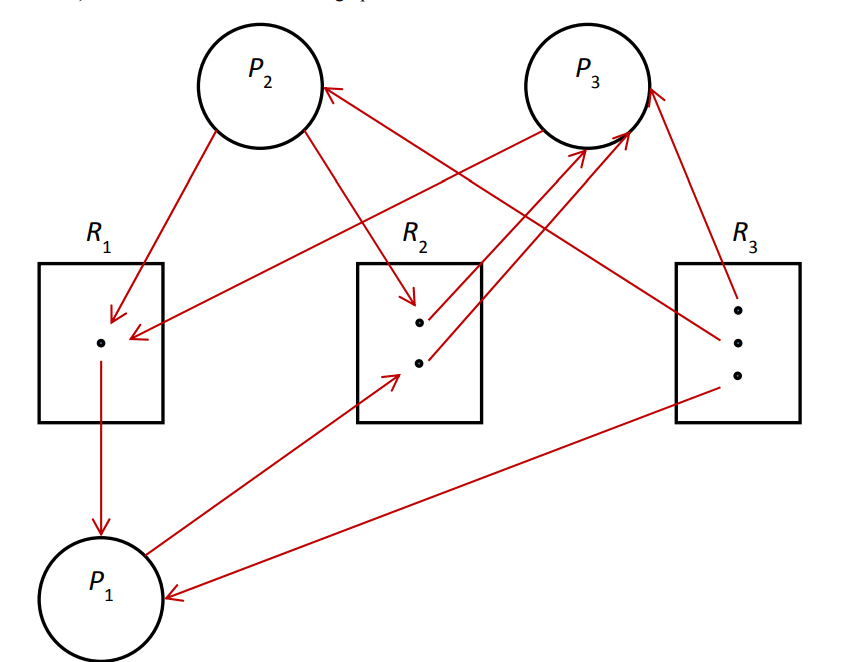
Needi <= Work.

Since Finish[i] ≠ true for all i, hence the system is not in a safe state.

Therefore, this request from process P2 should not be granted.

**Q2)** A system has three processes (P1, P2, P3) and three reusable resources (R1, R2, R3). There is one instance of R1, two instances of R2 and three instances of R3. P1 holds an R1 and an R3 and is requesting an R2. P2 holds an R3 and is requesting an R1 and an R2. P3 holds two R2 and an R3 and is requesting an R1.

**a)** Draw the resource allocation graph for this situation.



**b)** Write all the cycle(s) in the graph.

P3->R1->P1->R2(1st instance)->P3

P3->R1->P1->R2(2nd instance)->P3

**c)** Does a deadlock exist? Why?

Yes, a deadlock does exist.

P3 is waiting for an instance of R1, the only instance of which is held by P1 and P1 is

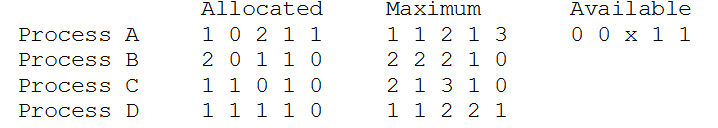
waiting for an instance of R2, both instances of which are held by P3. Since neither an

instance of R1 nor that of R2 will ever become available, hence, neither P1 nor P3 will

ever be able to continue executing. Therefore, a deadlock exists.

**Q3)**

**A system has four processes and five allocatable resources. The current allocation and maximum needs are as follows:**

****

**What is the smallest value of *x* for which this is a safe state?**

Start with x = 0 and make Need table

it will not satisfy any process in first step

then set x=1

it will satisfy process D need but in second step it will not able satisfy any process need

set x=2

This value will satisfy need with sequence of Process D, C, B, A

Answer should be x=2