Deadlocks (4)

Dr. Jun Zheng

CSE325 Principles of Operating Systems

10/14/2019

NEW MEXICO TECH
SCIENCE · ENGINEERING · RESEARCH · UNIVERSITY

Example of Banker’ s Algorithm

5 processes P_0 through P_4;

3 resource types:

A (10 instances), B (5instances), and C (7 instances)

Snapshot at time T_0:

	Allocation	Max	Available
	A B C	A B C	A B C
P_0	0 1 0	7 5 3	3 3 2
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	

Example (Cont.)

The content of the matrix ***Need*** is defined to be ***Max – Allocation***

	Need
	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

The system is in a safe state since the sequence < P_1, P_3, P_4, P_2, P_0> satisfies safety criteria

Example: P_1 Request (1,0,2)

Check that Request ≤ Available (that is, (1,0,2) ≤ (3,3,2) ⇒ true

	Allocation	*Need*	*Available*
	A B C	*A B C*	*A B C*
P_0	0 1 0	7 4 3	2 3 0
P_1	3 0 2	0 2 0	
P_2	3 0 2	6 0 0	
P_3	2 1 1	0 1 1	
P_4	0 0 2	4 3 1	

Executing safety algorithm shows that sequence < P_1, P_3, P_4, P_0, P_2> satisfies safety requirement

Can request for (3,3,0) by P_4 be granted?

Can request for (0,2,0) by P_0 be granted?

NEW MEXICO TECH
SCIENCE · ENGINEERING · RESEARCH · UNIVERSITY

In-Class Work 4

Consider the following snapshot of a system:

	Allocation	***Max***	***Available***
	ABCD	*ABCD*	*ABCD*
P0	0012	0012	1520
P1	1000	1750	
P2	1354	2356	
P3	0632	0652	
P4	0014	0656	

Answer the following questions using the banker's algorithm:

a. What is the content of the matrix ***Need***?

b. Is the system in a safe state?

c. If a request from process P1 arrives for (0,4,2,0), can the request be granted immediately?

NEW MEXICO TECH
SCIENCE · ENGINEERING · RESEARCH · UNIVERSITY

Answer

a. The values of **Need** for processes *P*0 through *P*4 respectively are (0, 0, 0, 0), (0, 7, 5, 0), (1, 0, 0, 2), (0, 0, 2, 0), and (0, 6, 4, 2).

b. Yes. With **Available** being equal to (1, 5, 2, 0), either process *P*0 or *P*3 could run. Once process *P*3 runs, it releases its resources, which allow all other existing processes to run.

c. This results in the value of **Available** being (1, 1, 0, 0). One ordering of processes that can finish is *P*0, *P*2, *P*3, *P*1, and *P*4.

Deadlock Detection

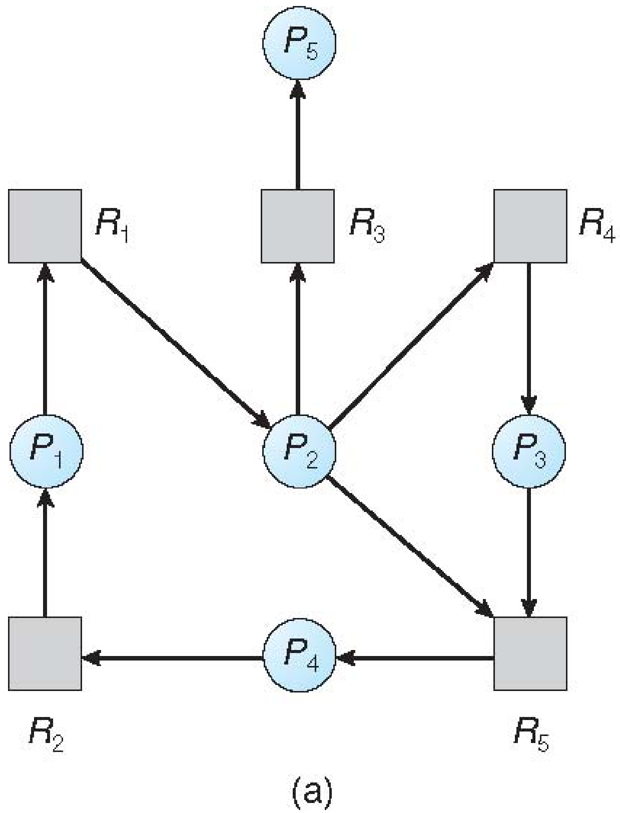
- Allow system to enter deadlock state
- Detection algorithm
- Recovery scheme

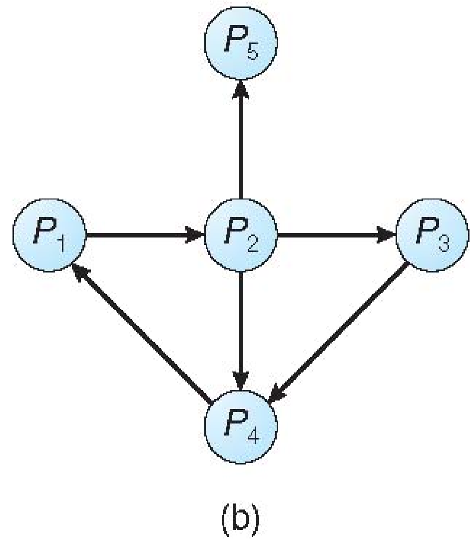
Single Instance of Each Resource Type

- Maintain **wait-for** graph
  - Nodes are processes
  - $P_i \rightarrow P_j$ if P_i is waiting for P_j
- Periodically invoke an algorithm that searches for a cycle in the graph. If there is a cycle, there exists a deadlock
- An algorithm to detect a cycle in a graph requires an order of n^2 operations, where n is the number of vertices in the graph

NEW MEXICO TECH
SCIENCE · ENGINEERING · RESEARCH · UNIVERSITY

Resource-Allocation Graph and Wait-for Graph





Resource-Allocation Graph Corresponding wait-for graph

Several Instances of a Resource Type

- **Available**: A vector of length ***m*** indicates the number of available resources of each type
- **Allocation**: An ***n* x *m*** matrix defines the number of resources of each type currently allocated to each process
- **Request**: An ***n* x *m*** matrix indicates the current request of each process. If ***Request* [*i*][*j*] = *k***, then process P_i is requesting ***k*** more instances of resource type R_j.

Detection Algorithm

1. Let ***Work*** and ***Finish*** be vectors of length ***m*** and ***n***, respectively Initialize:
   - (a) ***Work = Available***
   - (b) For ***i* = 1,2, …, *n***, if $\boldsymbol{Allocation_i \neq 0}$, then ***Finish*[i] = *false***; otherwise, ***Finish*[i] = *true***

2. Find an index ***i*** such that both:
   - (a) ***Finish[i] == false***
   - (b) $\boldsymbol{Request_i \leq Work}$

   - If no such ***i*** exists, go to step 4

NEW MEXICO TECH
SCIENCE · ENGINEERING · RESEARCH · UNIVERSITY

Detection Algorithm (Cont.)

3. $\boldsymbol{Work = Work + Allocation_i}$
$\boldsymbol{Finish[i] = true}$
go to step 2

4. If ***Finish[i] == false***, for some $\boldsymbol{i}$, $1 \leq \boldsymbol{i} \leq \boldsymbol{n}$, then the system is in deadlock state. Moreover, if ***Finish[i] == false***, then $\boldsymbol{P_i}$ is deadlocked

- **Algorithm requires an order of O(m x n^2) operations to detect whether the system is in deadlocked state**

Example of Detection Algorithm

Five processes P_0 through P_4; three resource types A (7 instances), B (2 instances), and C (6 instances)

Snapshot at time T_0:

	Allocation	*Request*	*Available*
	$A\ B\ C$	$A\ B\ C$	$A\ B\ C$
P_0	0 1 0	0 0 0	0 0 0
P_1	2 0 0	2 0 2	
P_2	3 0 3	0 0 0	
P_3	2 1 1	1 0 0	
P_4	0 0 2	0 0 2	

Sequence <P_0, P_2, P_3, P_1, P_4> will result in ***Finish[i]* = *true*** for all ***i***