SNS COLLEGE OF TECHNOLOGY

Coimbatore-35.
An Autonomous Institution

Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

COURSE NAME : 23CST202 – OPERATING SYSTEMS

II YEAR/ IV SEMESTER

UNIT – II PROCESS SCHEDULING AND SYNCHRONIZATION

Topic: Deadlock Detection and Recovery

Dr.V.Savitha
Associate Professor
Department of Computer Science and Engineering

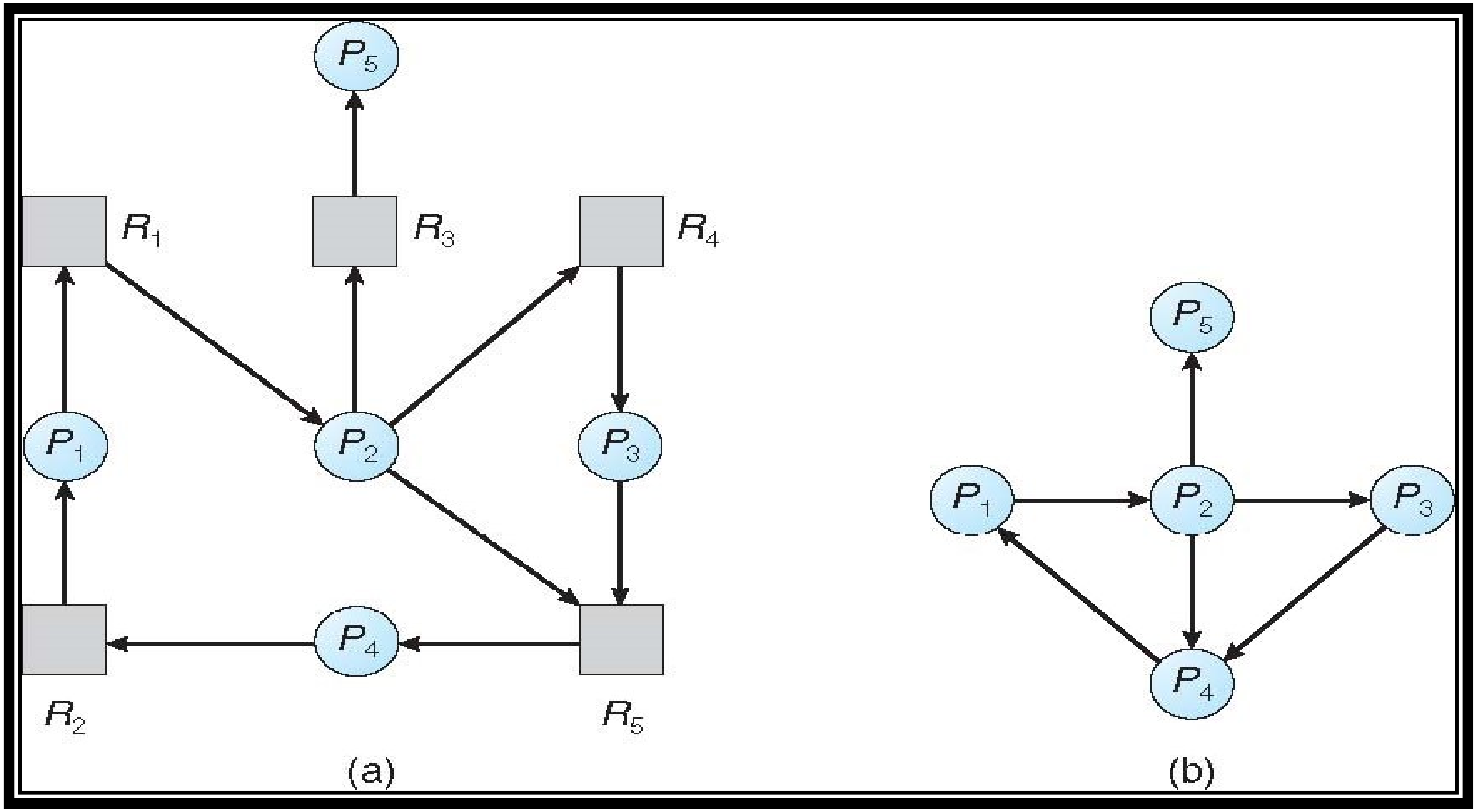
Deadlock Detection

- 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

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 $\boldsymbol{Allocation_i} \neq \mathbf{0}$, 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

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

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

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***

Example (Cont.)

- P_2 requests **an additional instance of type *C***

	Request
	A B C
P_0	0 0 0
P_1	2 0 2
P_2	0 0 1
P_3	1 0 0
P_4	0 0 2

- **State of system?**
  - Can reclaim resources held by process P_0, but insufficient resources to fulfill other processes; requests
  - Deadlock exists, consisting of processes P_1, P_2, P_3, and P_4

Recovery from Deadlock

- Process Termination
- Resource Preemption

Recovery from Deadlock: Process Termination

- Abort all deadlocked processes
- Abort one process at a time until the deadlock cycle is eliminated

- In which order should we choose to abort?
  1. Priority of the process
  2. How long process has computed, and how much longer to completion
  3. Resources the process has used
  4. Resources process needs to complete
  5. How many processes will need to be terminated
  6. Is process interactive or batch?

Recovery from Deadlock: Resource Preemption

- **Selecting a victim** – minimize cost
- **Rollback** – return to some safe state, restart process for that state
- **Starvation** – same process may always be picked as victim, include number of rollback in cost factor