Information Technology Institute

Operating System Fundamentals

Chapter Six

DEADLOCKS

Table of Content

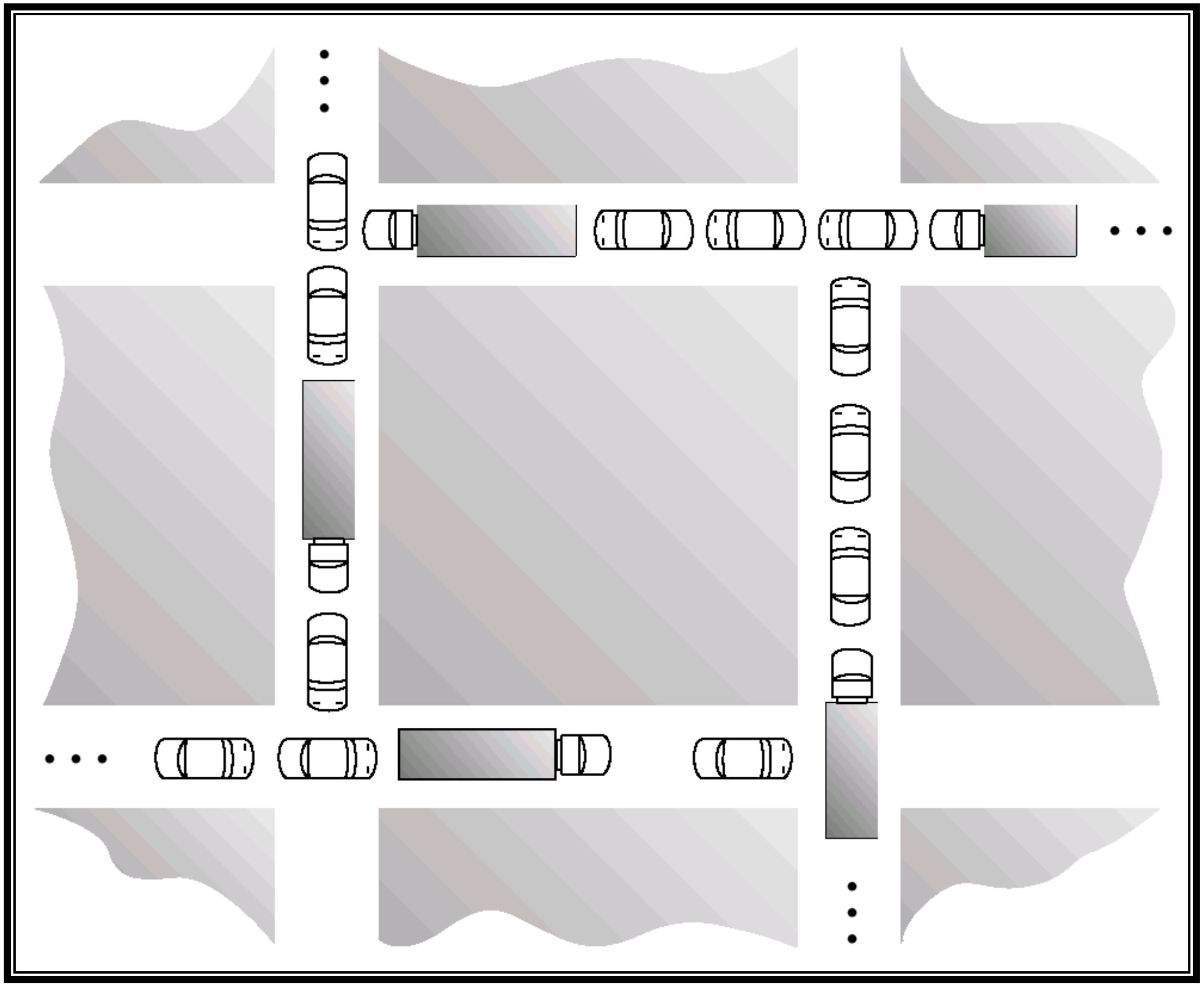
INTRODUCTION

The Deadlock Problem

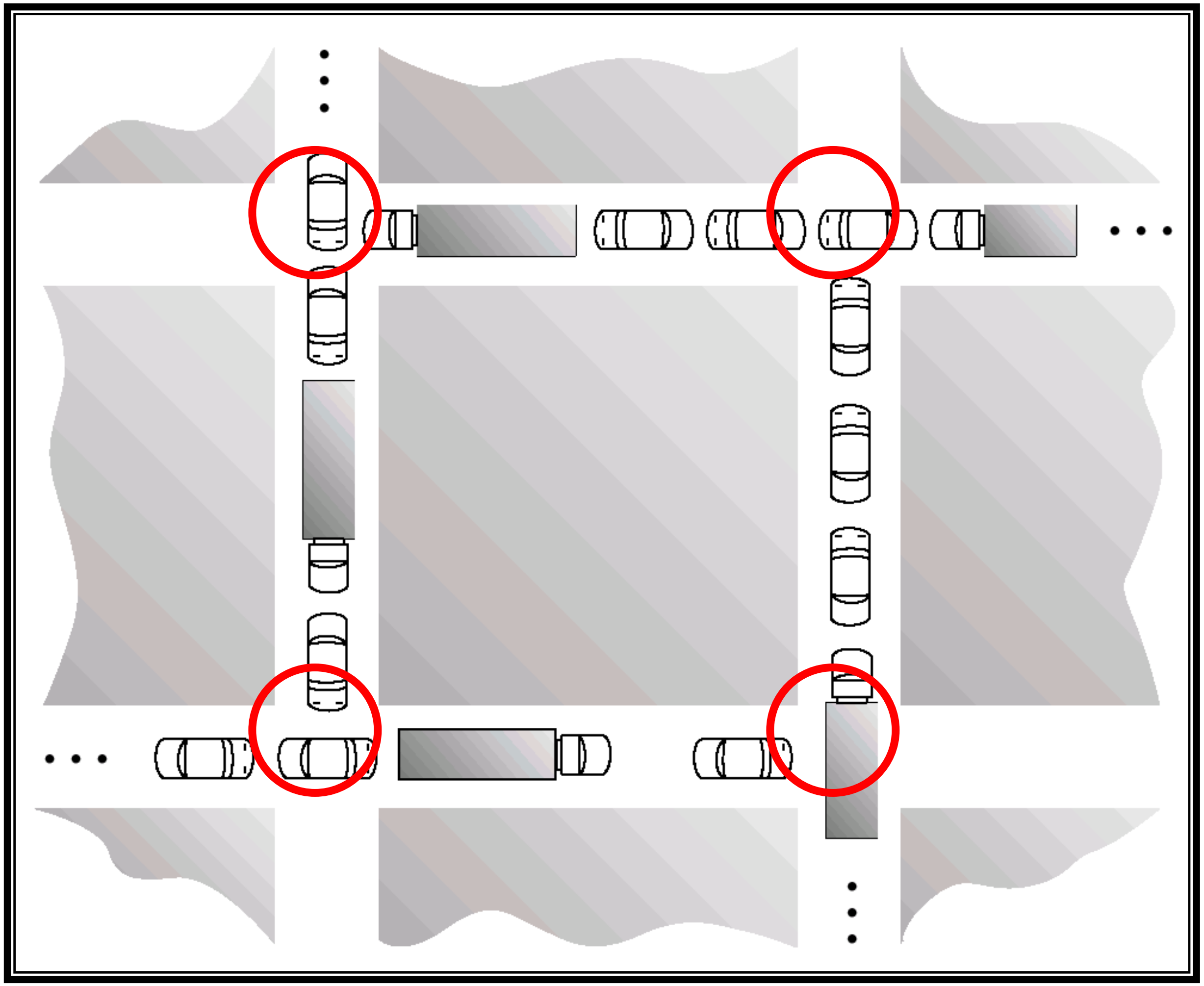
- A set of blocked processes each holding a resource and waiting to acquire a resource held by another process in the set.
- Example
  - System has 2 tape drives.
  - P1 and P2 each hold one tape drive and each needs another one.
  - semaphores A and B, initialized to 1

P1	P2
wait (A);	wait(B)
wait (B);	wait(A)

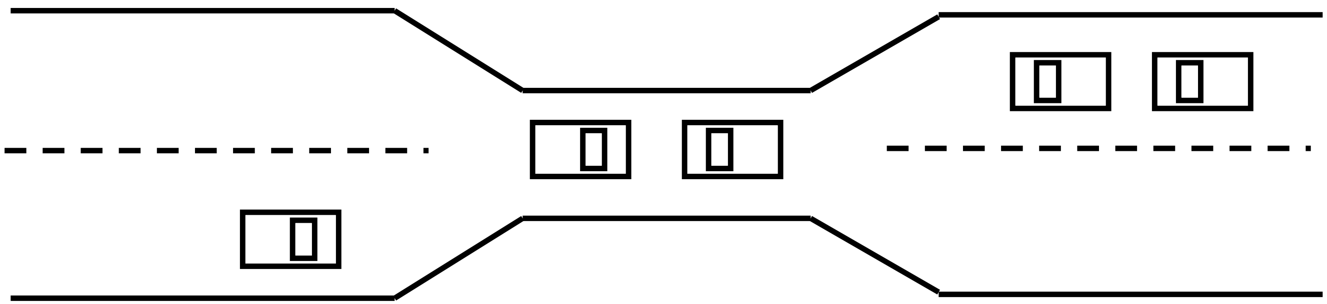
Is this Deadlock?



Yes, How to prevent it?



Bridge Crossing Example



- Traffic only in one direction.
- Each section of a bridge can be viewed as a resource.
- If a deadlock occurs, it can be resolved if one car backs up (preempt resources and rollback).
- Several cars may have to be backed up if a deadlock occurs.
- Starvation is possible

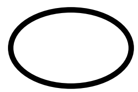
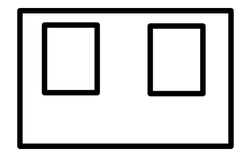
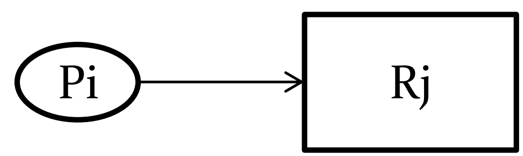
DEADLOCK CHARACTERIZATION

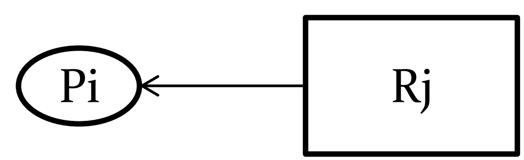
Deadlock Characterization

- Deadlock can arise if four conditions hold simultaneously.

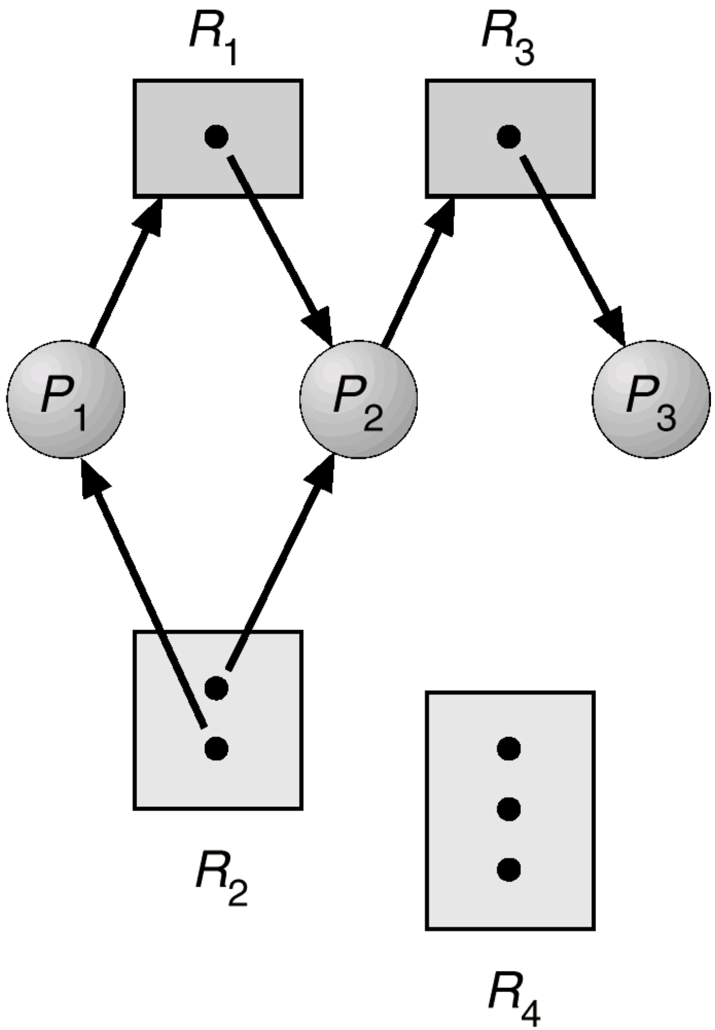
1. Mutual exclusion: only one process at a time can use a resource.
2. Hold and wait: a process holding at least one resource is waiting to acquire additional resources held by other processes.
3. No preemption: a resource can be released only voluntarily by the process holding it, after that process has completed its task.
4. Circular wait: there exists a set {P0, P1, …, P0} of waiting processes such that P0 is waiting for a resource that is held by P1, P1 is waiting for a resource that is held by P2, …, Pn–1 is waiting for a resource that is held by Pn, and Pn is waiting for a resource that is held by P0

Resource-Allocation Graph

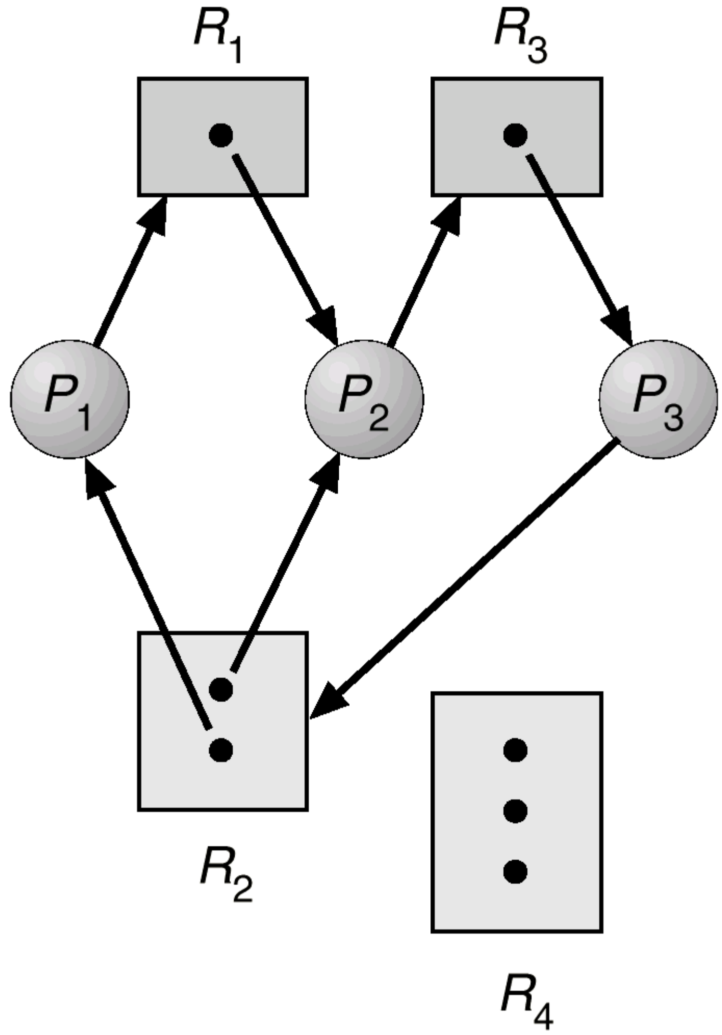
- Process 
- Resource Type with 2 instances 
- *Pi* requests instance of *Rj* 

- *Pi* is holding an instance of *Rj* 

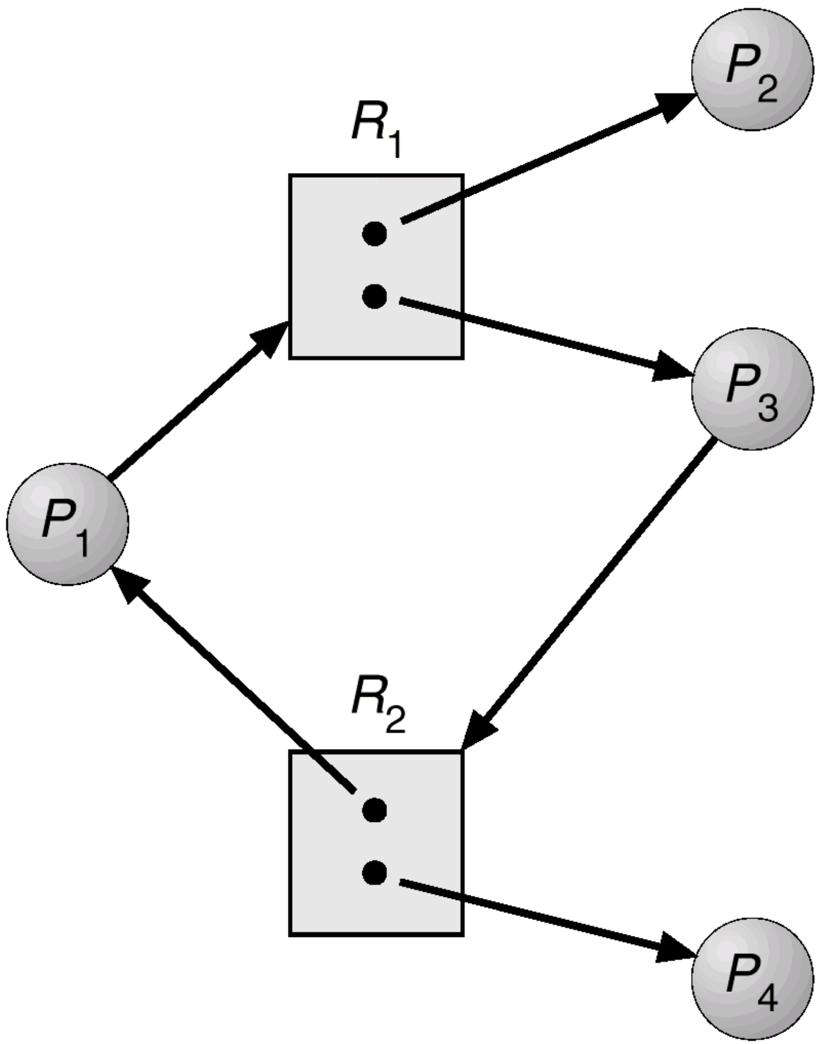
Example of a Resource Allocation Graph



Resource Allocation Graph With A Deadlock



Resource Allocation Graph With A Cycle But No Deadlock



Basic Facts

- If graph contains no cycles no deadlock.
- If graph contains a cycle
  - if only one instance per resource type, then deadlock.
  - if several instances per resource type, possibility of deadlock.

METHODS FOR HANDLING DEADLOCKS

Methods for Handling Deadlocks

- Ensure that the system will *never* enter a deadlock state.
- Allow the system to enter a deadlock state and then recover.
- Ignore the problem and pretend that deadlocks never occur in the system; used by most operating systems.

DEADLOCK PREVENTION

Deadlock Prevention

- Mutual Exclusion
  - Not required for sharable resources; must hold for non-sharable resources.
- Hold and Wait
  - must guarantee that whenever a process requests a resource, it does not hold any other resources.
    - Require process to request and be allocated all its resources before it begins execution, or allow process to request resources only when the process has none.
    - Low resource utilization; starvation possible.

Deadlock Prevention Cont'd

- No Preemption
  - If a process that is holding some resources requests another resource that cannot be immediately allocated to it, then all resources currently being held are released.
  - Preempted resources are added to the list of resources for which the process is waiting.
  - Process will be restarted only when it can regain its old resources, as well as the new ones that it is requesting.
- Circular Wait
  - Impose a total ordering of all resource types, and require that each process requests resources in an increasing order of enumeration.

RECOVERY FROM DEADLOCK

Recovery from Deadlock

- Abort all deadlocked processes.
- Abort one process at a time until the deadlock cycle is eliminated.
- In which order should we choose to abort?
  - Priority of the process.
  - How long process has computed, and how much longer to completion.
  - Resources the process has used.
  - Resources process needs to complete.
  - How many processes will need to be terminated.
  - 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.


Thank you

