

IE 411: Operating Systems

Deadlock detection

Managing deadlocks

- Deadlock detection algorithms find instances of deadlock and try to recover
- Admit the possibility of deadlock occurring and periodically check for it


System Modeling


- Set of resource types: R_1, R_2, \dots, R_m
 - There are multiple resource of each type: e.g., 3 NICs, 4 disks
- Set of processes (or threads): P_1, P_2, \dots, P_n

System Modeling

- Set of resource types: R_1, R_2, \dots, R_m
 - There are multiple resource of each type: e.g., 3 NICs, 4 disks
- Set of processes (or threads): P_1, P_2, \dots, P_n
- Each process can:
 - Request a resource of a given type and block/wait until one resource instance of that type becomes available
 - Use a resource
 - Release a resource

Resource-allocation graph

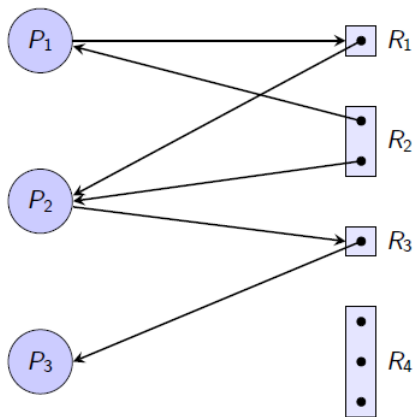
Process: 

Resource: 

Request: 

Allocation: 

Example Graph



Cycle and deadlock

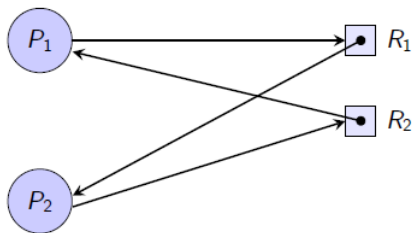
- Theorem [Holt]:
If the resource allocation graph contains no (directed) cycle, then there is no deadlock in the system
 - If cycles do exist then a deadlock is possible

Cycle and deadlock

- Theorem [Holt]:
If the resource allocation graph contains no (directed) cycle, then there is no deadlock in the system
 - If cycles do exist then a deadlock is possible
- If there is only one resource instance (black dot) per resource type then we have a stronger theorem:
The existence of a cycle is a necessary and sufficient condition for the existence of a deadlock

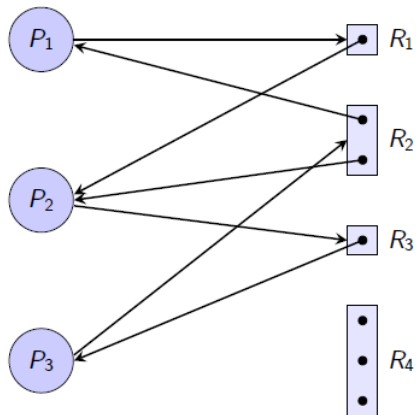
Cycle and deadlock: our 2-lock example

Clearly, there is a cycle



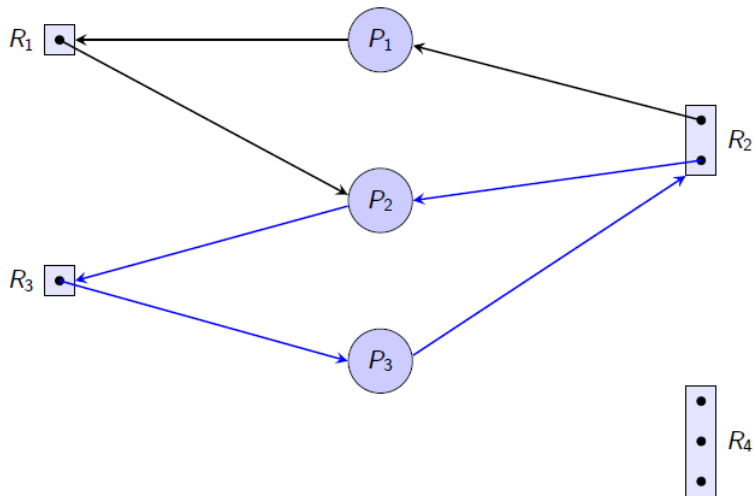
Another Example

Can you see the cycle(s)?



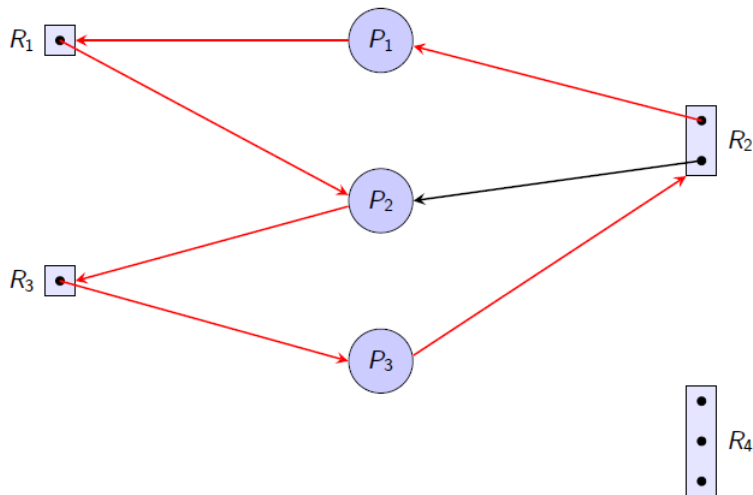
Moving vertices around

Can you see the cycle(s) now?



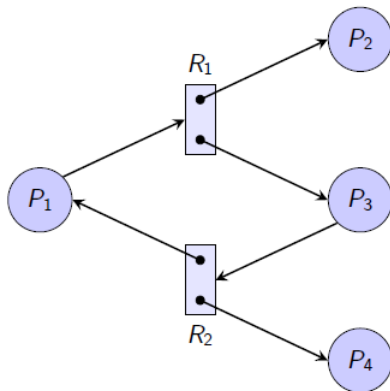
Moving vertices around

Can you see the cycle(s) now?



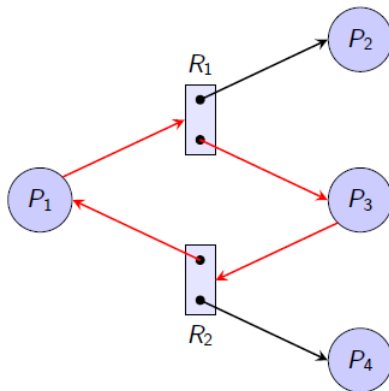
Example: Cycle and No Deadlock

There is a cycle ...



Example: Cycle and No Deadlock

There is a cycle ... but there is no deadlock

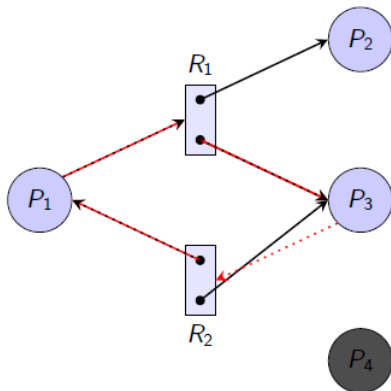


Example: Cycle and No Deadlock

When P_4 terminates it will release the instance of R_2 it locked, and that resource will be locked by P_3 .

P_3 will then be able to complete.

(Another option is that P_2 completes first.)



Rule of Thumb

- A cycle in the resource allocation graph
 - Is a necessary condition for a deadlock
 - But not a sufficient condition