CS & IT ENGINEERING

Operating System

Deadlock

Lecture -2

By- Vishvadeep Gothi sir

Recap of Previous Lecture

Topic Deadlock

Topic Deadlock Avoidance

Topic Banker's Algorithm

Topics to be Covered

Topic Banker's Algorithm

Topic Deadlock Detection

Safe $\Rightarrow$ <P1, P3, P2, P4, P5>

#Q.

Process	Allocation A B C D	Max A B C D	Available A B C D
P1	0 0 1 2	0 0 1 2	1 5 2 0
P2	1 0 0 0	1 7 5 0	
P3	1 3 5 4	2 3 5 6	
P4	0 6 3 2	0 6 5 4	
P5	0 0 1 4	0 6 5 6	

Topic : Banker's Algorithm

Process	Allocation	Max	Available	Need
	A B C	A B C	A B C	A B C
P_0	0 1 0	7 5 3	3 3 2	7 4 3
P_1	2 0 0	3 2 2		1 2 2
P_2	3 0 2	9 0 2		6 0 0
P_3	2 1 1	2 2 2		0 1 1
P_4	0 0 2	4 3 3		4 3 1

[NAT]

#Q. What will happen if process P3 requests one additional instance of resource type B ?

without checking ⇒ granted

⇓

because in safe sequence P3 is first process.

Topic : Resource Request Algorithm

m

$\rightarrow$ R.T.C. $= n^2 \times m$

no. of processes $= n$
no. of resources $= m$

1. If $Request_i <= Need_i$
   Goto step (2) ; otherwise, raise an error condition, since the process has exceeded its maximum claim.

2. If $Request_i <=$ Available m
   Goto step (3); otherwise, P_i must wait, since the resources are not available.

3. Have the system pretend to have allocated the requested resources to process P_i by modifying the state as follows:
   Available = Available – Requesti $3 * m$
   $Allocation_i = Allocation_i + Request_i$
   $Need_i = Need_i – Request_i$

4. Run safety algo $\Rightarrow n^2 \times m$

Topic : Banker's Algorithm

$Req_{P_1} = \langle 1, 0, 2 \rangle \Rightarrow$ granted

Process	Allocation	Max	Available	Need
	A B C	A B C	A B C	A B C
P_0	0 1 0	7 5 3	~~3 3 2~~ 2 3 0	7 4 3
P_1	~~2 0 0~~ 3 0 2	3 2 2	after P1	~~1 2 2~~ 0 2 0
P_2	3 0 2	9 0 2	= 5 3 2	6 0 0
P_3	2 1 1	2 2 2		0 1 1
P_4	0 0 2	4 3 3	safe	4 3 1

Topic : Banker's Algorithm

$Req_{P_1} = \langle 1, 0, 2 \rangle \Rightarrow$ granted

$Req_{P_3} = \langle 0, 1, 0 \rangle$

Process	Allocation	Max	Available	Need
	A B C	A B C	A B C	A B C
P_0	0 1 0	7 5 3	~~3 3 2~~	7 4 3
P_1	~~2 0 0~~ 3 0 2	3 2 2	~~2 3 0~~ 2 2 0	~~1 2 2~~ 0 2 0
P_2	3 0 2	9 0 2	after P1 5 2 2	6 0 0
P_3	~~2 1 1~~ 2 2 1	2 2 2	after P3 7 4 3	~~0 1 1~~ 0 0 1
P_4	0 0 2	4 3 3	⋮	4 3 1

safe

[NAT]

both req are granted together

#Q. What will happen if process P1 requests one additional instance of resource type A and two instances of resource type C ? ⇒ Req 1

What will happen if process P3 requests one additional instance of resource type B? ⇒ Req 2

Case 1 :- only req1 can be granted but req2 rejected

Case 2 :- only req2 —"— but req1 —"—

Case 3 :- Both req1 and req2 granted together

Case 4 :- Both —"— rejected

Case 5 :- Either of req1 and req2 granted but not together

Topic : Banker's Algorithm

Process	Allocation	Max	Available	Need
	A B C	A B C	A B C	A B C
P_0	0 1 0	7 5 3	3 3 2	7 4 3
P_1	2 0 0	3 2 2		1 2 2
P_2	3 0 2	9 0 2		6 0 0
P_3	2 1 1	2 2 2		0 1 1
P_4	0 0 2	4 3 3		4 3 1

[NAT]

H.W.

#Q. What will happen if process P0 requests one additional instance of resource type C ?

What will happen if process P3 requests one additional instance of resource type B and one instance of resource type C?

Topic : Deadlock Detection

1. When all resources have single instance
2. When resources have multiple instances

Topic : Deadlock Detection

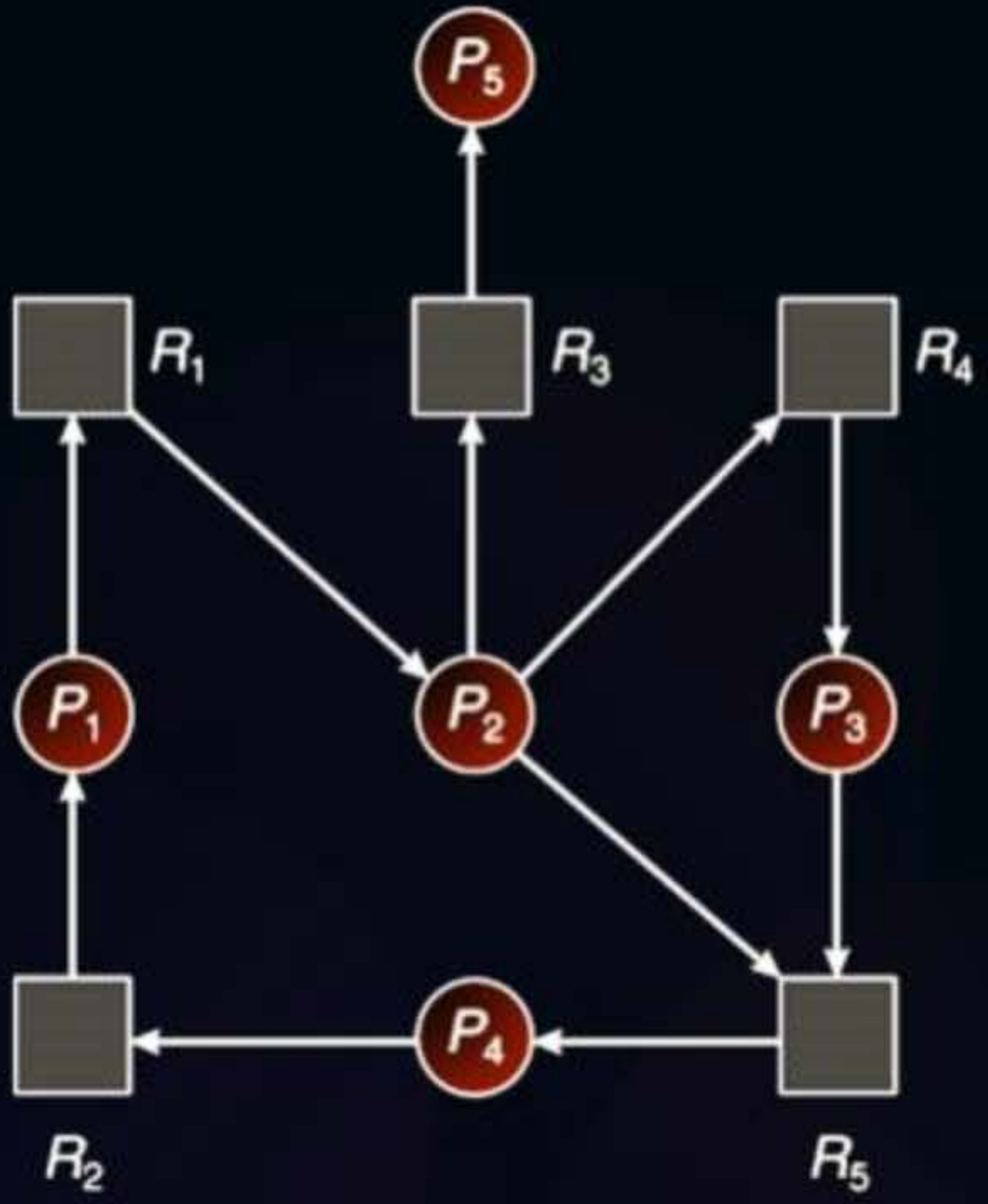
When all resources have single instance:

Deadlock detection is done using wait-for-graph

Topic : Wait For Graph

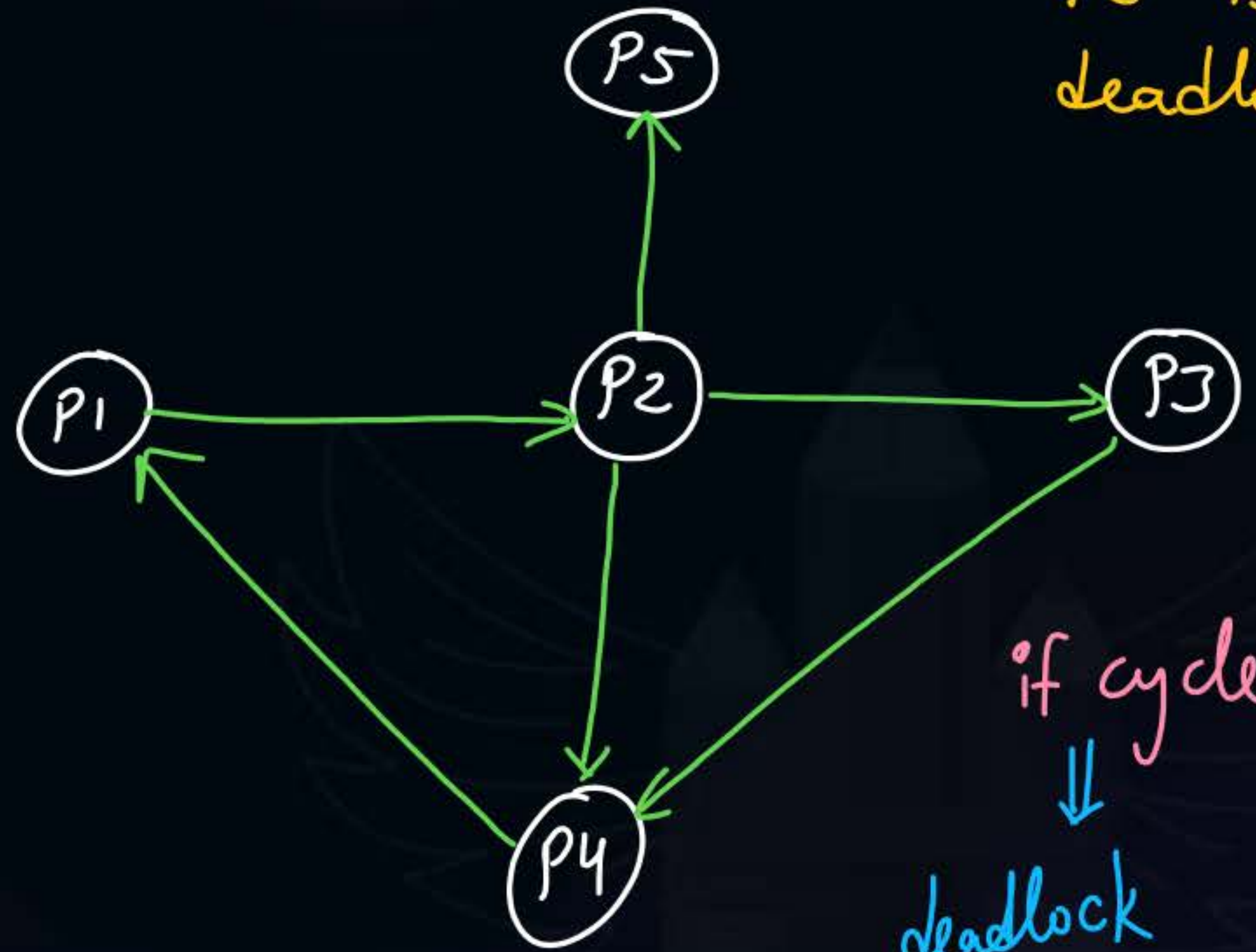
It is created from resource allocation graph

Topic : Wait For Graph



wait-for-graph :- only process

P5 is not deadlock



if cycle in graph

⇓

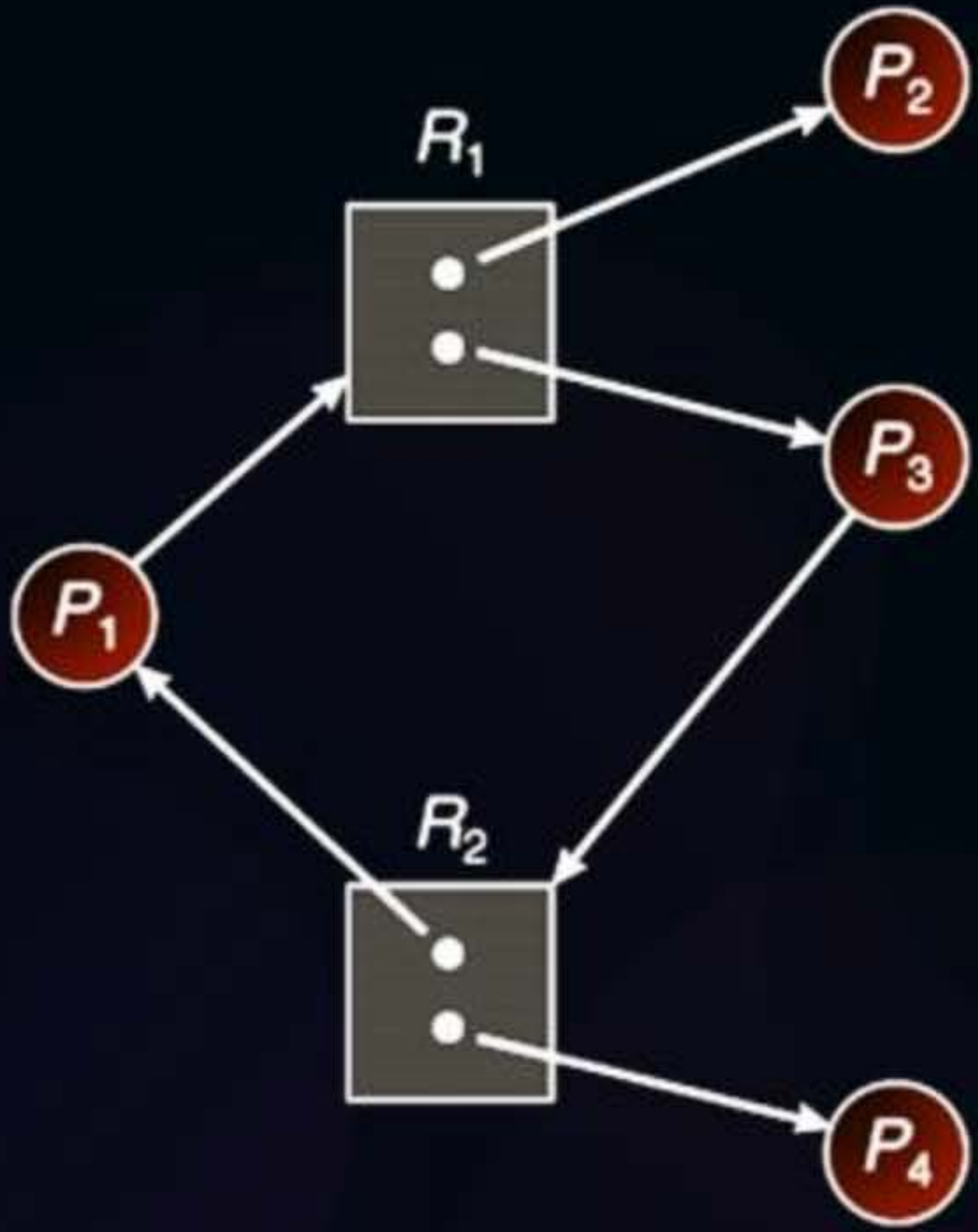
deadlock

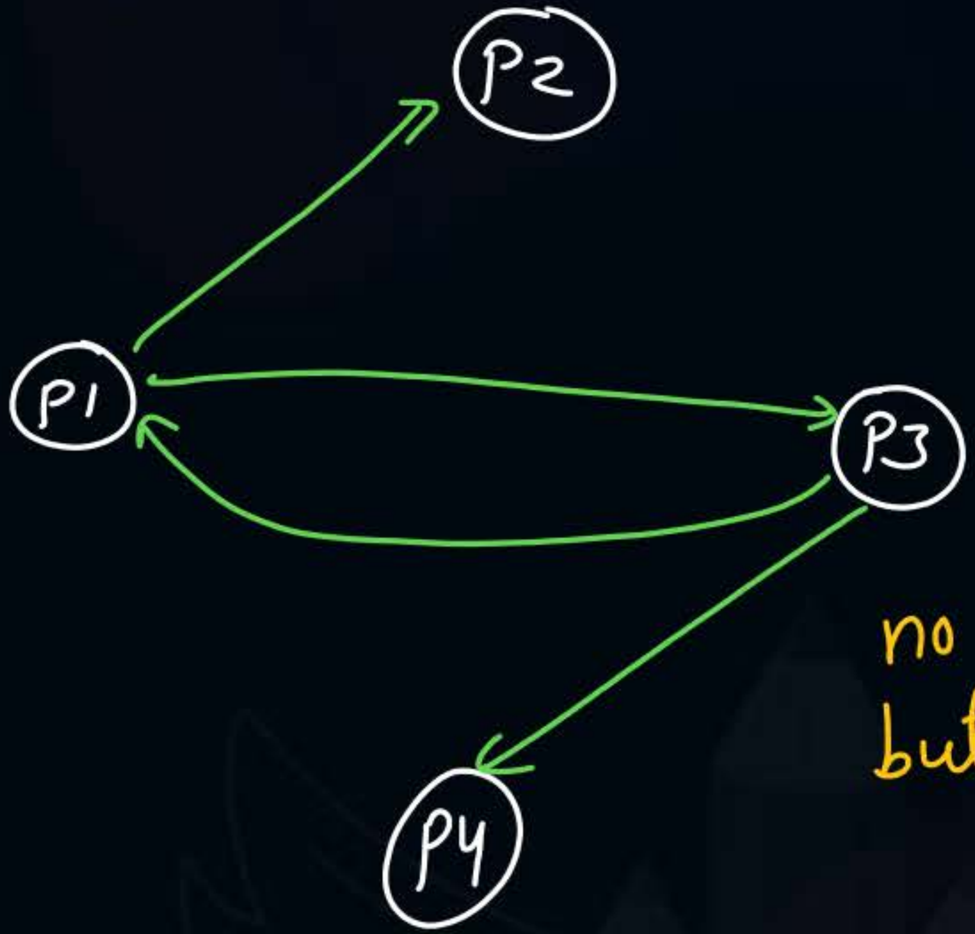
$P1 \rightarrow P2 \rightarrow P3 \rightarrow P4 \rightarrow P1$

Topic : Wait For Graph

If a resource category contains more than one instance, then the presence of a cycle in the resource-allocation graph indicates the possibility of a deadlock, but does not guarantee one.

Topic : Wait For Graph: Example





no deadlock
but only cycle

Topic : Deadlock Detection

When resources have multiple instance:

Deadlock detection is done using a specific algorithm

Topic : Deadlock Detection Algorithm

	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	After P_0 ~~0 1 0~~
P_2	3 0 3	0 0 0	After P_2 3 1 3
P_3	2 1 1	1 0 0	after P1 5 1 3 after P3 7 2 4
P_4	0 0 2	0 0 2	after P4 7 2 6

all processes executed

no deadlock

find process P_i for which $Request_i \le Available$

Ques)

	Allocation	Request	Available
P_0	1	1	1
P_1	2	5	
P_2	2	4	
P_3	4	4	
P_4	3	3	

after 2

$\Downarrow$

deadlock $\Rightarrow$ $\underbrace{P1, P2, P3, P4}_{\text{are deadlocked}}$

Topic : Detection-Algorithm Usage

When should the deadlock detection be done? Frequently, or infrequently?

time consume

Processes may be in deadlock for long time without being noticed.

Topic : Detection-Algorithm Usage

1. Do deadlock detection after every resource allocation
2. Do deadlock detection only when there is some clue

Topic : Recovery From Deadlock

There are three basic approaches to recovery from deadlock:

1. Inform the system operator and allow him/her to take manual intervention
2. Terminate one or more processes involved in the deadlock
3. Preempt resources.

Topic : Process Termination

Terminate all processes involved in the deadlock

Terminate processes one by one until the deadlock is broken

Topic : Resource Preemption

Important issues to be addressed when preempting resources to relieve deadlock:

1. Selecting a victim
2. Rollback
3. Starvation

[NAT]

#Q. Consider a system with 3 processes A, B and C. All 3 processes require 4 resources each to execute. The minimum number of resources the system should have such that deadlock can never occur?

	Requirement	Allocate (Req − 1)
A	4	3
B	4	3
C	4	3

Available

1

Ans = 3 + 3 + 3 + 1

= 10

if n processes and their max requirements are $\Rightarrow max_i$

$$\left[\sum_{i=1}^{n}(max_i - 1)\right] + 1 \leq \text{Total available resources}$$

[NAT]

#Q. Consider a system with 4 processes A, B, C and D. All 4 processes require 6 resources each to execute. The maximum number of resources the system should have such that deadlock may occur?

	allocation
A	$6-1=5$
B	$=5$
C	$=5$
D	$=5$
	20

Ans = 20

[NAT]

#Q. Consider a system with 3 processes that share 4 instances of the same resource type. Each process can request a maximum of K instances. Resource instances can be requested and released only one at a time. The largest value of K that will always avoid deadlock is ____.

$$3(k-1) + 1 \leq 4$$

$$3k - 3 \leq 3$$

$$3k \leq 6$$

$$k \leq 2$$

$$\boxed{k_{max} = 2}$$

Ques) n processes, each require 4 resources.
Total 24 resources.
max value of n for which deadlock will never occur?

$$n(4-1)+1 \le 24$$

$$3n \le 23$$

$$n \le 7.66$$

$$\boxed{n_{max} = 7}$$

Ques) In this questⁿ max value of n for which deadlock may occur?

Ans ⇒ Infinite

Ques) 4 processes.

	Allocation	max Requirement
P1	2	4
P2	1	3
P3	3	6
P4	1	5

7 allocated

3
2
5
4

no deadlock

$14 + 1 = 15$

allocated $= 7$

free $= \underline{\underline{8}}$

min freely available resources for which deadlock never occurs?

Ans = 8

Topic : Types of Locks

1. Spinlock
2. Livelock
3. Deadlock
4. Semaphores
5. Reentrant Locks

2 mins Summary

Topic Banker's Algorithm

Topic Deadlock Detection

Happy Learning

THANK - YOU