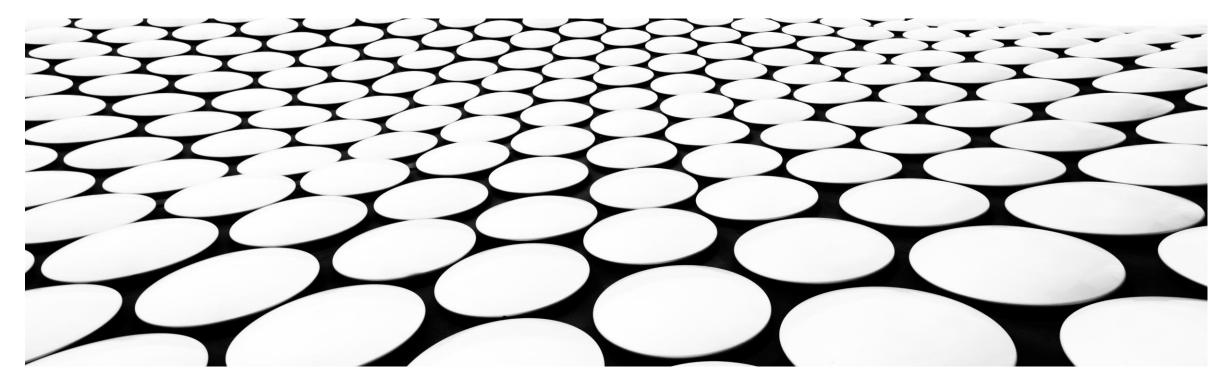
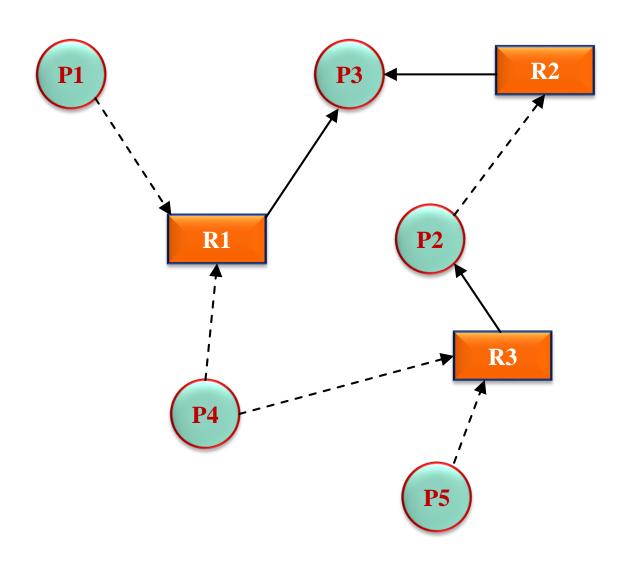
# **EXAMPLES: DEADLOCKS**

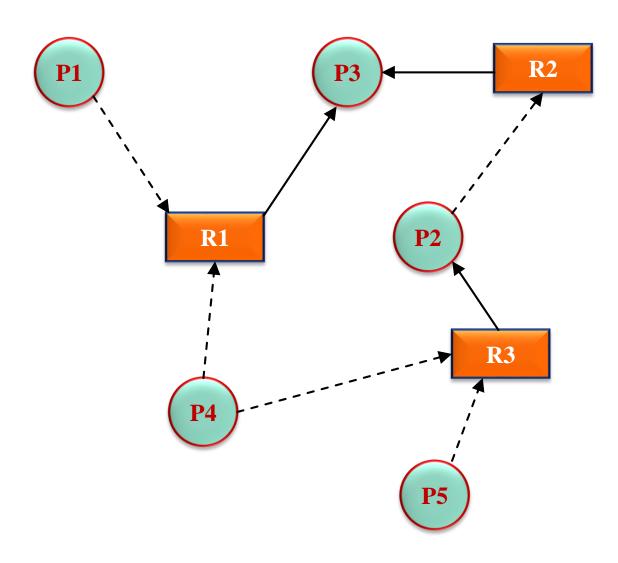
DR. ARIJIT ROY

COMPUTER SCIENCE AND ENGINEERING GROUP INDIAN INSTITUTE OF INFORMATION TECHNOLOGY

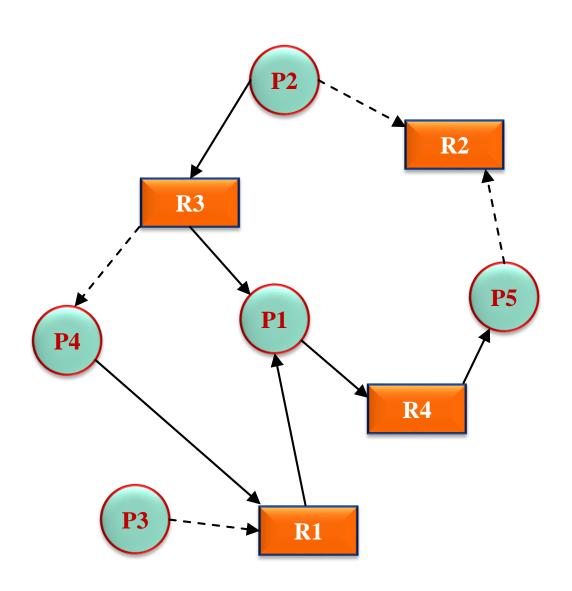


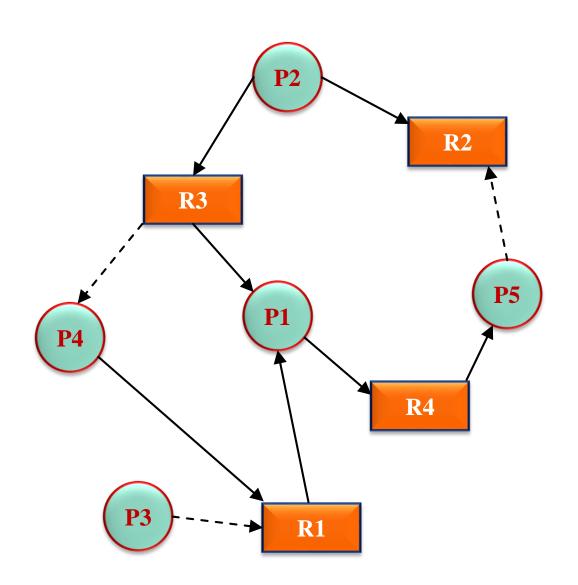




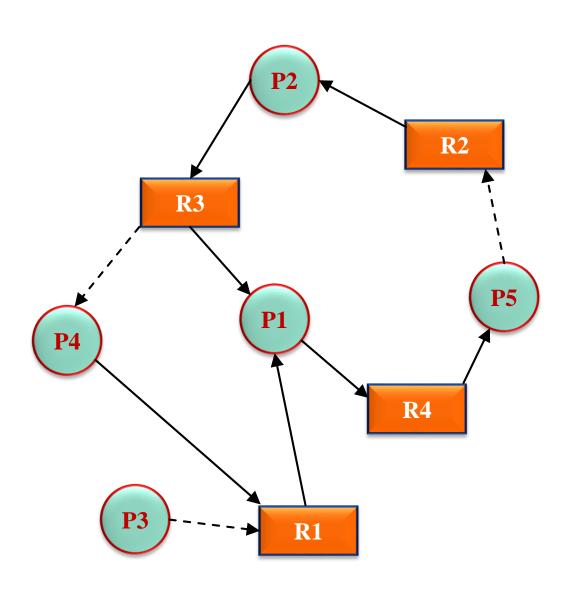


- No deadlock
- Most of the processes are independent
- Only possibility is P2, as it is holding R3 and requesting for R2 (which is holding by P3)
- P3 has already R1 and R3, so it can complete its execution and release the resources
- On releasing R2 by P3, P2 may acquire it.

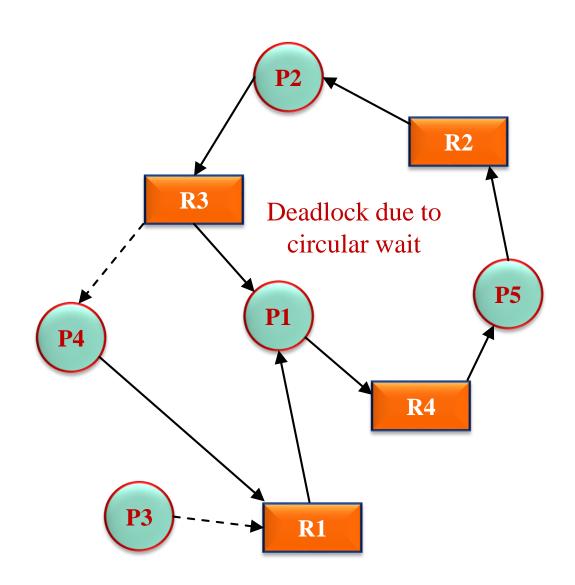




• P2 requests for R2



- P2 requests for R2
- R2 is allocated to P2



- P2 requests for R2
- R2 is allocated to P2
- P5 requests for R2

- Find out the safe sequence
- Total number of resources: 3
- A=7, B=7, C=8
- Process:  $4(P_0-P_3)$

Process	Allo	catio	n	Max	K		Ava	ailab	le
	A	В	C	A	В	C	A	В	C
P0	2	2	0	5	7	7	2	2	5
P1	0	1	2	2	2	3			
P2	3	1	0	4	2	3			
P3	0	1	1	4	5	6			

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4 (P_0 - P_3)$ 

Process	All	ocat	ion		Max		A	vaila	ble	Need			
	A	В	C	A	В	C	A	В	C	A	В	C	
P0	2	2	0	5	7	7	2	2	5	3	5	7	
P1	0	1	2	2	2	3				2	1	1	
P2	3	1	0	4	2	3				1	1	3	
P3	0	1	1	4	5	6				4	4	5	

• Calculate Need matrix

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4 (P_0 - P_3)$ 

Available resource

Process	All	Allocation			Max		A	vaila	ble	Need			
	A	В	C	A	В	C	A	В	C	A	В	C	
P0	2	2	0	5	7	7	2	2	5	3	5	7	
P1	0	1	2	2	2	3				2	1	1	
P2	3	1	0	4	2	3				1	1	3	
P3	0	1	1	4	5	6				4	4	5	

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4(P_0-P_3)$ 

Available resource

Process	Allocation				Max		A	vaila	ble	Need			
	A	В	C	A	В	C	A	В	C	A	В	C	
P0	2	2	0	5	7	7	2	2	5	3	5	7	
P1	0	1	2	2	2	3				2	1	1	
P2	3	1	0	4	2	3				1	1	3	
P3	0	1	1	4	5	6				4	4	5	

With the available resources, P1 and P2 can be served

• Total number of resources: 3

• A=7, B=7, C=8

**Process** 

P0

P1

P2

P3

• Process: 4 (P<sub>0</sub>-P<sub>3</sub>)

Max

3

3

Allocation

В

C

	1 <b>1 V</b>	ana		CSOU	
A	vaila	ble		Need	
A	В	C	A	В	C
2	2	5	3	5	7
			2	1	1
			1	1	3

Available resource

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4(P_0-P_3)$ 

Available resource

Process	Allocation				Max		A	aila	ble /	Need			
	A	В	C	A	В	C	A	В	Q	A	В	C	
P0	2	2	0	5	7	7				3	5	7	
P1	0	1	2	2	2	3	2	3	7				
P2	3	1	0	4	2	3				1	1	3	
P3	0	1	1	4	5	6				4	4	5	

Already done

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4 (P_0-P_3)$ 

Available resource

										/		
Process	All	ocat	ion		Max		A	vaila	ble		Need	
	A	В	C	A	В	C	A	В	C	A	В	C
P0	2	2	0	5	7	7				3	5	7
P1	0	1	2	2	2	3	2	3	7			
P2	3	1	0	4	2	3				1	1	3
P3	0	1	1	4	5	6				4	4	5

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4(P_0-P_3)$ 

Available resource

Process	All	Allocation			Max		A	vaila	ble	Need			
	A	В	C	A	В	C	A	В	C	A	В	C	
P0	2	2	0	5	7	7				3	5	7	
P1	0	1	2	2	2	3							1
P2	3	1	0	4	2	3	5	4	7				1
P3	0	1	1	4	5	6				4	4	5	

Already done

Already done

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4 (P_0-P_3)$  Available resource

	(-	0 -	3/										
Process	All	ocat	ion		Max	<u> </u>	A	vaila	ble		Need	1	
	A	В	C	A	В	C	A	В	C	A	В	C	
P0	2	2	0	5	7	7				3	5	7	
P1	0	1	2	2	2	3							Already done
P2	3	1	0	4	2	3	5	4	7				Already done
P3	0	1	1	4	5	6				4	4	5	Next we can allocate to P3
													<p1, p2,="" p3=""></p1,>

• Total number of resources: 3

• A=7, B=7, C=8

• Process:  $4 (P_0-P_3)$ 

Available resource

Process	Allocation				Max		Av	vaila	ble				
	A	В	C	A	В	C	A	В	C	A	В	C	
P0	2	2	0	5	7	7				3	5	7	
P1	0	1	2	2	2	3							A
P2	3	1	0	4	2	3							A
P3	0	1	1	4	5	6	5	5	8				A

Already done Already done

Already done

• Total number of resources: 3

• A=7, B=7, C=8

• Process: 4 (P<sub>0</sub>-P<sub>3</sub>) Available resource

Pro	ocess	All	ocat	ion		Max		A	aila	ble		Need		
		A	В	C	A	В	C	A	В	C	A	В	C	F' 11 11 11 11 10 10 10 10 10 10 10 10 10
]	P0	2	2	0	5	7	7				3	5	7	Finally we can allocate to P0 <p1, p0="" p2,="" p3,=""></p1,>
]	P1	0	1	2	2	2	3							Already done
	P2	3	1	0	4	2	3			<b>I</b>				Already done
]	P3	0	1	1	4	5	6	5	5	8				Already done

Total number of resources: 3

•	A=7	. B=	=7.	C=8
	<b>I I</b> - /	<b>, _</b> _	_ / 🤊	$\mathbf{C}$

Process:  $4(P_0-P_3)$ Available resource Allocati Available **Process** Max Need on B C A В В Already done 2 0 5 P0 Already done 2 2 3 P1 Already done 1 0 4 3 P2 Already done 5 P3 4 6

One of the safe sequences: <P1, P2, P3, P0>

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

Process	A	Alloc	catio	n		M	ax		1	Avai	labl	e
	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2				
$P_1$	3	0	1	1	3	1	3	4				
$P_2$	1	2	0	2	6	4	2	3				
$P_3$	1	1	3	0	5	3	7	5				
$P_4$	0	3	1	1	7	4	4	3				

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

Process	A	Alloc	catio	n		M	ax		ı	Avai	lable	e/		Ne	ed	
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2	1	3	3	4	6	3	2	2
$P_1$	3	0	1	1	3	1	3	4					0	1	2	3
$P_2$	1	2	0	2	6	4	2	3					5	2	2	1
$P_3$	1	1	3	0	5	3	7	5					4	2	4	5
$\mathbf{P}_{4}$	0	3	1	1	7	4	4	3					7	1	3	2

Available resources

Calculate Need matrix

Available resources

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5(P_0-P_4)$

Process	A	Alloc	catio	n		M	ax		1	Avai	labl	e/		Ne	ed	
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2	1	3	3	4	6	3	2	2
$P_1$	3	0	1	1	3	1	3	4					0	1	2	3
$P_2$	1	2	0	2	6	4	2	3					5	2	2	1
$P_3$	1	1	3	0	5	3	7	5					4	2	4	5
$P_4$	0	3	1	1	7	4	4	3					7	1	3	2

3 4 6 3 2 2

The available resources can

be allocated to P<sub>1</sub>

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

Process	A	Alloc	catio	n		M	ax		,	Avai	labl	e /		Ne	ed	
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2					6	3	2	2
$P_1$	3	0	1	1	3	1	3	4	4	3	4	5				
$P_2$	1	2	0	2	6	4	2	3					5	2	2	1
$P_3$	1	1	3	0	5	3	7	5					4	2	4	5
$P_4$	0	3	1	1	7	4	4	3					7	1	3	2

Available resources

Already done

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

Process	A	Alloc	catio	n		M	ax		1	Avai	labl	e /		Ne	ed	
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2					6	3	2	2
$P_1$	3	0	1	1	3	1	3	4	4	3	4	5				
$P_2$	1	2	0	2	6	4	2	3					5	2	2	1
$P_3$	1	1	3	0	5	3	7	5					4	2	4	5
$P_4$	0	3	1	1	7	4	4	3					7	1	3	2

Available resources

Already done

The available resources can be allocated to  $P_3$ 

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0 P_4)$

Process	A	Allo	catio	n		M	ax			Avai	labl	e		Ne	ed		
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	8	3	4	2					6	3	2	2	
$\mathbf{P}_{1}$	3	0	1	1	3	1	3	4									Already done
$P_2$	1	2	0	2	6	4	2	3	5	4	7	5	5	2	2	1	
$P_3$	1	1	3	0	5	3	7	5									Already done
$P_4$	0	3	1	1	7	4	4	3					7	1	3	2	

Available resources

<P1, P3>

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

Process	A	Alloc	catio	n		M	ax			Avai	labl	e /		Ne	ed	
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2					6	3	2	2
$P_1$	3	0	1	1	3	1	3	4								
$P_2$	1	2	0	2	6	4	2	3	5	4	7	5	5	2	2	1
$P_3$	1	1	3	0	5	3	7	5								
$P_4$	0	3	1	1	7	4	4	3					7	1	3	2

Available resources

Already done

The available resources can be allocated to Parallel Already done

$$<$$
P<sub>1</sub>, P<sub>3</sub>, P<sub>2</sub> $>$ 

Available resources

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

Process	A	Alloc	catio	n		M	ax			Avai	labl	e		Ne	ed		
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	8	3	4	2					6	3	2	2	
$P_1$	3	0	1	1	3	1	3	4									Already done
$P_2$	1	2	0	2	6	4	2	3									Already done
$P_3$	1	1	3	0	5	3	7	5	6	6	7	7					Already done
$P_4$	0	3	1	1	7	4	4	3					7	1	3	2	

$$<$$
P<sub>1</sub>, P<sub>3</sub>, P<sub>2</sub> $>$ 

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5(P_0-P_4)$

Process	A	Alloc	catio	n		M	ax		1	Avai	labl	e /		Ne	ed	
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2					6	3	2	2
$P_1$	3	0	1	1	3	1	3	4								
$P_2$	1	2	0	2	6	4	2	3								
$P_3$	1	1	3	0	5	3	7	5	6	6	7	7				
$P_4$	0	3	1	1	7	4	4	3					7	1	3	2

Available resources

The available resources can be allocated to P<sub>0</sub>

Already done

Already done

Already done

$$<$$
P<sub>1</sub>, P<sub>3</sub>, P<sub>2</sub>, P<sub>0</sub> $>$ 

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5(P_0-P_4)$

Process	A	Alloc	catio	n		M	ax		1	Avai	labl	e		Ne	ed	
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
$P_0$	2	0	2	0	8	3	4	2								
$P_1$	3	0	1	1	3	1	3	4								
$P_2$	1	2	0	2	6	4	2	3								
$P_3$	1	1	3	0	5	3	7	5			1					
$P_4$	0	3	1	1	7	4	4	3	8	6	9	7	7	1	3	2

Available resources

Already done
Already done
Already done
Already done

$$<$$
P<sub>1</sub>, P<sub>3</sub>, P<sub>2</sub>, P<sub>0</sub> $>$ 

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

													1				
Process	P	Alloc	catio	n		M	[ax			Avai	ilabl	e		Ne	ed		
	P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	8	3	4	2									Already
$P_1$	3	0	1	1	3	1	3	4									Already
$P_2$	1	2	0	2	6	4	2	3									Already
$P_3$	1	1	3	0	5	3	7	5			,						Already
$P_4$	0	3	1	1	7	4	4	3	8	6	9	7	7	1	3	2	The avail
P <sub>3</sub>	1 0	1			5	3	7	5	8	6	9	7	7	1	3	2	Alrea

Available resources

v done

v done

v done

v done

ilable resources can be allocated to P<sub>2</sub>

$$<$$
P<sub>1</sub>, P<sub>3</sub>, P<sub>2</sub>, P<sub>0</sub>, P<sub>4</sub> $>$ 

- Find out the safe sequence
- Total types of resources:
  - Printer (P): 8 instances
  - Scanner (S): 9 instances
  - File (F): 10 instances
  - Keyboard (K): 8 instances
- Process:  $5 (P_0-P_4)$

A	Alloc	catio	n		M	ax		1	Avai	labl	e		Ne	ed	
P	S	F	K	P	S	F	K	P	S	F	K	P	S	F	K
2	0	2	0	8	3	4	2								
3	0	1	1	3	1	3	4								
1	2	0	2	6	4	2	3								
1	1	3	0	5	3	7	5								
0	3	1	1	7	4	4	3								
	P 2 3 1 1 1	P S 2 0 3 0 1 2 1 1	P     S     F       2     0     2       3     0     1       1     2     0       1     1     3	P     S     F     K       2     0     2     0       3     0     1     1       1     2     0     2       1     1     3     0	P       S       F       K       P         2       0       2       0       8         3       0       1       1       3         1       2       0       2       6         1       1       3       0       5	P         S         F         K         P         S           2         0         2         0         8         3           3         0         1         1         3         1           1         2         0         2         6         4           1         1         3         0         5         3	P         S         F         K         P         S         F           2         0         2         0         8         3         4           3         0         1         1         3         1         3           1         2         0         2         6         4         2           1         1         3         0         5         3         7	P         S         F         K         P         S         F         K           2         0         2         0         8         3         4         2           3         0         1         1         3         1         3         4           1         2         0         2         6         4         2         3           1         1         3         0         5         3         7         5	P         S         F         K         P         S         F         K         P           2         0         2         0         8         3         4         2           3         0         1         1         3         1         3         4           1         2         0         2         6         4         2         3           1         1         3         0         5         3         7         5	P         S         F         K         P         S         F         K         P         S           2         0         2         0         8         3         4         2         2           3         0         1         1         3         1         3         4         4           1         2         0         2         6         4         2         3           1         1         3         0         5         3         7         5	P         S         F         K         P         S         F         K         P         S         F           2         0         2         0         8         3         4         2         2           3         0         1         1         3         1         3         4         1           1         2         0         2         6         4         2         3           1         1         3         0         5         3         7         5	P         S         F         K         P         S         F         K         P         S         F         K           2         0         2         0         8         3         4         2         2         3         4         1         3         1         3         4         4         4         1         1         1         1         3         1         3         4         4         2         3         3         4         1         1         1         3         0         5         3         7         5         3         7         5         3         7         5         3         7         5         3         7         5         3         4         2         3         4         2         3         4         2         3         3         4         2         3         3         4         2         3         3         4         2         3         3         4         2         3         3         4         2         3         3         4         2         3         3         4         2         3         3         4         2         3	P         S         F         K         P         S         F         K         P         S         F         K         P           2         0         2         0         8         3         4         2         3         4         3         4	P         S         F         K         P         S         F         K         P         S         F         K         P         S           2         0         2         0         8         3         4         2         3         4	P         S         F         K         P         S         F         K         P         S         F         K         P         S         F           2         0         2         0         8         3         4         2         0

Available resources

10

Already done

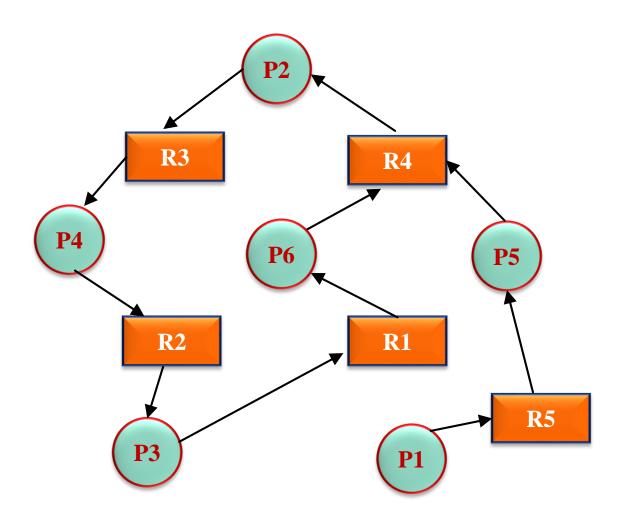
Already done

Already done

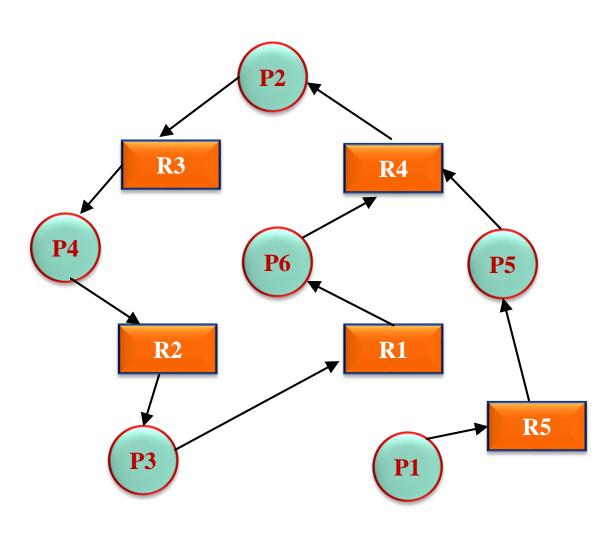
Already done One of the safe sequences

Already done

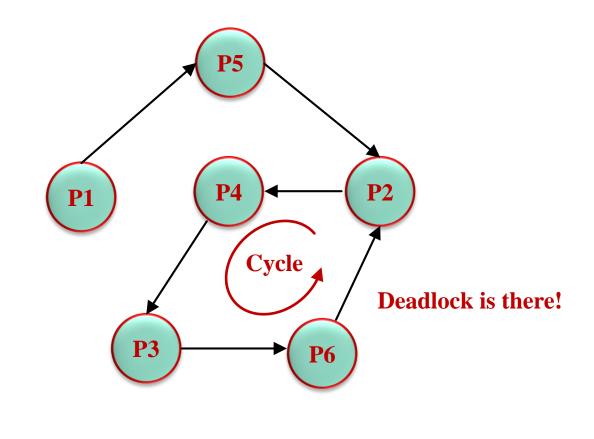
$$<$$
P<sub>1</sub>, P<sub>3</sub>, P<sub>2</sub>, P<sub>0</sub>, P<sub>4</sub>



Resource-Allocation Graph (RAG)



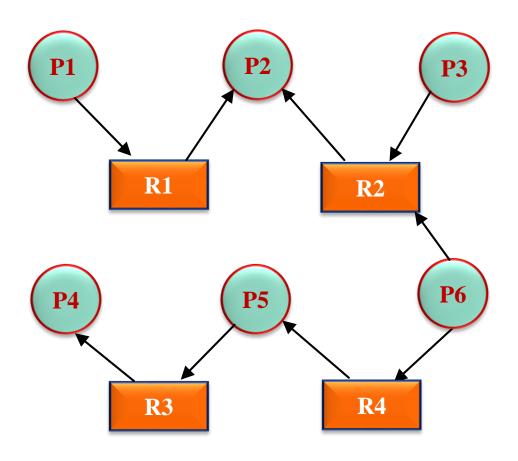
We can generate Wait-for-Graph from the RAG



Resource-Allocation Graph (RAG)

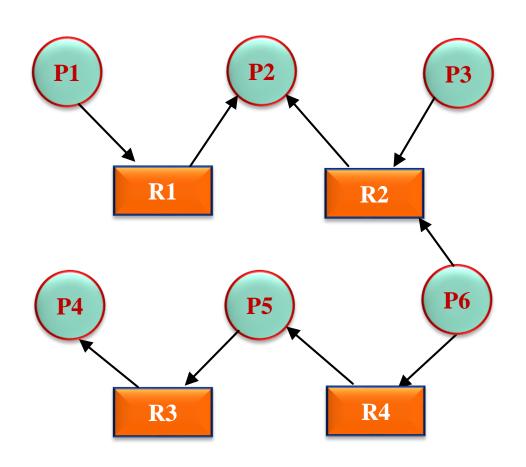
Wait-for-Graph

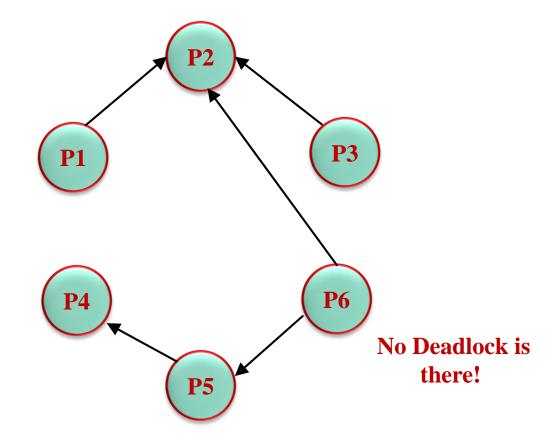
We can generate Wait-for-Graph from the RAG



Resource-Allocation Graph (RAG)

We can generate Wait-for-Graph from the RAG





• Need to detect if there is a deadlock when we have multiple instances of a resource

• A: 7 Instances

• B: 8 Instances

• C: 9 Instances

Process	Allo	catio	n	Req	uest		Ava	ailab	le
	A	В	C	A	В	C	A	В	C
P0	1	3	1	5	5	7	0	2	1
P1	2	1	2	0	2	1			
P2	3	1	4	4	6	3			
P3	1	1	1	4	5	6			

• Need to detect if there is a deadlock when we have multiple instances of a resource

• A: 7 Instances

• B: 8 Instances

• C: 9 Instances

Process	Allo	catio	n	Req	uest		Ava	ailab	le
	A	В	C	A	В	C	A	В	C
P0	1	3	1	5	5	7	0	2	1
P1	2	1	2	0	2	1			
P2	3	1	4	4	6	3			
P3	1	1	1	4	5	6			

P1 can execute with the available resources

• Need to detect if there is a deadlock when we have multiple instances of a resource

• A: 7 Instances

• B: 8 Instances

• C: 9 Instances

Process	Allo	catio	n	Req	uest		Ava	ailab	le
	A	В	C	A	В	C	A	В	C
P0	1	3	1	5	5	7	0	2	1
P1	2	1	2						
P2	3	1	4	4	6	3			
P3	1	1	1	4	5	6			

Let P1 is executed

• Need to detect if there is a deadlock when we have multiple instances of a resource

• A: 7 Instances

• B: 8 Instances

• C: 9 Instances

Process	Allo	catio	n	Req	uest		Ava	ailab	le
	A	В	C	A	В	C	A	В	C
P0	1	3	1	5	5	7			
P1	2	1	2				2	3	3
P2	3	1	4	4	6	3			
P3	1	1	1	4	5	6			

With the available resources, none of the processes can be executed – Deadlock occurs

• Total types of resources:

• Printer (P): 6 instances

• Scanner (S): 9 instances

• File (F): 12 instances

• Keyboard (K): 5 instances

• Process:  $5 (P_0 - P_4)$ 

Process	A	Alloc	catio	n		Req	uest	;	Available				
	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	3	3	4	5	0	3	6	1	
$P_1$	3	0	1	1	0	2	4	1					
$P_2$	0	2	0	2	6	4	2	3					
$P_3$	1	1	3	0	5	3	8	6					
$P_4$	0	3	1	1	3	3	7	1					

• Total types of resources:

• Printer (P): 6 instances

• Scanner (S): 9 instances

• File (F): 13 instances

• Keyboard (K): 5 instances

• Process:  $5 (P_0 - P_4)$ 

Process	A	Alloc	catio	n		Req	uest	;	Available				
	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	3	3	4	5	0	3	6	1	
$P_1$	3	0	1	1	0	2	4	1					
$P_2$	0	2	0	2	6	4	2	3					
$P_3$	1	1	3	0	5	3	8	6					
$P_4$	0	3	1	1	3	3	7	1					

• Total types of resources:

• Printer (P): 6 instances

• Scanner (S): 9 instances

• File (F): 13 instances

• Keyboard (K): 5 instances

• Process:  $5 (P_0 - P_4)$ 

Process	A	Alloc	catio	n		Req	uest		Available				
	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	3	3	4	5	0	3	6	1	
$P_1$	3	0	1	1	0	2	4	1					
$P_2$	0	2	0	2	6	4	2	3					
$P_3$	1	1	3	0	5	3	8	6					
$P_4$	0	3	1	1	3	3	7	1					

With the available resources, P<sub>1</sub> can execute

• Total types of resources:

• Printer (P): 6 instances

• Scanner (S): 9 instances

• File (F): 13 instances

• Keyboard (K): 5 instances

• Process:  $5 (P_0 - P_4)$ 

Process	A	Alloc	atio	n		Req	uest		Available				
	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	3	3	4	5					
$P_1$	3	0	1	1					3	3	7	2	
$P_2$	0	2	0	2	6	4	2	3					
$P_3$	1	1	3	0	5	3	8	6					
$P_4$	0	3	1	1	3	3	7	1					

Let P<sub>1</sub> is executed

Total types of resources:

• Printer (P): 6 instances

• Scanner (S): 9 instances

• File (F): 13 instances

• Keyboard (K): 5 instances

• Process:  $5 (P_0 - P_4)$ 

Process	A	Alloc	catio	n		Req	uest		Available				
	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	3	3	4	5					
$P_1$	3	0	1	1					3	3	7	2	
$P_2$	0	2	0	2	6	4	2	3					
$P_3$	1	1	3	0	5	3	8	6					
$P_4$	0	3	1	1	3	3	7	1					

Let P<sub>1</sub> is executed

With the available resources, P<sub>4</sub> can execute

• Total types of resources:

• Printer (P): 6 instances

• Scanner (S): 9 instances

• File (F): 13 instances

• Keyboard (K): 5 instances

• Process:  $5 (P_0 - P_4)$ 

Process	A	Alloc	catio	n		Req	uest			Avai	ilabl	e	
	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	3	3	4	5					
$P_1$	3	0	1	1									Let P1 is executed
$P_2$	0	2	0	2	6	4	2	3	3	6	8	3	
$P_3$	1	1	3	0	5	3	8	6					
$P_4$	0	3	1	1									Let P <sub>4</sub> is executed

 $< P_1, P_4 >$ 

• Total types of resources:

• Printer (P): 6 instances

• Scanner (S): 9 instances

• File (F): 13 instances

• Keyboard (K): 5 instances

• Process:  $5 (P_0-P_4)$ 

Process	A	Alloc	catio	n		Req	uest			Avai	labl	e	
	P	S	F	K	P	S	F	K	P	S	F	K	
$P_0$	2	0	2	0	3	3	4	5					
$P_1$	3	0	1	1									Let P1 is executed
$P_2$	0	2	0	2	6	4	2	3	3	6	8	3	
$P_3$	1	1	3	0	5	3	8	6					
$P_4$	0	3	1	1									Let P <sub>4</sub> is executed

With the available resources, none of the processes can be served. So, there is a deadlock

$$<$$
P<sub>1</sub>, P<sub>4</sub> $>$ 

# **THANK YOU!**