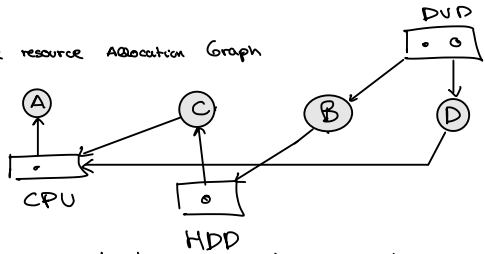
RESOURCE ALLOCATION GRAPH

(2)

a) Draw the resource Allocation Graph



There are no cycles in this graph, so there is no deadlock

b) It can not have an interlock because there are no cycles.

(2)

a) It's impossible to have a deadlock because we allowed "hold and wait"; "mutual exclusion" and "preemption" which are

b) Matrix A

	R_1	R_2	R_3
P_1	1	0	0
P_2	0	1	0
P_3	0	0	1

c) Vector R

	R_1	R_2	R_3
E	1	1	1

(3)

a) P_4 request (0, 1, 1, 0) ?

Req i ≤ Vi <0, 1, 1, 0> < 3, 3, 3, 3> ✓

N = C - A

	A	B	C	D
P_0	1	1	1	1
P_1	4	0	3	3
P_2	0	1	3	0
P_3	3	1	3	1
P_4	0	1	0	1

The OS does not let what it is requested happen

Req i ≤ Ni <0 1 1 0> ≤ ? <0 1 0 1> ✗

Not grant the request.

Virtual state represent how it would be if we grant the request.

b) P_2 request (0, 1, 2, 0) ?

Req i ≤ Vi <0, 1, 2, 0> < 3, 3, 3, 3> ✓

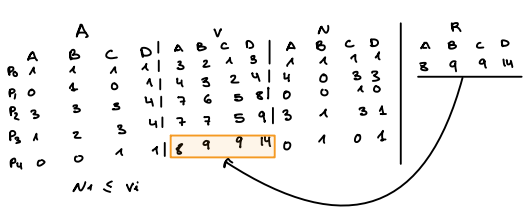
N = C - A

	A	B	C	D
P_0	1	1	1	1
P_1	4	0	3	3
P_2	0	1	3	0
P_3	3	1	3	1
P_4	0	1	0	1

<0, 1, 2, 0> ≤ ? <0, 1, 3, 0> ✓

3 > 2

Grant the request



④ Status consisting of 4 processes (P_0, P_1, P_2, P_3) and 3 resources (A, B and C)

PROCESS	A ABC	C ABC	V ABC
P_0	110	121	011
P_1	100	121	
P_2	001	021	
P_3	101	101	

a)

$N = C - A$

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

$E = D + A = (0,1,1) + (3,1,2) = (3,2,3)$

R		
A	B	C
3	2	3

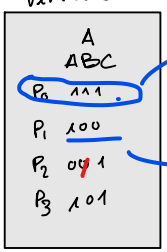
b) Safe system if P_0 makes request for an instance C

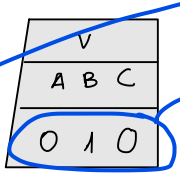
P_0 : $(0,0,1)$

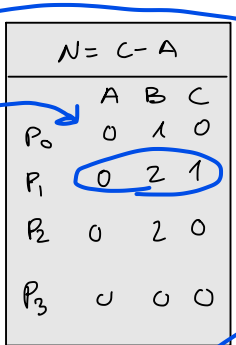
$Req_i < V_i \rightarrow (0,0,1) < (0,1,1)$ ✓

$Req_i < N_i \rightarrow (0,0,1) < (0,1,1)$

Virtual state:







Chosen	V	Finished
———	{0, 1, 0}	{ }
P_0	{0,1,0} + {1, 1, 1} = {1,2,1}	{P_0}
P_1	{1,2,1} + {1,0,0} = {2,2,1}	{P_0, P_1}
P_2	{2,2,1} + {0,0,1} = {2,2,2}	{P_0, P_1, P_2}
P_3	{2,2,2} + {1,0,1} = {3,2,3}	{P_0, P_1, P_2, P_3}

The system is safe if P_0 makes a request for an instance

c) P_1 makes a request for resource C

$(0,0,1) > (0,1,0)$ (V) ✗

$(0,0,1) > (0,1,0)$ (N) ✗

} It cannot happened.

d) P_2 request $(0,1,0)$

$(0,1,0) \leq (0,1,0)$ V

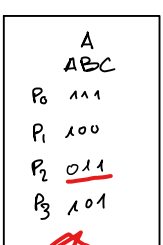
$(0,1,0) \leq (0,2,0)$ Ni

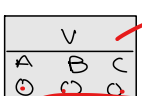
} ✓

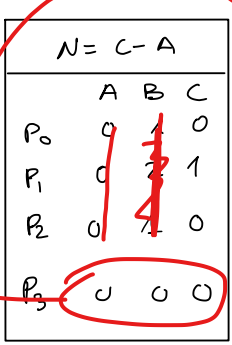
(0,1/0)

0,0,1 0

Virtual state







DEADLOCK
NOT SAFE

Chosen	V	Finished
———	{0, 0, 0}	{ }
P_2	{1, 0, 1}	{P_2}

✗

⑤

	A				C			
	A	B	C	D	A	B	C	D
P_0	4	1	0	1	6	1	0	2
P_1	2	1	2	0	4	2	1	1
P_2	0	0	0	0	0	0	1	0
P_3	0	0	0	0	3	1	0	4

As we can see, the problem is that P_2 is "using" 2 instances of C when it can use a maximum of 1. In order to guarantee a request, the first thing that Banker's Algorithm does is to check if $req_i \leq (C-A)_i$ and this is the same as checking $req_i + A_i \leq C_i$

Imagine that P_2 is allocating 1 instance of C and then it requests (0, 0, 1, 0) for example. That request is going to be added to AP_2 and then the question is: Is this addition less or equal than CP_2?

Obviously not, because in this addition we have 2 instances of C while the maximum to be used is only 1. We can't continue because this condition is not true and this "process of request" is stopped.

⑥

Process	A ABCD	Q ABCD	V ABCD
0	1031	0021	0102
1	5120	0103	
2	0231	5311	
3	2310	0411	
4	2011	6021	
5	0403	2103	

a) There are no processes with no recurses allocated.

2) $Q_x \leq V$

Process	V	M
———	{0, 1, 0, 2}	{}
P_1	{0,1,0,2} + {3,1,2,0} = {3,2,2,2}	{8}
P_0	{3,2,2,2} + {1,0,3,1} = {4,2,5,3}	{8, 8}
P_3	{4,2,5,3} + {0,4,0,3} = {4,6,5,6}	{8, 8, 8}
P_2	{4,6,5,6} + {0,2,3,1} = {4,8,8,7}	{8, 8, 8, 8}
P_3	{4,8,8,7} + {2,3,1,0} = {6,11,9,7}	{8, 8, 8, 8, 8}
P_4	{6,11,9,7} + {2,0,1,1} = {8, 11, 10, 8}	{8, 8, 8, 8, 8, 8}

NO DEADLOCK

b) Safe sequence: {P_1, P_0, P_3, P_2, P_3, P_4}

always — you can switch

⑦ 5 processes that use 6 resources (A, B, C, D, E, F)

Resources						
A	B	C	D	E	F
3	6	7	4	5	8

V					
A	B	C	D	E	F
1	1	0	0	2	1

	A						Q = R					
	A	B	C	D	E	F	A	B	C	D	E	F
P_0	1	1	0	1	1	1	0	1	2	2	0	1
P_1	1	0	0	2	1	2	0	0	3	0	0	1
P_2	0	2	0	1	1	0	1	3	2	3	2	5
P_3	0	0	5	0	0	0	1	0	0	0	2	1
P_4	0	2	2	0	0	4	1	3	4	0	3	3

a) In order to know if there is a deadlock we use Banker's algorithm.

b) $Q_i < V$

Chosen	V	Finished
———	{1, 1, 0, 2, 1}	{}
P_3	{1,1,0,2,1} + {0,0,5,0,0,0} = {1,1,5,0,2,1}	{P_3}
P_1	{1,1,5,0,2,1} + {1,2,0,2,1,1} = {2,1,5,2,3,2}	{P_3, P_1}
P_0	{2,1,5,2,3,2} + {1,1,0,1,1,1} = {3,2,5,3,4,4}	{P_3, P_1, P_0}

Q_2 y $Q_4 > V$ so there is a DEADLOCK no safe sequence can be found.

c) There is a deadlock produced by P_2 and P_4 because $Q_2 > V$ and $Q_4 > V$.

d) In order to avoid the deadlock we add one to resources B and F so we have (3, 3, 5, 3, 4, 5)

so now $Q_2 \leq V$, and when we perform $V + A_2 = (3, 5, 5, 4, 5, 5)$ and P_4 is the process that completes the safe sequences $Q_4 \leq V$ and $V + A_4 = (3, 7, 7, 4, 8, 9)$ which is the same as the total value of resources +1 in B and F