Conditions for deadlocks

$\boxed{1}$ Mutual Exclusion

$\boxed{2}$ Hold and wait

$\boxed{3}$ No preemption

$\boxed{4}$ Circular wait

Deal w/ Deadlocks

$\boxed{1}$ Prevention

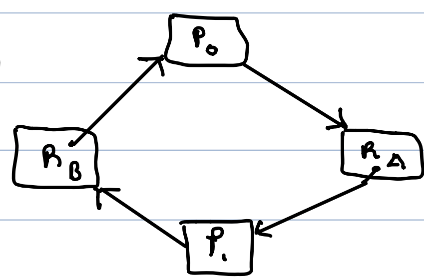
$\boxed{2}$ Avoidance

$\boxed{3}$ Detection

$\boxed{1}$ Prevention

$\boxed{1}$ × $\boxed{2}$ ✓ $\boxed{3}$ ✓ $\boxed{4}$

4



P_0	P_1
wait (s)	wait (a)
wait (a)	wait (s)
C.S.	C.S.
sign (a)	sign (s)
sign (s)	sign (a)

1	2	3	4
R_A	R_B	R_C	R_D

Define a linear order of resources

Process request resources in increasing order

$R_i \rightarrow P_i \rightarrow R_{i+1} \rightarrow P_{i+1} \longrightarrow \dots$

$$O(R_i) < O(R_{i+1}) \quad \forall i \longleftarrow \text{for all}$$

$$O(R_1) < O(R_2) < O(R_3) < \dots < O(R_1)$$

$$O(R_1) < O(R_1) \implies \text{contradiction}$$

$\boxed{2}$ Avoidance

Process provide extra information to the system to avoid deadlocks.

eg// Processes submit their maximum need of resources.

- State of the system

$\boxed{1}$ Current Allocation (How much you've spent from credit limit)

$\boxed{2}$ Maximum Need

$\boxed{3}$ Avaliable resources

Safe

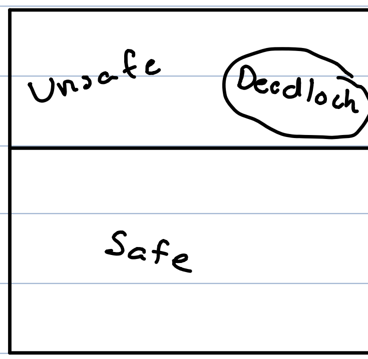
- System can allocate resources up to their maximum and still be able to finish.

(i.e)

System can allocate resources to each process up to its maximum need and still avoids a deadlock

- Safe Sequence: Order of process that can

- Unsafe: No safe sequence exist



May lead to a deadlock

ex//

12 tapes

Currently Allocated

it's going to request

(Maximum - Current)

	Current	Maximum	Maximum Need
P_0	5	10	5
P_1	2	4	2
P_2	2	9	7

↓

9

Avcilcble : 12 - 9 = 3 Resources.

P_1 can finish due to $2 < 3$

$P_1 \xrightarrow{\text{avclible} = 5}$

b/c P_1 released 2 $\therefore$ $2+3 = 5$

$P_1 \xrightarrow{\text{avclible} = 5} P_0 \xrightarrow{\text{Avclible} = 10} P_2$

$5 + 5 = 10$

$P_1 \, P_0 \, P_2$, safe sequence

$\therefore$ system is safe

es//	12 tapes	Current	Maximum	Need
	P_0	5	10	5
	P_1	2	4	2
	P_2	3	9	6
		= 10		

Avclible = 2

$P_1 \xrightarrow{\text{avalible} = 4}$ No one can finish due to $4 < 5$ && $4 < 6$

- May lead to a deadlock

$\boxed{P_0,\ P_2 \quad \text{May be deadlock}}$

Avoidance $\vdots$ being in a safe state.

Grant requests if allocations leave the system in a safe state. (Pretending we do the allocation)

<u>Resoure Allocation Graph.</u>

$\boxed{P_0} \longrightarrow \boxed{R_A}$ Req.

$\boxed{R_A} \longrightarrow \boxed{P_1}$ Assignment

(when it holds a resource)

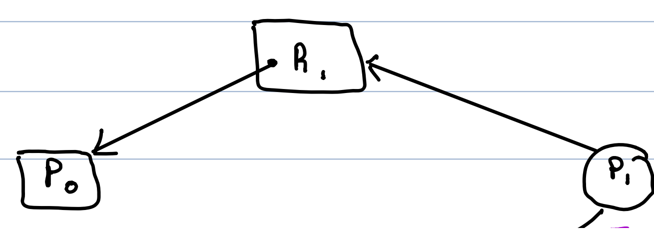
Works only with single instances of resources.

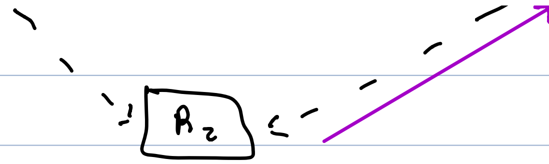
Claim edge :

$P_i \dashrightarrow R_j$

$\boxed{P_i} \dashrightarrow \boxed{R_j}$

(P_i may request R_j)





- P_1 request R_2 [could turn into a deadlock]
- Resource to P_1 is allocation ↓ Create a cycle May lead to deadlock
- If you see a cycle, it means unsafe.

Bankers Algorithim

- Saftey of state
- Handling request of Allocation

n process m resources

1 Saftey

Avalible : Vector of length m for the number of avalible resoures

Allocation : n x m matrix of the number of resoures allocated
(snapshot of Allocated)

Max : n x m matrix of the maximum demand
(Maximum demand)

Need : $n \times m$ matrix for number of

(How much) resources needed.
more you
can request

$$X \le Y$$

$$[\quad]_m \le [\quad]_m$$

$$X_i \le Y_i \quad \forall i$$

$$\Rightarrow X \le Y$$

need's same
pattern

$$[2 \quad 3 \quad 5] \le [3 \quad 4 \quad 8]$$

Steps

$\boxed{1}$ Work = avalible Finish [process] = False

$\boxed{2}$ Find any i such that

Finish [i] = FALSE;

$Need_i \le Work$;

if no i exists go to $\boxed{4}$

$\boxed{3}$ $Work = work + Allocation_i$

$Finish[i] = True;$

goto $\boxed{2}$

$\boxed{4}$ If $finish[i] == True \ \forall i \Rightarrow$ safe

else

Unsafe

$\boxed{2}$ Resource Allocation Algorithm

$Request_i \rightarrow R_i$: Vector of resources requested by process i

vector ⤴

$\boxed{1}$ If $R_i > Need_i \rightarrow$ error

$\boxed{2}$ If $R_i > Avalible \rightarrow$ wait

$\boxed{3}$ <u>Pretend</u> (check for softey part)

- Grant & check
- While remaining in safe state

we grant

$Allocation\ of_i = Allocation_i + request_i$

$audible = avalible - request_i$

$Need_i = Need_i - request_i$

$\boxed{4}$ Run softey part

if safe $\Rightarrow$ grant

else

deny

3 Resources A B C

5 Processes $P_0 \to P_5$

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

$n \times m$