CSC 139 Operating System Principles March 26, 2020

For i=0

$Need_0 = 743 \le work = 745$

P_0 can be kept in the safety sequence!

work = work + allocation[0]

= 745 + 010 = 755

For i=2

$Need_2 = 600 \le work = 755$

P_2 can be kept in the safety sequence!

Safe sequence = $\langle P_1, P_3, P_4, P_0, P_2 \rangle$ -- not the only safe sequence

Resource Request Algorithm

can we approve $request_1 = (1,0,2)$?

step 1. If $Request_i <= Need_i$

$request_1 = (1,0,2) <= Need_1 = (\phi, 2, 0)$

step 2. If $Request_i <= Available$

$request_1 = (1,0,2) <= available = (3,3,2)$

step 3. Assume $request_i$ is approved.

3i. available = available - $Request_i$

3ii. $allocation_i = allocation_i + Request_i$

3iii. $Need_i = Need_i - Request_i$

New state S' and then run the safety algorithm, if yes in safe state, we can approve the request!

allocation

	A	B	C
P_0	0	1	0
P_1	3	0	2
P_2	3	0	2
P_3	2	1	1
P_4	0	0	2

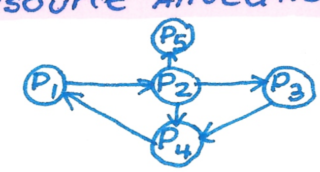
Deadlock Detection

allow system to enter to deadlock state

detection algorithm → if there is a cycle there exists a deadlock

recovery scheme

Resource Allocation Graph & Wait-for Graph



Maintain wait-for graph

$P_i \to P_j$ if P_i is waiting for P_j

Detection Algorithm:

Available: indicates # of resources

Allocation: # of resources of each type currently allocated to each process

request: current request of the process

1a. Work = Available

1b. If allocation ≠ 0, Finish[i] = false else Finish[i] = true

2a. Finish[i] = False] Find a index i that...

2b. $Request_i \le work$]

3. work = work + $allocation_i$

Finish[i] = true

4. Finish[i] == false for any i the system is in a deadlocked state & P_i is deadlocked