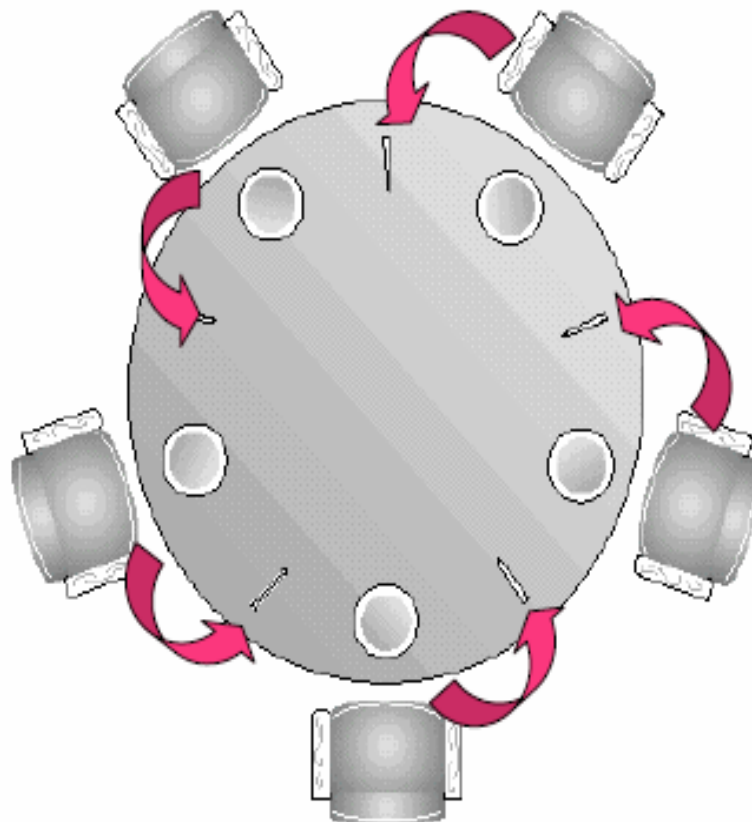




Vấn đề tắc nghẽn

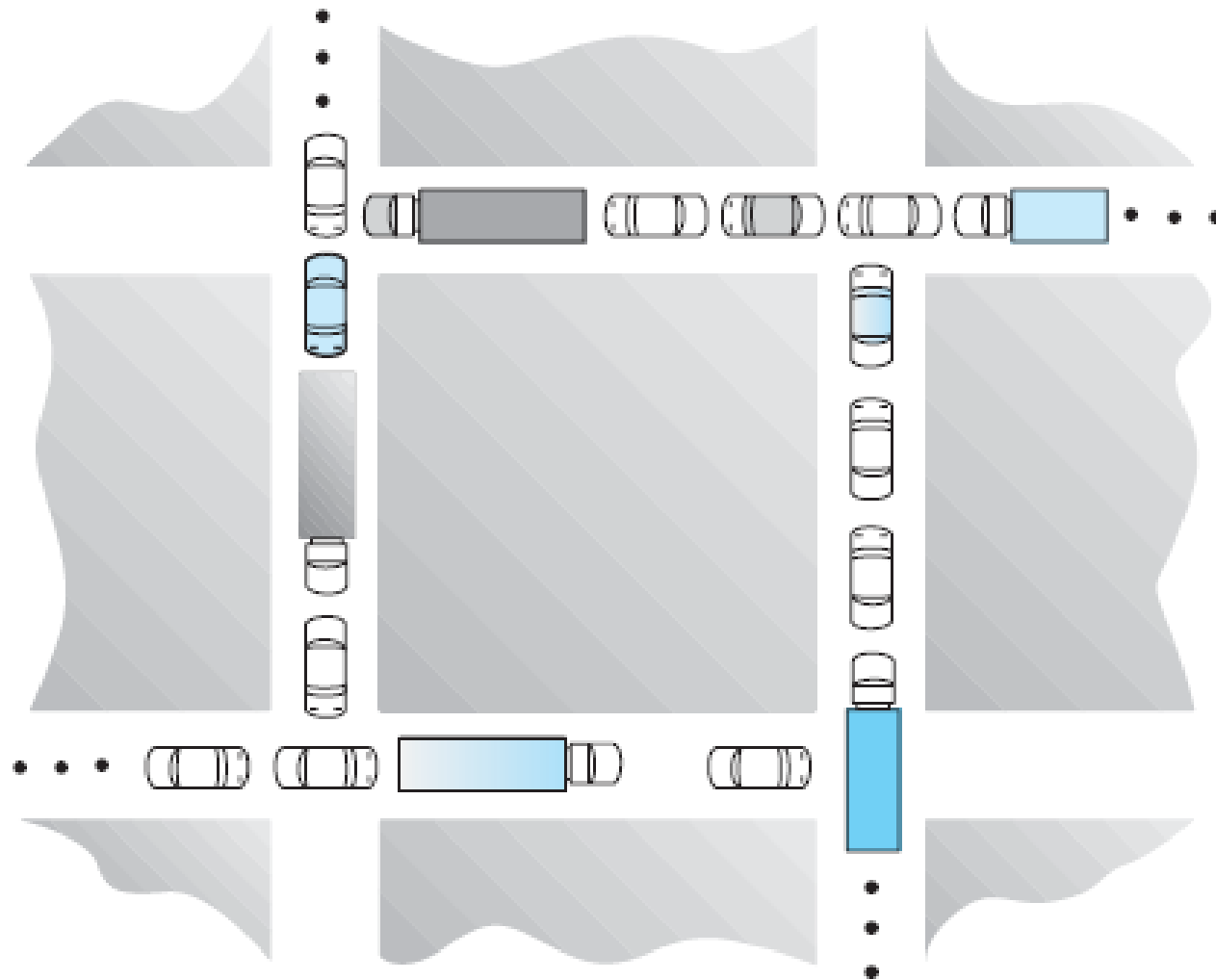
Deadlock

Cả 5 triết gia
cùng đồng loạt
muốn ăn



Mỗi người chỉ
lấy được đúng
1 chiếc nĩa

Deadlock



Deadlock trong tổ chức tiến trình

- Mỗi tiến trình trong nhóm đều chờ được cấp phát một tài nguyên.
- Tài nguyên đang tranh chấp bị một tiến trình khác cũng ở trạng thái blocked chiếm giữ.
- Không có tiến trình nào có thể tiếp tục xử lý để giải phóng tài nguyên.
- Tất cả các tiến trình trong nhóm đều bị khóa vĩnh viễn !

Điều kiện của tắc nghẽn

1. Truy xuất loại trừ (mutual exclusion): một tiến trình duy nhất được truy xuất
2. Chiếm giữ và yêu cầu thêm (Hold and wait)
3. Tiến trình chiếm giữ tài nguyên theo chế độ độc quyền (No preemption)
4. Tồn tại chu kỳ (Circulation wait)

Đồ thị cấp phát tài nguyên

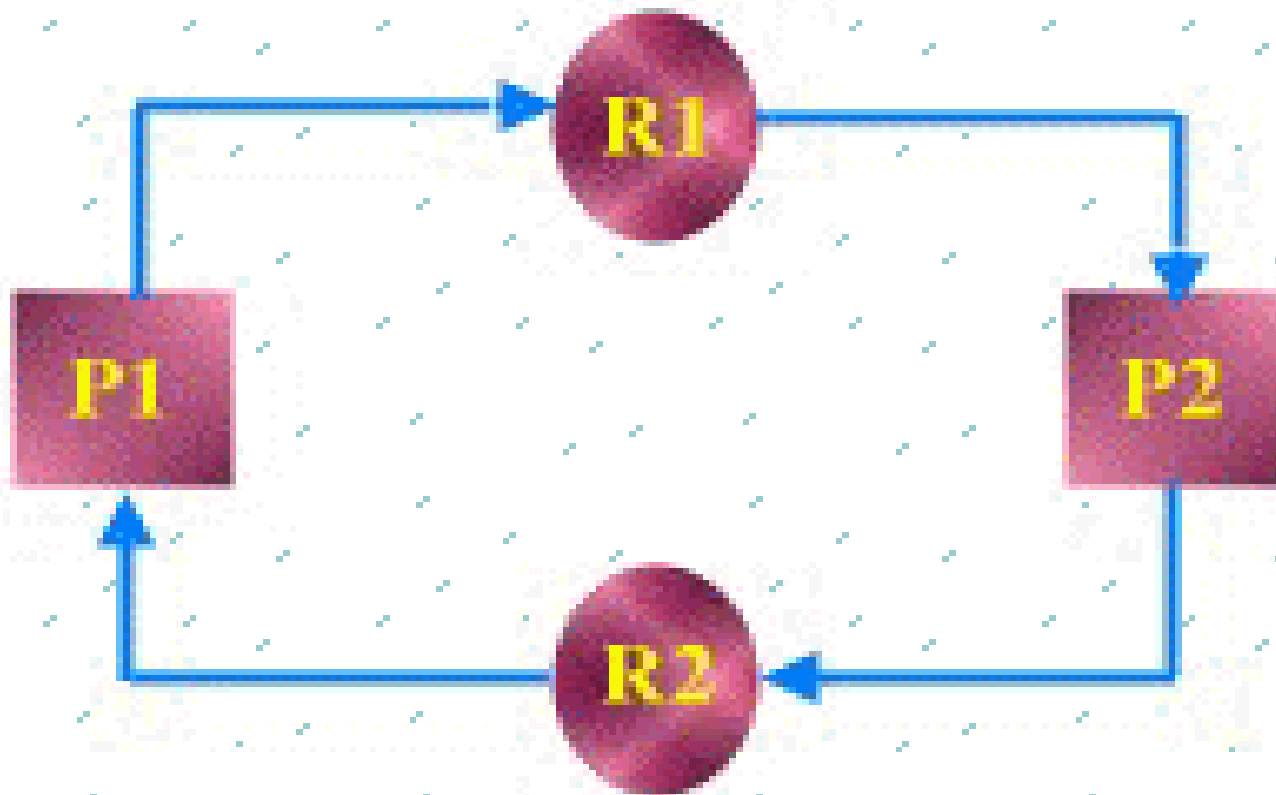


P: Tiến trình

R: Tài nguyên

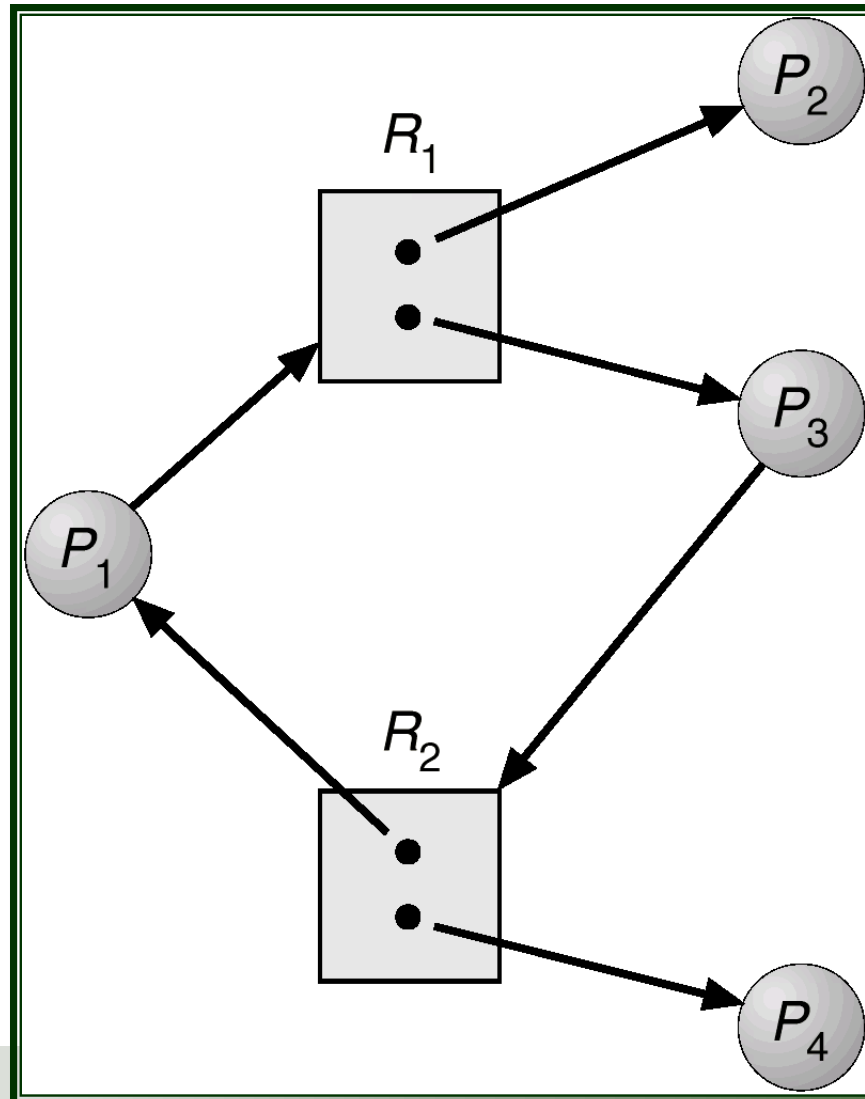


Đồ thị cấp phát tài nguyên



Đồ thị cấp phát tài nguyên

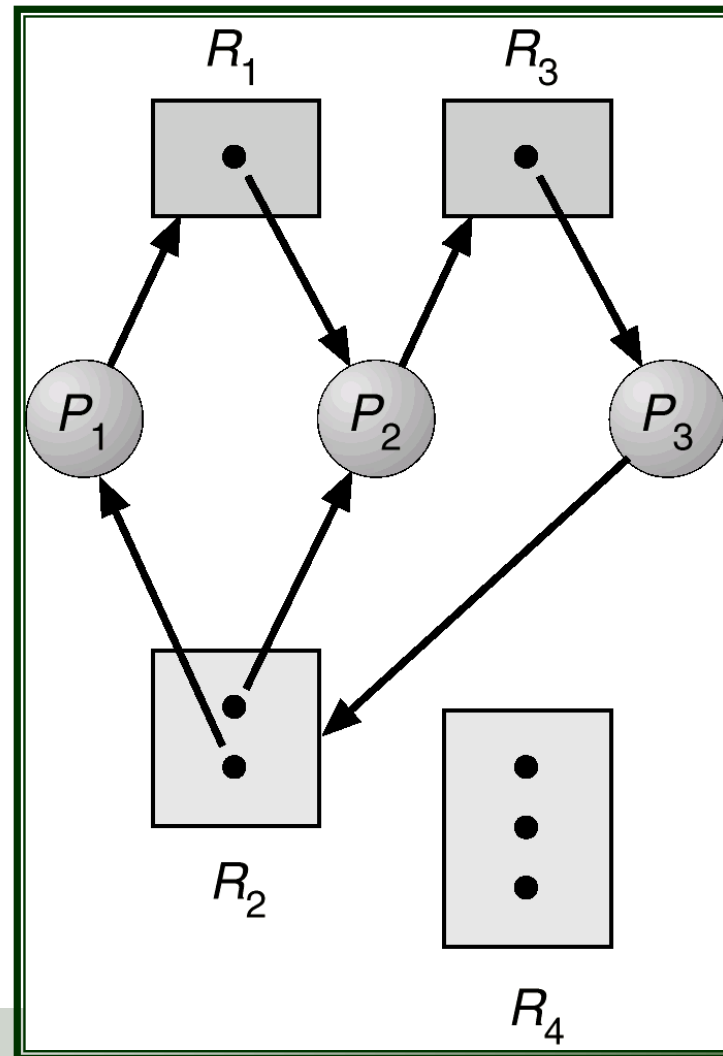
Có xảy ra
deadlock
không?



Đồ thị cấp phát tài nguyên

Có xảy ra
deadlock
không?

Kết luận???



Giải quyết vấn đề tấn công

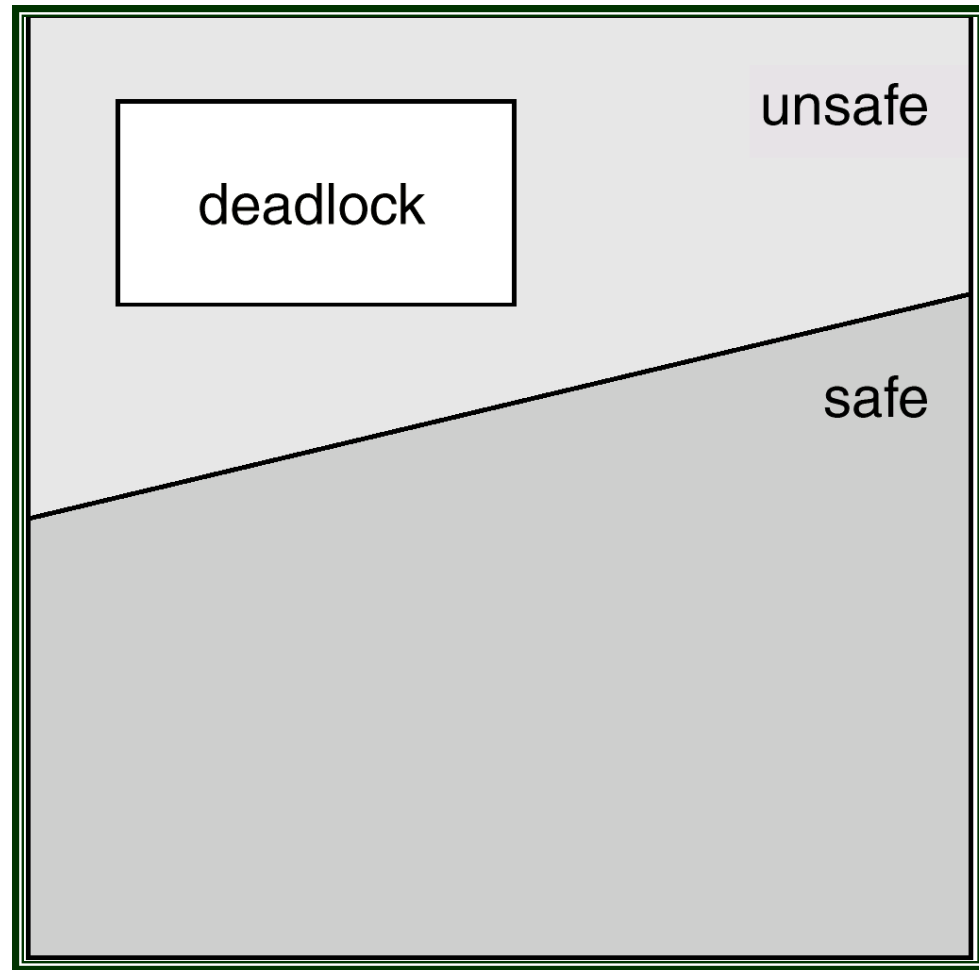
- Ngăn chặn (Prevention):
 - Loại bỏ 4 điều kiện của tấn công
- Xử lý (Detection and recovery)
 - Chấp nhận cho tấn công xảy ra, thực hiện các thủ tục để phát hiện tấn công, nếu có thì xử lý.
- Không quan tâm (Ignore)

Ngăn chặn tắc nghẽn

- Loại bỏ các điều kiện tắc nghẽn
 - Mutual Exclusion
 - Hold and Wait
 - No Preemption
 - Circular Wait

Trạng thái an toàn của hệ thống

Trạng thái an toàn (safe state): hệ thống có thể cấp phát tài nguyên cho các tiến trình mà không bị tắc nghẽn.



Trạng thái an toàn của hệ thống

- Chuỗi tiến trình $\langle P_1, P_2, \dots, P_n \rangle$ là an toàn đối với từng tiến trình P_i nếu các tài nguyên mà P_i cần sẽ được đáp ứng bởi các tài nguyên đang có cùng với các tài nguyên đang chiếm dụng bởi các tiến trình P_j , với $j < i$.

Xác định trạng thái an toàn

- *Available [r]*: Số tài nguyên sẵn sàng ứng với từng loại tài nguyên r .
- *Max[p,r]*: Nhu cầu tài nguyên của tiến trình p đối với tài nguyên r .
- *Allocation [p,r]*: số tài nguyên loại r đã cấp cho tiến trình p ;
- *Need [p,r]*: số tài nguyên loại r mà tiến trình p cần sử dụng.
- *Finish [p]*: tiến trình p đã thực hiện xong

Xác định trạng thái an toàn

- B1: Tìm tiến trình i thỏa:
 - $\text{Finish}[i] = \text{false};$
 - $\text{Need}[i,j] \leq \text{Available}[j];$
 - Nếu không tồn tại qua bước 3.
- B2: Cấp phát tài nguyên cho tiến trình i
 - $\text{Allocation}[i,j] = \text{Allocation}[i,j] + \text{Need}[i,j];$
 - $\text{Need}[i,j] = 0;$
 - $\text{Available}[j] = \text{Available}[j] - \text{Need}[i,j];$
 - $\text{Finish}[i] = \text{true};$
 - Thu hồi các tài nguyên đã cấp cho P_i và quay lại B1
- B3: Nếu $\text{Finish}[i] = \text{true}$ với mọi i thì hệ thống an toàn.

Thuật toán Banker

- Nguyên tắc: Khi tiến trình yêu cầu tài nguyên, hệ thống cấp phát “thử”, sau đó xác định xem hệ thống có an toàn không. Nếu an toàn thì cấp phát “thật”, ngược lại thì không cấp phát

Thuật toán Banker

- Tiến trình P_i yêu cầu k tài nguyên loại r .
 - B1: Nếu $k \leq \text{Need}[i, r]$ thì qua B2, ngược lại thì báo lỗi.
 - B2: Nếu $k \leq \text{Available}[r]$ thì qua B3, ngược lại P_i phải chờ.
 - B3: Thử cấp phát tài nguyên cho P_i :
 - $\text{Available}[r] = \text{Available}[r] - k$;
 - $\text{Allocation}[i, r] = \text{Allocation}[i, r] + k$;
 - $\text{Need}[i, r] = \text{Need}[i, r] - k$;
 - B4: Kiểm tra trạng thái an toàn của hệ thống

Ví dụ 1

	Max			Allocation			Available		
	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1	3	2	2	1	0	0	4	1	2
P2	6	1	3	2	1	1			
P3	3	1	4	2	1	1			
P4	4	2	2	0	0	2			

Nếu tiến trình P2 yêu cầu 4 cho R1, 1 cho R3. Hãy cho biết yêu cầu này có thể đáp ứng mà bảo đảm không xảy ra tình trạng deadlock hay không ?

Ví dụ 1

	Need			Allocation			Available		
	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1				1	0	0	4	1	2
P2				2	1	1			
P3				2	1	1			
P4				0	0	2			

Xác định nhu cầu tài nguyên còn lại của từng tiến trình trong bảng Need

Ví dụ 1

	Need			Allocation			Available		
	R1	R2	R3	R1	R2	R3	R1	R2	R3
P1							4	1	2
P2									
P3									
P4									

Thử cấp tài nguyên theo yêu cầu của P2 và cập nhật trạng thái hệ thống

Kiểm tra trạng thái an toàn

Ví dụ 2

- 5 tiến trình ($P_0 - P_4$).
- 3 loại tài nguyên: A (10), B (5), C (7).
- Tại thời điểm T_0 :

	<u>Allocation</u>	<u>Max</u>	<u>Available</u>
	$A \ B \ C$	$A \ B \ C$	$A \ B \ C$
P_0	0 1 0	7 5 3	3 3 2
P_1	2 0 0	3 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	

Ví dụ 2

- Xét tình huống

	<u>Need</u>		
	<i>A</i>	<i>B</i>	<i>C</i>
P_0	7	4	3
P_1	1	2	2
P_2	6	0	0
P_3	0	1	1
P_4	4	3	1

- Chuỗi truy xuất $\langle P_1, P_3, P_4, P_2, P_0 \rangle$ có an toàn?

Ví dụ 2

- Request \leq Available : $(1,0,2) \leq (3,3,2)$

	<u>Allocation</u>	<u>Need</u>	<u>Available</u>
	A B C	A B C	A B C
P_0	0 1 0	7 4 3	2 3 0
P_1	3 0 2	0 2 0	
P_2	3 0 1	6 0 0	
P_3	2 1 1	0 1 1	
P_4	0 0 2	4 3 1	

- Chuỗi truy xuất $\langle P_1, P_3, P_4, P_0, P_2 \rangle$ có an toàn không?
- Nếu P_4 yêu cầu tài nguyên $(3,3,0)$ thì có được đáp ứng không?
- Nếu P_0 yêu cầu tài nguyên $(0,2,0)$ thì có được đáp ứng không?