Bế tắc

Thuật toán Banking

Đề bài:

Một hệ thống có 5 tiến trình với trạng thái tài nguyên như sau:

Dùng giải thuật banking để trả lời các câu hỏi sau:

- Xác định ma trận Need của các tiến trình ?
- Hệ thống có an toàn hay ko ? (nếu có hãy cho biết một chuỗi an toàn)

	Max				Allocation			
	A	B	C	D	A	B	C	D
P1	4	4	1	3	2	0	1	2
P2	0	6	6	6	0	0	1	2
P3	5	4	5	6	1	3	5	2
P4	0	6	5	2	0	6	3	2
P5	1	6	5	0	1	0	4	0

Available			
A	B	C	D
2	6	2	1

Cách làm chung:

Xác định ma trận Need (= Max – Allocation)

So sánh ma trận Available với Need_P1

a. Nếu Available ≥ Need_P1:

- Thực thi tiến trình P1 và giải phóng tài nguyên đã phân bổ cho P1 (Allocation_P1)
- Cập nhật thông tin trong ma trận Available = Available + Allocation_P1
- Thực hiện bước tiếp theo

b. Nếu Available < Need_P1, thực hiện bước tiếp theo

So sánh ma trận Available với Need_P2, lặp lại quá trình so sánh như ở trên

Tiếp tục thực hiện quá trình cho đến khi so sánh hết các tiến trình

Lưu ý: Mỗi lần cập nhật Available thì phải so sánh lại với các tiến trình chưa được hoàn thành

- Nếu đã thực hiện hết mà vẫn còn tiến trình ko được thực hiện, thì hệ thống bị Bế tắc

- Nếu ko có Bế tắc thì ghi kết quả thu được (ít nhất một chuỗi an toàn)

Cách làm chi tiết:

Xác định ma trận Need = Max – Allocation

Max				
	A	B	C	D
P1	4	4	1	3
P2	0	6	6	6
P3	5	4	5	6
P4	0	6	5	2
P5	1	6	5	0

Allocation				
	A	B	C	D
P1	2	0	1	2
P2	0	0	1	2
P3	1	3	5	2
P4	0	6	3	2
P5	1	0	4	0

Need = Max - Allocation					
	A	B	C	D	
P1	2	4	0	1	Need_P1
P2	0	6	5	4	Need_P2
P3	4	1	0	4	Need_P3
P4	0	0	2	0	Need_P4
P5	0	6	1	0	Need_P5

- Xác định Need_Pi (với i = 1, 2, 3, 4, 5). VD: Need_P1 = [2 4 0 1]
- Trong đó Max_P1 = [4 4 1 3]; Allocation_P1 = [2 0 1 2]

Thực hiện phép trừ hai ma trận: Available - Need_P1

- Ma trận thu được là [0 2 2 0] > 0
- Thực hiện cấp tài nguyên và thực hiện xong P1
- Giải phóng tài nguyên đã phân bổ cho P1 (Allocation_P1)
- Cập nhật thông tin trong ma trận Available sau khi kết thúc P1. Avaible (P1) = Available (T0) + Allocation_P1
- Kiểm tra xem Availeble (P1) có thể đủ cấp cho việc thực hiện P2 ko?

Available (T0)			
A	B	C	D
2	6	2	1

Need_P1			
A	B	C	D
2	4	0	1

Available (T0) - Need_P1			
A	B	C	D
0	2	2	0

Available (T0) = Available ban đầu (đề bài cho)

Ma trận kq > 0, nên có đủ tài nguyên thực hiện P1

Allocation_P1			
A	B	C	D
2	0	1	2

Thực hiện xong P1, giải phóng Allocation_P1

Available (P1)			
A	B	C	D
4	6	3	3

Need_P2			
A	B	C	D
0	6	5	4

Available (P1) - Need_P2			
A	B	C	D
4	0	-2	-1

Available sau khi thực hiện xong P1, Available (P1) = Avaible (T0) + Allocation_P1

So sánh Available (P1) với Need_P2

Không đủ tài nguyên thực hiện P2

Thực hiện phép trừ hai ma trận: Available (P1) - Need_P2

- Ma trận thu được là [4 0 -2 -1] < 0
- Không đủ tài nguyên thực hiện P2
- Kiểm tra xem Availeble (P1) có thể đủ cấp cho việc thực hiện P3 ko?

Thực hiện phép trừ hai ma trận: Available (P1) - Need_P3

- Ma trận thu được là [0 5 3 -1] < 0
- Không đủ tài nguyên thực hiện P3
- Kiểm tra xem Availeble (P1) có thể đủ cấp cho việc thực hiện P4 ko?

Available (P1)			
A	B	C	D
4	6	3	3

Need_P3			
A	B	C	D
4	1	0	4

Available (P1) - Need_P3			
A	B	C	D
0	5	3	-1

So sánh Available (P1) với Need_P3
Ma trận kq < 0, nên không đủ tài nguyên thực hiện P3

Available (P1)			
A	B	C	D
4	6	3	3

Need_P4			
A	B	C	D
0	0	2	0

Available (P1) - Need_P4			
A	B	C	D
4	6	1	3

So sánh Available (P1) với Need_P4
Ma trận kq > 0, nên đủ tài nguyên thực hiện P4

Thực hiện phép trừ hai ma trận: Available (P1) - Need_P4

- Ma trận thu được là [4 6 1 3] > 0
- Thực hiện cấp tài nguyên và thực hiện xong P4
- Giải phóng tài nguyên đã phân bổ cho P4 (Allocation_P4)
- Cập nhật thông tin trong ma trận Available sau khi kết thúc P4. Avaible (P14) = Available (P1) + Allocation_P4
- Kiểm tra xem Availeble (P14) có thể đủ cấp cho việc thực hiện P5 ko?

Available (P1)			
A	B	C	D
4	6	3	3

Need_P4			
A	B	C	D
0	0	2	0

Available (P1) - Need_P4			
A	B	C	D
4	6	1	3

So sánh Available (P1) với Need_P4
Ma trận kq > 0, nên đủ tài nguyên thực hiện P4

Allocation_P4			
A	B	C	D
0	6	3	2

Thực hiện xong P4, giải phóng Allocation_P4

Available (P14)			
A	B	C	D
4	12	6	5

Need_P5			
A	B	C	D
0	6	1	0

Available (P14) - Need_P5			
A	B	C	D
4	6	5	5

Available sau khi thực hiện xong P4, Available (P14) = Avaible (P1) + Allocation_P4
So sánh Available (P14) với Need_P5
Ma trận kq > 0, nên đủ tài nguyên thực hiện P5

Thực hiện phép trừ hai ma trận: Available (P14) - Need_P5

- Ma trận thu được là [4 6 5 5] > 0
- Thực hiện cấp tài nguyên và thực hiện xong P5
- Giải phóng tài nguyên đã phân bổ cho P5 (Allocation_P5)
- Cập nhật thông tin trong ma trận Available sau khi kết thúc P5. Avaible (P145) = Available (P14) + Allocation_P5
- Kiểm tra xem Availeble (P145) có thể đủ cấp cho việc thực hiện P2 ko?

Available (P14)			
A	B	C	D
4	12	6	5

Need_P5			
A	B	C	D
0	6	1	0

Available (P14) - Need_P5			
A	B	C	D
4	6	5	5

Available sau khi thực hiện xong P4, Available (P14) = Avaible (P1) + Allocation_P4
So sánh Available (P14) với Need_P5
Ma trận kq > 0, nên đủ tài nguyên thực hiện P5

Allocation_P5			
A	B	C	D
1	0	4	0

Thực hiện xong P5, giải phóng Allocation_P5

Available (P145)			
A	B	C	D
5	12	10	5

Need_P2			
A	B	C	D
0	6	5	4

Available (P145) - Need_P2			
A	B	C	D
5	6	5	1

Available sau khi thực hiện xong P5, Available (P145) = Avaible (P14) + Allocation_P5
So sánh Available (P145) với Need_P2
Ma trận kq > 0, nên đủ tài nguyên thực hiện P2

Thực hiện phép trừ hai ma trận: Available (P145) - Need_P2

- Ma trận thu được là [5 6 5 1] > 0
- Thực hiện cấp tài nguyên và thực hiện xong P2
- Giải phóng tài nguyên đã phân bổ cho P2 (Allocation_P2)
- Cập nhật thông tin trong ma trận Available sau khi kết thúc P2. Avaible (P1452) = Available (P145) + Allocation_P2
- Kiểm tra xem Availeble (P1452) có thể đủ cấp cho việc thực hiện P3 ko?

Available (P145)			
A	B	C	D
5	12	10	5

Need_P2			
A	B	C	D
0	6	5	4

Available (P145) - Need_P2			
A	B	C	D
5	6	5	1

Available sau khi thực hiện xong P5, Available (P145) = Avaible (P14) + Allocation_P5
So sánh Available (P145) với Need_P2
Ma trận kq > 0, nên đủ tài nguyên thực hiện P2

Allocation_P2			
A	B	C	D
0	0	1	2

Thực hiện xong P2, giải phóng Allocation_P2

Available (P1452)			
A	B	C	D
5	12	11	7

Need_P3			
A	B	C	D
4	1	0	4

Available (P1452) - Need_P3			
A	B	C	D
1	11	11	3

Available sau khi thực hiện xong P2, Available (P1452) = Avaible (P145) + Allocation_P2
So sánh Available (P1452) với Need_P3
Ma trận kq > 0, nên đủ tài nguyên thực hiện P3

Thực hiện cấp tài nguyên và thực hiện xong P3

- Giải phóng tài nguyên đã phân bổ cho P3 (Allocation_P3)

Available (P1452)			
A	B	C	D
5	12	11	7

Need_P3			
A	B	C	D
4	1	0	4

Available (P1452) - Need_P3			
A	B	C	D
1	11	11	3

Available sau khi thực hiện xong P2, Available (P1452) = Avaible (P145) + Allocation_P2
So sánh Available (P1452) với Need_P3
Ma trận kq > 0, nên đủ tài nguyên thực hiện P3

Allocation_P3			
A	B	C	D
1	3	5	2

Thực hiện xong P3, giải phóng Allocation_P3

Available (P14523)			
A	B	C	D
6	15	16	9

Hệ thống an toàn. Chuỗi an toàn là: P1 - P4 - P5 - P2 - P3

Kết luận:

- Hệ thống an toàn.
- Chuỗi an toàn là: P1 - P4 - P5 - P2 - P3

Bài tập SV làm nộp

Đề bài:

Một hệ thống có 5 tiến trình với trạng thái tài nguyên như sau:

Dùng giải thuật banking để trả lời các câu hỏi sau:

- Xác định ma trận Need của các tiến trình ?
- Hệ thống có an toàn hay ko ? (nếu có hãy cho biết một chuỗi an toàn)

	MAX				ALLOCATION				AVAILABLE			
	A	B	C	D	A	B	C	D	A	B	C	D
P1	5	3	2	2	3	0	2	1	2	5	3	4
P2	1	5	7	5	1	1	0	1				
P3	4	5	4	7	1	3	4	3				
P4	0	5	4	1	0	5	2	0				
P5	2	7	4	0	2	2	3	0				