Có 4 ổ băng từ và 3 tiến trình P1, P2, P3 yêu cầu cấp phát nhiều nhất theo thứ tự là: 2, 4, 2. Hiện tại P1, P2, P3 đã được cấp phát theo thứ tự là: 1, 2, 0.

a)chứng minh trạng thái này an toàn

Ta có:

Available = 4 - 3 =1

Need = Max- Allocation

P[i] Allocation Max Need Available

P1 1 2 1 1

P2 2 4 2

P3 0 2 2

Xét tại thời điểm Ti

Work >= Need[i] P[i] Allocation[i]

1 1 P1 1

2 2 P2 2

4 2 P3 0

Vậy tồn tại chuỗi an toàn < p1, p2, p3 > .Suy ra trạng thái hệ thống ở thời điểm Ti là an toàn.

b) Xác định có nên đáp ứng hay không yêu cầu xin thêm 1 ổ nữa của p2.

Giả sử p2 bây giờ nêu yêu cầu mới là 1

Yêu cầu này phải thỏa các điều kiên sau:

1. Request1 <= Need1 vì 1 <= 1

2. Request1 <= Available vì 1 <= 1

Như vậy đủ để cấp nhưng nếu đáp ứng, hệ sẽ chuyển sang trạng thái không an toàn do không tồn tại chuỗi an toàn.

thuật toán Banker chứng minh không tồn tại trạng thái an toàn thay vì phải tính ra hết tất cả các trường hợp thì mình có ý này :

ta cứ đi theo một hướng bất kì, miễn nó còn tồn tại tiến trình kế tiếp cho tới khi nó không còn tiến trình nào có thể thêm vào chuỗi nữa.
Giả sử ta có chuỗi < P1,P0,P3> rồi còn lại 2 tiến trình P4 và P2 không đủ tài nguyên để đi vào. Lúc này ta có thể suy ra không tồn tại chuỗi an toàn.
Vì nếu giả sử tồn tại một chuỗi an toàn chẳng hạn là
thì ta bắt đầu xét từ tiến trình Pa xem Pa có thể thêm vào sau P3 trong chuỗi trên hay không, vì Pa là tiến trình đầu tiên của chuỗi an toàn nên nó không cần sự trao trả tài nguyên của tiến trình nào trước nó. Nếu Pa không thêm vào được thì do Pa đã tồn tại sau P3 rồi có thể là P1 hay P0, và như vậy nếu Pa đã thêm vào rồi thì ta có thể thêm Pb vì Pb cần Pa thực hiện xong thì nó mới có thể thực hiện. Cứ như vậy thì suy ra sau P3 luôn tồn tại một tiến trình có thể thêm vào --> Nếu tồn tại một chuỗi an toàn thì ta luôn tìm ra chuỗi an toàn khác cho dù có đi từ tiến trình nào trước và chọn lựa như thế nào. Ngược lại thì --- > không tồn tại chuỗi an toàn.

1. Một hệ thống có 5 tiến trình với tình trạng tài nguyên như sau:
2. Process Allocation Max Available
3. A B C D A B C D A B C D
4. P0 0 0 1 2 0 0 1 2 1 5 2 0
5. P1 1 0 0 0 1 7 5 0
6. P2 1 3 5 4 2 3 5 6
7. P3 0 6 3 2 0 6 5 2
8. P4 0 0 1 4 0 6 5 6

9. Duøng thuaät giaûi Nhaø baêng ñeå:
10. a. Chứng minh trạng thái này an toàn. (1 điểm)
11. b. Xác định có nên đáp ứng yêu cầu (0, 4, 3, 0) của P1 ? (1 điểm)
12. Giải:
13. a. Xét tại thời điểm T0 mà 5 tiến trình được cấp phát như đề bài ta có:
14. Need[i] = Max[i] – Allocation[i]
15. Process Need
16. A B C D
17. P0 0 0 0 0
18. P1 0 7 5 0
19. P2 1 0 0 2
20. P3 0 0 2 0
21. P4 0 4 4 2
22.
23. Tìm chuỗi an toàn:
24. Work >= Need[i] P[i] Allocation[i]
25. A B C D A B C D A B C D
26. 1 5 2 0 0 0 0 0 P0 0 0 1 2
27. 1 5 3 2 1 0 0 2 P2 1 3 5 4
28. 2 8 8 6 0 0 2 0 P3 0 6 3 2
29. 2 14 11 8 0 4 4 2 P4 0 1 1 4
30. 2 15 12 12 0 7 5 0 P1 1 0 0 0
31. Vậy tại thời điểm T0 tồn tại chuỗi an toàn {P0, P2, P3, P4, P1}. Suy ra, hệ thống tại thời điểm T0 ở trạng thái an toàn.
32.
33. Available vì tài nguyên C trong hệ thống chỉ còn 2 mà yêu cầu 3. Do vậy, không thể cấp phát thêm (0, 4, 3, 0) cho P1 được. ≤ Need1, nhưng không thoả điều kiện: Request1 ≤ b. Ta thấy, yêu cầu thêm (0, 4, 3, 0) của P1 thoả điều kiện Request1

Xem giúp mình 2 bài tập về giải thuật tìm chuỗi an toàn và giải thuật cấp phát tài nguyên…..

Bài 1: Cho 5 tiến trình và các thông số sau: (Đây là bài thầy mình đã chữa).

Process Allocation Max Available
ABC ABC ABC
P0 010 753 332
P1 200 322
P2 302 902
P3 211 222
P4 002 433
1. Hãy chỉ ra trạng thái an toàn.
Giải:
Để xét độ an toàn của hệ ta tính ma trận Need theo công thức:

Need = Max – Allocation
Prrocess Need
ABC
P0 743
P1 122
P2 600
P3 011
P4 431

Theo thuật toán kiểm tra tính an toàn của hệ ta có:
Work := Available = (3,3,2)
Finish[i]=False với i= 0,1,2,3,4.
Xét :
Need[0] >= Work => finish[0]=false
Need[2] >= Work => finish[0]=false
Need[4] >= Work => finish[0]=false
{Có phải vì :
Need[0]=(7,4,3) > Work=(3,3,2)
Need[2]=(6,0,0) > Work=(3,3,2)
Need[4]=(4,3,1) > Work=(3,3,2)}

Need[1] < = Work => Finish[1] = True và Work=Work + Allocation[1] = (3,3,2) + (2,0,0) = (5,3,2).
Need[3] < = Work => Finish[3] = True và Work=Work + Allocation[3] = (5,3,2) + (2,1,1) = (7,4,3).
Need[0] < = Work => Finish[0] = True và Work=Work + Allocation[0] = (7,4,3) + (0,1,0) = (7,5,3).
Need[2] < = Work => Finish[2] = True và Work=Work + Allocation[2] = (7,5,3) + (3,0,2) = (10,5,5).
Need[4] < = Work => Finish[4] = True và Work=Work + Allocation[4] = (10,5,5) + (0,0,2) = (10,5,7).
{Có phải vì :
Need[1],Need[3] < Work Ta xét lần lượt từ trên xuống.
Need[1]=(1,2,2) < Work = (3,3,2)
Need[3]=(0,1,1) < Work = (3,3,2)
Sau đó quay lại xét các Need[0],Need[2],Need[4] lần lượt từ trên xuống dưới.
Need[0]=(7,4,3) < Work = (3,3,2)
Need[2]=(6,0,0) < Work = (3,3,2)
Need[4]=(4,3,1) < Work = (3,3,2)}
Như vậy finish[i] = true với I = 1,3,0,2,4 dẫn đến dãy tiến trình P1, P3, P0, P2, P4 là dãy an toàn => hệ thống ở trạng thái an toàn.

Bài 2 : Cho 5 tiến trình và các thông số sau: (Bài này em tự làm)

Process Allocation Max Available
ABC ABC ABC
P0 252 753 253
P1 221 323
P2 511 932
P3 121 223
P4 412 522
Hãy chỉ ra trạng thái an toàn.
Giải:
Để xét độ an toàn của hệ ta tính ma trận Need theo công thức:
Need = Max – Allocation
Prrocess Need
ABC
P0 501
P1 102
P2 421
P3 102

P4 110

Theo thuật toán kiểm tra tính an toàn của hệ ta có:
Work := Available = (2,5,3)
Finish[i]=False với i= 0,1,2,3,4.
Xét :
Need[0] >= Work => finish[0]=false
Need[2] >= Work => finish[0]=false
{Có phải vì :
Need[0]=(5,0,1) > Work=(2,5,3)
Need[2]=(4,2,1) > Work=(2,5,3)}
Need[1] < = Work => Finish[1] = True và Work=Work + Allocation[1] = (2,5,3) + (2,2,1) = (4,7,4).
Need[3] < = Work => Finish[3] = True và Work=Work + Allocation[3] = (4,7,4) + (1,2,1) = (5,9,5).
Need[4] < = Work => Finish[4] = True và Work=Work + Allocation[4] = (5,9,5) + (4,1,2) = (9,10,7).
Need[0] < = Work => Finish[0] = True và Work=Work + Allocation[0] = (9,10,7) + (2,5,2) = (11,15,9).
Need[2] < = Work => Finish[2] = True và Work=Work + Allocation[2] = (11,15,9) + (5,1,1) = (16,16,10).
{Có phải vì :
Need[1],Need[3],Need[4] < Work Ta xét lần lượt từ trên xuống.
Need[1]=(1,0,2) < Work = (2,5,3)
Need[3]=(1,0,2) < Work = (2,5,3)
Need[4]=(1,1,0)< Work = (2,5,3)
Sau đó quay lại xét các Need[0],Need[2] lần lượt từ trên xuống dưới.
Need[0]=(5,0,1) < Work = (2,5,3)
Need[2]=(4,2,1) < Work = (2,5,3)}
Như vậy finish[i] = true với I = 1,3,4,0,2 dẫn đến dãy tiến trình P1, P3, P4, P0, P2 là dãy an toàn => hệ thống ở trạng thái an toàn.

Bài tập này mình lấy đề bài trên mạng và họ cho đáp án trạng thái an toàn là : P1, P2, P0, P3, P4 vậy em đã làm sai ở đâu ạ????

Cách giải bài tập của mình như vậy về thuật toán và cách hiểu như vậy có sai ko ????
Mình mong các bác xem giúp em với ạ.