







>>> COURSE MATERIAL <<<

INTRODUCTION TO OPERATING SYSTEM / Course ID 502047

BÀI TẬP CHƯƠNG 8 – TẮC NGHẼN

Hướng dẫn tự học, chỉ sử dụng để tham khảo và có thể thay đổi mà không báo trước.

| | | |
|---|-----------------------------------|---------------------------------|
|  | Programming Exercises | --3-----9-----1--45-78-01 |
|  | Exam multichoice questions | 123-56--9-12-4-67---1234--78--- |
|  | Discussion questions | 12-4-6-8---2----7--0----5----01 |
|  | Just for reference | ----- |

8.1 Chỉ ra 03 ví dụ của tắc nghẽn không liên quan đến môi trường máy tính điện tử. Ví dụ như: các xe kẹt ở giao lộ.

8.2 Giả định rằng một hệ thống đang ở trạng thái không an toàn. Hãy chứng minh rằng một tiểu trình vẫn có thể hoàn thành quá trình thực thi mà không bị rơi vào trạng thái tắc nghẽn.

8.3 Cho thông tin cấp phát trong một hệ thống như sau:

| | <i>Allocation</i> | <i>Max</i> | <i>Available</i> |
|-----------|-------------------|----------------|------------------|
| Process | <i>A B C D</i> | <i>A B C D</i> | <i>A B C D</i> |
| <i>T0</i> | 0 0 1 2 | 0 0 1 2 | 1 5 2 0 |
| <i>T1</i> | 1 0 0 0 | 1 7 5 0 | |

| | | | |
|-------|---------|---------|--|
| T_2 | 1 3 5 4 | 2 3 5 6 | |
| T_3 | 0 6 3 2 | 0 6 5 2 | |
| T_4 | 0 0 1 4 | 0 6 5 6 | |

Bảng 1. Dữ liệu câu hỏi 8.2

Áp dụng giải thuật Nhà băng để trả lời các câu hỏi sau:

- Giá trị các phần tử trong ma trận Need?
- Hệ thống có đang ở trong trạng thái an toàn không?
- Tiêu trình T_1 yêu cầu cần cấp thêm (0,4,2,0), hệ thống có thể đáp ứng ngay lập tức không?

8.4 Một phương thức khả thi để phòng ngừa tắc nghẽn là đặt ra một tài nguyên đơn lẻ, thứ tự sắp xếp trước và nó cần được yêu cầu trước tiên (trước khi yêu cầu tài nguyên khác). Ví dụ, nếu nhiều tiêu trình cố gắng truy cập các đối tượng đồng bộ A, B, C D, E, tắc nghẽn có thể xảy ra. (Các đối tượng đồng bộ hóa đó có thể bao gồm các mutex, các semaphore, biến số điều kiện và tương tự). Chúng ta có thể ngăn chặn sự tắc nghẽn bằng cách thêm một đối tượng thứ sáu F. Bất cứ một tiêu trình tại bất cứ lúc nào muốn có khóa đồng bộ cho bất kỳ đối tượng nào (trong số A đến E), nó phải yêu cầu khóa cho đối tượng F trước tiên. Giải pháp này được biết đến như là một ngăn chặn: các khóa cho các đối tượng A đến E được chứa trong khóa của đối tượng F.

So sánh thiết kế này với thiết kế chờ vòng tròn của Phần 8.5.4.

8.5 Chứng minh rằng thuật toán An toàn được trình bày trong Mục 8.6.3.1 cần thực thi thứ tự có $m \times n^2$ thao tác.

8.6 Hãy xem xét một hệ thống máy tính chạy 5.000 công việc mỗi tháng và không có thiết kế phòng ngừa tắc nghẽn hoặc tránh tắc nghẽn. Tắc nghẽn xảy ra khoảng hai lần mỗi tháng và người vận hành phải chấm dứt rồi chạy lại khoảng mười tác vụ mỗi lần tắc nghẽn. Mỗi công việc có giá trị khoảng \$2 (tính theo thời gian CPU) và các tác vụ bị chấm dứt có xu hướng đã hoàn thành được một nửa khi chúng bị hủy bỏ.

Một lập trình viên hệ thống đã ước tính rằng thuật toán tránh tắc nghẽn (như thuật toán Nhà băng) có thể được cài đặt trong hệ thống với mức tăng khoảng 10% thời gian thực hiện trung bình cho mỗi tác vụ. Vì máy tính đó hiện có 30% thời gian nhàn rỗi, tất cả 5.000 tác vụ mỗi tháng vẫn có thể được chạy, mặc dù thời gian quay vòng sẽ tăng trung bình khoảng 20%.

- Các lý lẽ bảo vệ việc cài đặt thuật toán tránh tắc nghẽn là gì?
- Các lý lẽ chống lại việc cài đặt thuật toán tránh tắc nghẽn là gì?

8.7 Một hệ thống có thể phát hiện ra rằng một số tiêu trình của nó đang bị cạn kiệt tài nguyên không? Nếu cho rằng có, hãy giải thích cần hiện thực như thế nào. Nếu cho rằng không, hãy giải thích cách hệ thống giải quyết vấn đề cạn kiệt tài nguyên ra sao.

8.8 Xem xét các chính sách phân bổ tài nguyên sau đây. Các Yêu cầu và Giải phóng tài nguyên có thể tiến hành bất cứ lúc nào. Nếu yêu cầu tài nguyên không thể được đáp ứng vì tài nguyên không có sẵn, thì hệ thống sẽ kiểm tra bất kỳ tiêu trình nào bị chặn chờ tài nguyên. Nếu một tiêu trình bị chặn có các tài nguyên mong muốn, thì các tài nguyên này sẽ bị thu hồi và được trao cho tiêu trình đang yêu cầu. Vector tài nguyên mà tiêu trình bị chặn đang chờ được tăng lên một giá trị chính là các tài nguyên đã bị thu hồi.

Ví dụ: một hệ thống có ba loại tài nguyên và vector *Available* được khởi tạo với giá trị (4,2,2). Nếu tiêu trình T_0 yêu cầu (2,2,1), nó sẽ nhận được chúng. Nếu T_1 yêu cầu (1,0,1), nó sẽ nhận được chúng. Sau đó, nếu T_0 yêu cầu

(0,0,1), nó sẽ bị chặn (loại tài nguyên thứ ba không có sẵn vì đã cấp toàn bộ). Nếu bây giờ T2 yêu cầu (2,0,0), thì nó nhận được tài nguyên đang sẵn sàng còn lại (1,0,0), và phần đã cấp phát cho T0 (vì T0 bị chặn). Vector *Allocation* của T0 giảm xuống (1,2,1) và vector *Need* của nó tăng lên (1,0,1).

a. Tắc nghẽn có thể xảy ra không? Nếu câu trả lời có, thì hãy đưa ra một ví dụ. Nếu câu trả lời không, thì chỉ định điều kiện cần thiết để có tắc nghẽn nào đã không xảy ra.

b. Hệ thống có thể bị chặn vô hạn định [indefinite blocking] không? Giải thích câu trả lời.

8.9 Cho thông tin cấp phát trong một hệ thống như sau:

| | <i>Allocation</i> | <i>Max</i> |
|---------|-------------------|----------------|
| Process | <i>A B C D</i> | <i>A B C D</i> |
| T0 | 3 0 1 4 | 5 1 1 7 |
| T1 | 2 2 1 0 | 3 2 1 1 |
| T2 | 3 1 2 1 | 3 3 2 1 |
| T3 | 0 5 1 0 | 4 6 1 2 |
| T4 | 4 2 1 2 | 6 3 2 5 |

Bảng 2. Dữ liệu cho câu hỏi 8.9

Sử dụng giải thuật Nhà băng để xác định hệ thống đang ở tình trạng an toàn hay không? Nếu hệ thống đang trong tình trạng an toàn, chỉ ra một thứ tự để các tiến trình hoàn thành. Nếu hệ thống không an toàn, hãy mô tả và giải thích.

Thực hiện câu hỏi cho từng tình huống sau đây.

a. *Available* = (0, 3, 0, 1)

b. *Available* = (1, 0, 0, 2)

8.10 Cho rằng chúng ta đã hiện thực thuật toán an toàn Tránh tắc nghẽn để xác định xem hệ thống có ở trạng thái an toàn hay không. Bây giờ nếu được yêu cầu thực hiện thuật toán Phát hiện tắc nghẽn. Có thể hay không thể thực hiện yêu cầu này bằng cách sử dụng mã thuật toán an toàn với chỉnh sửa thông số của $Max_i = Waiting_i + Allocation_i$, trong đó $Waiting_i$ là một vector chỉ định các tài nguyên cho tiến trình i đang chờ được cấp và $Allocation_i$ như đã được định nghĩa trong Phần 8.6?

Giải thích câu trả lời của bạn.

8.11 Có thể xảy ra một tắc nghẽn với một tiến trình đơn luồng duy nhất không? Giải thích câu trả lời.

8.12 Xem xét sự tắc nghẽn giao thông được mô tả trong hình sau.

a. Chỉ ra rằng bốn điều kiện cần thiết để có tắc nghẽn đã xuất hiện trong ví dụ này.

b. Nêu một quy tắc đơn giản để tránh tắc nghẽn trong giao lộ này.

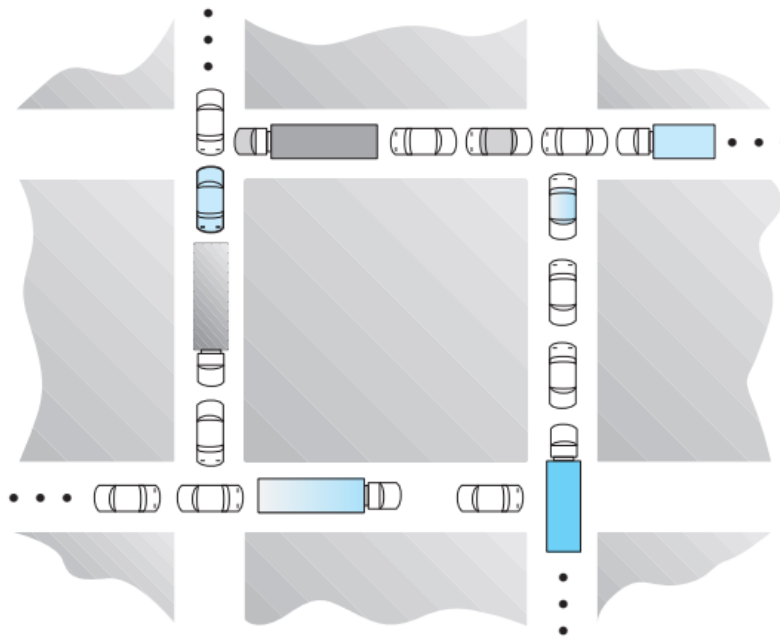


Figure 8.11 Traffic deadlock for Exercise 8.12.

- 8.13 Vẽ biểu đồ phân bổ tài nguyên minh họa sự tắc nghẽn từ chương trình ví dụ trong Hình 8.1 trong Phần 8.2.
- 8.14 Trong Phần 6.8.1, một kịch bản tắc nghẽn tiềm năng liên quan đến các quá trình P0 và P1 và semaphores S và Q đã được mô tả. Vẽ biểu đồ phân bổ tài nguyên minh họa tắc nghẽn theo kịch bản được trình bày trong phần đó.
- 8.15 Giả sử rằng một ứng dụng đa luồng chỉ sử dụng các khóa Bộ đọc - Bộ ghi để đồng bộ hóa. Áp dụng bốn điều kiện cần thiết cho tắc nghẽn, liệu tắc nghẽn vẫn có thể xảy ra nếu nhiều khóa Bộ đọc - Bộ ghi được sử dụng?
- 8.16 Chương trình được ví dụ trong Hình 8.1 không phải lúc nào cũng dẫn đến tắc nghẽn. Mô tả vai trò của bộ lập lịch CPU và cách thức mà nó có thể đóng góp vào tắc nghẽn trong chương trình này.
- 8.17 Trong Mục 8.5.4, một tình huống trong đó việc ngăn chặn tắc nghẽn bằng cách đảm bảo rằng tất cả các khóa được thu thập theo một thứ tự nhất định đã được mô tả. Tuy nhiên, trong tình huống đó, tắc nghẽn vẫn có thể xảy ra nếu hai tiểu trình đồng thời gọi hàm `transaction()`. Hãy khắc phục hàm `transaction()` để ngăn chặn tắc nghẽn.
- 8.18 Trong sáu biểu đồ phân bổ tài nguyên trong Hình 8.12, hình nào minh họa sự tắc nghẽn? Đối với những tình huống bị tắc nghẽn, hãy cung cấp chu trình chứa các tiến trình và tài nguyên. Trong trường hợp không có tình huống tắc nghẽn, hãy minh họa thứ tự mà các tiến trình có thể hoàn thành thực thi.

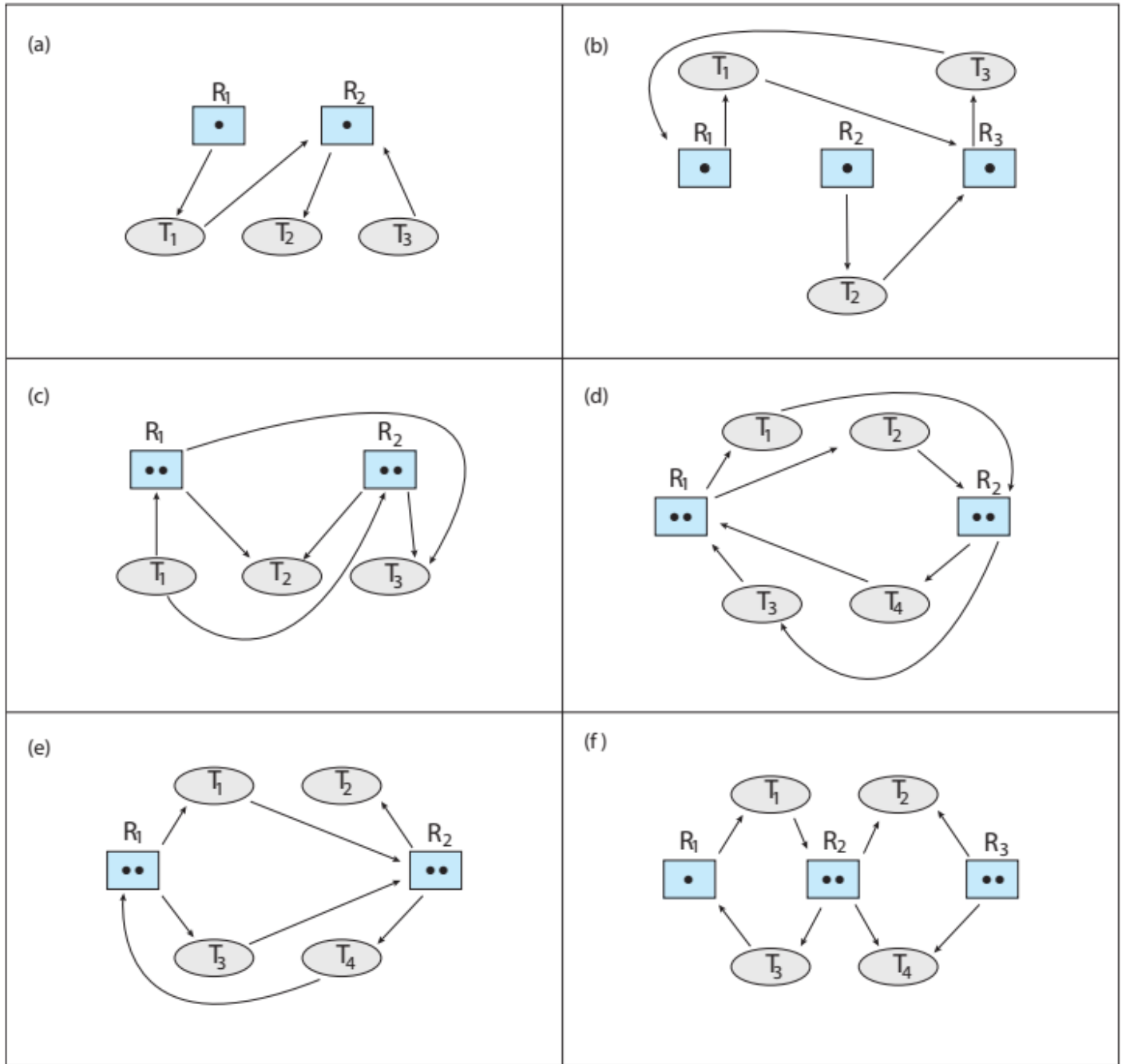


Figure 8.12 Resource-allocation graphs for Exercise 8.18.

8.19 So sánh biểu diễn chờ vòng tròn với các biểu diễn tránh tắc nghẽn khác (như thuật toán Nhà băng) với các vấn đề sau:

- Thời gian chạy.
- Thông lượng hệ thống.

8.20 Trong một hệ thống máy tính thời gian thực, các tài nguyên có sẵn cũng như nhu cầu dùng tài nguyên của tiểu trình đều nhất quán trong thời gian dài (hàng tháng). Tài nguyên bị phá vỡ hoặc được thay thế, các tiến trình và tiểu trình mới đến và đi, và các tài nguyên mới được mua và thêm vào hệ thống. Nếu tắc nghẽn được kiểm soát bởi thuật toán Nhà băng, thì những thay đổi nào sau đây có thể được thực hiện một cách an toàn hay không (mà không đưa ra khả năng tắc nghẽn), xem xét từng trường hợp sau đây?

- Tăng Available (tài nguyên mới được thêm vào).
- Giảm Available (tài nguyên bị xóa vĩnh viễn khỏi hệ thống).
- Tăng Max cho một tiểu trình (tiểu trình cần nhiều hơn tài nguyên hơn mức cho phép).
- Giảm Max cho một tiểu trình (tiểu trình quyết định không cần nhiều tài nguyên).
- Tăng số lượng trình.
- Giảm số lượng trình.

8.21 Cho thông tin cấp phát trong một hệ thống như sau:

| Process | Allocation | Max |
|-----------|----------------|----------------|
| | <i>A B C D</i> | <i>A B C D</i> |
| <i>T0</i> | 2 1 0 6 | 6 3 2 7 |
| <i>T1</i> | 3 3 1 3 | 5 4 1 5 |
| <i>T2</i> | 2 3 1 2 | 6 6 1 4 |
| <i>T3</i> | 1 2 3 4 | 4 3 4 5 |
| <i>T4</i> | 3 0 3 0 | 7 2 6 1 |

Bảng 3. Dữ liệu cho câu hỏi 8.21

Cho biết giá trị các phân tử của ma trận Need.

8.22 Hãy xem xét một hệ thống bao gồm bốn tài nguyên cùng loại được chia sẻ bởi ba tiểu trình, mỗi tiểu trình cần tối đa hai tài nguyên. Cho thấy hệ thống đang trong tình trạng không thể bị tắc nghẽn.

8.23 Hãy xem xét một hệ thống bao gồm tài nguyên cùng loại được chia sẻ bởi n tiểu trình. Một tiểu trình có thể yêu cầu hoặc chỉ phát hành một tài nguyên tại bất cứ thời điểm nào. Cho thấy hệ thống không có tắc nghẽn nếu hai điều kiện sau được đảm bảo:

- Nhu cầu tối đa của mỗi tiểu trình là giữa một tài nguyên và m tài nguyên.
- Tổng của tất cả các nhu cầu tối đa nhỏ hơn $m + n$.

8.24 Hãy xem xét phiên bản của bài toán Triết gia ăn tối trong đó đĩa được đặt ở giữa bàn và bất kỳ hai trong số chúng có thể được sử dụng bởi một triết gia. Giả sử rằng các yêu cầu cho đĩa được thực hiện cùng một lúc. Mô tả một quy tắc đơn giản để quyết định xem một yêu cầu lấy đĩa cụ thể có thể được đáp ứng ngay lập tức mà không gây ra tắc nghẽn hay không.

8.25 Xem xét lại cài đặt trong bài tập trước. Giả sử bây giờ mỗi triết gia cần 3 chiếc đĩa để ăn. Yêu cầu tài nguyên vẫn được phát hành tại cùng một thời điểm. Mô tả một số quy tắc đơn giản để xác định xem một yêu cầu cụ thể có thể được đáp ứng mà không gây ra bế tắc khi phân bổ đĩa hiện tại cho các nhà triết học hay không.

8.26 Chúng ta có thể có được giải thuật Nhà băng cho một loại tài nguyên duy nhất bằng cách giảm giảm số chiều của các ma trận xuống 1.

Hãy chỉ ra một ví dụ cho thấy không thể hiện thực được giải thuật Nhà băng cho hệ thống nhiều loại tài nguyên bằng cách áp dụng giải thuật Nhà băng cho một loại tài nguyên lên từng loại tài nguyên riêng lẻ.

8.27 Cho thông tin cấp phát trong một hệ thống như sau:

| <i>Allocation</i> | <i>Max</i> | |
|-------------------|----------------|---------|
| <i>A B C D</i> | <i>A B C D</i> | |
| <i>T0</i> | 1 2 0 2 | 4 3 1 6 |
| <i>T1</i> | 0 1 1 2 | 2 4 2 4 |
| <i>T2</i> | 1 2 4 0 | 3 6 5 1 |
| <i>T3</i> | 1 2 0 1 | 2 6 2 3 |
| <i>T4</i> | 1 0 0 1 | 3 1 1 2 |

Bảng 4. Dữ liệu cho câu hỏi 8.27

Sử dụng giải thuật Nhà băng để xác định hệ thống đang ở tình trạng an toàn hay không? Nếu hệ thống đang trong tình trạng an toàn, chỉ ra một thứ tự để các tiến trình hoàn thành. Nếu hệ thống không an toàn, hãy mô tả và giải thích.

Thực hiện câu hỏi cho từng tình huống sau đây.

- Available* = (2, 2, 2, 3)
- Available* = (4, 4, 1, 1)
- Available* = (3, 0, 1, 4)
- Available* = (1, 5, 2, 2)

8.28 Cho thông tin cấp phát trong một hệ thống như sau:

| <i>Allocation</i> | <i>Max</i> | <i>Available</i> | |
|-------------------|----------------|------------------|---------|
| <i>A B C D</i> | <i>A B C D</i> | <i>A B C D</i> | |
| <i>T0</i> | 3 1 4 1 | 6 4 7 3 | 2 2 2 4 |
| <i>T1</i> | 2 1 0 2 | 4 2 3 2 | |
| <i>T2</i> | 2 4 1 3 | 2 5 3 3 | |
| <i>T3</i> | 4 1 1 0 | 6 3 3 2 | |
| <i>T4</i> | 2 2 2 1 | 5 6 7 5 | |

Bảng 5. Dữ liệu cho câu hỏi 8.28

Trả lời các câu hỏi sau đây bằng cách áp dụng thuật toán Nhà băng:

- Chỉ ra rằng hệ thống ở trạng thái an toàn bằng cách chứng minh thứ tự trong đó các tiến trình có thể hoàn thành.
- Nếu tiến trình T4 yêu cầu cấp thêm (2, 2, 2, 4), yêu cầu có thể được chấp thuận ngay không?
- Nếu tiến trình T2 yêu cầu cấp thêm (0, 1, 1, 0), yêu cầu có thể được chấp thuận ngay không?
- Nếu tiến trình T3 yêu cầu cấp thêm (2, 2, 1, 2), yêu cầu có thể được chấp thuận ngay không?

8.29 Giả định lạc quan được đưa ra trong thuật toán phát hiện tắc nghẽn là gì? Giả định này sẽ bị vi phạm trong trường hợp nào?

8.30 Một cây cầu chỉ có một làn đường nối hai làng Vermont ở Bắc và Nam. Nông dân ở hai làng sử dụng cây cầu này để giao sản phẩm của họ đến thị trấn lân cận. Cây cầu có thể bị tắc nghẽn nếu một nông dân ở phía bắc và phía nam lên cầu cùng lúc, cùng thời điểm. (Nông dân Vermont buống binh nên sẽ không lùi bước). Sử dụng semaphores và / hoặc khóa mutex, thiết kế một thuật toán bằng mã giả để ngăn ngừa tắc nghẽn. Ban đầu, đừng quan tâm về vấn đề cạn kiệt tài nguyên (do nhiều nông dân từ một hướng liên tục đến cầu và phía bên kia phải chờ vô hạn định).

8.31 Sửa đổi giải pháp của Bài tập 8.30 để loại bỏ vấn đề cạn kiệt tài nguyên.