Chương 4: Quản lý tiến trình

Hệ điều hành

ThS. Đinh Xuân Trường

truongdx@ptit.edu.vn

Posts and Telecommunications Institute of Technology

Faculty of Information Technology 1

CNTT1

Học viện Công nghệ Bưu chính Viễn thông

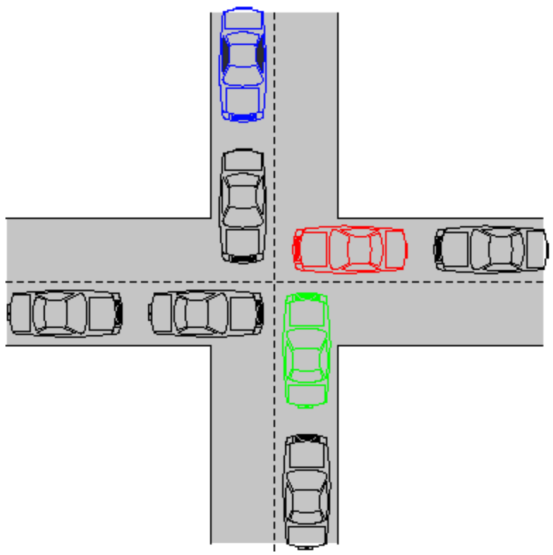
August 15, 2022

Mục lục

Nội dung chương 4

1. Các khái niệm liên quan đến tiến trình
2. Luồng - Thread
3. Điều độ tiến trình
4. Đồng bộ hóa các tiến trình đồng thời
5. Tình trạng bế tắc và đói

Bế tắc là tình trạng một nhóm tiến trình có cạnh tranh về tài nguyên hay có hợp tác phải dừng vô hạn
Do tiến trình phải chờ đợi một sự kiện chỉ có thể sinh ra bởi tiến trình khác cũng đang trong trạng thái chờ đợi



Ví dụ: Hai tiến trình P và Q thực hiện đồng thời. Mỗi tiến trình cần sử dụng đồng thời hai tài nguyên X và Y trong một khoảng thời gian:

- X và Y chỉ có khả năng phục vụ 1 tiến trình tại 1 thời điểm
- Mỗi tiến trình cần thực hiện thao tác Request yêu cầu tài nguyên
- Thực hiện xong, tiến trình giải phóng tài nguyên bằng Release.

Tiến trình P	**Tiến trình Q**
…	…
Request X	Request Y
…	…
Request Y	Request X
…	…
Release X	Release Y
…	…
Release Y	Release X
…	…

Giả sử do kết quả điều độ, thứ tự thực hiện của hai tiến trình diễn ra như sau:

...
P: Request X
Q: Requets Y
...
P: Request Y
Q: Request X
...

- Thao tác "P:Request Y"sẽ khiến P phải chờ cho đến khi Q giải phóng Y.
- Q phải dừng lại ở thao tác "Q: Request X"để đợi tài nguyên X do P đang giữ.
- *Kết quả hai tiến trình rơi vào bế tắc.*

Tiến trình đồng thời hay còn được gọi *tiến trình tương tranh* là các tiến trình cùng tồn tại.
Quản lý tiến trình đồng thời bao gồm các vấn đề :

- Liên lạc giữa các tiến trình
- Cạnh tranh và chia sẻ tài nguyên
- Phối hợp và đồng bộ hóa các tiến trình
- Vấn đề bế tắc
- Đói/Thiếu tài nguyên (starvation)

Tình trạng bế tắc xẩy ra khi 4 điều kiện sau đồng thời thỏa mãn:

- **Loại trừ tương hỗ**: có tài nguyên nguy hiểm, tại 1 thời điểm duy nhất 1 tiến trình sử dụng. Tiến trình khác yêu cầu tài nguyên này sẽ phải dừng lại chờ khi tài nguyên được giải phóng.
- **Giữ và chờ**: tiến trình giữ tài nguyên trong khi chờ đợi, chẳng hạn chờ đợi để được cấp thêm tài nguyên khác.
- **Không có phân phối lại (no preemption)**: tài nguyên do tiến trình giữ không thể phân phối lại cho tiến trình khác trừ khi tiến trình đang giữ tự nguyện giải phóng tài nguyên
- **Chờ đợi vòng tròn**: tồn tại nhóm tiến trình P1, P2, . . . , Pn sao cho P1 chờ đợi tài nguyên do P2 đang giữ, P2 chờ tài nguyên do P3 đang giữ, . . . , Pn chờ tài nguyên do P1 đang giữ.

Giải quyết vấn đề bế tắc theo cách:

- **Ngăn ngừa** (deadlock prevention): đảm bảo để một trong bốn điều kiện xẩy ra bế tắc không bao giờ thỏa mãn
- **Phòng tránh** (deadlock avoidance): cho phép một số điều kiện bế tắc được thỏa mãn nhưng đảm bảo để không đạt tới điểm bế tắc
- P**hát hiện và giải quyết** (deadlock detection): cho phép bế tắc xẩy ra, phát hiện bế tắc và khôi phục hệ thống về tình trạng không bế tắc.

Để ngăn ngừa bế tắc thì phải đảm bảo ít nhất 1 trong 4 điều kiện không xảy ra:

- Loại trừ tương hỗ
- Giữ và chờ
- Không có phân phối lại
- Chờ đợi vòng tròn

Loại trừ tương hỗ:

- Có thể tránh loại trừ tương hỗ với những tài nguyên cho phép nhiều tiến trình sử dụng một lúc ví dụ các file trong chế độ đọc.
- Tuy nhiên, trên thực tế luôn tồn tài tài nguyên không có khả năng chia sẻ đồng thời như vậy.
- *Không thể ngăn ngừa điều kiện về loại trừ tương hỗ.*

Giữ và chờ: Có hai cách ngăn ngừa điều kiện này:

- Cách 1:
  - Yêu cầu tiến trình phải nhận đủ toàn bộ tài nguyên cần thiết trước khi thực hiện tiếp
  - Nếu không nhận đủ, tiến trình bị phong tỏa để chờ cho đến khi có thể nhận đủ tài nguyên
- Cách 2
  - Tiến trình chỉ được yêu cầu tài nguyên nếu không giữ tài nguyên khác
  - Trước khi yêu cầu thêm tài nguyên, tiến trình phải giải phóng tài nguyên đã được cấp và yêu cầu lại (nếu cần) cùng với tài nguyên mới.

Không có phân phối lại:

- Cách 1:
  - Khi một tiến trình yêu cầu tài nguyên nhưng không được do đã bị cấp phát, HDH sẽ thu hồi lại toàn bộ tài nguyên nó đang giữ
  - Tiến trình chỉ có thể thực hiện tiếp sau khi lấy được tài nguyên cũ cùng với tài nguyên mới yêu cầu
- Cách 2:
  - Khi tiến trình yêu cầu tài nguyên, nếu còn trống, sẽ được cấp phát ngay
  - Nếu tài nguyên do tiến trình khác giữ mà tiến trình này đang chờ cấp thêm tài nguyên thì thu hồi lại để cấp cho tiến trình yêu cầu
  - Nếu hai điều kiện trên đều không thỏa thì tiến trình yêu cầu tài nguyên phải chờ

Chờ đợi vòng tròn:

- Xác định thứ tự cho các dạng tài nguyên và chỉ cho phép tiến trình yêu cầu tài nguyên sao cho tài nguyên mà tiến trình yêu cầu sau có thứ tự lớn hơn tài nguyên mà nó yêu cầu trước
- Giả sử trong hệ thống có n dạng tài nguyên ký hiệu R1, R2, . . . , Rn
- Giả sử những dạng tài nguyên này được sắp xếp theo thứ tự tăng dần của chỉ số
- Nếu tiến trình đã yêu cầu một số tài nguyên dạng Ri thì sau đó tiến trình chỉ được phép yêu cầu tài nguyên dạng Rj nếu $j > i$
- Nếu tiến trình cần nhiều tài nguyên cùng dạng thì tiến trình phải yêu cầu tất cả tài nguyên dạng đó cùng một lúc

Giải pháp ngăn ngừa bế tắc tập trung vào việc sử dụng quy tắc hay ràng buộc để ngăn ngừa điều kiện xảy ra bế tắc.
Nhược điểm: làm cho việc sử dụng tài nguyên kém hiệu quả, giảm hiệu năng của tiến trình.
Ngăn ngừa bế tắc:

- Sử dụng quy tắc hay ràng buộc khi cấp phát tài nguyên để ngăn ngừa điều kiện xẩy ra bế tắc
- Sử dụng tài nguyên kém hiệu quả, giảm hiệu năng của tiến trình

Phòng tránh bế tắc:

- Cho phép 3 điều kiện đầu xẩy ra và chỉ đảm bảo sao cho trạng thái bế tắc không bao giờ đạt tới
- Mỗi yêu cầu cấp tài nguyên của tiến trình sẽ được xem xét và quyết định tùy theo tình hình cụ thể
- HĐH yêu cầu tiến trình cung cấp thông tin về việc sử dụng tài nguyên (số lượng tối đa tài nguyên tiến trình cần sử dụng)

Thuật toán người cho vay: *Thuật toán người cho vay vì sự tương tự giữa việc quyết định cho vay tiền với việc cấp phát tài nguyên trong máy tính.*

- Người cho vay giỏi là người cho vay được nhiều
- Tuy nhiên, khi cho vay vượt quá số tiền thực có sẽ gặp rủi ro do mỗi người vay không thể vay đủ số tiền cần thiết để phục vụ kinh doanh và do đó, cả ngân hàng và ngườii cho vay đều rơi vào tình trạng bế tắc.

- HĐH yêu cầu tiến trình cung cấp thông tin về việc sử dụng tài nguyên của mình và sử dụng thông tin này khi cấp phát
- Khi tiến trình muốn khởi tạo, thông báo dạng tài nguyên và số lượng tài nguyên tối đa cho mỗi dạng sẽ yêu cầu
- Nếu số lượng yêu cầu không vượt quá khả năng hệ thống, tiến trình sẽ được khởi tạo
- Trạng thái được xác định bởi tình trạng sử dụng tài nguyên hiện thời trong hệ thống:
  - Số lượng tối đa tài nguyên mà tiến trình yêu cầu:
    - Dưới dạng ma trận M[n][m]: n là số lượng tiến trình, m: số tài nguyên
    - M[i][j]: số lượng tài nguyên tối đa dạng j mà tiến trình i yêu cầu

- Trạng thái được xác định bởi tình trạng sử dụng tài nguyên hiện thời trong hệ thống:
  - Số lượng tài nguyên còn lại:
    - Dưới dạng vectơ A[m]
    - A[j] là số lượng tài nguyên dạng j còn lại và có thể cấp phát
  - Lượng tài nguyên đã cấp cho mỗi tiến trình:
    - Dưới dạng ma trận D[n][m]
    - D[i][j] là lượng tài nguyên dạng j đã cấp cho tiến trình i
  - Lượng tài nguyên còn cần cấp
    - Dưới dạng ma trận C[n][m]
    - C[i][j]=M[i][j]-D[i][j] là lượng tài nguyên dạng j mà tiến trình i còn cần cấp

Thuật toán người cho vay:
Trạng thái an toàn: trạng thái mà từ đó có ít nhất một phương án cấp phát sao cho bế tắc không xẩy ra.

- Cách phòng tránh bế tắc:
  - Khi tiến trình có yêu cầu cấp tài nguyên, hệ thống giả sử tài nguyên được cấp
  - Cập nhật lại trạng thái và xác định xem trạng thái đó có an toàn?
    - Nếu an toàn, tài nguyên sẽ được cấp thật
    - Ngược lại, tiến trình bị phong tỏa và chờ tới khi có thể cấp phát an toàn

Thuật toán người cho vay:

Thuật toán xác định trạng thái an toàn

1. Khai báo mảng W kích thước m và mảng F kích thước n. (F[i] chứa tình trạng tiến trình thứ i đã kết thúc hay chưa, W chứa lượng tài nguyên còn lại)

   Khởi tạo W=A và F[i]=false (i=0,…,n-1)

2. Tìm i sao cho:

   F[i] = false và C[i][j] ≤ W[j] với mọi j=0,…,m-1

   Nếu không có i như vậy thì chuyển sang bước 4

3. a) W = W + D[i]

   b) F[i] = true

   c) Quay lại bước 2

4. If F[i] = true với mọi i =0,…,n-1 thì trạng thái an toàn

   Else trạng thái không an toàn

Thuật toán người cho vay: *Hệ thống có 3 dạng tài nguyên X, Y, Z với số lượng ban đầu X=10, Y=5, Z=7*

- 4 tiến trình P1, P2, P3, P4 có yêu cầu tài nguyên tối đa cho trong bảng

		X	Y	Z
	P1	7	5	3
	P2	3	2	2
M	P3	9	0	2
	P4	2	2	2

Yêu cầu tối đa

- Xét trạng thái sau của hệ thống:
  - Là trạng thái an toàn
  - Có thể tìm ra phương pháp cấp phát không dẫn đến bế tắc ví dụ như: P2, P4, P1, P3

Thuật toán người cho vay: *Hệ thống có 3 dạng tài nguyên X, Y, Z với số lượng ban đầu X=10, Y=5, Z=7*

Trạng thái này là trạng thái an toàn do có thể tìm ra cách cấp phát không dẫn đến bế tắc, ví dụ theo thứ tự P2, P4, P3, P1.

D

	X	Y	Z
P1	0	1	0
P2	2	0	0
P3	3	0	2
P4	2	1	1

Đã cấp

A

X	Y	Z
3	3	4

Còn lại

C=M-D

	X	Y	Z
P1	7	4	3
P2	1	2	2
P3	6	0	0
P4	0	1	1

Còn cần cấp

Thuật toán người cho vay: *Hệ thống có 3 dạng tài nguyên X, Y, Z với số lượng ban đầu X=10, Y=5, Z=7*
P1 yêu cầu cấp phát 3 tài nguyên dạng Y, tức là yêu cầu = (0,3,0). Nếu yêu cầu thỏa mãn, hệ thống chuyển sang trạng thái:

X	Y	Z
3	0	4

Còn lại

	X	Y	Z
P1	7	1	3
P2	1	2	2
P3	6	0	0
P4	0	1	1

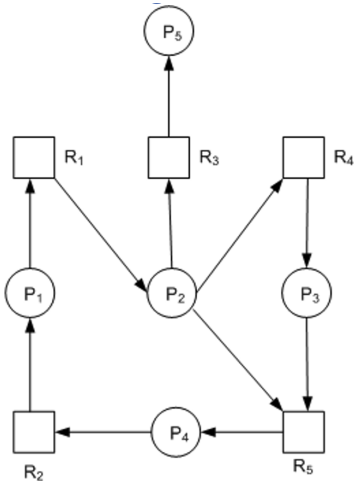
Còn cần cấp

Trạng thái không an toàn nên yêu cầu (0,3,0) bị từ chối.

- Hệ thống không thực hiện biện pháp ngăn ngừa/phòng tránh bế tắc có thể xảy ra
- Hệ thống định kỳ kiểm tra để phát hiện có tình trạng bế tắc hay không?
- Nếu có, hệ thống sẽ xử lý để khôi phục lại trạng thái không có bế tắc

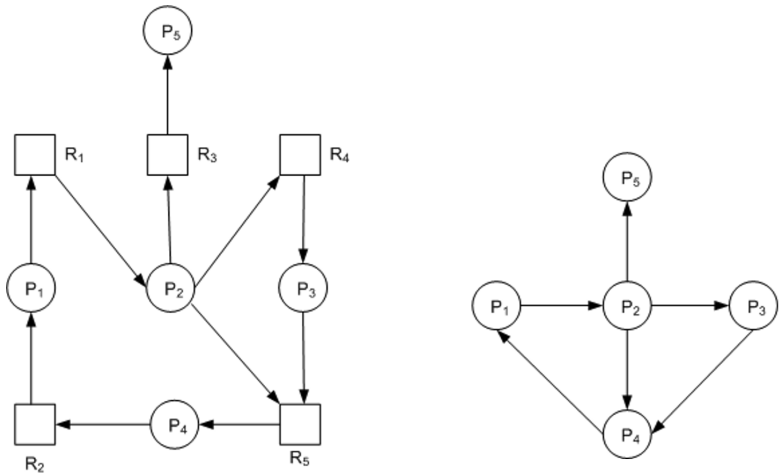
Phát hiện bế tắc:

- Trường hợp mỗi dạng tài nguyên chỉ có một tài nguyên duy nhất => sử dụng đồ thị biểu diễn quan hệ chờ đợi lần nhau giữa tiến trình
- Xây dựng đồ thị cấp phát tài nguyên:
  - Các nút là tiến trình và tài nguyên
  - Tài nguyên được nối với tiến trình bằng cung có hướng nếu tài nguyên được cấp cho tiến trình đó
  - Tiến trình được nối với tài nguyên bằng cung có hướng nếu tiến trình đang được cấp tài nguyên đó


P5
R1
R3
R4
P1
P2
P3
P4
R2
R5

Đồ thị chờ đợi:

- Được xây dựng từ đồ thị cấp phát tài nguyên bằng cách bỏ đi các nút tương ứng với tài nguyên và nhập các cung đi qua nút bị bỏ
- Cho phép phát hiện tình trạng tiến trình chờ đợi vòng tròn là điều kiện đủ để sinh ra bế tắc
- Sử dụng thuật toán phát hiện chu trình trên đồ thị có hướng để phát hiện bế tắc trên đồ thị chờ đợi.



Thời điểm phát hiện bế tắc:

- Bế tắc chỉ có thể xuất hiện sau khi một tiến trình nào đó yêu cầu tài nguyên và không được thỏa mãn
- Chạy thuật toán phát hiện bế tắc mỗi khi có yêu cầu cấp phát tài nguyên không được thỏa mãn
- Cho phép phát hiện bế tắc ngay khi vừa xẩy ra
- Chạy thường xuyên làm giảm hiệu năng hệ thống
- Giảm tần suất chạy thuật toán phát hiện bế tắc:
  - Sau từng chu kỳ từ vài chục phút tới vài giờ
  - Khi có một số dấu hiệu như hiệu suất sử dụng CPU giảm xuống dưới một ngưỡng nào đó

- Xử lý khi bị bế tắc:
  - Kết thúc tất cả tiến trình đang bị bế tắc
  - Kết thúc lần lượt từng tiến trình đang bị bế tắc đến khi hết bế tắc:
    - HDH phải chạy lại thuật toán phát hiện bế tắc sau khi kết thúc 1 tiến trình
    - HDH có thể chọn thứ tự kết thúc tiến trình dựa trên tiêu chí nào đó
  - Khôi phục tiến trình về thời điểm trước khi bị bế tắc sau đó cho các tiến trình thực hiện lại từ điểm này:
    - Đòi hỏi HDH lưu trữ trạng thái để có thể thực hiện quay lui và khôi phục về các điểm kiểm tra trước đó
    - Khi chạy lại, các tiến trình có thể lại rơi vào bế tắc tiếp
  - Lần lượt thu hồi lại tài nguyên từ các tiến trình bế tắc cho tới khi hết bế tắc

Tổng kết

Chương 4 Quản lý tiến trình

- Đồng bộ hóa các tiến trình đồng thời
- Tình trạng bế tắc và đói

Tổng kết

- Trình bày bài tập lớn
- Chữa đề thi các năm
- Các câu hỏi thắc mắc

Câu hỏi và bài tập

1. **Câu 1**: Xét trạng thái cấp phát tài nguyên của hệ thống như sau: P2 yêu cầu cấp phát 1 tài nguyên Y, 2 tài nguyên Z. Sử dụng thuật toán người cho vay để xác định xem yêu cầu của P2 có được đáp ứng hay không?

	X	Y	Z
P1	0	1	0
P2	2	0	0
P3	3	0	2
P4	2	1	1

Đã cấp

X	Y	Z
3	3	4

Còn lại

	X	Y	Z
P1	7	4	3
P2	1	2	2
P3	6	0	0
P4	0	1	1

Còn cần cấp

2. **Câu 2**:Xét trạng thái cấp phát tài nguyên của hệ thống như sau: P1 yêu cầu cấp phát 1 tài nguyên X, 1 tài nguyên Y, 2 tài nguyên Z. Sử dụng thuật toán người cho vay để xác định xem yêu cầu của P1 có được đáp ứng hay không? Trạng thái là gì?

	X	Y	Z
P1	0	1	0
P2	2	0	0
P3	3	0	2
P4	2	1	1

Đã cấp

X	Y	Z
3	3	4

Còn lại

	X	Y	Z
P1	7	4	3
P2	1	2	2
P3	6	0	0
P4	0	1	1

Còn cần cấp