TÀI LIỆU HỆ ĐIỀU HÀNH **CHƯƠNG 6: DEADLOCKS_Phần 1**

DÀNH CHO KHÓA 13 KHÓA NGÀY: 26 April 2020

HỆ THỐNG LÝ THUYẾT

1. Định nghĩa deadlock

- Một tiến trình được gọi là deadlock nếu nó đang đợi một sự kiện mà không bao giờ xảy ra.
- Một tiến trình được gọi là trì hoãn vô hạn định nếu nó trì hoãn một khoản thời gian lặp đi lặp lại trong khi hệ thống đáp ứng cho những tiến trình khác.
- **Điều kiện xảy ra deadlock**
  - Ít nhất một tài nguyên được giữ ở chế độ không chia sẻ (Máy in)
  - Giữ và chờ cấp phát tài nguyên
  - Không trưng dụng:
  - Có chu trình đợi

    P1 đợi tài nguyên P2 giữ, P2 đợi tài nguyên P3 giữ, P3 đợi tài nguyên P1 giữ → Deadlock.

2. Đồ thị cấp phát tài nguyên RAG (Phát hiện deadlock)

- Cạnh yêu cầu Pi → Rj
- Cạnh cấp phát Rj → Pi
- Kí hiệu (Xem slide)
- **Nhận xét**
  - RAG không chứa chu trình → Không deadlock

- RAG có chu trình và mỗi loại tài nguyên chỉ chứa một thực thể → Xảy ra deadlock
- RAG có chu trình mỗi loại tài nguyên chứa nhiều hơn 1 loại thực thể → Có thể xảy ra deadlock.

3. Giải quyết daedlock

- Bằng cách ngăn hoặc tránh deadlock
  - Ngăn deadlock: Không cho thỏa 4 điều kiện
  - Tránh deadlock: Quá trình cấp phát tài nguyên phải thích hợp
- Cho phép hệ thống vào trạng thái deadlock, nhưng sau đó phục hồi hệ thống
- Bỏ qua mọi vấn đề xem như deadlock không bao giờ xảy ra.
- **Ngăn deadlock**
  - Ngăn 4 điều kiện xảy ra deadlock
  - Ngăn loại trừ tương hỗ
    - Tài nguyên không chia sẻ (máy in): Không làm được
    - Tài nguyên chia sẻ: Không cần thiết
  - Ngăn no preemption: Nếu A đang giữ tài nguyên và đang yêu cầu tài nguyên khác nhưng tài nguyên này chưa được cấp phát ngay thì
    - **Cách 1:** Hệ thống lấy lại mọi tài nguyên A đang giữ
    - **Cách 2:** Hệ thống xem tài nguyên A yêu cầu
  - Ngăn Circular wait: Gán thứ tự cho tài nguyên trong hệ thống.
- **Tránh deadlock**
  - Ngăn deadlock sử dụng tài nguyên không hiệu quả

- Một trạng thái của hệ thống được gọi là an toàn nếu tồn tại một chuỗi an toàn.
- Một chuỗi <P1,P2,…Pn> được gọi là an toàn khi
    - Tài nguyên hệ thống sẵn có
    - Cùng tài nguyên với tất cả các Pj(j<i) đang giữ

Ví dụ

	MAX	**Allocation**	**NEED**
P0	10	5	5
P1	4	2	2
P2	9	2	7

Còn 3 tape sẵn sàn

P0 cần 5 nhưng còn 3 tape, P0 không thêm → P1 cần 2 tape, P1 thêm 2, còn dư 1. Sau khi giải phóng P1 giải phóng 4 và thêm 1 còn dư tổng cộng là 5 tape.

Chuỗi lúc này là <P1

P2 cần 7 nhưng hiện tại có 5, P2 không thêm →P0 cần 5 tape, P0 thêm 5, còn dư 0. Sau khi giải phóng P0 giải phóng 10.

Chuỗi lúc này là <P1;P0

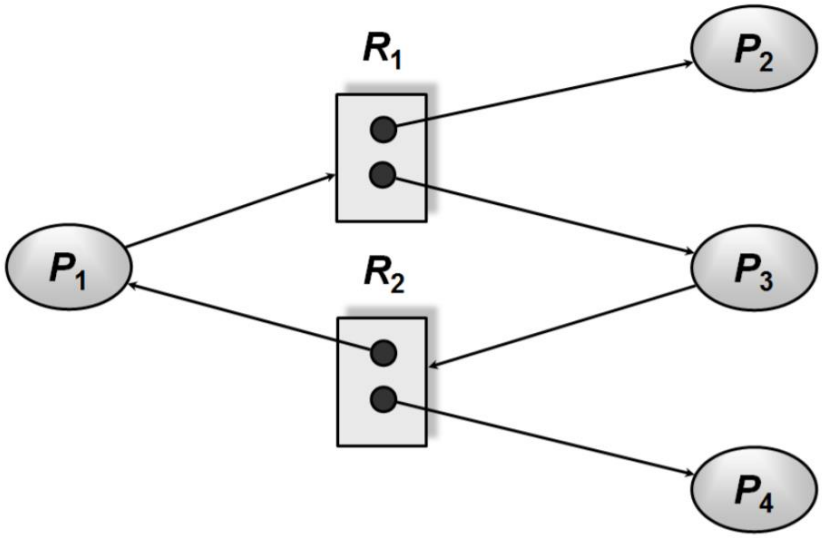
Tương tự vậy với P2

Ta được chuỗi an toàn <P1;P0;P2> → Hệ thống an toàn → Không xảy ra deadlock.

- Nếu hệ thống đang ở trạng thái safe → Không deadlock
- Nếu hệ thống ở trạng thái unsafe → Có thể xảy ra deadlock.
- Tránh deadlock là làm cho hệ thống luôn ở trạng thái an toàn.

Bài tập chương 6.1

Bài tập 1. Đồ thị RAG sau đây có thể ra deadlock không



Bài tập 2. Hệ thống có 18 tape và 4 tiến trình P0; P1; P2; P3. Hệ thống sau đây có an toàn không?

	Max	Allocation	Need	Available
P0	10	5	5	5
P1	4	2	2	3
P2	15	2	13	16
P3	10	6	4	10

Công thức: Need = Max – Allocation

Ta có chuỗi an toàn <P1;P0;P3;P2> → Hệ thống an toàn → Không xảy ra deadlock

---------- Hết ----------