DUY TAN UNIVERSITY

CHƯƠNG 5

ĐIỀU KHIỂN TƯƠNG TRANH (tt)

NỘI DUNG

1. Điều Khiển Tương Tranh Bằng Cơ Chế khóa
2. Các Tình Trạng Của Khóa
3. Các loại khóa
4. Giải quyết DeadLock

- **Đường Link you tube**
  - https://www.youtube.com/watch?v=jj4IjCH1I4U&list=PL67CJL04EcjN5PQrippgbVWz9L06a3Atz&index=2

1- Điều Khiển Tương Tranh Bằng Cơ Chế Khoá

- **Khoá (Lock)**

Lock là một đặc quyền truy xuất lên các đơn vị dữ liệu của các giao dịch. Khi một giao dịch đã lock trên một đơn vị dữ liệu nào đó thì các giao dịch khác không được phép truy cập đến đơn vị dữ liệu đó cho đến khi đơn vị dữ liệu đó được unlock.

T1	T2	Chi chú
Lock A		
Read A		
	Lock A	T2 không thể lock trên đơn vị dữ liệu A, vì T1 đã lock A trước, nên phải chờ T1 unlock trên A.
	Read A	
A = A+1		
	A = A+1	
Write A		
Unlock A		
	Unlock A	

2- Các Tình Trạng Của Khóa

✸ **LiveLock**:

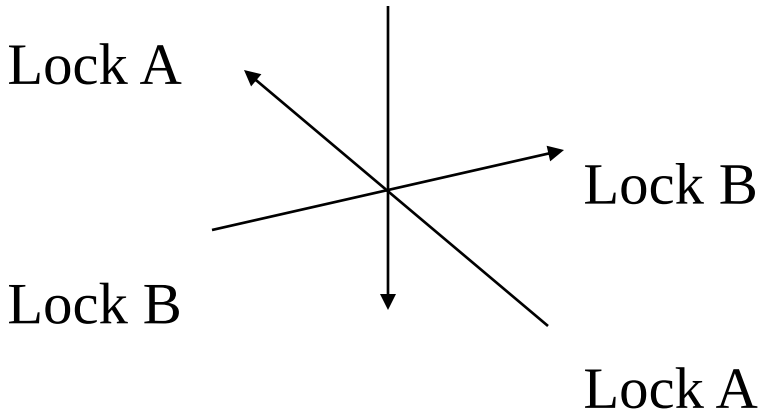
Tình trạng một giao dịch chờ được cấp quyền lock trên 1 đơn vị dữ liệu nào đó mà không xác định được thời điểm được đáp ứng yêu cầu (giống hiện tượng Starvation)

✸ **DeadLock:**

Deadlock là tình trạng trong đó những giao dịch có liên quan không thể thực hiện tiếp các thao tác của nó mà phải chờ nhau mãi

Ví dụ 1: Hai giao dịch T1, T2 thực hiện các công việc sau:

Giao dịch T1(t1 = 100) Giao dịch T2(t2 = 110)



▪ Tính khả tuần tự của lịch thao tác (trong cơ chế khoá)

Chỉ xét các thao tác lock/unlock để xác định tính khả tuần tự của lịch thao tác.

**** Đồ thị khả tuần tự của lịch thao tác:***

Cho S là 1 tập các giao dịch T1, T2 , ..., Tn. Xét các thao tác Oij có dạng **lockA** và **UnlockA** của các giao dịch.

Nếu 1 giao dịch $\mathbf{T_i}$ có 1 thao tác có dạng UnlockA

$\mathbf{T_j}$ có 1 thao tác tiếp sau đó có dạng LockA

Thì ta vẽ 1 cung có hướng đi từ $\mathbf{T_i} \rightarrow \mathbf{T_j}$

Nếu đồ thị có chu trình thì lịch thao tác không khả tuần tự.

Cách xây dựng đồ thị khả tuần tự:

• Đánh dấu tất cả các thao tác unlock

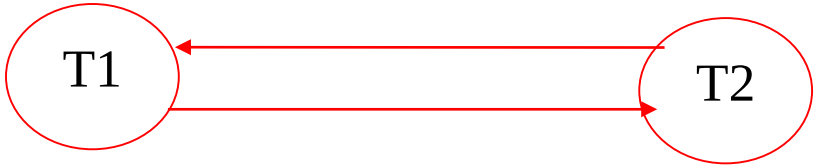
• Xét tuần tự từng lệnh unlock, giả sử đang xét lệnh unlockA của giao dịch Ti. Với mỗi lệnh lockA của giao dịch Tj khác thực hiện sau lệnh này, thì vẽ 1 cung có hướng đi từ Ti đến Tj.

Ví dụ 1: **Xét tính khả tuần tự của lịch thao tác sau:**

t (thời gian)

Giao dịch T1	Giao dịch T2
Lock A	
Read A -> a1	
Unlock A	
	Lock A
	Read A -> a2
	a2 +1 -> a2
	Write a2 -> A
	Unlock A
a1 +1 -> a1	
Lock A	
Write a1 -> A	
Unlock	

* Đồ thị khả tuần tự như sau:



Đồ thị có chu trình

=> Lịch thao tác không khả tuần tự.

3 - Các loại khóa

* ReadLock (Shared lock_khoá chia sẻ): Khi giao dịch yêu cầu đọc đơn vị dữ liệu thì dùng RLock

* WriteLock (Exclusive lock_khoá độc quyền): Khi giao dịch yêu cầu đọc và viết đơn vị dữ liệu hay chỉ viết trên đơn vị dữ liệu thì dùng WLock

Ma trận tương thích các loại lock:

		Lock đang được giữ	
		R-lock	W-lock
Giao dịch yêu cầu lock	R-lock	Yes	No
	W-lock	No	No

Đồ thị khả tuần tự trong cơ chế khoá có phân biệt ReadLock, WriteLock:

S là 1 tập các giao dịch T1, T2, ..., Tn. Xét các thao tác Oij có dạng **RLock A**, **WLock A**, **Unlock A** (không quan tâm các thao tác khác)

- Nếu giao dịch **Ti** thực hiện **Wlock A** hay **Rlock A**, giao dịch **Tj** thực hiện **Wlock A** sau đó thì vẽ 1 cung từ **Ti** đến **Tj** (**W/R → W**)

- Nếu giao dịch **Ti** thực hiện **Wlock A**, giao dịch **Tj** thực hiện **Rlock A** sau khi Ti thực hiện **Unlock A** nhưng trước khi các giao dịch khác thực hiện **Wlock A** thì thì vẽ 1 cung từ **Ti** đến **Tj** (**Wgần nhất → R**)

- **Đồ thị không có chu trình → Lịch thao tác khả tuần tự**.

Nghi thức lock 2 giai đoạn

- Một giao dịch tuân theo nghi thức lock 2 giai đoạn có nghĩa là nó không thực hiện một thao tác lock nào nữa sau khi đã dùng unlock
- Mục đích: Đảm bảo tính khả tuần tự
- Phát biểu: Mọi thực hiện các giao dịch 2 phase là khả tuần tự.

4 - Giải quyết DeadLock

Sử dụng nghi thức lock 2 giai đoạn chỉ có thể đảm bảo tính khả tuần tự cho lịch thao tác nhưng không đảm bảo không xảy ra tình trạng deadlock

- **Các bước tiếp cận để giải quyết deadlock**

- **Bỏ qua deadlock:**

  Khi gặp deadlock thì hủy, làm lại tất cả với nhãn thời gian khác.

Ngăn ngừa deadlock

*** Thuật toán DIE-WAIT:**

ưu tiên cho các giao dịch già (thời gian chờ đợi lâu)

Ti , Tj có timestamp là tTi ,tTj

Ti yêu cầu lock trên 1 đơn vị dữ liệu đang bị giữ lock bởi Tj thì

If $t_{Ti} < t_{Tj}$ then

Ti chờ

Else

Rollback **Ti** , **Ti** bắt đầu lại

End if

*** Thuật toán WOUD-WAIT:**

ưu tiên cho các giao dịch già (thời gian chờ đợi lâu)

Ti , Tj có timestamp là tTi ,tTj

Ti yêu cầu lock trên 1 đơn vị dữ liệu đang bị giữ lock bởi Tj thì

If $t_{Ti} < t_{Tj}$ then

Rollback Tj

Else

Ti chờ

End if.

Phát hiện và phục hồi sau deadlock

Phát hiện deadlock: Dùng đồ thị chờ để phát hiện deadlock**.**

Cho 1 lịch thao tác S có n giao dịch T1, T2 , ...,Tn.

Đồ thị chờ có :

Đỉnh là các giao dịch

Cung có hướng Ti -> Tj nếu Ti chờ Tj

Nếu đồ thị có chu trình thì có deadlock.

Giải quyết deadlock: hủy giao dịch (đỉnh) nào có nhiều cung ra vào lớn nhất VÀ thời gian vào sau.

Phân biệt đồ thị khả tuần tự với đồ thị chờ trong điều khiển truy xuất đồng thời bằng cơ chế khóa.

- Đồ thị khả tuần tự: cung có hướng đi từ Ti -> Tj nếu Ti thực hiện trước Tj (Ti<Tj)

- Đồ thị chờ: cung có hướng đi từ Ti -> Tj nếu Ti chờ Tj khi có yêu cầu lock chưa được đáp ứng.

Nghi thức lock 2 giai đoạn nghiêm ngặt

- Chỉ cho phép unlock ở cuối giao tác cho dù giao tác không còn thao tác trên đơn vị dữ liệu đó nữa.

Q & A

1. Phân biệt 2 loại khoá: Rlock, Wlock
2. Phân biệt các tình trạng của khoá: deadlock, livelock
3. Phân biệt các loại đồ thị: ưu tiên; khả tuần tự không khoá; khả tuần tự có khoá không phân biệt Rlock, Wlock; khả tuần tự có khoá có phân biệt Rlock, Wlock; và đồ thị chờ
4. So sánh 2 thuật toán ngăn ngừa deadlock: DIE-WAIT, WOUD-WAIT

- **Bài tập nhóm**
  - Các sinh viên chia nhóm. Mỗi nhóm 3 sinh viên thực hiện yêu cầu sau:
  - Sinh viên 1: Lập trình giải quyết deadlock bằng thuật toán die wait
  - Sinh viên 2: Lập trình giải quyết deadlock bằng thuật toán Wout wait
  - Sinh viên 3: Lập trình cài đặt giải quyết deadlock bằng đồ thị chờ

- **Tóm tắt**

  Có 3 thuật toán điều khiển tương tranh bằng cơ chế khóa :

  – Thuật toán Die Wait
  – Thuật toán Wout Wait
  – Thuật toán dựa trên đồ thị chờ