TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
BỘ MÔN CÔNG NGHỆ THÔNG TIN

QUẢN TRỊ DỮ LIỆU - CT467

Chương 4: ĐIỀU KHIỂN CẠNH TRANH

Biên soạn:

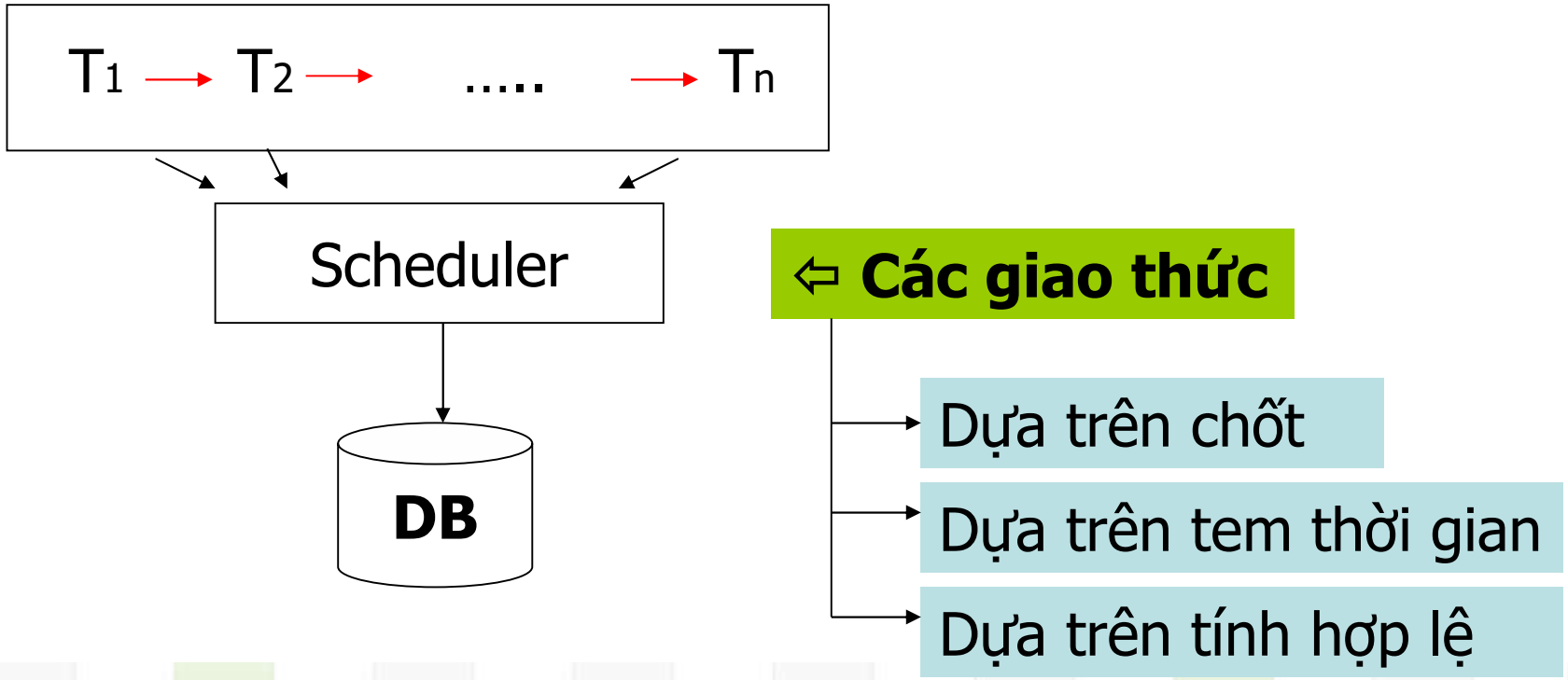
Ths. Nguyễn Thị Kim Yến **Ntkyen@ctu.edu.vn**

NỘI DUNG

1. Kỹ thuật khóa (chốt)
2. Kỹ thuật nhãn (tem) thời gian
3. Giao thức dựa trên tính hợp lệ
4. Quản lý deadlock

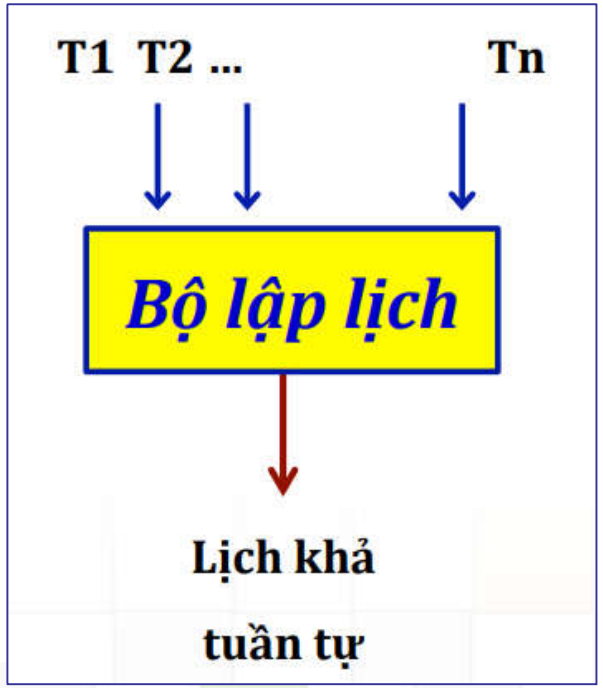
GIỚI THIỆU

- Trong chương trước, ta đã có các giải thuật để xác định một lịch trình có khả tuần tự hay không.
- Nhưng làm thế nào để sinh ra được các lịch trình khả tuần tự?



GIỚI THIỆU (tt)

- Các kỹ thuật điều khiển đồng thời: những kỹ thuật cho phép bộ lập lịch (shedule) sử dụng để tạo một lịch trình khả tuần tự n giao dịch thực hiện đồng thời.



⇦ **Các kỹ thuật**

- Kỹ thuật khóa (chốt)
- Kỹ thuật nhãn (tem) thời gian
- Kỹ thuật xác nhận (tính) hợp lệ

1. KỸ THUẬT KHÓA (CHỐT)

Các loại:

1.1 Khóa đơn giản

1.2 Khóa 2 giai đoạn (giao thức chốt 2 kỳ)

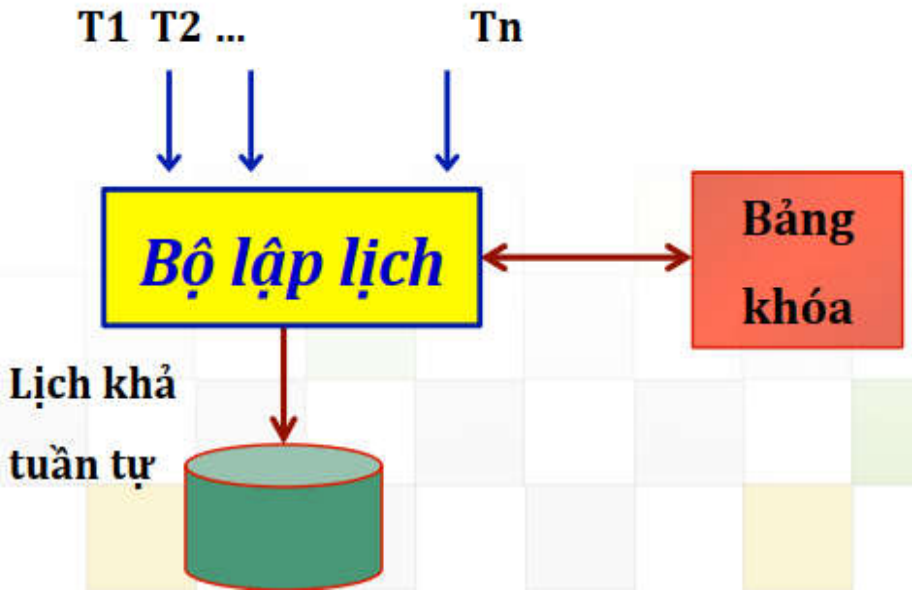
1.3 Khóa đọc ghi (chốt và cấp chốt)

1.4 Khóa đa hạt

1.5 Khóa đồ thị

Kỹ thuật khoá đơn giản

- Kỹ thuật khoá đơn giản còn gọi khoá nhị phân (Binary locks)
- Bộ lập lịch với cơ chế khóa đơn giản (locking scheduler)
  - Là bộ lập lịch với thêm **2** hành động :
    - **Lock** : Phát khoá
    - **Unlock** : Giải phóng khóa
  - Các khóa được ghi nhận trong bảng khóa (Lock Table)



Kỹ thuật khóa đơn giản

- **Quy định:**
  - Các giao tác trước khi muốn đọc/ghi lên 1 đơn vị dữ liệu phải phát ra 1 yêu cầu xin khóa (lock) đơn vị dữ liệu đó
    - Ký hiệu: **Lock(A)** hay **l(A)**
  - Yêu cầu này được bộ phận quản lý khóa xử lý (Lock Manager)
    - Nếu yêu cầu được chấp thuận thì giao tác mới được phép đọc/ghi lên đơn vị dữ liệu
    - Yêu cầu xin khóa được bộ cấp phát chấp thuận nếu đơn vị dữ liệu chưa bị khóa bởi một giao tác nào khác
  - Sau khi thao tác xong thì giao tác phải phát ra lệnh giải phóng đơn vị dữ liệu (unlock)
    - Ký hiệu: **Unlock(A)** hay **u(A)**

BẢNG KHÓA

Element	Transaction
A	T1

Bảng khóa ghi nhận giao tác T1 đang giữ khóa trên đơn vị dữ liệu A

Kỹ thuật khóa đơn giản (tt)

- **Quy tắc**:
  - 1. **Giao tác đúng đắn**: Việc giao tác Ti đọc hay ghi lên đơn vị dữ liệu A phải sau khi Ti phát khoá trên A và trước khi Ti giải phóng khoá trên A. Phát khoá và giải phóng khoá phải đi đôi với nhau (lock trước, unlock sau)
    - Ti: ... l(A) ... r(A) / w(A) ... u(A) ...
  - 2. **Lịch thao tác hợp lệ**: Khi Ti đang giữ khoá trên một đơn vị dữ liệu A thì không Ti nào khác được phát khoá trên A.
    - S: ... li(A) .....ui(A) ...

Kỹ thuật khóa đơn giản (tt)

- Ví dụ:
  - Giao tác T1 và T2 có đúng đắn hay không ?
  - Lịch S có hợp lệ hay không ?

S

T1	T2
Lock(A)	
Read(A, t)	
t := t + 100	
Write(A, t)	
Unlock(A)	
	Lock(A)
	Read(A, s)
	s := s * 2
	Write(A, s)
	Unlock(A)
	Lock(B)
	Read(B, s)
	s := s * 2
	Write(B, s)
	Unlock(B)
Lock(B)	
Read(B, t)	
t := t + 100	
Write(B, t)	
Unlock(B)	

Kỹ thuật khóa đơn giản (tt)

- **Bài toán**: Kiểm tra tính khả tuần tự của lịch S
  - *Input*: Lịch **S** được lập từ n giao tác xử lý đồng thời T_1, T_2, ..., T_n theo kỹ thuật khóa đơn giản
  - *Output*: **S** khả tuần tự hay không?

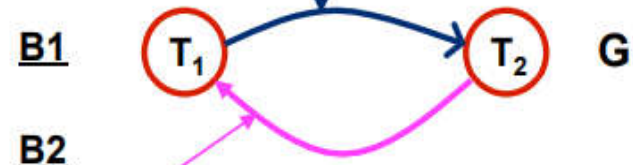
- **Phương pháp**: Xây dựng 1 đồ thị có hướng G
  - *B1*: Mỗi giao tác T_i là 1 đỉnh của đồ thị
  - *B2*: Nếu một giao tác T_j phát ra $Lock_j(A)$ sau một giao tác T_i phát ra $Unlock_i(A)$ thì sẽ vẽ cung từ T_i đến T_j , $i \neq j$
  - *B3*: S khả tuần tự nếu G không có chu trình

Kỹ thuật khóa đơn giản (tt)

S

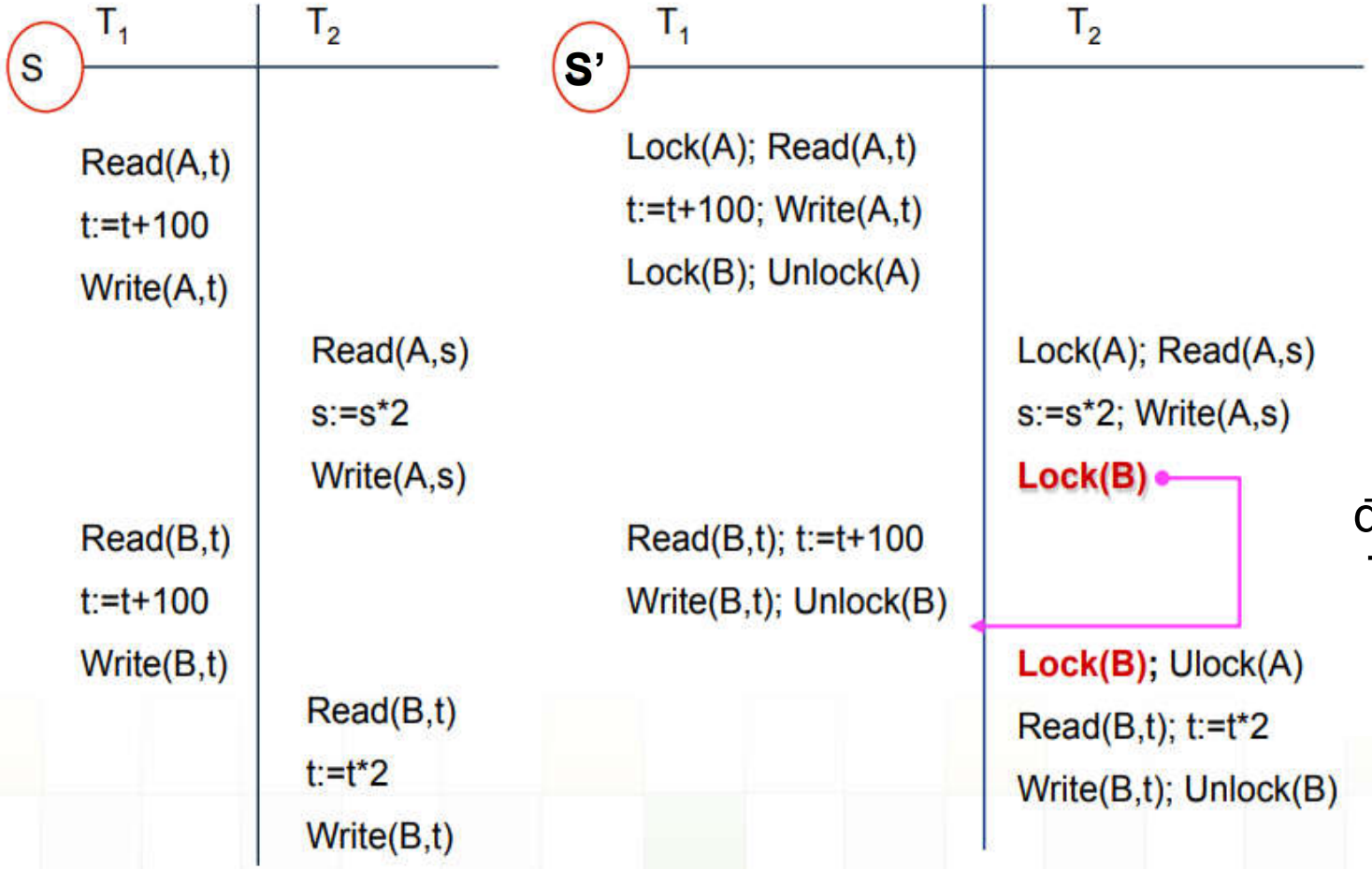
T1	T2
Lock(A)	
Read(A, t)	
t := t + 100	
Write(A, t)	
Unlock(A)	
	Lock(A)
	Read(A, s)
	s := s * 2
	Write(A, s)
	Unlock(A)
	Lock(B)
	Read(B, s)
	s := s * 2
	Write(B, s)
	Unlock(B)
Lock(B)	
Read(B, t)	
t := t + 100	
Write(B, t)	
Unlock(B)	

Lịch S có khả tuần tự hay không ?



G có chu trình → S không khả tuần tự

Kỹ thuật khóa đơn giản (tt)



Không xin được khóa. Tình trạng "chết đói" (starved)

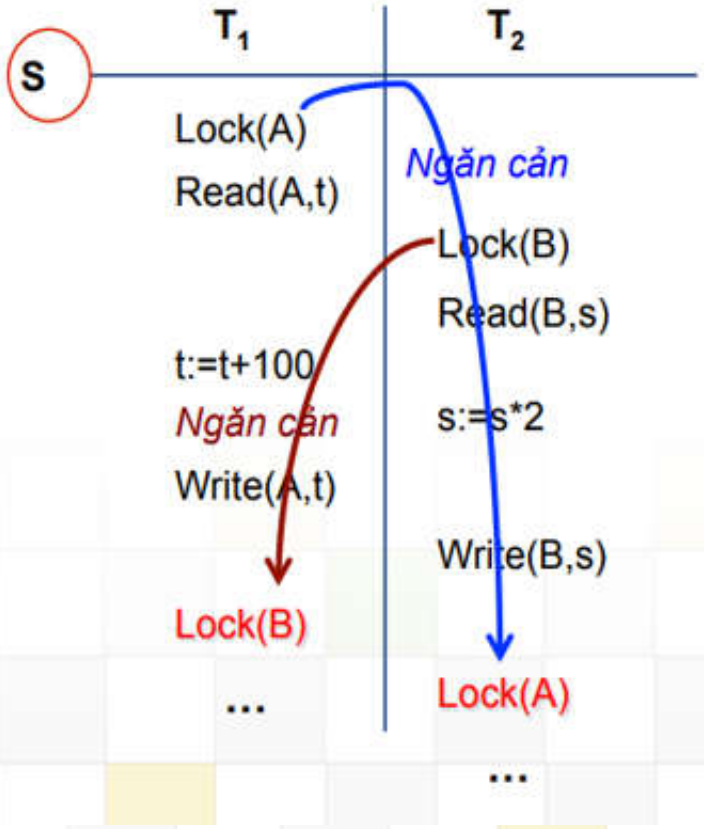
Tình trạng một GD yêu cầu khóa nhưng cứ chờ mãi mà không được cấp

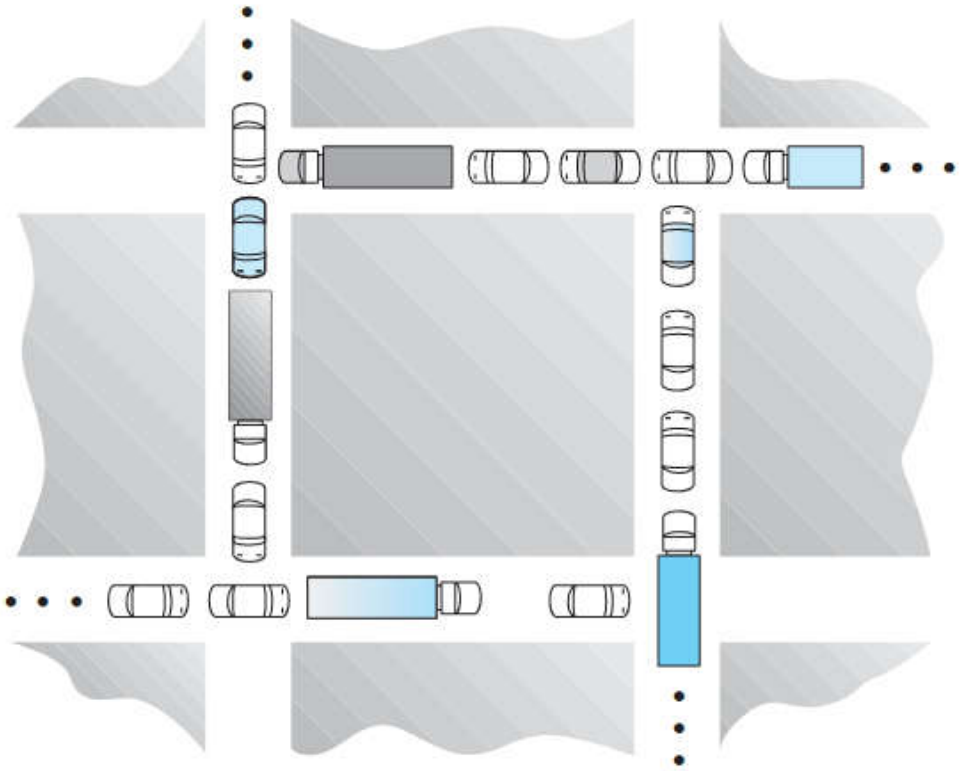
Kỹ thuật khoá đơn giản (tt)

- Để tránh chết đói, ta có thể dùng giao thức sau:
  - GD T_i chỉ được cấp khóa trên hạng mục dữ liệu Q ở phương thức M, khi cả hai điều kiện sau được thỏa:
    1. Không có GD khác giữ khóa không tương thích với M trên Q
    2. Không có GD khác đang chờ khóa trên Q và đã đưa ra yêu cầu cấp khóa trước T_i
- Một GD sẽ được cấp khóa nếu như không có GD nào giữ khóa ở phương thức không tương thích

Kỹ thuật khóa đơn giản (tt)

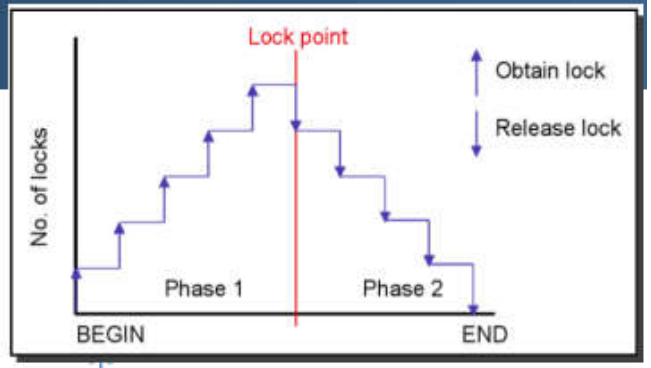
- **Vấn đề khoá chết (Dead lock):** Hiện tượng chờ khi cần phát khóa có thể dẫn đến chờ lẫn nhau vĩnh viễn



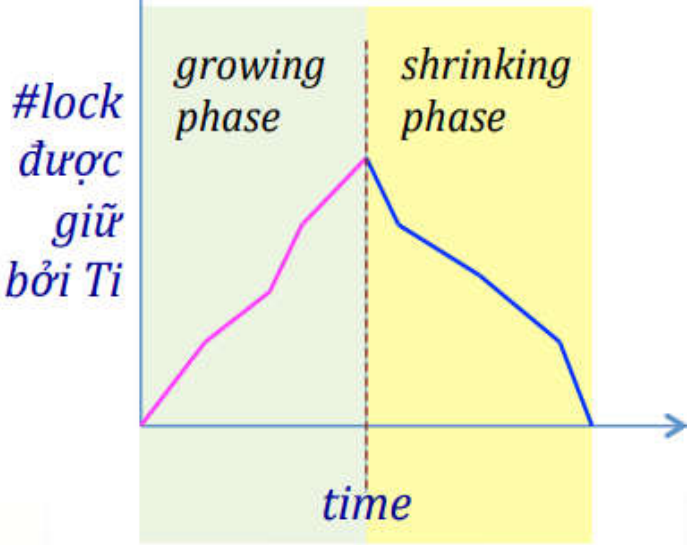


Nghi thức khoá 2 giai đoạn

(1.2 Giao thức chốt 2 kỳ)



- Nghi thức khoá 2 giai đoạn chia việc xin khoá và giải phóng khoá của giao tác thành 2 giai đoạn (phase) phân biệt:
  - **Growing** phase (pha phát khoá): (xin khóa)
    - Giao tác chỉ được phép xin khoá chứ không được phép giải phóng khoá trong pha này (số lượng khoá được giữ chỉ được phép ngày càng tăng) (kỳ phình to)
    - Giai đoạn này kết thúc ở lệnh xin khoá cuối cùng. (trả khóa)
  - **Shrinking** phase (pha giải phóng khoá): (kỳ thu nhỏ)
    - Giao tác chỉ được phép giải phóng khoá chứ không được phép xin khoá trong pha này
    - Giao tác này bắt đầu từ khi lệnh giải phóng khoá đầu tiên.



Nghi thức khoá 2 giai đoạn (tt)

- **Mục tiêu**: Vậy làm sao để kỹ thuật khóa cho ta lịch khả tuần tự
- **Giải pháp** : Tuân theo quy tắc sau:
  - (1) và (2): Giống như cũ (giao tác đúng đắn - thao tác hợp lệ)
  - (3) Giao tác phải thỏa **nghi thức khóa 2 giai đoạn** (**2PL: two phases locking)** : Trong 1 giao tác, tất cả các thao tác phát khóa đều xảy ra trước tất cả các thao tác giải phóng khóa.

- **Định lý**:
  - Nếu lịch S thoả nghi thức 2PL thì S conflict-serializable

Nghi thức khoá 2 giai đoạn (tt)

T1	T2	T3	T4
Lock(A)	**Lock(B)**	**Lock(B)**	**Lock(A)**
Read(A)	Read(B)	Read(B)	Read(A)
Lock (B)	**Lock(A)**	B=B-50	**Unlock(A)**
Read(B)	Read(A)	Write(B)	**Lock(B)**
B:= B + A	**Unlock(B)**	**Unlock(B)**	Read(B)
Write(B)	A:= A+B	**Lock(A)**	**Unlock(B)**
Unlock(A)	Write(A)	Read(A)	Print (A+B)
Unlock(B)	**Unlock(A)**	A:=A+50	
		Write(A)	
		Unlock(A)	

Thỏa nghi thức khóa 2 giai đoạn

Không thỏa nghi thức khóa 2 giai đoạn

Nghi thức khoá 2 giai đoạn (tt)

- Các lịch trình tuân theo giao thức chốt 2 kỳ: lịch trình khả tuần tự xung đột ➔ tuy nhiên, không tránh được deadlock và bị cuộn lại hàng loạt.
- **Giải pháp:** tránh cuộn lại hàng loạt là thêm ràng buộc
  - Giao thức chốt 2 kỳ **nghiêm ngặt**: tất cả các chốt X phải được trả ở cuối giao dịch
  - Giao thức chốt 2 kỳ **nghiêm khắc**: tất cả các chốt phải được trả ở cuối giao dịch.

Kỹ thuật khóa đọc ghi

(1.3 Chốt và cấp chốt)

- Dựa trên sự loại trừ hỗ tương: khi 1 GD đang truy xuất một hạng mục DL, không một GD nào khác được phép cập nhật hạng mục DL đó.

- Do đó, bộ lập lịch cần các hành động:
  - **Khóa đọc** (Read lock, Shared lock)
    - Ký hiệu : **RLock(A)** hay **rl(A)** hay **Lock_S(A)** hoặc **L_S(A)**
  - **Khóa ghi** (Write lock, Exclusive lock)
    - Ký hiệu : **WLock(A)** hay **wl(A)** hay **Lock_X(A)** hoặc **L_X(A)**
  - **Giải phóng khóa**
    - Ký hiệu : **Unlock(A)** hay **u(A)**

Kỹ thuật khóa đọc ghi (tt)

- Cho 1 đơn vị dữ liệu A bất kỳ
  - WLock(A) hay **Lock_X(A)** hay **L_X(A)** hoặc **LX(A)**
    - Hoặc có 1 khóa ghi duy nhất lên A
    - Hoặc không có khóa ghi nào lên A
  - RLock(A) hay **Lock_S(A)** hay **L_S(A)** hoặc **LS(A)**
    - Có thể có nhiều khóa đọc được thiết lập lên A

- Ma trận tương thích:

Trạng thái hiện hành \ *Yêu cầu khoá*	**RLock(A)**	**WLock(A)**
RLock(A)	✔	✖
WLock(A)	✖	✖

✔ *Tương thích* ✖ *Không tương thích*

Kỹ thuật khóa đọc ghi (tt)

- Giao tác T muốn Write(A):
  - Yêu cầu WLock(A) hay **Lock_X(A)** hay **L_X(A)** hoặc **LX(A)**
    - WLock(A) sẽ được chấp thuận nếu hiện không có khóa nào trên A
    - Từ đó sẽ không có giao tác nào khác nhận được WLock(A) hay RLock(A) cho đến khi T giải phóng khóa trên A
- Giao tác muốn Read(A): hay **Lock_S(A)** hay **L_S(A)** hoặc **LS(A)**
  - Yêu cầu RLock(A) hoặc WLock(A)
    - RLock(A) sẽ được chấp thuận nếu A **không** đang giữ một WLock nào
    - Từ đó sẽ không có giao tác nào khác nhận được WLock(A) cho đến khi T giải phóng khóa trên A. Nhưng không ngăn chặn các thao tác khác cùng xin Rlock(A) nên các giao tác không cần phải chờ nhau khi đọc A
- Sau khi thao tác xong thì giao tác phải giải phóng khóa trên đơn vị dữ liệu A
  - Tháo chốt sớm: cạnh tranh vs. không nhất quán.
  - Tháo chốt trễ: nhất quán vs. ít cạnh tranh + deadlock.

Kỹ thuật khóa đọc ghi (tt)

- **Qui tắc**
  - ***(1) Giao tác đúng đắn*** :
    - Đã có phát khóa thì sau đó phải có giải phóng khóa, giải phóng khóa chỉ có khi trước đó có phát khóa mà chưa giải phóng
    - Thao tác đọc chỉ được thực hiện sau khi phát khóa đọc hoặc ghi và trước khi giải phóng khóa ấy
    - Thao tác ghi chỉ được thực hiện sau khi phát khóa ghi và trước khi giải phóng khóa ghi ấy
    - Các thao tác đọc, ghi, phát khóa và giải phóng khóa đề cập trên đây là xét trong cùng một giao tác và trên cùng 1 đơn vị dữ liệu

Kỹ thuật khóa đọc ghi (tt)

- **Qui tắc**
  - ***(2) - Lịch thao tác hợp lệ***
    - Khi T_i đang giữ khóa đọc trên 1 đơn vị Dữ liệu A thì **không** một T_j nào khác được phép ghi trên A
    - Khi T_i đang giữ khóa ghi trên 1 đơn vị Dữ liệu A thì **không** một T_j nào khác được phép đọc hay ghi trên A

Kỹ thuật khóa đọc ghi (tt)

- **Qui tắc**
  - *(3) - Giao tác 2PL* (Giao thức chốt 2 kỳ)
    - Ngoại trừ trường hợp nâng cấp khóa, các trường hợp còn lại đều giống với nghi thức khóa hai giai đoạn
    - T : ... **rli(A)** ... **wli(A)** ..... **ui(A)** ...

Không có giải phóng bất kỳ khóa nào

Không có phát ra bất kỳ khóa nào

    - Trường hợp nâng cấp khóa được giải phóng khóa đọc trong pha phát khóa
    - T : ... rli(A) ..... **uli(A)** ..... wli(A) ..... ui(A) ...

Chấp nhận giải phóng khóa đọc khi nâng cấp khóa

- Định lý :
  - S thoả (1), (2) và (3) → S **conflict-serializable** (Khả tuần tự xung đột)

Ví dụ

S1

T_1	T_2
RLock(A)	
Read(A)	
U(A)	
	RLock(B)
	Read(B)
	U(B)
	WLock(A)
	Read(A)
	A:=A+B
	Write(A)
	U(A)
WLock(B)	
Read(B)	
B:=B+A	
Write(B)	
U(B)	

1. Giao tác nào đúng đắn ?
2. Lịch thao tác có hợp lệ hay không ?
3. Giao tác nào không thoả nghi thức 2PL ?
4. S có khả tuần tự ?

Trả lời:

1. Giao tác T1 và T2 đúng đắn
2. Lịch thao tác có hợp lệ
3. Giao tác T1 và T2 KHÔNG thỏa nghi thức khóa 2 giao đoạn
4. S khả tuần tự

Kỹ thuật khóa đọc ghi (tt)

- Để tăng tính cạnh tranh cho GD, nâng cấp cho giao thức chốt 2 kỳ dựa trên việc chuyển đổi chốt (lock conversion)
  - **Nâng cấp** (upgrade): Khóa chia sẻ (L_S) ➡ Khóa loại trừ (L_X) ➔ Chỉ thực hiện trong kỳ phình to
  - **Hạ cấp** (downgrade): Khóa loại trừ (L_X) ➡ Khóa chia sẻ (L_S) ➔ Chỉ thực hiện trong kỳ thu nhỏ

Kỹ thuật khóa đọc ghi (tt)

- Khi 1 GD T_i phát ra 1 chỉ thị Read(A), hệ thống sẽ sinh ra 1 chỉ thị Lock_S(A) ngay trước chỉ thị Read(A)
- Khi 1 GD T_i phát ra 1 chỉ thị Write(A), hệ thống sẽ **kiểm tra** xem T_i có giữ khóa chia sẻ trên A hay chưa:
  - Nếu có thì sẽ sinh ra 1 chỉ thị Upgrade(A) ngay trước Write(A)
  - Ngược lại, phát ra 1 chỉ thị Lock_X(A) ngay trước Write(A)

– Ma trận tương thích

Trạng thái hiện hành \ *Yêu cầu khoá*	RLock	WLock	ULock
RLock	✔	✖	✔
WLock	✖	✖	✖
ULock	✖	✖	✖

Kỹ thuật khóa đọc ghi (tt)

(Chốt và cấp chốt)

Ví dụ: Xét 2 giao dịch như sau:

T_1	T_2
Read(A_1)	Read(A_1)
Read(A_2)	Read(A_2)
.....	Display (A_1+A_2)
Read(A_n)	
Write(A_1)	

Nâng cấp

	T_1	T_2
1	L_S(A_1)	
2		L_S(A_1)
3	L_S(A_2)	
4		L_S(A_2)
5	L_S(A_3)	
6	L_S(A_4)	
7		Unlock(A_1)
8		Unlock(A_2)
9	L_S(A_n)	
10	Upgrade(A_1)	
11	Wite(A_1)	

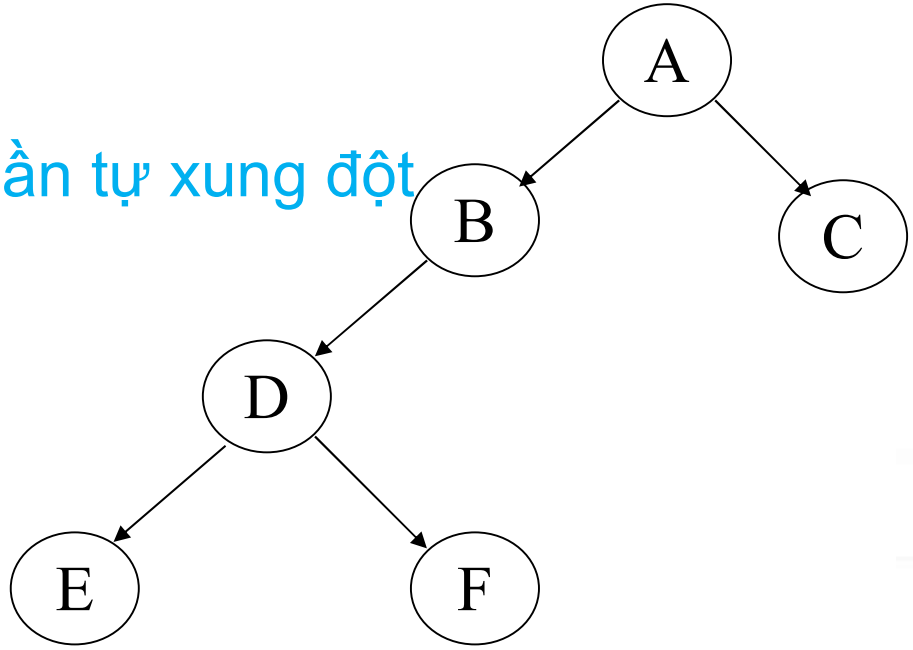
Kỹ thuật khóa - Đồ thị

(1.4 Giao thức dựa trên đồ thị)

- Có thể thực hiện nếu biết trước thứ tự mà các hạng mục DL được truy xuất. Xem như đồ thị có hướng phi chu trình

  ⇒ Đồ thị CSDL

- Tính chất:
  - LT tuân theo GT cây khả tuần tự xung đột
  - Tránh deadlock
  - Chỉ sử dụng chốt Lock_X.



A → B có nghĩa là nếu GD nào truy xuất cả A và B phải truy xuất A trước B

Kỹ thuật khóa - Đồ thị (tt)

- **Quy tắc:**

  (1) Mỗi hạng mục có thể chốt nhiều nhất 1 lần

  (2) Chốt đầu tiên bởi T có thể trên bất kỳ hạng mục DL nào

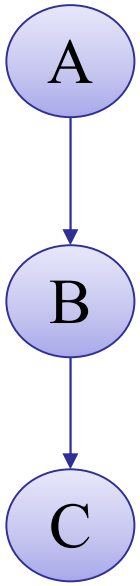
  (3) Sau đó, Q có thể được chốt bởi T nếu T đã chốt cha Q

  (4) Các hạng mục DL có thể tháo chốt bất kỳ lúc nào

- Giao thức này có thuận lợi hơn GT chốt 2 kỳ là ta có thể tháo chốt sớm nhưng có hạn chế là phải chốt các hạng mục DL không truy xuất (3) ➔ lãng phí khóa, tăng thời gian chờ và giảm tính cạnh tranh
- GT cây cũng không đảm bảo tính chất khả phục hồi và tránh cuộn lại hàng loạt => tất cả các khóa phải giữ đến cuối GD

Kỹ thuật khóa - Đồ thị (tt)

- **Ví dụ:** GD thỏa GT cây nhưng không thỏa GT chốt 2 kỳ và ngược lại



T1	T2
L_X(A)	
L_X(B)	
U(A)	
	L_X(A)
L_X(C)	
U(B)	
	L_X(B)
	U(A)
	U(B)
U(C)	
Tree, !2PL	

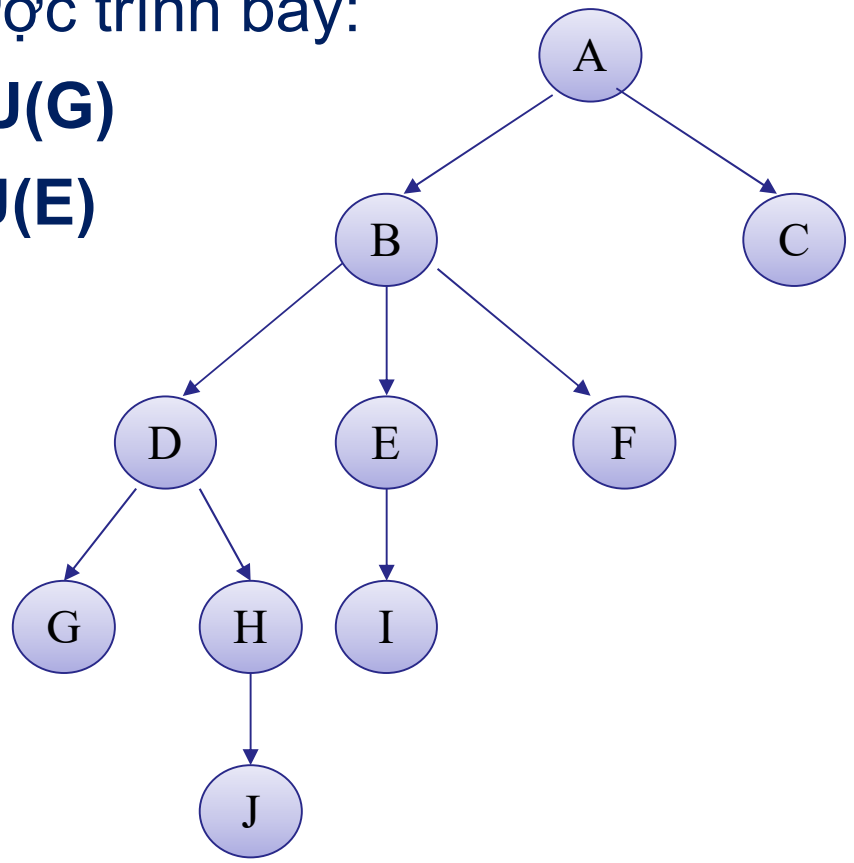
T1	T2
L_X(A)	
	L_S(B)
L_X(C)	
	U_S(B)
U(A)	
U(C)	
2PL, !Tree	

Kỹ thuật khóa - Đồ thị (tt)

(Giao thức dựa trên đồ thị)

- Chỉ thị xin/tháo khóa của GD được trình bày:
  - T_{10}: **L_X(D); L_X(G); U(D); U(G)**
  - T_{11}: **L_X(B); L_X(E); U(B); U(E)**

Cây CSDL:



T_{10}	T_{11}
	L_X(B)
L_X(D)	
	L_X(E)
L_X(G)	
U(D)	
	U(B)
	U(E)
U(G)	

Lịch trình khả tuần tự dưới giao thức cây

Kỹ thuật khóa đa hạt (Multiple Granularity)

(1.5 Đa hạt)

- Xét ví dụ hệ thống ngân hàng
  - Quan hệ TaiKhoan (maTK, soDu)
  - Giao tác gửi tiền và rút tiền

❖ **Khóa Relation:**

- Các giao tác thay đổi giá trị của số dư của 1 tài khoản X sẽ yêu cầu khóa độc quyền
- Vì khóa ở mức độ quan hệ nên toàn bộ bảng bị khóa
- Các giao tác khác muốn truy cập tài khoản Y (Y khác X) cũng phải chờ → vô lý, tốc độ xử lý đồng thời chậm

Kỹ thuật khóa đa hạt

❖ **Nên sử dụng chốt nhỏ hay lớn?**

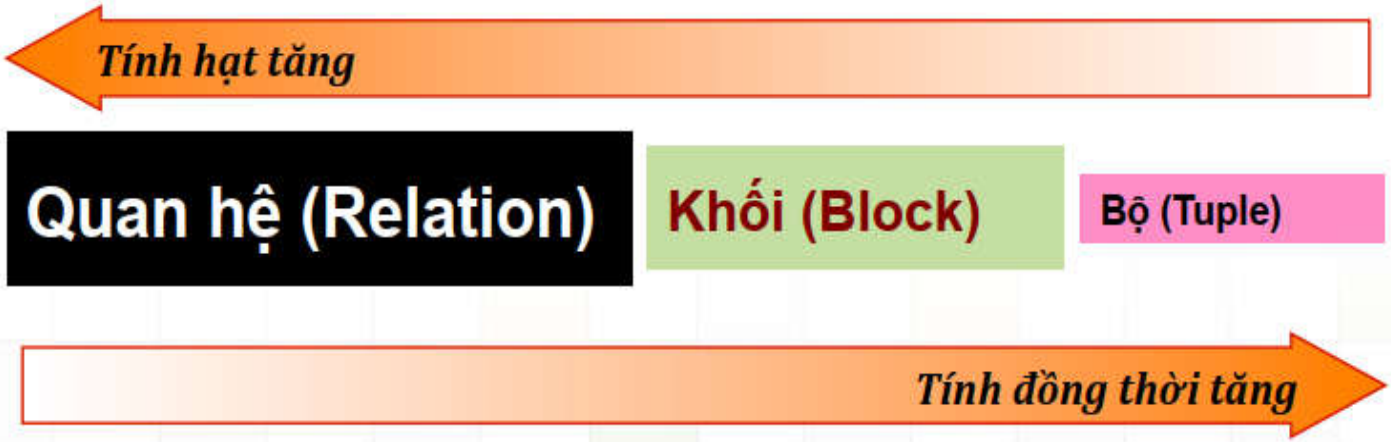
- Chốt nhỏ (mẩu tin), phù hợp với các GD truy xuất ít hạng mục DL => cho phí chốt cao nhưng tăng tính cạnh tranh
- Chốt lớn (bảng, cả CSDL), phù hợp với các GD truy xuất nhiều hạng mục DL => cần ít khóa nhưng ít cạnh tranh

❖ **Giải pháp:**

- Đa hạt (multiple granulatiry)
- Hạt ở đây có nghĩa là đơn vị cấp chốt: hệ thống hỗ trợ nhiều đơn vị cấp chốt hay nhiều mức hạt khác nhau.

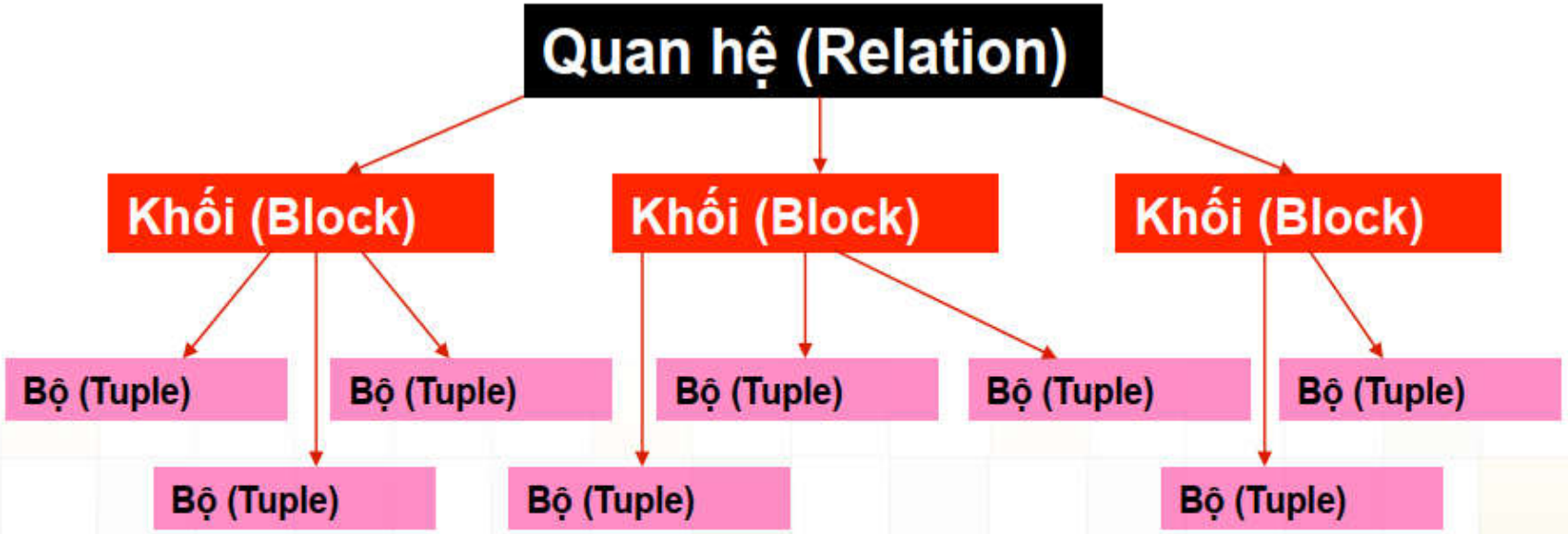
Kỹ thuật khóa đa hạt (tt)

- Phải quản lý khóa ở nhiều mức độ
  - Tính chất hạt (granularity) : Tính hạt càng tăng khi đơn vị dữ liệu bị khóa càng lớn
  - Tính đồng thời (concurrency) : Tính đồng thời càng tăng khi đơn vị dữ liệu bị khóa càng nhỏ
  - Tình hạt tăng thì tính đồng thời giảm và ngược lại → phải thỏa hiệp



Kỹ thuật khóa đa hạt (tt)

- Phân cấp Dữ liệu
  - Relations là đơn vị dữ liệu khóa lớn nhất (mức cao nhất: toàn bộ CSDL)
  - Một relation gồm 1 hoặc nhiều blocks (pages) (vùng, không gian bảng)
  - Một block gồm 1 hoặc nhiều tuples (mức thấp nhất: các mẩu tin)



Kỹ thuật khóa đa hạt (tt)

❖ **Gồm các khóa:**

- **Khóa thông thường (khóa tường minh):**
  - Shared lock: S
  - Exclusive lock: X
- **Khóa cảnh báo (khóa tăng cường):**
  - Warning (intensive) shared lock: IS cho khóa chia sẻ S. Khi 1 nút được khóa ở phương thức chia sẻ S, thì khóa IS nó cũng sẽ hiện diện trên tất cả các nút tổ tiên.
  - Warning (intensive) exclusive lock: IX cho khóa loại trừ X
  - Shared intensive exclusive: SIX kết hợp của khóa S và IX

Kỹ thuật khóa đa hạt (tt)

- Ma trận tương thích trên cùng một node
  - Cho biết các khóa có thể cùng tồn tại trên 1 node dữ liệu

	IS	IX	S	X
IS	✔	✔	✔	✖
IX	✔	✔	✖	✖
S	✔	✖	✔	✖
X	✖	✖	✖	✖

Kỹ thuật khóa đa hạt (tt)

- Ma trận tương thích giữa node cha và node con
  - Cho biết các khóa có thể phát trên node con khi node cha đang có một khoá nào đó → Cho biết muốn phát 1 khóa trên node con thì trước đó phải phát khóa nào trên node cha

Node cha đã khoá bằng phương thức	*Node con có thể khoá bằng các phương thức*
IS	**IS, S**
IX	**IS, S, IX, X**
S	**S, IS**
X	**Không có**

Kỹ thuật khóa đa hạt (tt)

- Quy tắc:
  - (1) Thỏa ma trận tương thích trên cùng một node
  - (2) Khóa nút gốc của cây trước
  - (3) Thỏa ma trận tương thích giữa node cha và node con
  - (4) T_i thỏa 2PL
  - (5) T_i có thể giải phóng nút Q khi không có nút con nào của Q bị khóa bởi T_i
  - (6) Trong trường hợp thêm mới hay xóa bỏ một node Q thì cha của Q phải được khóa bằng X trước → tránh Phantom

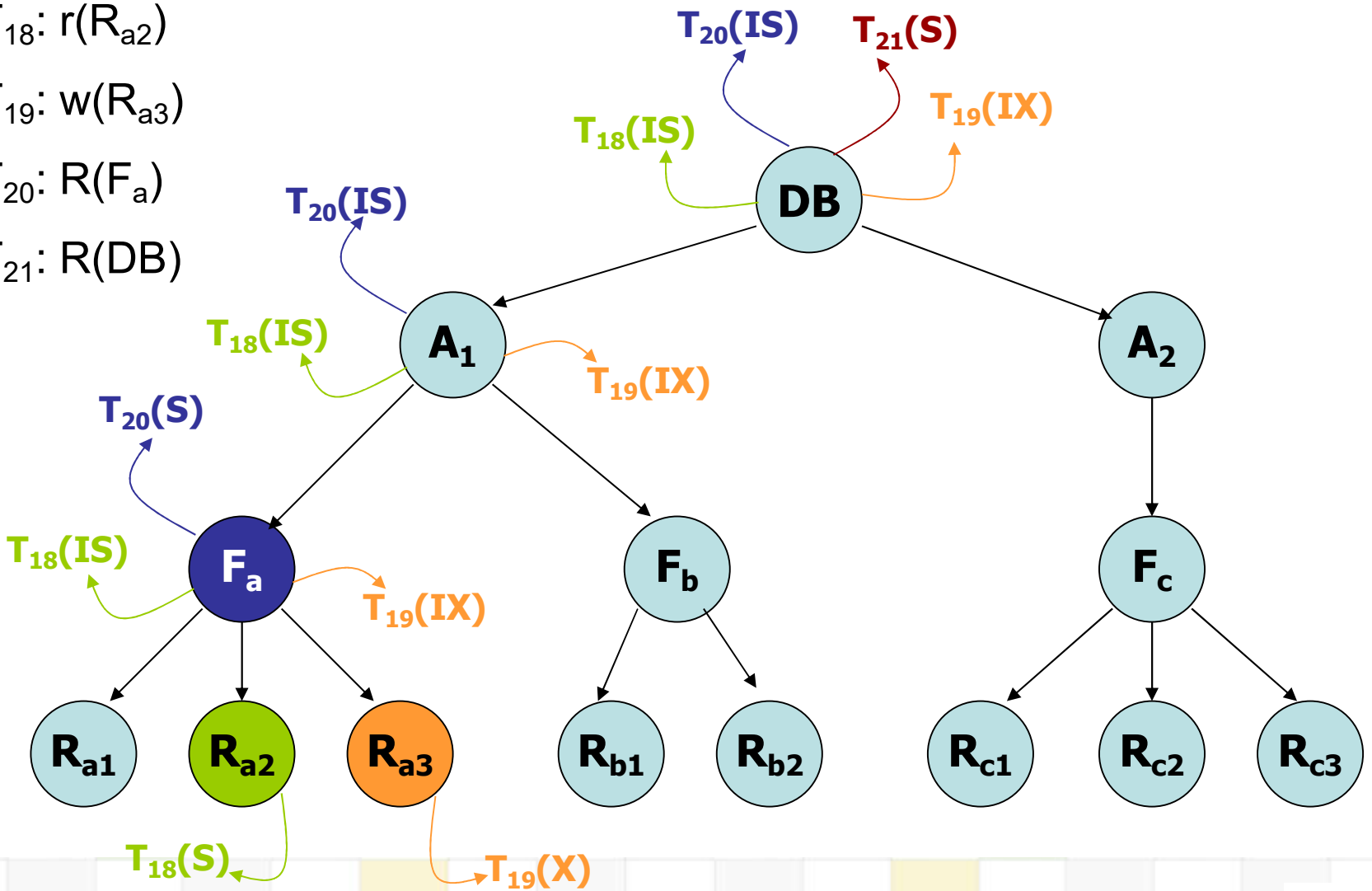
Kỹ thuật khóa đa hạt (tt)

- Qui tắc cấp chốt:
  1. Kiểm chứng hàm tương thích
  2. Gốc của cây phải được chốt đầu tiên
  3. Q có thể chốt bởi T ở {S, IS} nếu cha Q đang bị chốt bởi T ở {IS, IX}
  4. Q có thể chốt bởi T ở {X, SIX, IX} nếu cha Q đang bị chốt bởi T ở {IX, SIX}
  5. T tuân theo giao thức chốt 2 kỳ
  6. T có thể tháo chốt Q nếu không có con của Q đang bị chốt bởi T

⇒ Tậu chốt: Top-Down, tháo chốt: Bottom-Up

Kỹ thuật khóa đa hạt (tt)

- T_{18}: $r(R_{a2})$
- T_{19}: $w(R_{a3})$
- T_{20}: $R(F_a)$
- T_{21}: R(DB)



Kỹ thuật khóa đa hạt (tt)

- T_{19}: w(R_{a3})
- T_{20}: R(F_a)
- T_{21}: R(DB)