II

Giao tác và Lịch giao tác

Nội dung chi tiết

- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

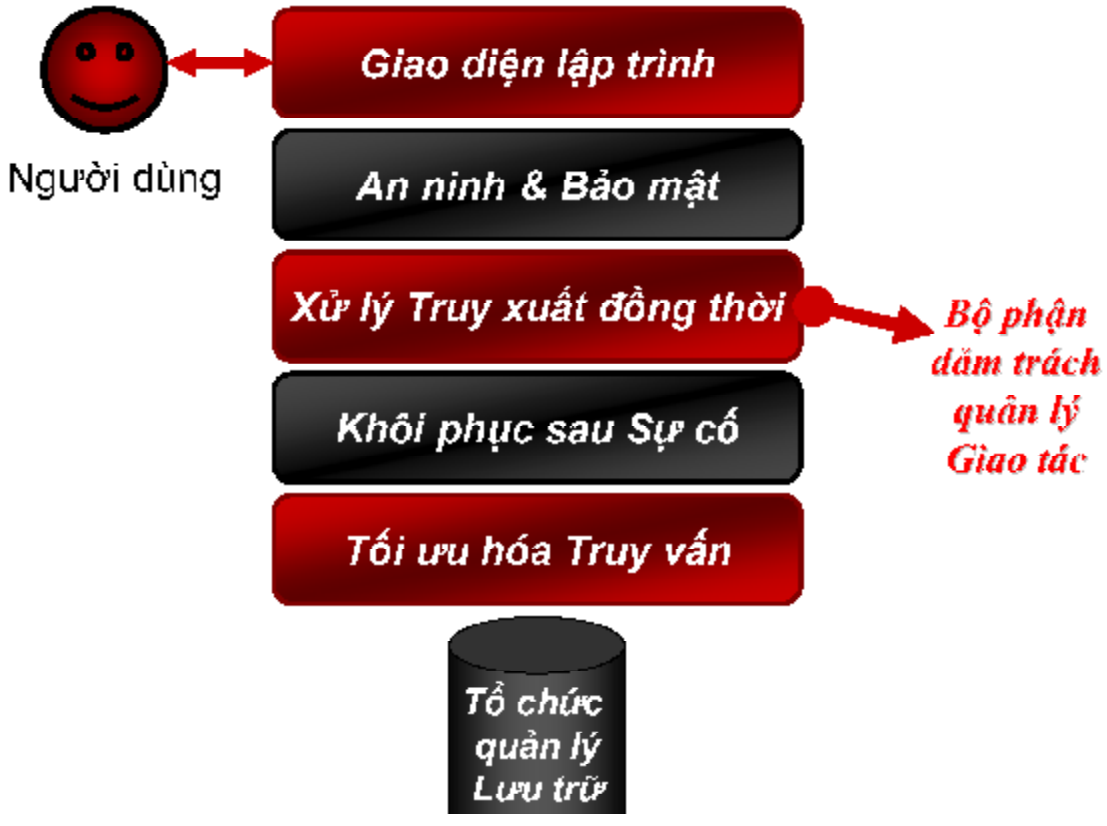
- ***Ví dụ mở đầu***
  - Hệ thống giao dịch ngân hàng
  - Hệ thống đặt vé bay
- ***Thực tế đòi hỏi phải cho phép nhiều người dùng đồng thời khai thác CSDL → Phải giải quyết ổn các tranh chấp***
  - Vd: Hai hành khách cùng đặt một ghế trống
- ***Thực tế phát sinh nhiều sự cố kỹ thuật → Phải bảo đảm nhất quán dữ liệu dù sự cố có sảy ra***
  - Vd: Chuyển tiền từ tài khoản A sang tài khoản B

Nội dung chi tiết

- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

- ***Giao tác là 1 đơn vị xử lý nguyên tố gồm 1 chuỗi các hành động thao tác trên CSDL***
  - Nguyên tố: không thể phân chia được nữa. Các hành động trong 1 giao tác sẽ hoặc là được thực hiện hết, hoặc là không thực hiện bất cứ hành động nào

Giao tác (tt)



- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

Tính chất ACID của giao tác

- ***Nguyên tố (Atomicity)***
  - Hoặc là toàn bộ hoạt động của giao tác được phản ánh đúng đắn trong CSDL hoặc không có hoạt động nào cả
- ***Nhất quán (Consistency)***
  - Khi một giao tác kết thúc (thành công hay thất bại), CSDL phải ở trạng thái nhất quán (đảm bảo mọi RBTV)
- ***Cô lập (Isolation)***
  - Một giao tác không quan tâm đến các giao tác khác xử lý đồng thời với nó
- ***Bền vững (Durability)***
  - Mọi thay đổi mà giao tác thực hiện trên CSDL phải được ghi nhận bền vững vào thiết bị lưu trữ

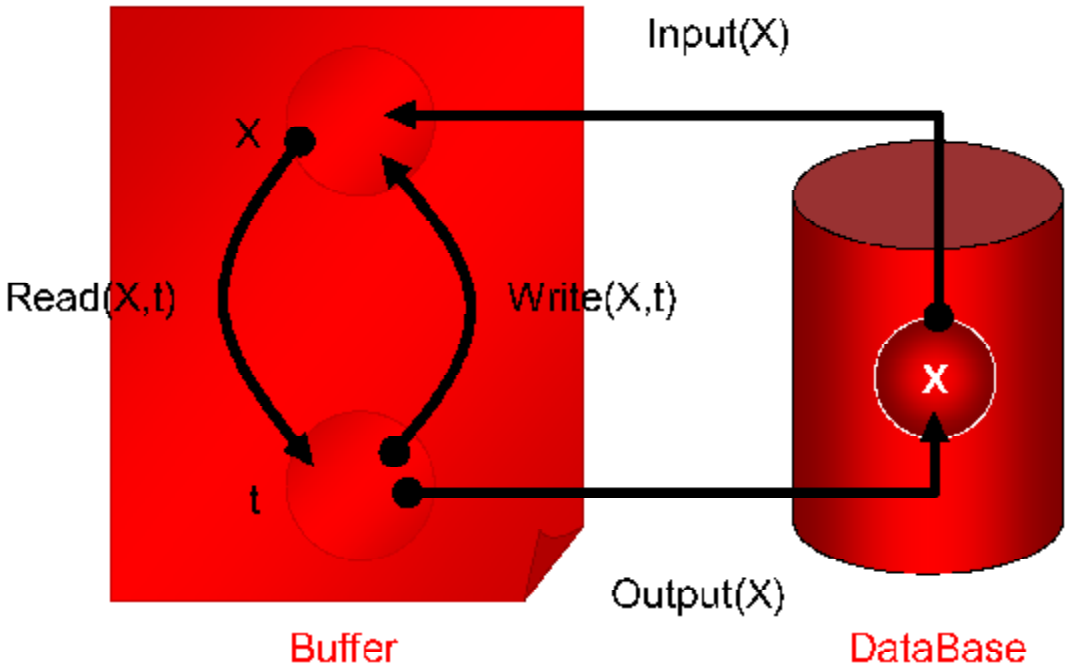
- ***Atomic***
  - Chuyển một khoản tiền từ tài khoản A sang tài khoản B
- ***Consistency***
  - Đăng ký học phần : Số tín chỉ tối đa 32tc / SV / Học kỳ
- ***Durability***
  - Khi T kết thúc thành công
  - Những cập nhật mà T thực hiện trên Dữ liệu sẽ không thể nào bị mất bất chấp có sự cố hệ thống xảy ra
- ***Isolation***
  - Đếm sĩ số lớp >< Thêm (bớt) sinh viên

Nội dung chi tiết

- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

- ***Giả sử CSDL gồm nhiều đơn vị dữ liệu***
- ***Một đơn vị dữ liệu (element)***
  - Có một giá trị
  - Được truy xuất và sửa đổi bởi các giao tác
  - Đơn vị dữ liệu có thể là các thành phần dữ liệu sau đây :
    - Quan hệ (relation) - Lớp (class)
    - Khối dữ liệu trên đĩa (block) - trang (page)
    - Bộ (tuple) - Đối tượng (object)

Các thao tác của giao tác (tt)



- *Giả sử CSDL có 2 đơn vị dữ liệu A và B với ràng buộc A=B trong mọi trạng thái nhất quán*
- *Giao tác T thực hiện 2 bước*
  - A:=A*2
  - B:=B*2
- *Biểu diễn T*
  - Read(A,t) ; t=t*2; Write(A,t);
  - Read(B,t) ; t=t*2; Write(B,t);

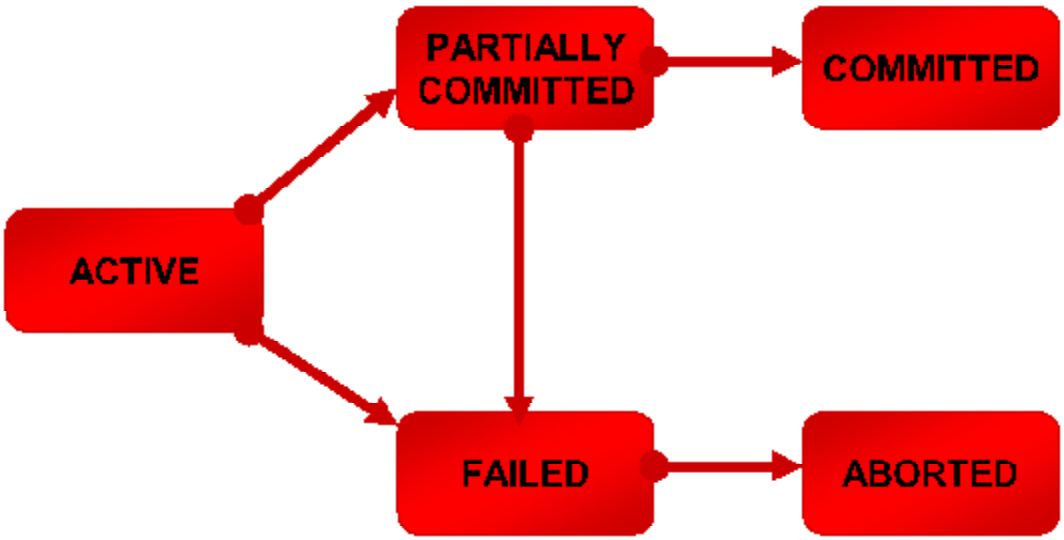
Ví dụ (tt)

Hành động	t	Mem A	Mem B	Disk A	Disk B
Input(A)		8		8	8
Read(A,t)	8	8		8	8
t:=t*2	16	8		8	8
Write(A,t)	16	16		8	8
Input(B)	16	16	8	8	8
Read(B,t)	8	16	8	8	8
t:=t*2	16	16	8	8	8
Write(B,t)	16	16	16	8	8
Output(A)	16	16	16	16	8
Output(B)	16	16	16	16	16

- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

Trạng thái của giao tác

- ***Active***
  - Ngay khi bắt đầu thực hiện thao tác đọc/ghi
- ***Partially committed***
  - Sau khi lệnh thi hành cuối cùng thực hiện
- ***Failed***
  - Sau khi nhận ra không thể thực hiện các hành động được nữa
- ***Aborted***
  - Sau khi giao tác được quay lui và CSDL được phục hồi về trạng thái trước trạng thái bắt đầu giao tác
- ***Committed***
  - Sau khi mọi hành động hoàn tất thành công



Nội dung chi tiết

- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

- ***Thực hiện tuần tự***
  - Các thao tác được gọi là thực hiện tuần tự khi chúng không giao nhau về mặt thời gian
    - Ưu : Nếu thao tác đúng đắn thì luôn bảo đảm nhất quán dữ liệu
    - Khuyết : Không tối ưu về sử dụng tài nguyên và tốc độ
- ***Thực hiện đồng thời***
  - Các lệnh của các giao tác khác nhau xen kẽ vào nhau trên trục thời gian
    - Khuyết : Gây ra nhiều phức tạp về nhất quán dữ liệu
    - Ưu :
      - *Tận dụng tài nguyên và thông lượng (throughput). Ví dụ : Trong khi 1 giao tác đang thực hiện đọc/ghi trên đĩa, 1 giao tác khác đang xử lý tính toán trên CPU*
      - *Giảm thời gian chờ. Ví dụ : Chia sẻ chu kỳ CPU và truy cập đĩa để làm giảm sự trì hoãn trong khi các giao tác thực thi*

Nội dung chi tiết

- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

- ***Một lịch thao tác S được lập từ n giao tác T1, T2, …, Tn được xử lý đồng thời là 1 thứ tự thực hiện xen kẽ các hành động của n giao tác này***
- ***Thứ tự xuất hiện của các thao tác trong lịch phải giống với thứ tự xuất hiện trong giao tác***
- ***Bộ lập lịch (Scheduler) : Là một thành phần của DBMS có nhiệm vụ lập 1 lịch để thực hiện n giao tác xử lý đồng thời***
- ***Các loại lịch thao tác :***
  - Lịch tuần tự (Serial)
  - Lịch khả tuần tự (Serializable)
    - Confict-Serializability
    - Vicw-Scrializability

Nội dung chi tiết

- ***Giới thiệu***
- ***Giao tác***
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- ***Lịch thao tác***
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

- ***Một lịch S được gọi là tuần tự nếu các hành động của các giao tác Ti (i=1..n) được thực hiện liên tiếp nhau, không có sự giao nhau về mặt thời gian***

Lịch tuần tự (tt)

S_1

T_1	T_2	A	B
		25	25
Read(A,t)			
t:=t+100			
Write(A,t)		125	
Read(B,t)			
t:=t+100			
Write(B,t)			125
	Read(A,s)		
	s:=s*2		
	Write(A,s)	**250**	
	Read(B,s)		
	s:=s*2		
	Write(B,s)		**250**

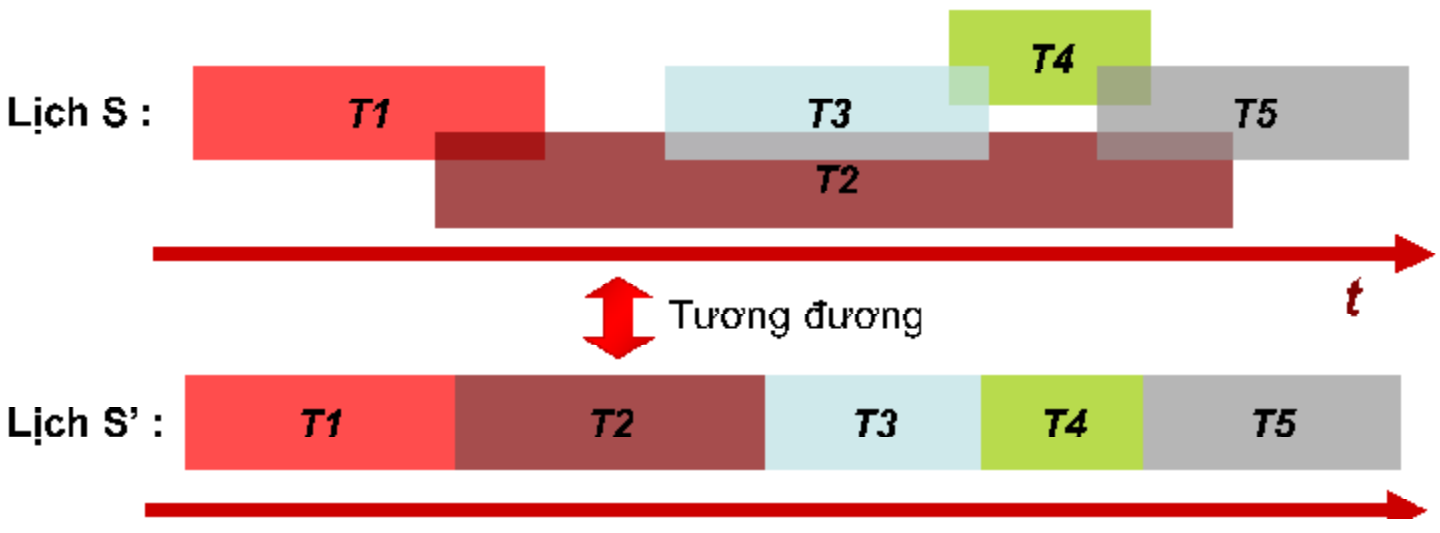
S_2

T_1	T_2	A	B
		25	25
	Read(A,s)		
	s:=s*2		
	Write(A,s)	50	
	Read(B,s)		
	s:=s*2		
	Write(B,s)		50
Read(A,t)			
t:=t+100			
Write(A,t)		**150**	
Read(B,t)			
t:=t+100			
Write(B,t)			**150**

- *Giới thiệu*
- *Giao tác*
  - Khái niệm
  - Tính ACID của Giao tác
  - Các thao tác của Giao tác
  - Các trạng thái của Giao tác
- *Lịch thao tác*
  - Giới thiệu
  - Khái niệm
  - Lịch tuần tự
  - Lịch khả tuần tự

Lịch khả tuần tự (Serializable)

- ***Lịch xử lý đồng thời là lịch mà các giao tác trong đó có giao nhau về mặt thời gian***
- ***Một lịch S được lập từ n giao tác T1, T2, …, Tn xử lý đồng thời được gọi là khả tuần tự nếu nó cho cùng kết quả với 1 lịch tuần tự nào đó được lập từ n giao tác này***



- ***Trước S3 khi thực hiện***
  - A=B=c
  - với c là hằng số
- ***Sau khi S3 kết thúc***
  - A=2*(c+100)
  - B=2*(c+100)
- ***Trạng thái CSDL nhất quán***
- ***S3 là khả tuần tự***

S_3

T_1	T_2	A	B
		25	25
Read(A,t)			
t:=t+100			
Write(A,t)		125	
	Read(A,s)		
	s:=s*2		
	Write(A,s)	**250**	
Read(B,t)			
t:=t+100			
Write(B,t)			125
	Read(B,s)		
	s:=s*2		
	Write(B,s)		**250**

Lịch khả tuần tự (tt)

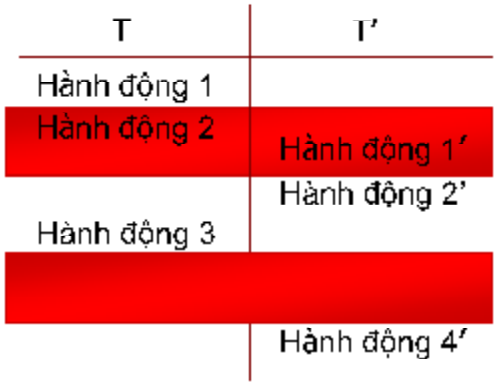
- ***Trước S4 khi thực hiện***
  - A=B=c
  - với c là hằng số
- ***Sau khi S4 kết thúc***
  - A = 2*(c+100)
  - B = 2*c + 100
- ***Trạng thái CSDL không nhất quán***
- ***S4 không khả tuần tự***

S_4

T_1	T_2	A	B
		25	25
Read(A,t)			
t:=t+100			
Write(A,t)		125	
	Read(A,s)		
	s:=s*2		
	Write(A,s)	**250**	
	Read(B,s)		
	s:=s*2		
	Write(B,s)		50
Read(B,t)			
t:=t+100			
Write(B,t)			**150**

Ý tưởng

- Xét 2 hành động liên tiếp nhau của 2 giao tác khác nhau trong 1 lịch thao tác, khi 2 hành động ấy được đảo thứ tự sẽ có thể dẫn đến 1 trong 2 hệ quả :
  - Hoạt động của cả hai giao tác chứa hai hành động ấy không bị ảnh hưởng gì
  - Hoạt động của cả ít nhất một trong hai giao tác chứa hai hành động ấy bị ảnh hưởng



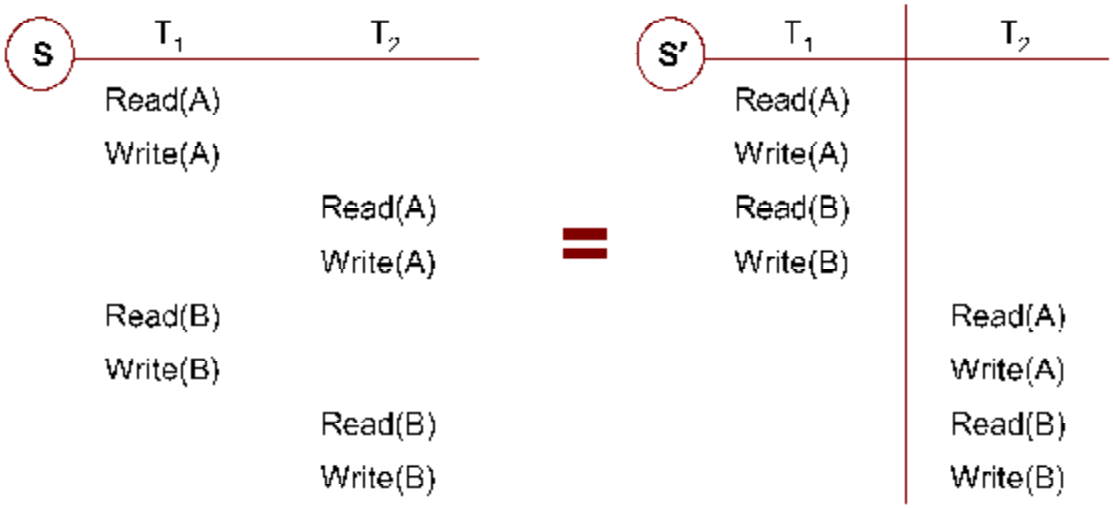
Conflict-Serializability (tt)

- ***Cho lịch S có 2 giao tác T_i và T_j, xét các trường hợp***
  - $r_i(X)$; $r_j(Y)$
    - Không bao giờ có xung đột, ngay cả khi X=Y
    - Cả 2 thao tác không làm thay đổi giá trị của X và Y
  - $r_i(X)$; $w_j(Y)$
    - Không xung đột khi X≠Y
      - *T_j không thay đổi gì dữ liệu đọc của T_i*
      - *T_i không sử dụng gì đến dữ liệu ghi của T_j*
    - Xung đột khi X=Y
  - $w_i(X)$; $r_j(Y)$
    - Không xung đột khi X≠Y, xung đột khi X=Y
  - $w_i(X)$; $w_j(Y)$
    - Không xung đột khi X≠Y, xung đột khi X=Y

- ***Hai hành động xung đột nếu chúng***
  - Thuộc 2 giao tác khác nhau
  - Truy xuất đến cùng 1 đơn vị dữ liệu
  - Trong chúng có ít nhất một hành động là ghi (write)
- ***Hai hành động xung đột thì không thể đảo thứ tự chúng trong lịch thao tác***

Conflict-Serializability (tt)

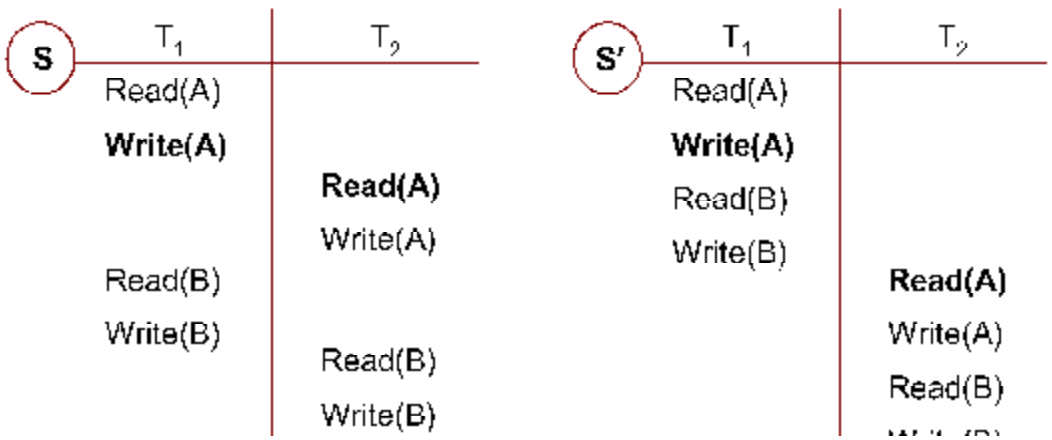
- ***Ví dụ***



- ***Định nghĩa***
  - S, S' là những lịch thao tác conflict-equivalent
    - Nếu S có thể được chuyển thành S' bằng một chuỗi những hoán vị các thao tác không xung đột
  - Một lịch thao tác S là conflict-serializable
    - Nếu S là conflict-equivalent với một lịch thao tác tuần tự nào đó
- ***S conflict-serializable thì S khả tuần tự***
- ***S khả tuần tự thì không chắc S conflict-serializable***

Kiểm tra Conflict-Serializability

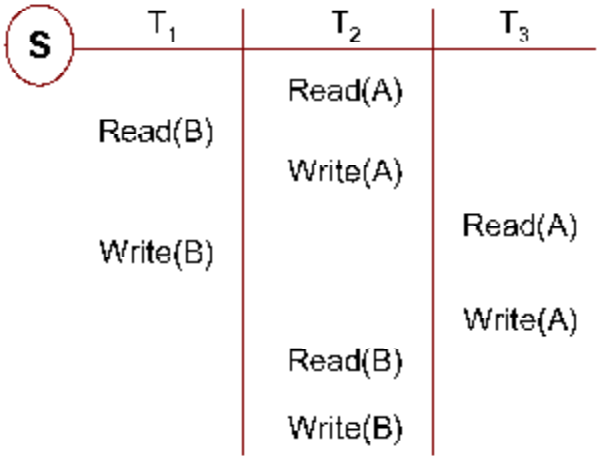
- ***Cho lịch S***
  - S có conflict-serializable không?
- ***Ý tưởng***
  - Các hành động xung đột trong lịch S được thực hiện theo thứ tự nào thì các giao tác thực hiện chúng trong S' (kết quả sau hoán vị) sẽ cũng ở thứ tự đó



- ***Cho lịch S có 2 giao tác T1, T2***
  - T1 thực hiện hành động A1
  - T2 thực hiện hành động A2
  - Ta nói T1 thực hiện trước T2 trong S, ký kiệu $T1 <_S T2$, khi
    - A1 được thực hiện trước A2 trong S
      - *A1 không nhất thiết phải liên tiếp A2*
    - A1 và A2 cùng thao tác lên 1 đơn vị dữ liệu và có ít nhất 1 hành động ghi trong A1 và A2 (nghĩa là A1 và A2 là hai hành động xung đột)

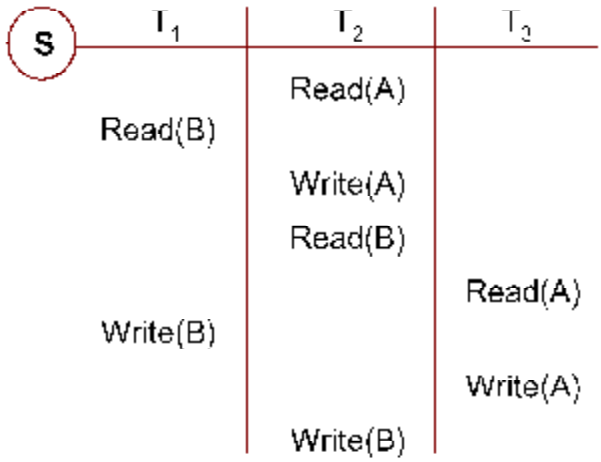
Precedence graph

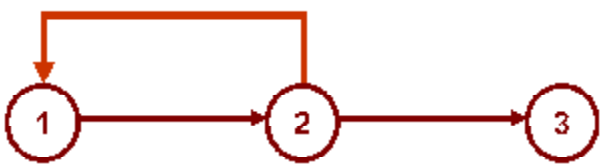
- ***Cho lịch S gồm các giao tác T1, T2, …, Tn***
- ***Đồ thị trình tự (Precedence graph) của S, ký hiệu P(S), có***
  - Đỉnh là các giao tác Ti
    - Ta có thể đặt nhãn cho đỉnh là i
  - Cung đi từ Ti đến Tj nếu $Ti <_S Tj$
- ***Nếu P(S) không có chu trình khi và chỉ khi S conflict-serializable***
- ***Thứ tự hình học (topological order) của các đỉnh là thứ tự của các giao tác trong lịch tuần tự tương đương với S***
- ***S và S' gọi là conflict equivalent khi và chỉ khi P(S) = P(S')***



- ***P(S) không có chu trình***
- ***S conflict-serializable theo thứ tự T_1, T_2, T_3***

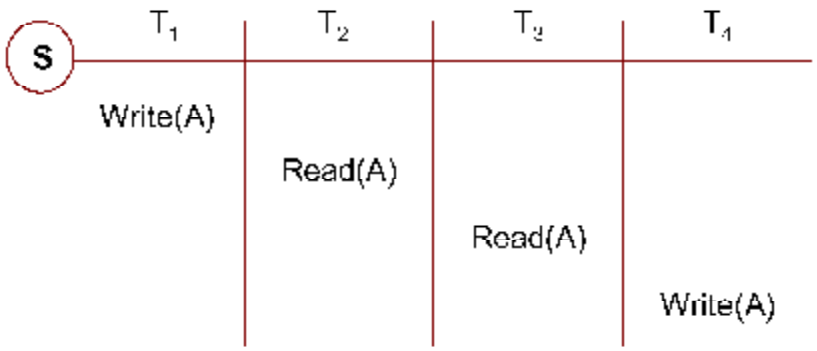
Ví dụ (tt)





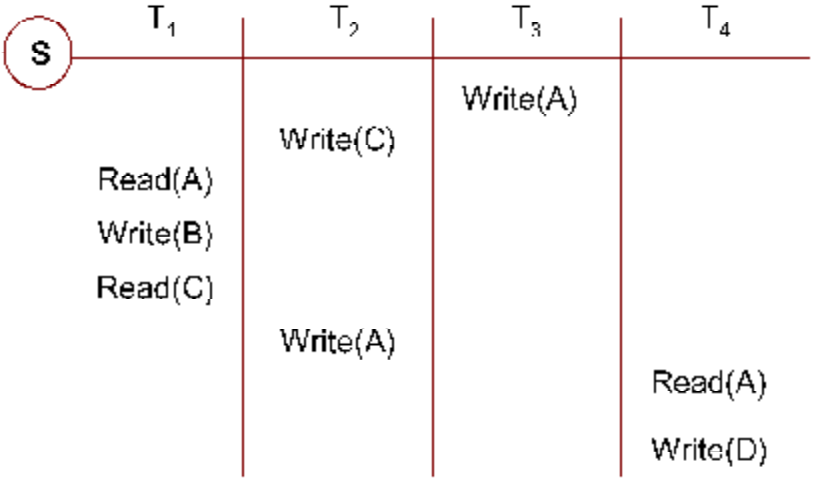
- ***P(S) có chu trình***
- ***S không conflict-serializable***

- *Vẽ P(S)*
- *S có conflict-serializable không?*

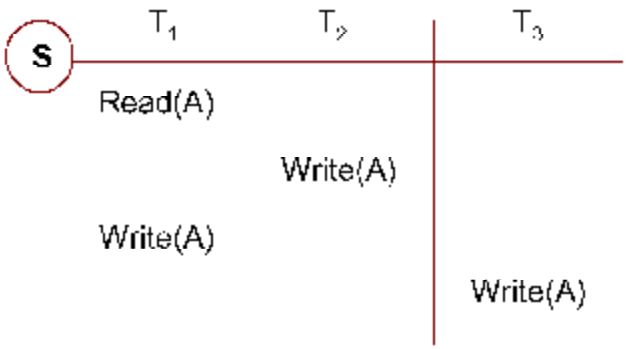


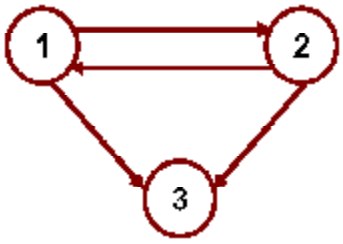
Bài tập (tt)

- *Vẽ P(S)*
- *S có conflict-serializable không?*



- ***Xét lịch S***

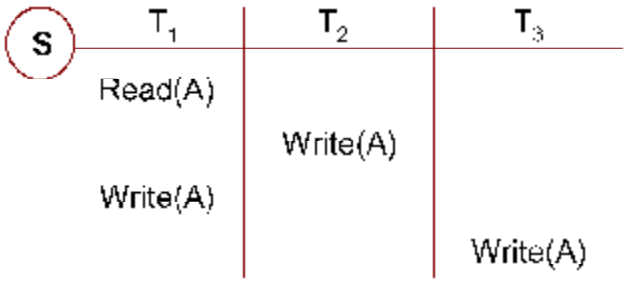




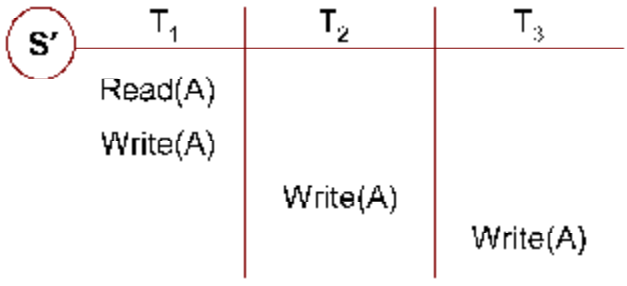
- ***P(S) có chu trình***
- ***S không conflict-serializable***

View-Serializability (tt)

- ***So sánh lịch S và 1 lịch tuần tự S'***
  - Trong S và S' đều có T1 thực hiện read(A)
  - T2 và T3 không đọc A
  - Kết quả của S và S' giống nhau → S vẫn serializable



Không conflict-serializable



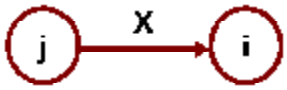
Serial

- ***Định nghĩa***
  - S, S' là những lịch thao tác view-equivalent nếu
    - 1- Thành phần và thứ tự thao tác của T_i bất kỳ là như nhau trong S và S'
    - 2- Nếu trong S có T_i đọc giá trị ban đầu của A thì trong S' cũng chính T_i đọc giá trị ban đầu của A
    - 3- Nếu trong S có T_j ghi giá trị sau cùng lên A thì trong S' cũng chính T_j ghi giá trị sau cùng lên A
  - Một lịch thao tác S là view-serializable
    - Nếu S là view-equivalent với một lịch thao tác tuần tự nào đó
- ***S conflict-serializable thì S view-serializable (chiều ngược lại không chắc đúng)***

Kiểm tra View-Serializability (tt)

- ***Cho 1 lịch thao tác S***
- ***Thêm 1 giao tác cuối Tf vào trong S sao cho Tf thực hiện việc đọc hết tất cả đơn vị dữ liệu ở trong S***
  - S = … w1(A)……w2(A) rf(A)
    - (bỏ qua điều kiện thứ 3 của định nghĩa view-equivalent)
- ***Thêm 1 giao tác đầu tiên Tb vào trong S sao cho Tb thực hiện việc ghi các giá trị ban đầu cho các đơn vị dữ liệu***
  - S = wb(A)… w1(A)……w2(A)…
    - (bỏ qua điều kiện thứ 2 của định nghĩa view-equivalent)

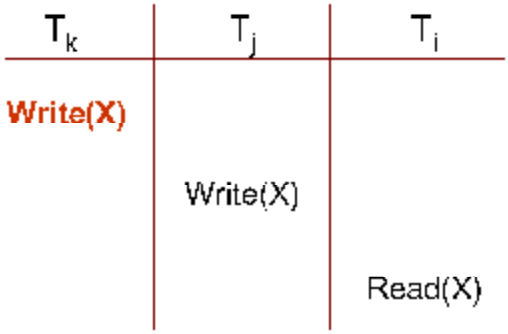
- ***Vẽ đồ thị phức (PolyGraph) cho S, ký hiệu G(S) với***
  - Đỉnh là các giao tác Ti (bao gồm cả Tb và Tf)
  - Cung
    - (1) Nếu giá trị mà ri(X) đọc được là do Tj ghi *(ri(X) có gốc là Tj)* thì vẽ cung đi từ Tj đến Ti

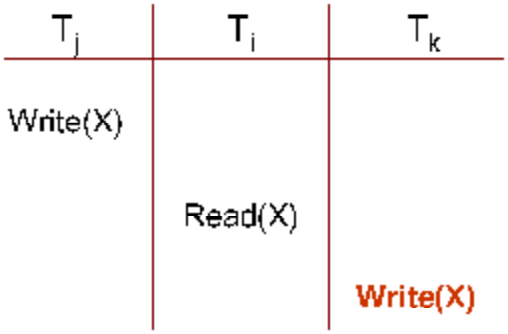


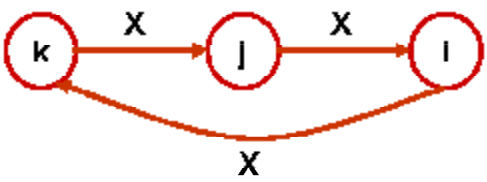
    - (2) Với mỗi wj(X) … ri(X), xét wk(X) khác Tb sao cho Tk không chèn vào giữa Tj và Ti

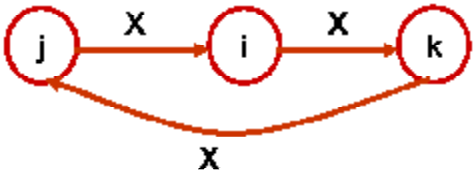
Kiểm tra View-Serializability (tt)

- (2a) Nếu Tj ≠ Tb và Ti ≠ Tf thì vẽ cung Tk → Tj và Ti → Tk

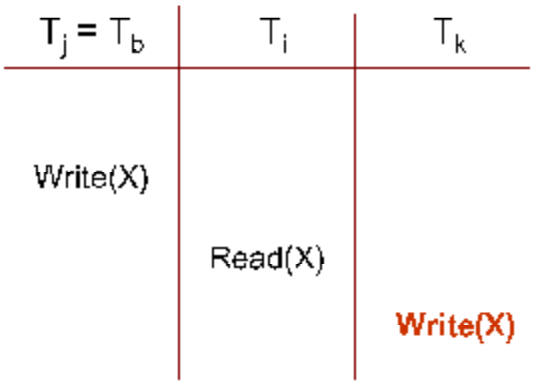


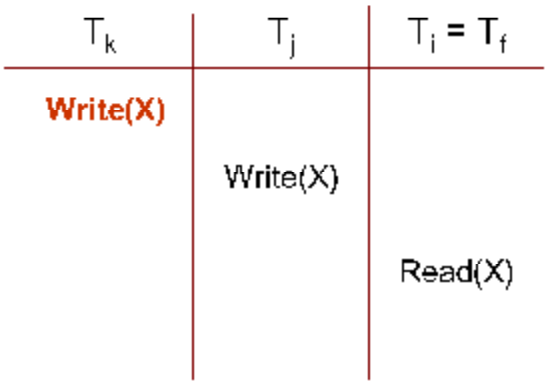




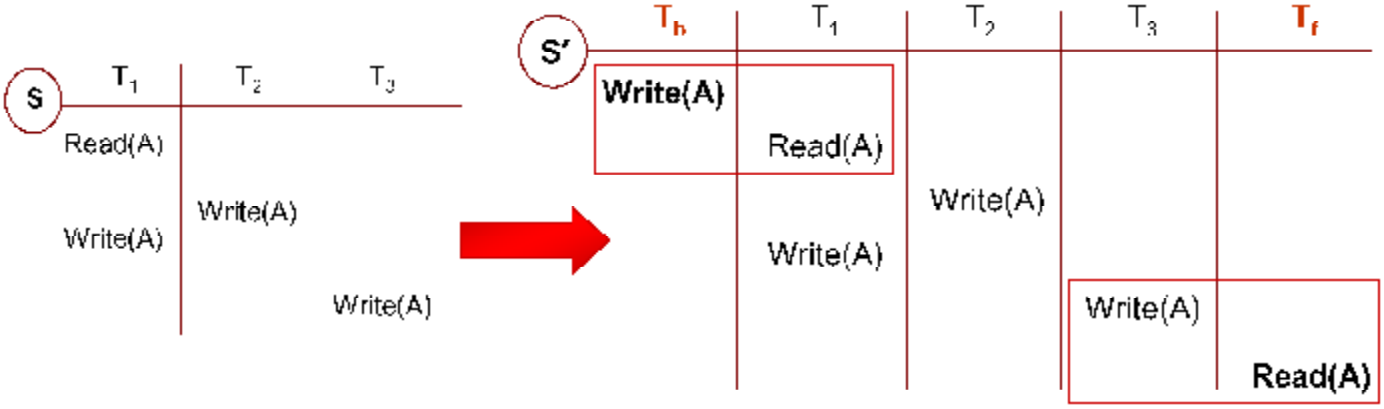


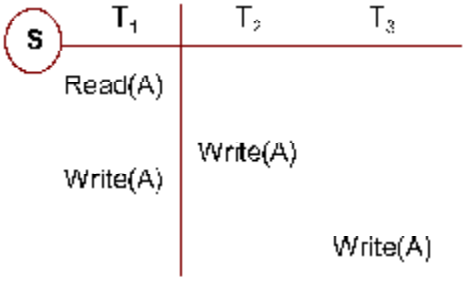
- (2b) Nếu Tj = Tb thì vẽ cung Ti → Tk
- (2c) Nếu Ti = Tf thì vẽ cung Tk → Tj

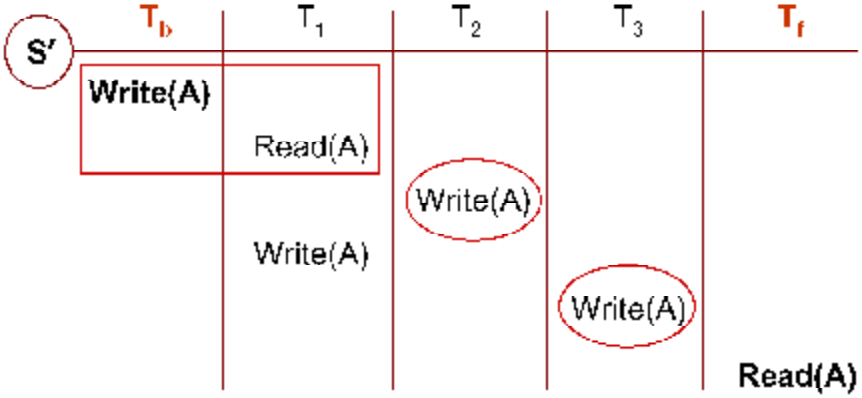


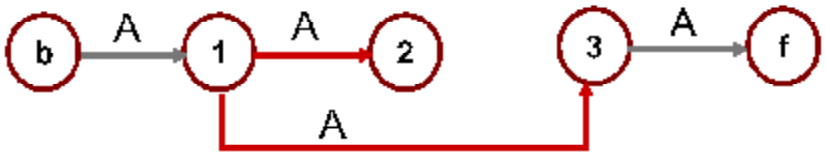


Ví dụ

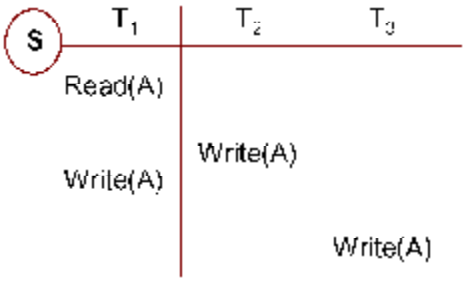


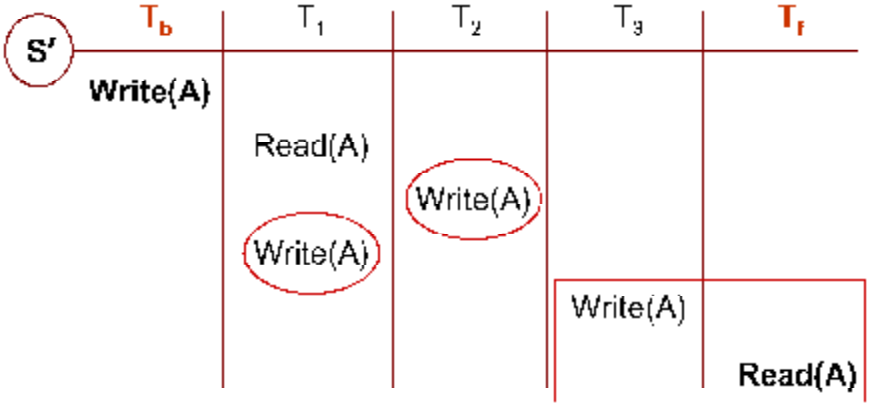




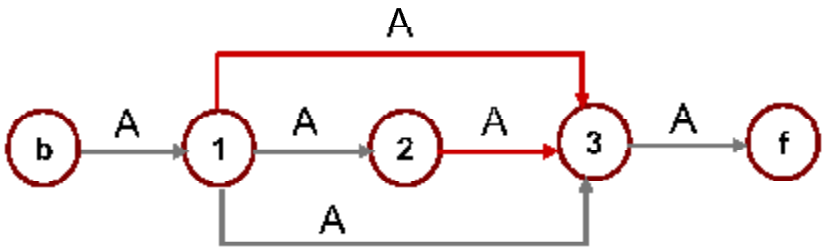


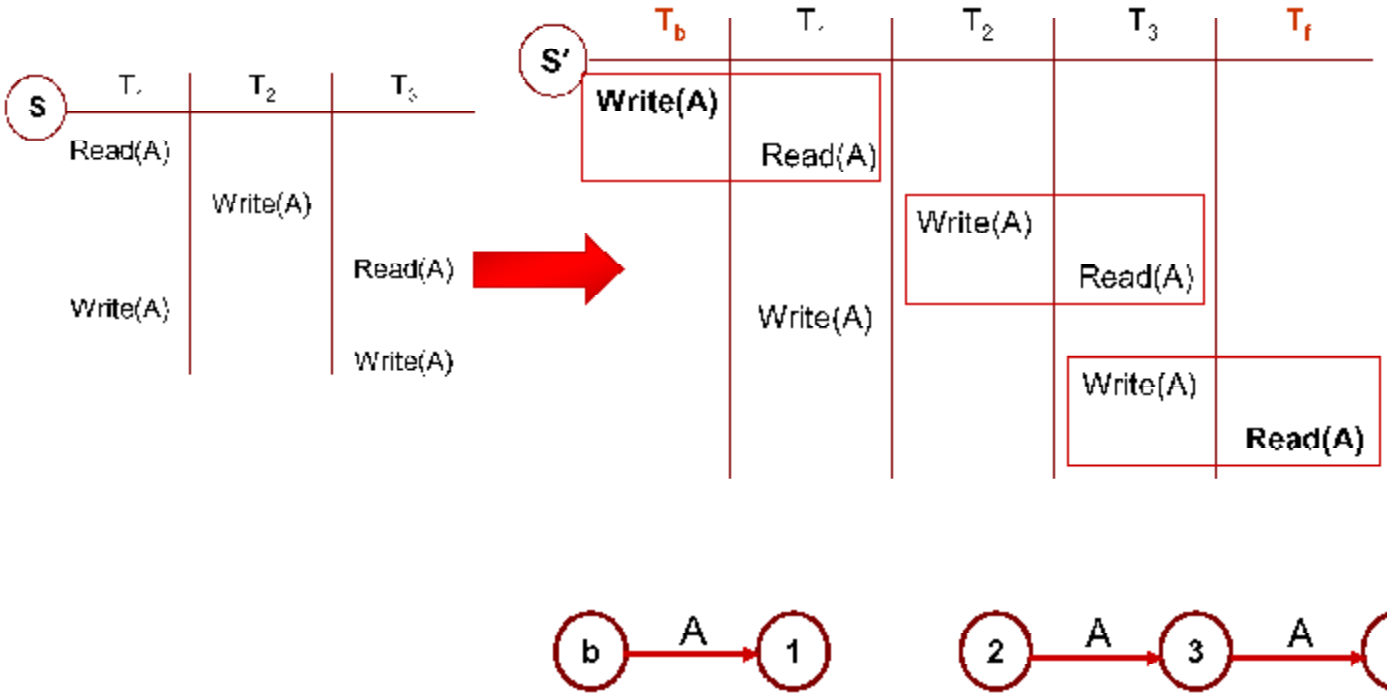
Ví dụ





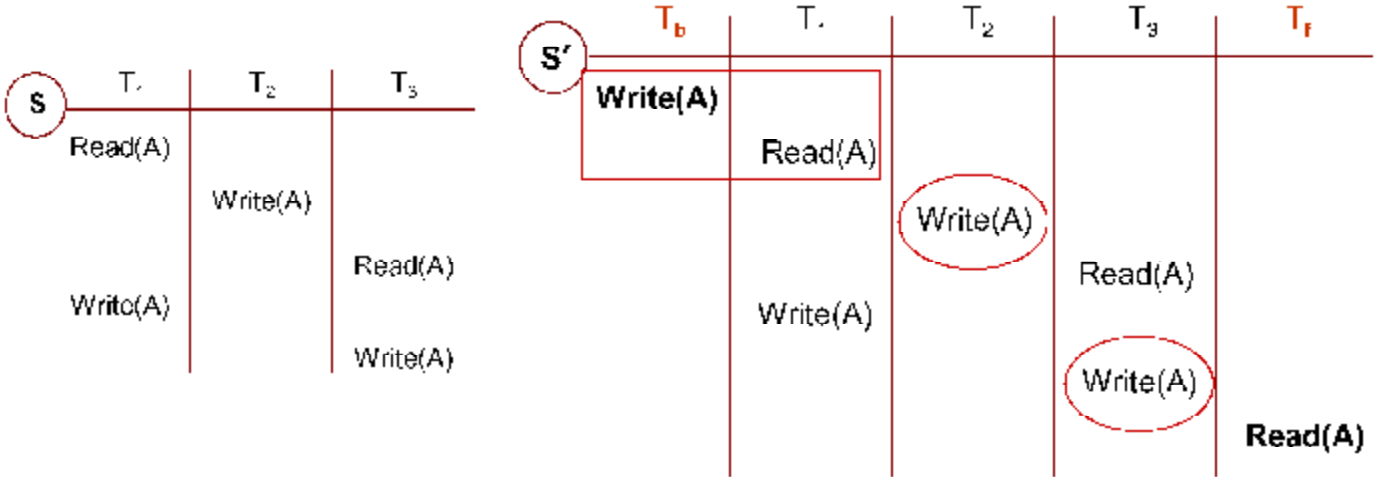
- *G(S) không có chu trình*
- *S view-serializable theo thứ tự T_b, T_1, T_2, T_3, T_f*

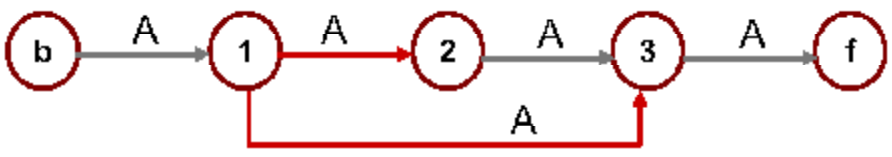


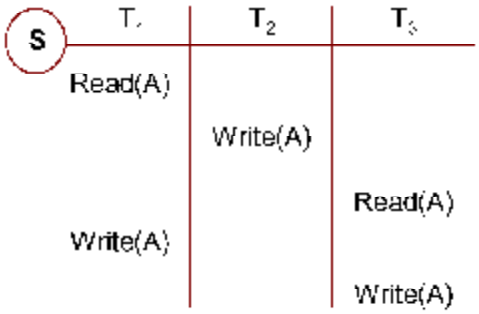

S
T1
T2
T3
Read(A)
Write(A)
Read(A)
Write(A)
Write(A)
S'
Tb
T1
T2
T3
Tf
Write(A)
Read(A)
Write(A)
Read(A)
Write(A)
Read(A)

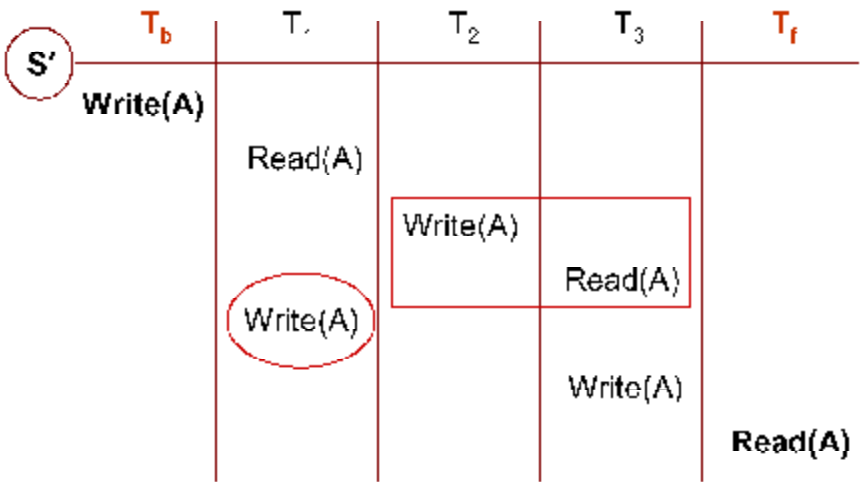
b
A
1
2
A
3
A
f

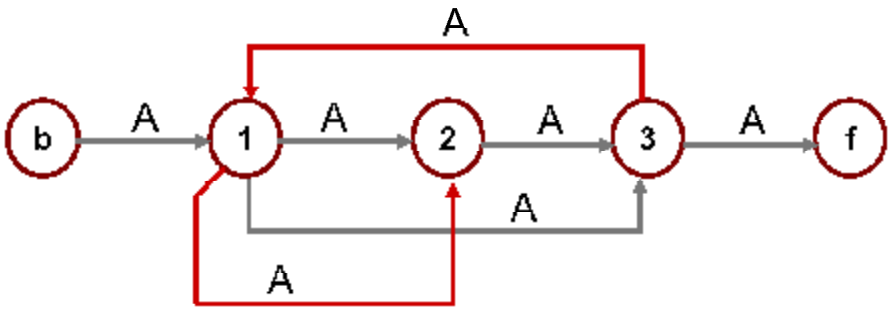
Ví dụ (tt)


S
T1
T2
T3
Read(A)
Write(A)
Write(A)
Read(A)
Write(A)
S'
Tb
T1
T2
T3
Tf
Write(A)
Read(A)
Write(A)
Read(A)
Write(A)
Read(A)

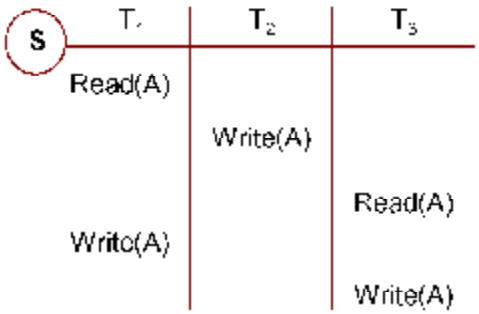

b
A
1
A
2
A
3
A
f
A

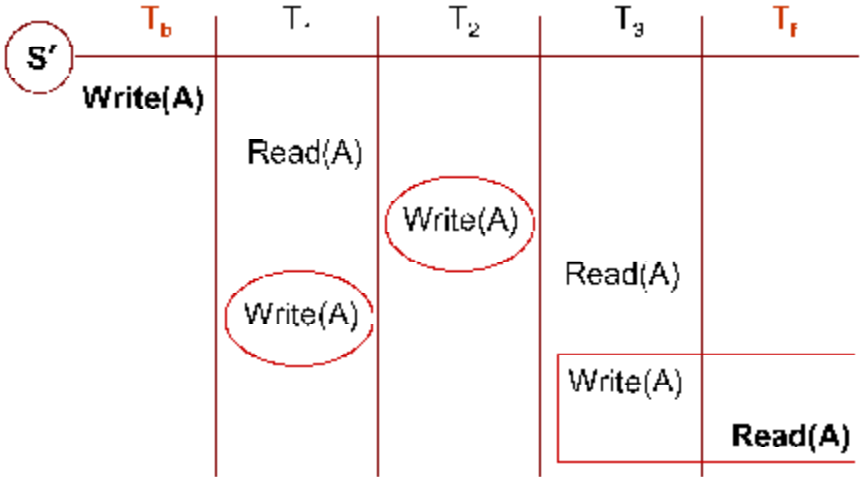


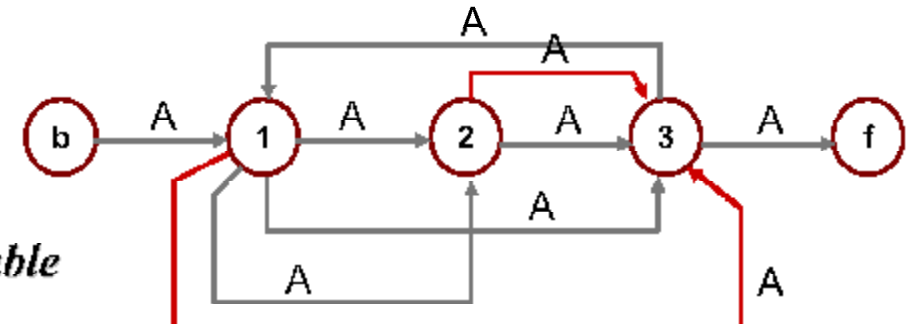




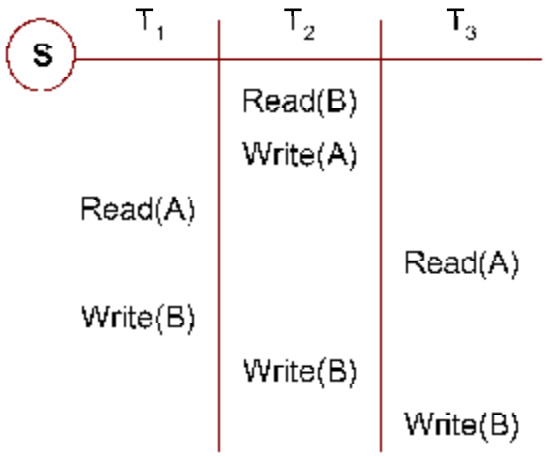
Ví dụ (tt)





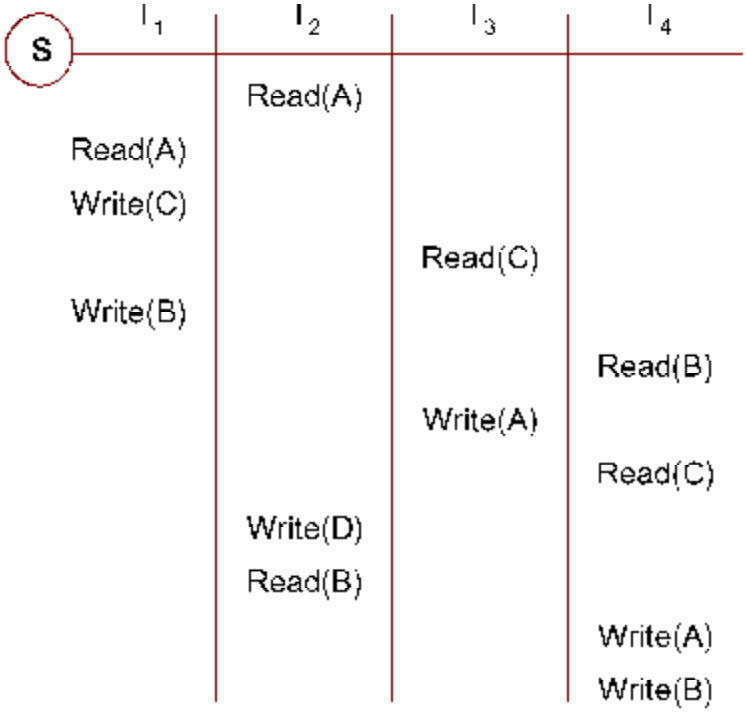


- ***G(S) có chu trình***
- ***S không view-serializable***



- *Vẽ G(S)*
- *S có view-serializable?*

Bài tập (tt)



- *Vẽ G(S)*
- *S có view-serializable?*