Serializabilidad

Planificación de transacciones

- Cada transacción comprende una secuencia de operaciones que incluyen acciones de lectura y escritura en la BD, que finaliza con una confirmación (commit) o anulación (rollback)

- Una **planificación** P **de** n **transacciones concurrentes** $T_1, T_2 \ldots T_n$ es una **secuencia de las operaciones** realizadas por dichas transacciones, sujeta a la restricción de que
  para cada transacción T_i que participa en P,
  sus operaciones aparecen **en** P
  en el mismo orden en el **que** ocurren **en** T_i

Serializabilidad

Planificación de transacciones

- Para el control de la concurrencia (y recuperación de fallos) interesa prestar mayor atención a estas **operaciones**:

operación	abreviatura
leer_elemento	ℓ
escribir_elemento	e
commit	c
rollback	r

- Ejemplos de planificaciones de transacciones
  - El subíndice de cada operación indica la transacción que la realiza

P_A: $\ell_1(X)$; $e_1(X)$; $\ell_1(Y)$; $e_1(Y)$; c_1 ; $\ell_2(X)$; $e_2(X)$; c_2 ;
P_B: $\ell_2(X)$; $e_2(X)$; c_2 ; $\ell_1(X)$; $e_1(X)$; $\ell_1(Y)$; $e_1(Y)$; c_1 ;
P_C: $\ell_1(X)$; $\ell_2(X)$; $e_1(X)$; $\ell_1(Y)$; $e_2(X)$; c_2 ; $e_1(Y)$; c_1 ;
P_D: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; c_2 ; $\ell_1(Y)$; $e_1(Y)$; c_1 ;
P_E: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; c_2 ; $\ell_1(Y)$; r_1 ;

Serializabilidad

Planificación serie

- Una **planificación serie** P es aquella en la que **las operaciones de cada transacción se ejecutan consecutivamente** sin que se intercalen operaciones de otras transacciones

P_A: $\ell_1(X)$; $e_1(X)$; $\ell_1(Y)$; $e_1(Y)$; c_1 ; $\ell_2(X)$; $e_2(X)$; c_2 ;
P_B: $\ell_2(X)$; $e_2(X)$; c_2 ; $\ell_1(X)$; $e_1(X)$; $\ell_1(Y)$; $e_1(Y)$; c_1 ;

- Toda planificación serie es **correcta** ▸ BD consistente
- Pero no se garantiza que los resultados de todas las ejecuciones en serie de las mismas transacciones sean idénticos
  - Ejemplo: cálculo del interés de una cuenta bancaria antes o después de realizar un ingreso considerable

▸ en general, son inaceptables en la práctica (ineficiencia)

Serializabilidad

Planificación no serie

- Una **planificación no serie** P es aquella en la que **las operaciones de un conjunto de transacciones concurrentes se ejecutan intercaladas**

P_C: $\ell_1(X)$; $\ell_2(X)$; $e_1(X)$; $\ell_1(Y)$; $e_2(X)$; c_2 ; $e_1(Y)$; c_1 ; KO
P_D: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; c_2 ; $\ell_1(Y)$; $e_1(Y)$; c_1 ; ☺

- Hemos de determinar **qué planificaciones no serie permiten llevar la BD a un estado al que pueda llegarse mediante una ejecución en serie**

Este es el objetivo de la Serializabilidad

Serializabilidad

Planificación serializable

- Una **planificación** P (no serie) es **serializable** si es **equivalente a alguna planificación serie de las mismas** n **transacciones**
  - Una planificación que no es equivalente a **ninguna** ejecución en serie, es una planificación **no serializable**
- Toda planificación serializable es **correcta**
  - Produce los mismos resultados que alguna ejecución en serie
- Dos maneras de definir la equivalencia entre planificaciones:
  - **Equivalencia por conflictos**
  - **Equivalencia de vistas**

Serializabilidad

Equivalencia por conflictos

- Si dos transacciones **únicamente leen** un determinado **elemento** de datos, no entran en conflicto entre sí y el orden de las operaciones no es importante
- Si hay dos transacciones que **leen o escriben elementos** de datos **independientes**, no entran en conflicto entre sí y el orden de las operaciones no es importante
- Si **una** de las transacciones **escribe** un elemento de datos **y la otra lee o escribe el mismo elemento**, entran en **conflicto** y **el orden de las operaciones sí es importante**

Serializabilidad

Equivalencia por conflictos

- En una planificación, 2 **operaciones** están **en conflicto** si
  - pertenecen a **diferentes transacciones**,
  - tienen acceso al **mismo elemento** X,
  - y al menos una de ellas es escribir_elemento(X)

Operaciones en conflicto en las planificaciones P_C y P_D:

P_C: $\ell_1(X)$; $\ell_2(X)$; $e_1(X)$; $\ell_1(Y)$; $e_2(X)$; c_2 ; $e_1(Y)$; c_1;

P_D: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; c_2 ; $\ell_1(Y)$; $e_1(Y)$; c_1 ;

- Dos **planes** son **equivalentes por conflictos** si **el orden de cualesquiera dos operaciones en conflicto es el mismo en ambos planes**

Serializabilidad

Planificación serializable por conflictos

- Una **planificación** P es **serializable por conflictos** si **equivale por conflictos a alguna planificación serie** S
  - Podremos intercambiar cada dos operaciones de P consecutivas de transacciones distintas y sin conflicto, hasta obtener la planificación serie equivalente

P_D : $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; c_2 ; $\ell_1(Y)$; $e_1(Y)$; c_1 ;

P_{D1}: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; $\ell_1(Y)$; c_2 ; $e_1(Y)$; c_1 ;

P_{D2}: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; $\ell_1(Y)$; $e_1(Y)$; c_2 ; c_1 ;

P_{D3}: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $e_2(X)$; $\ell_1(Y)$; $e_1(Y)$; c_1 ; c_2 ;

P_{D4}: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $\ell_1(Y)$; $e_2(X)$; $e_1(Y)$; c_1 ; c_2 ;

P_{D5}: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $\ell_1(Y)$; $e_1(Y)$; $e_2(X)$; c_1 ; c_2 ;

P_{D6}: $\ell_1(X)$; $e_1(X)$; $\ell_2(X)$; $\ell_1(Y)$; $e_1(Y)$; c_1 ; $e_2(X)$; c_2 ;

P_{D7}: $\ell_1(X)$; $e_1(X)$; $\ell_1(Y)$; $\ell_2(X)$; $e_1(Y)$; c_1 ; $e_2(X)$; c_2 ;

P_{D8}: $\ell_1(X)$; $e_1(X)$; $\ell_1(Y)$; $e_1(Y)$; $\ell_2(X)$; c_1 ; $e_2(X)$; c_2 ;

P_{D9}: $\ell_1(X)$; $e_1(X)$; $\ell_1(Y)$; $e_1(Y)$; c_1 ; $\ell_2(X)$; $e_2(X)$; c_2 ;

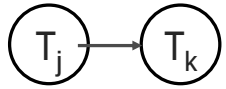
¡Es una planificación serie! P_D es serializable

Serializabilidad

Detección de la serializabilidad por conflictos

- Construcción del **grafo de precedencia** (o de serialización)
  - Es un grafo dirigido G = (N, A)
  - N es un conjunto de **nodos** y A es un conjunto de **aristas dirigidas**
  - Algoritmo:
    - Crear **un nodo por** cada **transacción** T_i en P

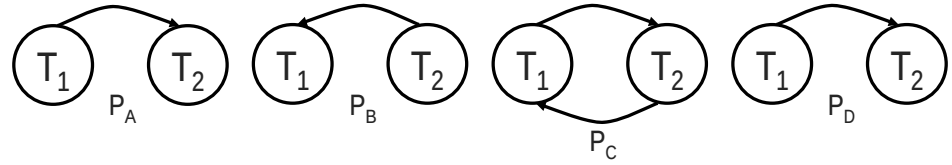
    - Crear una arista $T_j \rightarrow T_k$ si T_k **lee** el valor de un elemento **después** de que T_j lo haya **escrito**
    - Crear una arista $T_j \rightarrow T_k$ si T_k **escribe** el valor de un elemento **después** de que T_j lo haya **leído**
    - Crear una arista $T_j \rightarrow T_k$ si T_k **escribe** el valor de un elemento **después** de que T_j lo haya **escrito**



Serializabilidad

Detección de la serializabilidad por conflictos (y 2)

- Una arista $T_j \rightarrow T_k$ indica que T_j debe aparecer **antes** que T_k **en una planificación serie equivalente** a P, pues dos operaciones en conflicto aparecen en dicho orden en P
- Si el grafo contiene un **ciclo**, P **no es serializable** por conflictos
  - Un **ciclo** es una secuencia de aristas $C=((T_j \rightarrow T_k), (T_k \rightarrow T_p),... (T_i \rightarrow T_j))$
- Si **no** hay **ciclos** en el grafo, **P es serializable**
  - Es posible obtener una planificación serie S equivalente a P, mediante una ordenación topológica de los nodos



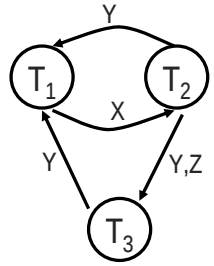
Serializabilidad

Ejemplo de planificación no serializable

Transacción T1	Transacción T2	Transacción T3
leer_elemento(X);	leer_elemento(Z);	leer_elemento(Y);
escribir_elemento(X);	leer_elemento(Y);	leer_elemento(Z);
leer_elemento(Y);	escribir_elemento(Y);	escribir_elemento(Y);
escribir_elemento(Y);	leer_elemento(X);	escribir_elemento(Z);
	escribir_elemento(X);	

T1	T2	T3
	leer_elemento(Z);	
	leer_elemento(Y);	
	escribir_elemento(Y);	
		leer_elemento(Y);
		leer_elemento(Z);
leer_elemento(X);		
escribir_elemento(X);		
		escribir_elemento(Y);
		escribir_elemento(Z);
	leer_elemento(X);	
leer_elemento(Y);		
escribir_elemento(Y);		
	escribir_elemento(X);	

Planificación E



Hay dos ciclos:
$T_1 \rightarrow T_2 \rightarrow T_1$ y
$T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$

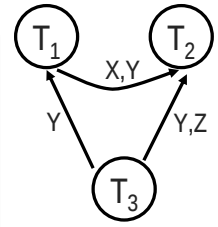
Serializabilidad

Ejemplo de planificación serializable

Transacción T1	Transacción T2	Transacción T3
leer_elemento(X);	leer_elemento(Z);	leer_elemento(Y);
escribir_elemento(X);	leer_elemento(Y);	leer_elemento(Z);
leer_elemento(Y);	escribir_elemento(Y);	escribir_elemento(Y);
escribir_elemento(Y);	leer_elemento(X);	escribir_elemento(Z);
	escribir_elemento(X);	

T1	T2	T3
		leer_elemento(Y);
		leer_elemento(Z);
leer_elemento(X);		
escribir_elemento(X);		
		escribir_elemento(Y);
		escribir_elemento(Z);
	leer_elemento(Z);	
leer_elemento(Y);		
escribir_elemento(Y);		
	leer_elemento(Y);	
	escribir_elemento(Y);	
	leer_elemento(X);	
	escribir_elemento(X);	

Planificación F



La **planificación serie equivalente** es T3 → T1 → T2