Two-Phase Locking

Gabriela Vargas
Dalia Camacho

Serializability

- Mecanismo utilizado para el control de concurrencias en bases de datos y sistemas de transacción.
  - Ejemplo: Sistema de reserva de vuelos que debe procesar varias transacciones al mismo tiempo (probablemente interrelacionadas).
    P1 - Cancela N asientos en el vuelo X y reserva N asientos en el vuelo Y.
    P2 - Reserva M asientos en el vuelo X.
- Es necesario para controlar las intervenciones de los procesos en ejecuciones que se dan de manera concurrente.
- Por lo tanto las acciones de los procesos no se entrelazan en el tiempo y no interfieren en la ejecución de otro proceso.

Serializability

T1	T2
R	R
W	W
W	R
R	

Primero T1 y después T2

Two-phase locking

- Algoritmo de control de sistemas concurrentes que garantiza serializabilidad.
- Ejecución del sistema $\alpha \rightarrow$ Secuencia de acciones. Donde cada acción pertenece a un solo proceso.
- Acciones
  - Acceso a variables compartidas.
  - Adquirir o liberar candados sobre las variables compartidas.

Two-phase locking

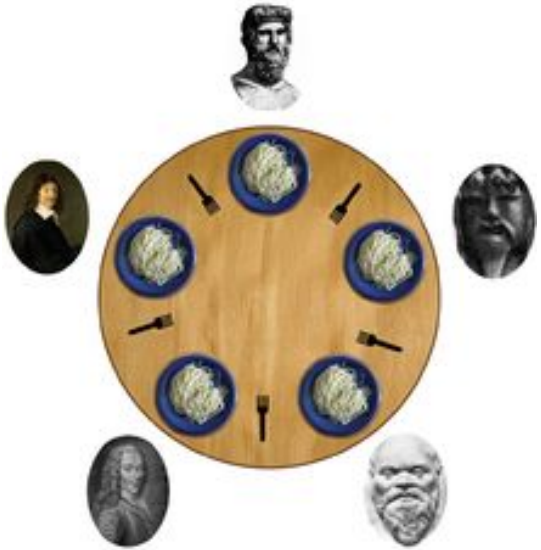
Indistinguibilidad

Dos ejecuciones α_1 y α_2 son indistinguibles a un proceso, si ambas ejecutan las mismas acciones del proceso en cuestión y se obtienen los mismos resultados.

Una ejecución es serializable si es indistinguible de una ejecución secuencial para todos los procesos, donde cada P_i completaría sus acciones sin la intervención de algún otro P_k.

El algoritmo two-phase locking garantiza serializabilidad mediante la aplicación de candados a variables globales.

Dining Philosophers Problem



- 5 filósofos se sientan a cenar en una mesa circular.
- Cada filósofo tiene un plato de fideos y un palillo a su izquierda. Pero para comer los fideos son necesarios dos palillos y únicamente pueden tomar los palillos que están a la izquierda o derecha de su plato.
- Si un filósofo toma un palillo y el otro está ocupado, se esperará esperando hasta que el filósofo que lo está ocupando haya terminado.
- Si todos los filósofos toman el tenedor que está a su derecha al mismo tiempo, entonces se quedarán esperando eternamente con un bloqueo mútuo (*deadlock*)

Two-phase locking

Si en vez de 5 filósofos fueran sólo dos, los procesos para cada uno se pueden describir como:

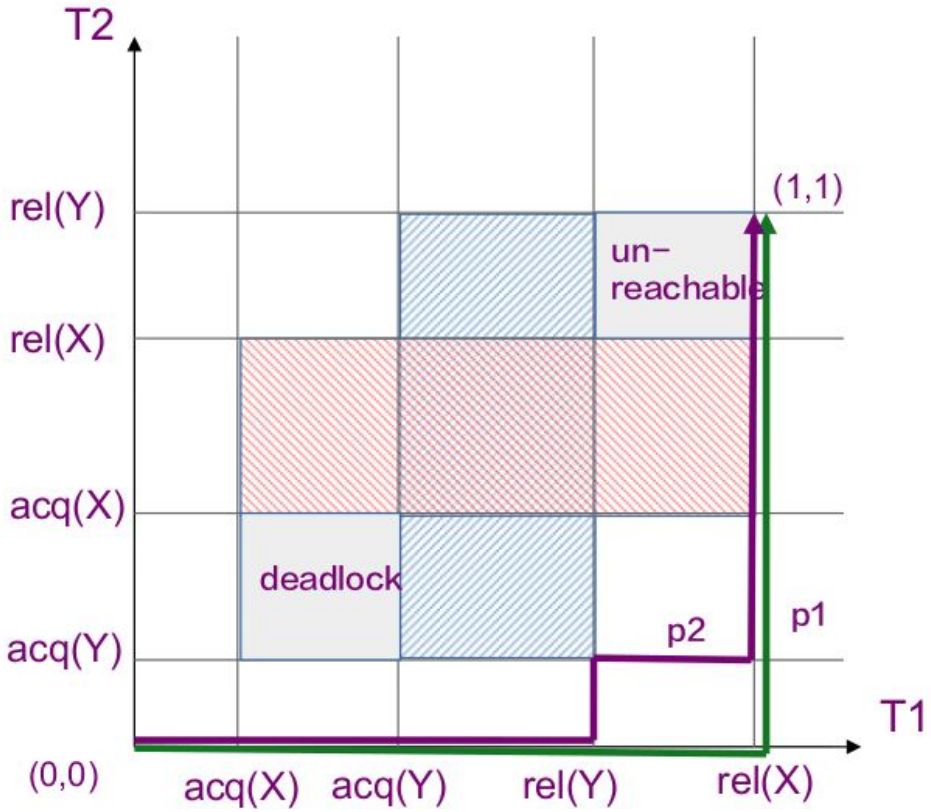
$$T1 : ack(X); ack(Y); rel(Y); rel(X)$$

$$T2 : ack(Y); ack(X); rel(X); rel(Y)$$

Donde sólo se tienen dos palillos el X y el Y

Two-phase locking

Las ejecuciones se pueden representar en un plano, donde las acciones de el proceso uno corresponden al eje X y las del proceso dos corresponden al eje Y.

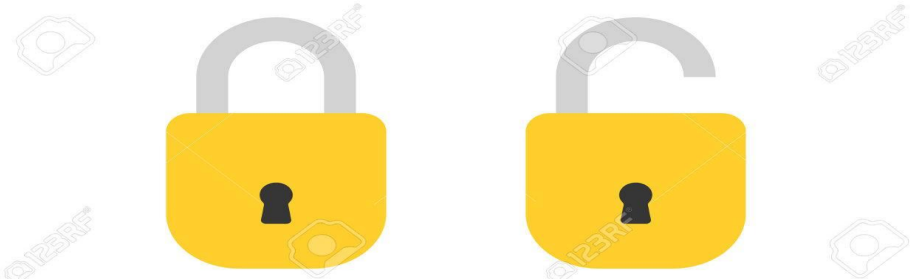


Two-phase locking

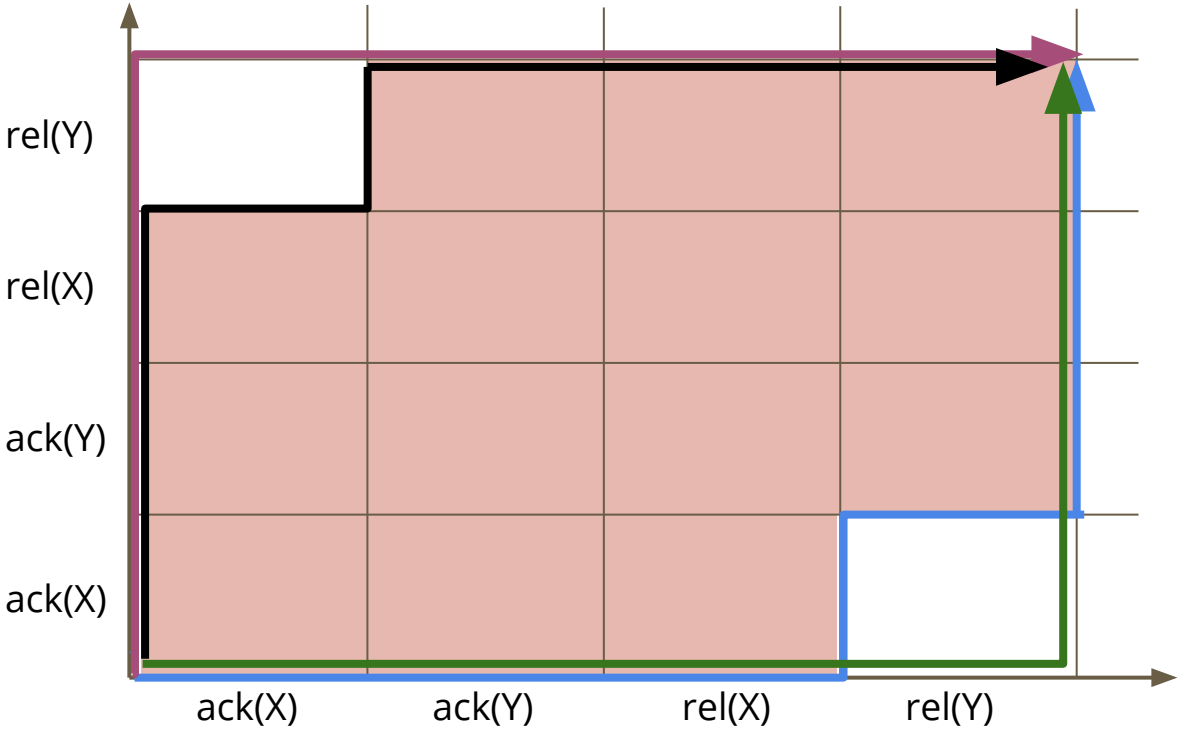
Ahora supongamos que el orden en que deben tomar los palillos es el mismo,

T1:ack(X); ack(Y); rel(X); rel(Y)
T2:ack(X); ack(Y); rel(X); rel(Y)

En este caso no habrá un bloqueo mutuo y lo podemos observar con el siguiente plano.



Two-phase locking



Verificación de Two-phase locking

La indistinguibilidad nos ayuda a verificar el two phase-locking, ya que todas las ejecuciones concurrentes que son indistinguibles a una ejecución secuencial cumplen las mismas propiedades.

Por lo tanto si la ejecución secuencial cumple el protocolo de two-phase locking, las ejecuciones indistinguibles también cumplen con el protocolo.

Verificación de Two-phase locking

Teorema: Si la ejecución de cada proceso satisface el protocolo cuando no se intercalan los procesos, entonces cualquier ejecución en que se intercalan los procesos también satisfacen el protocolo de *locking.*