**Control de Concurrencia**

Profesor Heider Sanchez

**P1. Detección de problemas de concurrencia:**

Analizar los siguientes planes de transacciones y deducir que problemas de concurrencia pueden ocurrir: actualización perdida, dependencia no confirmada (lectura sucia) y lectura no repetible. Mostrar los valores del recurso compartido X en cada instante de tiempo. Resalte claramente en donde se producen los conflictos.

|  |  |  |
| --- | --- | --- |
| **T1** | **T2** | **X=100** |
| READ(X)  SHOW(X)  READ(X)  SHOW(X) | READ(X)  X= X + 1  WRITE(X) | X=100 X=100 X=100 X=100  X=101  X=101 X=101 |

* **Existe lectura no repetible (Primero cuando leemos tenemos 100 y en la siguiente lectura tenemos 101)**
* **No es critico**

|  |  |  |
| --- | --- | --- |
| **T1 (N=5)** | **T2 (M=3)** | **X=100** |
| READ(X)  X= X - N  WRITE(X)  READ(Y)  Y=Y+N  WRITE(Y) | READ(X)  X= X + M  WRITE(X) | X=100  X= 100 95  X=100  X=100 103  X=95  X=95  X=103  X=103  X=103 |

* **Tenemos una lectura sucia ya que hemos perdido la actualización de la Transacción 1.**

|  |  |  |
| --- | --- | --- |
| **T1 (N=5)** | **T2 (M=3)** | **X=100** |
| READ(X)  X= X - N  WRITE(X)  READ(Y)  ROLLBACK | READ(X)  X= X + M  WRITE(X) | X=100  X= 100 95  X= 95  X= 95  X= 100 98  X= 100  X = 98  X= 100 |

**P2. Detección de problemas de concurrencia:**

Analice el siguiente plan de ejecución de transacciones, ¿Qué problema se presenta si se tiene que mantener la **restricción A=B** al finalizar la ejecución del plan? ¿Qué modificación haría a dicho plan concurrente para garantizar la restricción?

|  |  |  |
| --- | --- | --- |
| **T1** | **T2** | **A=100, B=100** |
| READ(A)  A = A + 1  WRITE(A)  READ(B)  B = B + 1  WRITE(B) | READ(A)  A = 2 \* A  WRITE(A)  READ(B)  B = 2 \* B  WRITE(B) | A=100  A= 100 101  A= 101  A= 101  A= 101 202  A= 202  B= 100  B= 100 200  B= 200  B= 200  B=200  B=201 |

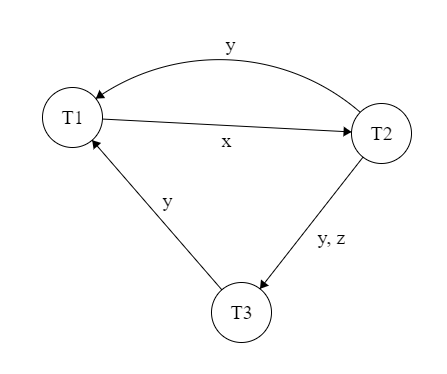
|  |  |  |
| --- | --- | --- |
| **T1** | **T2** | **A=100, B=100** |
| READ(A)  A = A + 1  WRITE(A)  READ(B)  B = B + 1  WRITE(B) | READ(A)  A = 2 \* A  WRITE(A)  READ(B)  B = 2 \* B  WRITE(B) | A=100  A= 100 101  A= 101  A= 101  A= 101 202  A= 202  B= 100  B= 100 101  B= 101  B= 101  B= 101 202  B=202 |

**P3. Grafo de Precedencia:**

Dada los siguientes planes de transacciones indique usted si corresponde a una planificación serializable por conflictos usando el grafo de precedencia. Caso de no ser serializable, permute las instrucciones para obtener un plan serializable (si es factible) y muestre el plan secuencial equivalente.

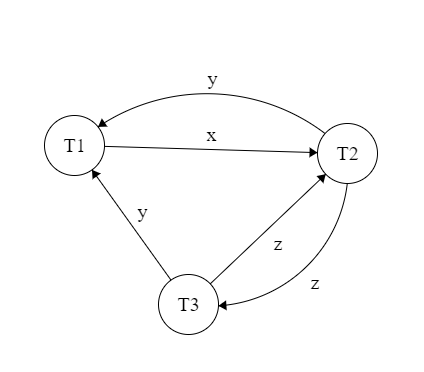
|  |  |  |
| --- | --- | --- |
| **T1** | **T2** | **T3** |
| READ (x)  WRITE (x)  READ (y)  WRITE (y) | READ(z)  READ(y)  WRITE(y)  READ (x)  WRITE (x) | READ (y)  READ (z)  WRITE (y)  WRITE (z) |

* **GRAFICO**



|  |  |  |
| --- | --- | --- |
| **T1** | **T2** | **T3** |
| READ (x)  WRITE (x)  READ (y)  WRITE (y) | READ(z)  READ(y)  WRITE(y)  WRITE (z)  READ (x)  WRITE (x) | READ (y)  READ (z)  WRITE (y) |

* **GRAFICO**



**P4. Protocolos de Bloqueos:**

Aplicando protocolos de bloqueo, rediseñe los siguientes planes (sin variar el orden cronológico) para garantizar una planificación serializable.

|  |  |
| --- | --- |
| **T1** | **T2** |
| READ(X)  X= X - N  WRITE(X)  READ(Y)  Y=Y+N  WRITE(Y) | READ(X)  X= X + M  WRITE(X) |

|  |  |
| --- | --- |
| **T1** | **T2** |
| READ(X)  X= X - 10  WRITE(X)  READ(Y)  Y=Y+10  WRITE(Y) | READ(Y)  Y = Y - 20  WRITE(Y)  READ(C)  C = C + 20  WRITE(C) |

**P5: Bloqueo en dos fases**

Considere el protocolo de bloque en dos fases e indique como dicho mecanismo de control podría manejar la siguiente secuencia de instrucciones:

T1:R(X), T2:W(X) T2:W(Y), T3:W(Y), T1:W(Y) T1: Commit,T2:Commit, T3:Commit

**P6.** Diseñe el algoritmo para verificar si una planificación es serializable por conflicto. Considere los tres pasos generales (slide 44).

**P7.** Diseñe el algoritmo de bloqueo para el **protocolo de actualización** (slide 65 -67).

**P7.** Diseñe el algoritmo de desbloqueo considerando el **protocolo de actualización** (slide 65 -67).