Esercizio1

Descrivere la ripresa a caldo, indicando la costituzione progressiva degli insiemi di UNDO e REDO

e le azioni di recovery, a fronte del seguente log:

DUMP, B(T1), B(T2), B(T3), I(T1, O1, A1), D(T2, O2, B2), B(T4), U(T4, O3, B3, A3),

U(T1, O4, B4, A4), C(T2), CK(T1, T3, T4), B(T5), B(T6), U(T5, O5, B5, A5), A(T3), CK(T1, T4, T5, T6),

B(T7), A(T4), U(T7, O6, B6, A6), U(T6, O3, B7, A7), B(T8), A(T7), guasto

Soluzione:

1) Per prima cosa bisogna percorrere il log a ritroso fino al più recente record di check-point:

CK(T1,T4,T5,T6)

Si costruiscono gli insiemi di UNDO e di REDO:

UNDO= { T1, T4, T5, T6 } REDO={}

2) Il log viene percorso in avanti, aggiornando i due insiemi:

B(T7) UNDO= { T1, T4, T5, T6, T7 } REDO={}

A(T4) UNDO= { T1, T4, T5, T6, T7 } REDO={}

B(T8) UNDO= { T1, T4, T5, T6, T7, T8 } REDO={}

A(T7) UNDO= { T1, T4, T5, T6, T7 } REDO={}

3) Il log viene ripercorso ancora a ritroso, fino all’operazione I(T1,O1,A1), eseguendo le seguenti

operazioni:

O3=B7

O6=B6

O5=B5

O4=B4

O3=B3

Delete O1

4) Il log viene ripercorso in avanti per rieseguire le operazioni di REDO, ma essendo vuoto questo

insieme, nessuna operazione verrà eseguita.

Esercizio 2

Si supponga che nella situazione precedente si verifichi un guasto di dispositivo che coinvolge gli

oggetti O1, O2, O3; descrivere la ripresa a freddo.

Soluzione:

La ripresa a freddo è articolata in tre fasi successive.

1. Il log viene percorso a ritroso fino al primo record DUMP e si ricopia selettivamente la parte

deteriorata della base dati.

2. Si ripercorre in avanti il log, applicando relativamente alla parte deteriorata della base di dati sia

le azioni sulla base di dati sia le azioni di commit o abort e riportandosi così nella situazione

precedente al guasto.

Insert O1=A1

Delete O2

O3=A3

Commit (T2)

Abort (T4)

O3=A7

3. Si svolge una ripresa a caldo.

Esercizio 3

r1(x), r4(x), w4(x), r1(y), r4(z), w4(z), w3(y), w3(z), w1(t), w2(z), w2(t)

Per classificare questo schedule si deve realizzare un grafo dei conflitti.

Consideriamo le operazioni relative a ogni risorsa separatamente:

T: w1, w2

X: r1, r4, w4

Y: r1, w3

Z: r4, w4, w3, w2

Visto che il grafo è aciclico è CSR, quindi anche VSR.

Lo schedule seriale equivalente è:

S: r1(x), r1(y), w1(t), r4(x), w4(x), r4(z), w4(z), w3(y), w3(z), w2(z), w2(t)

Esercizio 4

Se gli schedule seguenti si presentassero a uno scheduler che usa il locking a due

fasi, quali transazioni verrebbero messe in attesa? Si noti che, una volta posta in attesa una

transazione, le sue successive azioni non vanno più considerate.

Soluzione:

1. r1(x), w1(x), r2(z), r1(y), w1(y), r2(x), w2(x), w2(z)

Nessuna transazione in attesa

1. r1(x), w1(x), w3(x), r2(y), r3(y), w3(y), w1(y), r2(x)

Le transazioni 3 e 1 e poi 2 vengono messe in attesa, producendo una situazione di deadlock;

infatti, l’azione r2(x) deve aspettare l’oggetto x, messo in lock dalla transazione 1, e la

transazione 1 è in attesa su w1(y) dell’oggetto y, messo in lock dalla transazione 2.

3) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), w3(y), w3(x), w1(y), w5(x), w1(z), w5(y), r5(z)

Le transazioni 2, 1, 3 e 5 sono messe in attesa a causa del lock in lettura su x. Si crea una

situazione di deadlock dovuta alla richiesta di lock in scrittura su x da parte delle transazioni 1 e

2, entrambe con un lock in lettura sulla stessa risorsa x.

4) r1(x), r3(y), w1(y), w4(x), w1(t), w5(x), r2(z), r3(z), w2(z), w5(z), r4(t), r5(t)

Le transazioni 1, 4, 5 e 2 sono in attesa. La transazione 1 deve aspettare per y (allocato da t3),

le transazioni 4 e 5 devono aspettare per x (allocato da t1) e la transazione 2 deve aspettare per z

(allocato da t3).

5) r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), r3(y), r3(x), w1(y), w5(x), w1(z), r5(y), r5(z)

Le transazioni 2, 1 e 5 sono in attesa. Devono aspettare per x (allocata da t1, t2 e t3)

6) r1(x), r1(t), r3(z), r4(z), w2(z), r4(x), r3(x), w4(x), w4(y), w3(y), w1(y), w2(t)

Le transazioni 2, 3 e 4 vengono messe in attesa. t2 e t4 devono aspettare per z (allocata da t3) e

t3 deve aspettare per y (allocata da t1)

7) r1(x), r4(x), w4(x), r1(y), r4(z), w4(z), w3(y), w3(z), w1(t), w2(z), w2(t)

Assumendo che t1 rilasci il lock esclusivo su t al termine delle sue operazioni, le transazioni 3 e

4 sono in attesa. Devono aspettare per x e y, allocate da t1.

Esercizio 5

Se gli schedule seguenti si presentassero a uno scheduler basato su timestamp, quali transazioni verrebbero abortite?

Soluzione:

r1(x), w1(x), r2(z), r1(y), w1(y), r2(x), w2(x), w2(z)

Operazione Risposta Nuovo Valore

read(x,1) Ok RTM(x)=1

write(x,1) Ok WTM(x)=1

read(z,2) Ok RTM(z)=2

read(y,1) Ok RTM(y)=1

write(y,1) Ok WTM(y)=1

read(x,2) Ok RTM(x)=2

write(z,2) Ok WTM(z)=2

Non viene abortita nessuna transazione.

r1(x), w1(x), w3(x), r2(y), r3(y), w3(y), w1(y), r2(x)

Operazione Risposta Nuovo Valore

read(x,1) Ok RTM(x)=1

write(x,1) Ok WTM(x)=1

write(x,3) Ok WTM(x)=3

read(y,2) Ok RTM(y)=2

read(y,3) Ok RTM(y)=3

write(y,3) Ok WTM(y)=3

write(y,1) t1 aborted

read(x,2) t2 aborted