

Esercitazione Schedule

Indicare se i seguenti schedule possono produrre anomalie; i simboli *ci* e *ai* indicano l'esito (commit o abort) della transazione.

1. $r_1(x), w_1(x), r_2(x), w_2(y), a_1, c_2$
2. $r_1(x), w_1(x), r_2(y), w_2(y), a_1, c_2$
3. $r_1(x), r_2(x), r_2(y), w_2(y), r_1(z), a_1, c_2$
4. $r_1(x), r_2(x), w_2(x), w_1(x), c_1, c_2$
5. $r_1(x), r_2(x), w_2(x), r_1(y), c_1, c_2$
6. $r_1(x), w_1(x), r_2(x), w_2(x), c_1, c_2$

- Si considerino i seguenti schedule:

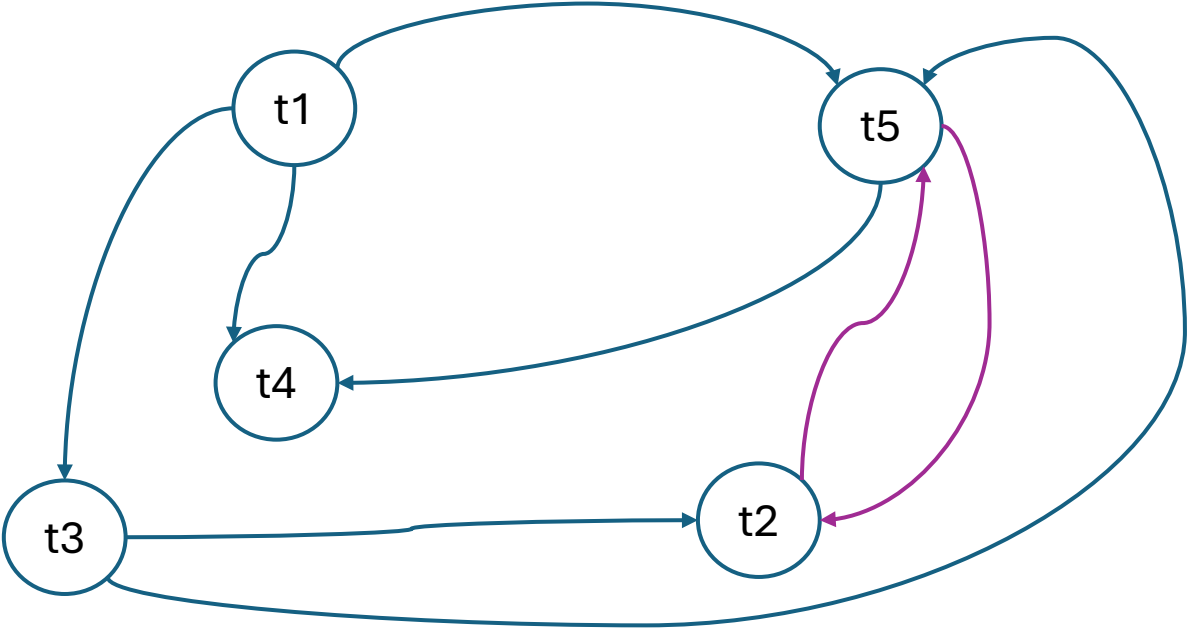
- $r3(y), w1(y), r1(x), w5(x), r2(z), w1(t), r3(z), w4(x), r5(t), w5(z), r4(t), w2(z)$
- $r2(y), w2(y), r3(z), r1(x), w1(x), r3(y), w3(y), r2(z), w3(z), r2(x), w2(x), r1(y), w1(y)$.

Stabilire se sono VSR o CSR

Se passati ad un 2PL producono deadlock?

$r_3(y), w_1(y), r_1(x), w_5(x), r_2(z), w_1(t), r_3(z), w_4(x), r_5(t), w_5(z), r_4(t), w_2(z)$

x	r1 w5 w4
y	r3 w1
z	r2 r3 w5 w2
t	w1 r5 r4



NO CSR!

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

$w2(z), w4(x), w1(t), w1(y)$ sono scritture finali

$r5(t)$ legge da $w1(t)$

$r4(t)$ legge da $w1(t)$

$t5 < t2$

$t5 < t4$

$t1 < t5 < t4$

$t3 < t1$

$t3 < t2$

$t1: w1(y), r1(x), w1(t)$

$t2: r2(z), w2(z)$

$t3: r3(y), r3(z)$

$t4: w4(x), r4(t)$

$t5: w5(x), r5(t), w5(z)$

$t3 \ t1 \ t5 \ t4 \ t2$

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

$r5(t)$ legge da $w1(t)$

$r4(t)$ legge da $w1(t)$

$r2(z)$ legge da $w5(z)$

$t3 \ t1 \ t5 \ t2 \ t4$

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

$r5(t)$ legge da $w1(t)$

$r4(t)$ legge da $w1(t)$

$r2(z)$ legge da $w5(z)$

NON è VSR

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

	R_lock	W_lock
x	t1	t5 t4
y	t3	t1
z	t2 t3	t2 t5
t	t5 t4	t1

La transazione 1 è in attesa della transazione 3 per un write lock su y

La transazione 3 termina e rilascia tutti i suoi lock e la t1 può essere risvegliata

La transazione 1 riprende e viene bloccata sulla lettura dell'oggetto x perché c'è un w_lock di t5

La transazione 1 è in attesa della transazione 5 per un write lock su x

La transazione 4 è in attesa della transazione 5 per un write lock su x

La transazione 5 è in attesa della transazione 2 per un write lock su z

La transazione 2 chiede ed ottiene una lock escalation su z e quindi completa la scrittura e rilascia i suoi lock

L'oggetto z non ha più lock e quindi la transazione 5 può essere risvegliata.

La transazione 5 completa e rilascia i suoi lock

Le transazioni 1 e 4 sono risvegliate

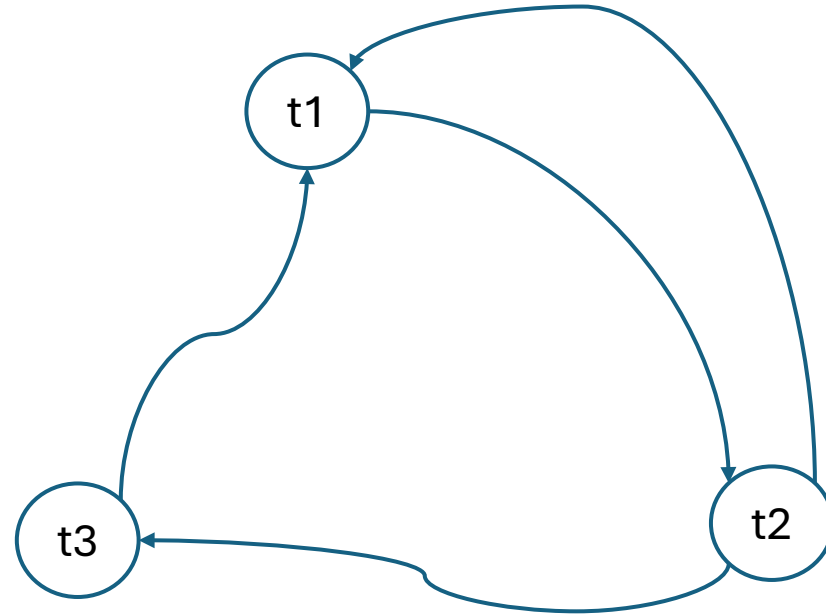
La transazione 1 completa e rilascia i suoi lock

La transazione 4 completa e rilascia i suoi lock

Non si creano deadlock!

$r_2(y), w_2(y), r_3(z), r_1(x), w_1(x), r_3(y), w_3(y), r_2(z), w_3(z), r_2(x), w_2(x), r_1(y), w_1(y)$

x	r1w1r2w2
y	r2w2r3w3r1w1
z	r3r2w3



$r_2(y), w_2(y), r_3(z), r_1(x), w_1(x), r_3(y), w_3(y), r_2(z), w_3(z), r_2(x), w_2(x), r_1(y), w_1(y)$

scritture finali: $w_3(z), w_2(x), w_1(y)$

$r_3(y)$ legge da $w_2(y)$

$r_2(x)$ legge da $w_1(x)$

$r_1(x)$ legge da $w_3(y)$

$t_1: r_1(x), w_1(x), r_1(y), w_1(y)$

$t_2: r_2(y), w_2(y), r_2(z), r_2(x), w_2(x),$

$t_3: r_3(z), r_3(y), w_3(y), w_3(z),$

$t_1 < t_2$

$t_3 < t_1$

$t_2 < t_1$

Dato che t_2 deve precedere t_1 affinché $w_1(y)$ sia scrittura finale, e che t_2 deve anche seguire t_1 affinché $w_2(x)$ sia scrittura finale, possiamo concludere che lo schedule non può essere VSR in quanto nessuno schedule seriale può soddisfare le condizioni viste prima.

$r_2(y), w_2(y), r_3(z), r_1(x), w_1(x), r_3(y), w_3(y), r_2(z), w_3(z), r_2(x), w_2(x), r_1(y), w_1(y)$

A
A

A
A
A

	R_lock	W_lock
x	t1	t1
y	t2	t2
z	t3 t2	

T2 effettua lock escalation su y

T1 effettua lock escalation su x

T3 vuole leggere y ma non può ottenere r_lock perché y ha un w_lock di t2, quindi T3 in attesa di T2

T2 vuole leggere x ma non può ottenere r_lock perché x ha un w_lock di t1, quindi T2 in attesa di T1

T1 vuole leggere y ma non può ottenere r_lock perché y ha un w_lock di t2, quindi T1 in attesa di T2

Abbiamo un dead lock perché T2 attende una risorsa da T1 e T1 attende una risorsa da T2

Classificare i seguenti schedule (come: NonSR, VSR, CSR); nel caso uno schedule sia VSR oppure CSR, indicare tutti gli schedule seriali e esso equivalenti.

1. $r1(x), w1(x), r2(z), r1(y), w1(y), r2(x), w2(x), w2(z)$
2. $r1(x), w1(x), w3(x), r2(y), r3(y), w3(y), w1(y), r2(x)$
3. $r1(x), r2(x), w2(x), r2(x), r4(z), w1(x), w3(y), w3(x), w1(y), w5(x), w1(z), w5(y), r5(z)$
4. $r1(x), r3(y), w1(y), w4(x), w1(t), w5(x), r2(z), r3(z), w2(z), w5(z), r4(t), r5(t)$
5. $r1(x), r2(x), w2(x), r3(x), r4(z), w1(x), r3(y), r3(x), w1(y), w5(x), w1(z), r5(y), r5(z)$
6. $r1(x), r1(t), r3(z), r4(z), w2(z), r4(x), r3(x), w4(x), w4(y), w3(y), w1(y), w2(t)$
7. $r2(x), r4(x), w4(x), r1(y), r4(z), w4(z), w3(y), w3(z), w1(t), w2(z), w2(t)$