Feladat 1.

Adjuk meg az X adatbáziselem lehetséges értékeit a tranzakciók lefutása után, feltéve, hogy a tranzakciók ütemezése soros és az X kezdeti értéke 100.

T1: READ(X, t); t := t+100; WRITE(X, t)

T2: READ(X, t); t := t*2; WRITE(X, t)

T3: READ(X, t); t := t+10; WRITE(X, t)

Hányféle soros ütemezése van a fenti 3 tranzakciónak?

Megoldás:

6 féle soros ütemezés lehetséges: 3! = 3 * 2 * 1 = 6. A lehetséges ütemezések: 123, 132, 213, 231, 312, 321 (az egyszerűség kedvéért csak számokkal jelölve a tranzakciókat)

Az X lehetséges értékei különböző soros ütemezéseket tekintve: 310, 320, 410, 420.

Feladat 2.

Szomszédos műveletek cseréjével készítsünk az alábbi ütemezésből soros ütemezést:

R1(A); W1(A); R2(A); W2(A); R1(B); W1(B); R2(B); W2(B);

Megoldás:

R1(A); W1(A); R2(A); **W2(A); R1(B);** W1(B); R2(B); W2(B);

R1(A); W1(A); **R2(A); R1(B);** W2(A); W1(B); R2(B); W2(B);

R1(A); W1(A); R1(B); R2(A); **W2(A); W1(B);** R2(B); W2(B);

R1(A); W1(A); R1(B); **R2(A); W1(B);** W2(A); R2(B); W2(B);

R1(A); W1(A); R1(B); W1(B); R2(A); W2(A); R2(B); W2(B);

A végső állapotban kapott ütemezés a T1, T2 soros ütemezés.

Feladat 3.

Adjuk meg a konfliktus-sorbarendezhető ütemezések számát az alábbi két tranzakcióra:

T1: R1(A); W1(A); R1(B); W1(B)
T2: R2(A); W2(A); R2(B); W2(B)

Megoldás:

A konfliktus-sorbarendezhető utemezések megkaphatóak, ha kiindulunk egy soros ütemezésből és konfliktus mentes cseréket végzünk. Minden különböző ütemezés egy konfliktus-sorbarendezhető ütemezés lesz.

Egyik kiinduló állapot: T1, T2

R1(A); W1(A); R1(B); W1(B); R2(A); W2(A); R2(B); W2(B)

Vegyük észre, hogy az W1(B) művelet nem vihető semmiképpen ez R2(B) jobb oldalára és az R2(A) művelet sem a W1(A) bal oldalára. Így négy műveletnyi hely van, ahol cseréket tudunk végezni:

R1(A); W1(A); | R1(B); W1(B); R2(A); W2(A); | R2(B); W2(B)

Mivel egy tranzakción belül a műveletek sorrendje kötött, ha a 4 helyből kiválasztunk kettőt, akkor oda elhelyezhetjük az egyik tranzakció két műveletét, a maradék két helyre pedig a megmaradt két műveletet egyértelműen el tudjuk helyezni. Így, a $\binom{4}{2}$ megadja a lehetséges kombinációk számát, ami 6.

Másik kiinduló állapot: T2, T1

Itt ugyanúgy lehet eljárni, mint a korábbi esetben, ugyanúgy 6 kombinációt kapunk

Következtetés: Mivel mindkét irányban 6 kombinációt kapunk, de mindkét irányban ebből az egyik a kiinduló állapot, ezért 5+5=10 az összes lehetséges konfliktus-sorbarendezhető ütemezés.

Feladat 4.

Rajzoljuk fel az alábbi ütemezések megelőzési gráfjait és adjuk meg a csúcsok topologikus sorrendjeit:

S1: R1(A); R2(B); W2(A); R2(B); R3(A); W1(B); W3(A); W2(B);

S2: R1(A); R2(A); R3(B); W1(A); W2(C); R2(B); W2(B); W1(C);

S3: R1(A); R2(A); W1(B); W2(B); R1(B); R2(B); W2(C); W1(D);

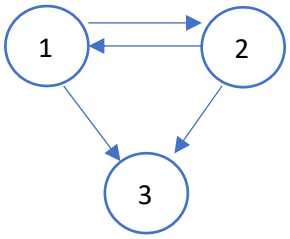
S4: R1(A); R2(A); R1(B); R2(B); R3(A); R4(B); W1(A); W2(B);

S5: R1(A); R2(A); R1(C); R2(B); R3(A); R4(B); W1(A); W2(B);

Megoldások:

S1)

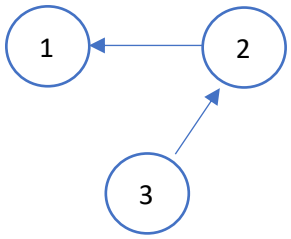
T1	T2	T3
R1(A)		
	R2(B)	
	W2(A)	
	R2(B)	
		R3(A)
W1(B)		
		W3(A)
	W2(B)	



Mivel a megelőzési gráfban van kör (1 <-> 2) a csúcsoknak nincs topologikus sorrendje, azaz az ütemezés nem konfliktus-sorbarendezhető.

S2)

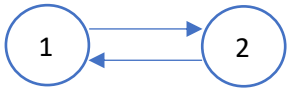
T1	T2	T3
R1(A)		
	R2(A)	
		R3(B)
W1(A)		
	W2(C)	
	R2(B)	
	W2(B)	
W1(C)		



Egy topologikus sorrendje van a csúcsoknak: T3, T2, T1. Az ütemezés konfliktus-sorbarendezhető.

S3)

T1	**T2**
R1(A)	
	R2(A)
W1(B)	
	W2(B)
R1(B)	
	R2(B)
	W2(C)
W1(D)	



A gráfban van kör, így nincs topologikus sorrend és az ütemezés nem konfliktus-sorbarendezhető.

Feladat 5.

Adott két tranzakció:

T1: L1(A); R1(A); W1(A); L1(B); R1(B); W1(B); U1(A); U1(B);

T2: L2(B); R2(B); W2(B); L2(A); R2(A); W2(A); U2(B); U2(A);

Kétfázisú-e a fenti két tranzakció?

Adjunk példát egy olyan tranzakcióra, amely nem kétfázisú.

Megoldások:

A T1 és a T2 is kétfázisú, mivel az összes LOCK művelete megelőzi az összes UNLOCK műveletet.

Példa nem kétfázisú tranzakcióra: L1(A); R1(A); W1(A); U1(A); L1(B); R1(B); W1(B); U1(B);

Feladat 6.

Adjuk meg az alábbi ütemezésekre, hogy jogszerűek-e, valamint a bennük szereplő tranzakciókról, hogy melyek konzisztensek, illetve kétfázisúak.

a) L1(A); W1(A); L1(B); U1(A); L2(A); W2(A); U2(A); R1(B); L2(C); W2(C); U2(C); U1(B)

b) L1(A); W1(A); U1(A); L2(A); W2(A); L1(B); U2(A); R1(B); L2(C); W2(C); U2(C); U1(B)

c) L1(A); W1(A); L2(A); W2(A); L1(B); U1(a); U2(A); R1(B); L2(C); W2(C); U2(C); U1(B)

Megoldások:

a)

Az ütemezésben két tranzakció van:

T1: L1(A); W1(A); L1(B); U1(A); R1(B); U1(B)

T2: L2(A); W2(A); U2(A); L2(C); W2(C); U2(C);

A T1 és a T2 tranzakció is konzisztens mivel olvasás/írás előtt zárolja az adatbáziselemet, utána pedig feloldja azokat. A T1 kétfázisú, a T2 viszont nem. Az ütemezés jogszerű, mivel egy adatbáziselemet egyszerre csak egy tranzakció zárol.

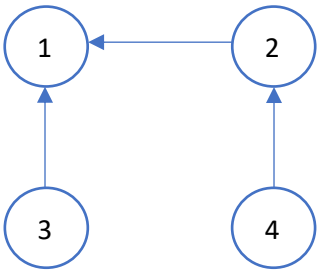
Feladat 7.

Rajzoljuk fel a következő ütemezéshez tartozó várakozási gráfot a 7., és 9. lépés után. Tegyük fel, hogy egy felszabaduló zárat az a várakozó fog megkapni, aki a legrégebben vár.

L1(A); L2(B); L3(C); L1(D); L2(A); L3(D); L4(B); U1(A); L2(C); ...

Megoldás:

Gráf a 7. lépés után:



Gráf a 9. lépés után: