ELTE | IK

INFORMATIKAI KAR

Adatbázisok 2. GY.

Ütemezések és sorba rendezhetőség

Definíció: ütemezés

- Az *ütemezés* (schedule) egy vagy több tranzakció által végrehajtott lényeges műveletek időrendben vett sorozata.
- Azt mondjuk, hogy egy ütemezés *soros* (serial schedule), ha úgy épül fel a tranzakciós műveletekből, hogy először az egyik tranzakció összes műveletét tartalmazza, majd azután egy másik tranzakció összes műveletét stb., miközben nem cseréli fel a műveleteket.

Példa: egy soros ütemezés

T1	T2	A	B
READ(A, t)		25	25
t = t*2			
WRITE(A, t)		50	
READ(B, t)			
T = t*2			
WRITE(B, t)			50
	READ(A, s)		
	s = s+100		
	WRITE(A, s)	150	
	READ(B, s)		
	S = s+100		
	WRITE(B, s)		150

Feladatok soros ütemezésre

- 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?

Definíció: sorba rendezhetőség

- Egy ütemezés *sorba rendezhető* (serializable schedule), ha ugyanolyan hatással van az adatbázis állapotára, mint valamelyik soros ütemezés, függetlenül attól, hogy mi volt az adatbázis kezdeti állapota.

Példa: egy sorba rendezhető ütemezés

T1	T2	A	B
READ(A, t)		25	25
t = t*2			
WRITE(A, t)		50	
	READ(A, s)		
	s = s+100		
	WRITE(A, s)	150	
READ(B, t)			
T = t*2			
WRITE(B, t)			50
	READ(B, s)		
	S = s+100		
	WRITE(B, s)		150

Példa: nem sorba rendezhető ütemezés

T1	T2	A	B
READ(A, t)		25	25
t = t*2			
WRITE(A, t)		50	
	READ(A, s)		
	s = s+100		
	WRITE(A, s)	150	
	READ(B, s)		
	S = s+100		
	WRITE(B, s)		125
READ(B, t)			
T = t*2			
WRITE(B, t)			250

Jelölések

- Egyszerűsítsük a jelölést.
- Mivel számunkra csak az olvasás és írás műveletek a fontosak, ezért használjunk egyszerűbb jelöléseket a tranzakciók műveleteire:
  - R1(A) – a T1 tranzakció olvassa az A adatbáziselemet.
  - W1(A) – a T1 tranzakció írja az A adatbáziselemet.

Példa: tranzakciók felírása

- T1: R1(A); W1(A); R1(B); W1(B)

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

- A T1 és T2 tranzakciók egy ütemezése:

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

Definíció: konfliktus

- A *konfliktus* (conflict): olyan egymást követő műveletpár az ütemezésben, amelynek ha a sorrendjét felcseréljük, akkor legalább az egyik tranzakció viselkedése megváltozhat.
- Egy műveletpár akkor konfliktusos, ha:
  - Ugyanannak a tranzakciónak két művelete. Pl: *R1*(*X*) és W1(*Y*) konfliktus.
  - Ha két művelet ugyanazt az adatbáziselemet használja és legalább az egyik művelet írás. Pl: W1(A) és R2(A) konfliktus.

Következtetés

- Különböző tranzakciók bármely két műveletének sorrendje felcserélhető, hacsak nem
  - Ugyanarra az adatbáziselemre vonatkoznak, és
  - Legalább az egyik művelet írás.

Feladat: konfliktus sorba rendezhetőség

- 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);

Definíciók

- Azt mondjuk, hogy két ütemezés *konfliktus ekvivalens* (conflict-equivalent), ha szomszédos műveletek nem konfliktusos cseréinek sorozatával az egyiket átalakíthatjuk a másikká.
- Azt mondjuk, hogy egy ütemezés *konfliktus sorbarendezhető* (conflict-serializable schedule), ha konfliktus ekvivalens valamely soros ütemezéssel.

Feladat: konfliktus-sorbarendezhetőség

- 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)

Definíció: megelőzés

- Adott a T1 és T2 tranzakcióknak, esetleg további tranzakcióknak, egy S ütemezése. Azt mondjuk, hogy T1 megelőzi T2-t, és T1 <S T2-vel jelöljük, ha van a T1-ben olyan A1 művelet, és a T2-ben olyan A2, hogy
  - A1 megelőzi A2-t az S-ben,
  - A1 és A2 ugyanarra az adatbáziselemre vonatkoznak, és
  - A1 és A2 közül legalább az egyik írás művelet.

Definíció: megelőzési gráf

- Ezeket a megelőzéseket a *megelőzési gráfban* (precedence graph) összegezhetjük. A megelőzési gráf csomópontjai az *S* ütemezés tranzakciói. Ha a tranzakciókat *Ti*-vel jelöljük az *i* függvényében, akkor a *Ti*-nek megfelelő csomópontot az *i* egésszel jelöljük. Az *i* csomópontból a *j* csomópontba vezet irányított él, ha *Ti* <*S Tj*.

Példa: megelőzési gráf

- S1: r2(A); r1(B); w2(A); r3(A); w1(B); w3(A); r2(B); w2(B);

T1	T2	T3
	r2(A)	
r1(B)		
	w2(A)	
		r3(A)
w1(B)		
		w3(A)
	r2(B)	
	w2(B)	

Feladat: megelőzési gráf

- 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);

Sorba rendezhetőség

- Van egy egyszerű szabály, amivel megmondhatjuk, hogy egy *S* ütemezés konfliktus sorba rendezhető-e.
- Rajzoljuk fel az *S* megelőzési gráfját, és nézzük meg tartalmaz-e kört!
- Ha igen, akkor *S* nem konfliktus sorba rendezhető. Ha pedig a gráf körmentes, akkor *S* konfliktus sorba rendezhető, továbbá a csomópontok bármelyik topologikus sorrendje megadja a konfliktus ekvivalens soros sorrendet.

Zárolások

- Két új műveletet vezetünk be:
  - L1(A) – a T1 tranzakció zárolja az A adatbáziselemet (lock)
  - U1(A) – a T1 tranzakció feloldja az A adatbáziselemet (unlock)

Definíciók

- Jogszerűség
  - Nem zárolhatja két tranzakció ugyanazt az elemet, csak úgy, ha az egyik már feloldotta a zárat
- Konzisztencia:
  - Egy tranzakció csak akkor írhat vagy olvashat egy elemet ha már zárolta, de még nem oldotta fel azt.
  - Ha egy tranzakció zárol egy elemet, akkor később fel kell oldania.
- 2PL (kétfázisú zárolás – two phase locking)
  - A tranzakciókban az összes zárolási művelet megelőzi az összes feloldási műveletet.

Feladatok

- 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.

Feladatok

- 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)

Definíció: várakozási gráf

- A várakozási gráf csúcsai tranzakciók.
- Akkor van él az „i” tranzakcióhoz tartozó csúcsból a „j” tranzakcióhoz tartozó csúcsba, ha az „i” tranzakció vár egy olyan zár feloldására, amelyet „j” tranzakció tart éppen.
- A várakozási gráf a zárolások és feloldások során folyamatosan változik.
- Ha a várakozási gráfban van kör, akkor holtpont van.

Feladatok

- Rajzoljuk fel a következõ 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); ...