AG Datenbanken und Informationssysteme

Wintersemester 2006 / 2007

Prof. Dr.-Ing. Dr. h. c. Theo Härder
Fachbereich Informatik
Technische Universität Kaiserslautern

http://wwwdvs.informatik.uni-kl.de

10. Übungsblatt

Für die Übung am Donnerstag, **18. Januar 2007**,
von 15:30 bis 17:00 Uhr in 13/222.

Aufgabe 1: Parallele Transaktionen auf unterschiedlichen Konsistenzebenen

In einem DBS werden hierarchische Sperrprotokolle für Datenbank, Segment, Relation und Tupel eingesetzt. Eine Relation R_{11}, die Konten beschreibt, bestehe aus folgenden 5 Tupeln mit den zugehörigen Kontenbeständen:

t_{111} mit 100 Euro
t_{112} mit 200 Euro
t_{113} mit 300 Euro
t_{114} mit 400 Euro
t_{115} mit 500 Euro

Transaktion T_1 überweist 100 Euro von t_{114} nach t_{111}, addiert 100 Euro auf t_{112} und zieht 100 Euro von t_{113} ab. Parallel dazu liest Transaktion T_2 sequentiell alle Kontenstände und bildet die Gesamtsumme.

a) T_1 läuft auf Konsistenzebene 3 und sperrt R_{11} exklusiv. T_2 läuft auf Konsistenzebene 1. Jeder Verarbeitungsschritt beträgt eine Zeiteinheit. Es werden nur die Aktionen mit dem Datenbanksystem als Verarbeitungsschritte berücksichtigt, dabei genügen die Operationen LOCK, UNLOCK, READ, MODIFY.

   (1) Welches Ergebnis erhält T_2, wenn sie zeitgleich mit T_1 startet?

   (2) Welches Ergebnis erhält T_2, wenn sie im 5. Verarbeitungsschritt, bzw. 8. Verarbeitungsschritt von T_1 startet?

b) T_1 und T_2 benutzen zur Erledigung der gleichen Aufgabe Tupelsperren. T_1 läuft wieder auf Konsistenzebene 3, T_2 diesmal auf Konsistenzebene 2. Kurze Lesesperren bedeuten, dass die Operationen LOCK, READ und UNLOCK (wenn möglich) direkt in drei aufeinander folgenden Verarbeitungsschritten durchgeführt werden.

   (1) Beide Transaktionen starten zur gleichen Zeit. Nach wie vielen Zeiteinheiten endet T_2? Welche Gesamtsumme wird ermittelt?

   (2) Was passiert, wenn T_2 ein Sperrprotokoll der Konsistenzebene 3 einhält?

Aufgabe 2: Escrow-Ansatz

Die Anzahl von freien Sitzplätzen in einem Flugzeug soll über ein Escrow-Feld mit LO=0 und HI=165 verwaltet werden. Eine Transaktion kann Plätze reservieren (negativer Wert in Klammern) oder zurückgeben (positiver Wert). Der Anfangszustand (INF, Q, SUP) sei (10, 10, 10), d.h. es sind noch 10 Plätze frei.

a) Bestimmen Sie für folgende Anforderungen/Rückgaben der Transaktionen T_1 bis T_5 das Werteintervall des Escrow-Feldes:
T_1(-3), T_2(+3), T_3(+2), T_4(-3), T_1.commit, T_3.rollback, T_5(-3), T_4.commit, T_2.commit, T_5.rollback

b) Wieviele Sitzplätze sind nach dem Commit von T_1 noch frei? Warum kann die Anzahl nicht exakt angegeben werden?

Aufgabe 3: Präzisionssperren

Gegeben sei folgendes DB-Schema:

Angestellte: PERS (PNR, NAME, GEHALT, BERUF, ANR, MNR, ORT)
Abteilung: ABT (ANR, ANAME, AORT)

PERS.ANR ist Fremdschlüssel auf ABT.ANR, PERS.MNR ist Fremdschlüssel auf PERS.MNR

Führen Sie die Prädikatliste und die Update-Liste, die beide zu Anfang leer sind, für die folgenden SQL-Anweisungen der Transaktionen T1-T5:

```
T1: SELECT * FROM PERS WHERE ORT="Kaiserslautern"
T2: SELECT * FROM PERS WHERE GEHALT >= 0
T3: INSERT INTO PERS VALUES (4711, "Maier", 45000, "Programmierer",
    P11, 0815, "Kaiserslautern")
T4: INSERT INTO ABT VALUES (P12, "Informationsintegration",
    "Karlsruhe")
T5: UPDATE PERS SET ORT="Kaiserslautern" WHERE PNR = 4711
T6: UPDATE PERS SET ORT="Karlsruhe" WHERE PNR = 4712
T7: SELECT * FROM ABT WHERE ANAME="Buchhaltung" OR AORT="Karlsruhe"
```

Welche Operationen werden ausgeführt, welche werden gesperrt?

Aufgabe 4: Synchronisation mit Mehrversionen-Verfahren

In einer Datenbank, die das in der Vorlesung vorgestellte Mehrversionen-Verfahren zur Synchronisation einsetzt, werden vier Schreibe- und fünf Lesetransaktionen nebenläufig ausgeführt, die auf die Datenbankobjekte A und B zugreifen. Alle Sperren werden bis zum Ende der Transaktion gehalten und dort atomar freigegeben (starkes 2PL). Die fünf Lesetransaktionen sind als mehrfache Ausführung des gleichen Transaktionsprogramms aufzufassen.

T_{x1}: BOT (t=0), [1], r(A), [4], w(A), [3], EOT

T_{x2}: BOT (t=2), [2], r(B), [5], w(B), [4], r(A), [2], w(A), [3], EOT

T_{x3}: BOT (t=3), [4], r(A), [2], w(A), [1], EOT

T_{x4}: BOT (t=14), [2], r(A), [2], w(A), [2], r(B), [2], w(B), [1], EOT

T_{r1}-T_{r5}: BOT, [6], r(A), [6], r(B), [6], EOT

Die Transaktionen T_{r1}-T_{r5} starten zu den Zeitpunkten t=0, t=9, t=12, t=19 bzw. t=26.

Hinweis: Die in eckigen Klammern angegebenen Zahlen geben die Verzögerung (durch interne Aktivitäten/Berechnungen der TA, Nutzerinteraktion etc.) zwischen der *tatsächlichen Ausführung* der

davor stehenden und der *versuchten Ausführung* der danach stehenden Operation an. Diese Verzögerung ist unabhängig von eventuellen Blockierungen/Wartezeiten der Transaktionen.

Synchronisieren Sie die Transaktionen mit dem in der Vorlesung vorgestellten Mehrversionen-Verfahren, d.h. ermitteln Sie den mit diesem Verfahren erzielten Schedule. Vermerken Sie dazu jede Änderung der vom System verwalteten Versionen der Objekte A und B. Welche Versionen der Objekte werden von den Transaktionen jeweils gelesen oder geschrieben?

Welche Serialisierbarkeitsreihenfolgen ergeben sich?

Aufgabe 5: Mehrversionen-Synchronisation

Eine Datenbank enthalte die Objekte a, b und c. Außerdem sei eine Integritätsbedingung definiert: a + b + c = konst. = 12 mit den anfänglichen Zahlenwerten a = 5, b = 4, c = 3. Die folgenden fünf Transaktionen sollen mit dem Mehrversionenverfahren synchronisiert werden:

T1: a := a - 1; b := b + 1; x1 := a + b + c;

T2: b := b - 1; c := c + 1; x2 := a + b + c;

T3: c := c - 1; a := a + 1; x3 := a + b + c;

T4: x4 := a + b + c;

T5: x5 := a + b + c;

Im unsynchronisierten Fall entstehe der folgende Ablaufplan für T1-T4. Wird dabei die Leseoperation nicht explizit für ein Datenobjekt angegeben, so erfolgt sie implizit beim ersten Zugriff.

	T1	**T2**	**T3**	**T4**
1	a - 1			
2		lesen(a)		lesen(c)
3			lesen(b)	
4	b + 1			
5		b - 1		
6			c - 1	
7	lesen(c)			lesen(b)
8		c + 1		
9			a + 1	
10	x1 := a + b + c			
11	Commit			
12				lesen(a)
13		x2 := a + b + c		x4 := a + b + c;
14		Commit		
15			x3 := a + b + c	
16			Commit	Commit

a) Welche Werte erhalten Sie für x1, x2 , x3 und x4, wenn die Leseoperationen mit R-Sperren synchronisiert werden?

b) Wie verändern sich diese Werte, wenn Sie die Leseoperationen wie Leser-Transaktionen (Leser) behandeln würden, d. h. die zum jeweiligen Transaktionsbeginn gültige Version der DB-Objekte lesen würden (ohne weitere Synchronisationsmaßnahmen)?

c) Welches Ergebnis erhält Leser T5 für x5 (und a, b, c), wenn T5 nach Commit von T1 bzw. Commit von T2 gestartet wird?

Aufgabe 6: Klassen von Historien (Wiederholung)

Ermitteln Sie für die folgenden Schedules, ob sie rücksetzbar sind (RC), kaskadierendes Rücksetzen vermeiden (ACA) oder strikt (ST) sind. Ermitteln Sie zusätzlich, ob die Historien serialisierbar oder seriell sind.

a) H1: $r_1(x)$, $w_1(x)$, $r_2(x)$, $r_2(y)$, $w_2(x)$, c_2, $r_1(z)$, c_1

b) H2: $w_1(x)$, $w_2(x)$, $w_2(y)$, c_2, $r_1(y)$, $w_1(y)$, c_1

c) H3: $r_1(x)$, $w_1(y)$, $r_2(y)$, $r_2(z)$, $w_1(x)$, $w_2(y)$, c_1, $w_2(z)$, c_2

d) H4: $r_2(z)$, $r_1(x)$, $r_1(y)$, $w_1(z)$, c_1, $w_2(z)$, c_2

e) H5: $w_2(x)$, $w_1(x)$, $w_1(z)$, $r_2(z)$, c_1, $w_2(z)$, c_2

f) H6: $r_1(x)$, $r_1(y)$, $w_1(y)$, c_1, $r_2(y)$, $r_2(z)$, c_2

g) H7: $w_1(y)$, $w_2(y)$, $r_2(z)$, $r_1(x)$, $w_1(x)$, c_1, $r_2(x)$, $w_2(z)$, c_2

h) H8: $r_2(y)$, $w_2(y)$, $r_1(x)$, $r_1(z)$, $w_1(z)$, c_1, $r_2(z)$, $w_2(z)$, c_2

i) H9: $r_1(x)$, $r_1(y)$, $w_1(z)$, $r_2(z)$, $w_2(x)$, c_2, $w_1(x)$, c_1