4. Übungszettel

Vorlesung Transaktionale Systeme

Abgabe *am Do. 27/05/2010 in der Übung*

H. Schweppe, J. Broß

Aufgabe 1 (Callback Locking) 5 Punkte

Gegeben sei ein Data-Sharing-System mit drei Servern A, B und C. Nehmen Sie an, dass Server C der Heimatserver für die (Daten-)Seiten a, b, c und d ist. Die Server A und B greifen dynamisch während der Transaktionsausführung auf die einzelnen Seiten zu. Geben Sie für unten stehende Ausführungsreihenfolge den Nachrichtenfluss im Callback-Locking-Protokoll an:

A: r1(a) w1(a) r1(c) c1 r3(a) w3(a) c3 r6(b)r6(d) c6
B: r2(c) c2 r4(a) r4(b) r4(d) c4 r5(b) w5(b) r5(d) c5

Aufgabe 2 (Timestamp Ordering) 5 Punkte

Unten stehend sind einige Sequenzen von Operationen angegeben. Hierbei bedeute st[i], dass die Transaktion Ti startet. Der Scheduler wird die Zeitstempel von Transaktionen in der Reihenfolge ihrer Starts festlegen. Sprich st[2] < st[1] in einer Sequenz bedeutet, dass der Zeitstempel von T2 kleiner (älter) als der von T1 ist.
Simulieren Sie das Timestamp-Ordering-Protokoll für folgende Sequenzen. Wenden Sie, wenn möglich die Thomas-Write-Rule an:

a) st[1] st[2] r1(a) r2(b) w2(a) w1(b)

b) st[1] r1(a) st[2] w2(b) r2(a) w1(b)

c) st[1] st[2] st[3] r1(a) r2(b) w1(c) r3(b) r3(c) w2(b) w3(a)

d) st[1] st[3] st[2] r1(a) r2(b) w1(c) r3(b) r3(c) w2(b) w3(a)

Aufgabe 3 (Optimistic Concurrency Control) 5 Punkte

Betrachten Sie den folgenden Schedule dreier Transaktionen t1, t2 und t3:

r1(x) r2(x) r1(y) r3(x) w1(x) w1(y) c1 r2(y) r3(z) w3(z) c3 r2(z) c2

1. Welchen Schedule produziert ein BOCC-Scheduler
2. Welchen Schedule produziert ein FOCC-Scheduler

Beachten Sie, dass Schreiboperationen zunächst auf lokalen Kopien und auf der Datenbank erst nach erfolgreicher Validierung stattfinden.

Projekt:

Fertigstellung des ResourceManagers:

a) Erweitern Sie bis zum 27.05.2010 den ResourceManager um einen Schattenspeichermechanismus, so dass nun auch die Dauerhaftigkeit der Daten gewährleistet werden kann. Nutzen Sie dazu zwei Dateien auf dem Hintergrundspeicher für die Datenbank. Eine Datei enthält stets den aktuellen konsistenten Zustand der DB. In einer weiteren Datei merken Sie sich einen Zeiger auf die Datei mit dem aktuellen Zustand.
Beim Commit einer Transaktion wird die DB zunächst im Hauptspeicher aktualisiert, dann in die Schattendatei kopiert und bei erfolgreichem Schreiben der Zeiger auf die Schattendatei umgesetzt. Die Schattendatei enthält nun den aktuellen konsistenten Zustand.
Eine detaillierte Aufgabenbeschreibung finden Sie auf der Projektwebseite.