Eidgenössische Technische Hochschule Zürich
Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Swiss Federal Institute of Technology Zurich

Dr. G. Alonso — *C. Hagen, B. Kemme*

AR 1: Transaktionsverwaltung

Theoretische Übung 2 — Abgabe: 15.5.1997

Aufgabe 1: Blind Writes

Ein Blind Write ist ein Write einer Transaktion auf ein Objekt, das diese Transaktion vorher nicht gelesen hat.

Beweisen Sie unter der Annahme, dass Transaktionen keine Blind Writes enthalten (d.h. wenn $w_i(x) \in T_i$ dann gilt $r_i(x) \in T_i$ und $r_i(x) <_i w_i(x)$), folgende Aussage:

Ein Schedule ist genau dann konfliktserialisierbar, wenn er viewserialisierbar ist.

Aufgabe 2: Inkrementelle Erkennung der Serialisierbarkeit

$$\begin{aligned} T_1 &= < r_1^1(a)\ r_2^1(c)\ r_3^1(b)\ w_4^1(b)\ w_5^1(c) > \\ T_2 &= < r_1^2(c)\ r_2^2(a)\ w_3^2(a)\ r_4^2(b) > \\ S &= < r_1^2(c)\ r_1^1(a)\ r_2^1(c)\ r_2^2(a) \ldots > \end{aligned}$$

Versuchen Sie den Schedule der Transaktionen T_1, T_2 so fortzusetzen, dass er

a) viewserialisierbar ist

b) konfliktserialisierbar ist

Entscheiden Sie in beiden Fällen, ob dies möglich ist und geben Sie das Entscheidungsvorgehen an.

Aufgabe 3: Rücksetzbarkeit / Striktheit

Zeigen Sie, dass die strikten Schedules eine echte Teilmenge der kaskadenfreien Schedules sind. Finden Sie dazu ein Beispiel für einen Schedule, der kaskadenfrei, aber nicht strikt ist.

Striktheit ist unabhängig von Serialisierbarkeit: Konstruieren Sie einen Schedule der strikt ist, aber nicht konfliktserialisierbar.

Aufgabe 4: Sperrverfahren

a) Jeder 2PL – Schedule ist konfliktserialisierbar. Zeigen Sie, dass die Umkehrung, d.h. $S \in \text{CPSR} \Rightarrow S \in \text{2PL}$, **nicht** gilt.

b) Betrachten Sie ein Sperrprotokoll, in dem keine Lesesperren gesetzt werden und in dem keine Lock-Konversion durchgeführt wird (wenn ein Datenobjekt geschrieben werden soll, wird eine Schreibsperre erworben). Schreibsperren werden bis zum Commit der Transaktion gehalten.

   Sind mit diesem Protokoll *lost updates* möglich? Können *Dirty Reads* auftreten? Sind Leseoperationen wiederholbar (*repeatable read*) ?

c) Betrachten Sie ein modifiziertes 2PL-Protokoll, in dem Lesesperren zu Beginn der Schrumpfungsphase freigegeben werden, sobald die Leseoperationen beendet sind. Schreibsperren werden bis zum Commit der Transaktion gehalten.

   Garantiert dieses Protokoll Serialisierbarkeit? Können *Dirty Reads* auftreten? Sind Leseoperationen wiederholbar (*repeatable read*) ? Sind alle so entstehenden Schedules strikt?

d) Betrachten Sie ein modifiziertes 2PL-Protokoll, bei dem in der Schrumpfungsphase Lock-Konversionen zugelassen sind, d.h. eine S-Sperre kann in eine X-Sperre umgewandelt werden. Zeigen Sie durch ein Gegenbeispiel, dass ein derart modifiziertes 2PL-Protokoll Konfliktserialisierbarkeit nicht mehr garantiert.