3.5 Dekompositions-Verfahren

4 Transaktionsverwaltung

4.1 Begriffe

Transaktion. Als Transaktion bezeichnet man die Ausführung eines Programmes, das Lese- und Schreibzugriffe auf die Datenbank durchführt.

Konflikt. Konfliktär sind zwei Operationen, deren Reihenfolge nicht vertauscht werden kann, ohne dass sich ihr Ergebnis ändert. Zwei Operationen o_1 und o_2 konfligieren, wenn sie auf das gleiche Datenobjekt zugreifen, und o_1 oder o_2 eine Schreiboperation ist.

$$o_1[x] \not\parallel o_2[x] \Leftrightarrow o_1[x] = w[x] \vee o_2[x] = w[x]$$

Eine Ausnahme bilden hier Inkrement- und Dekrement-Operationen, die gegenseitig kompatibel sind.

History. Eine History ist eine Menge von Transaktionen, deren Operationen nebenläufig ablaufen.

$$H = \{T_1, \ldots, T_n\}$$

Eine vollständige History Zu Scheduling-Zwecken wird als History ein Präfix einer vollständigen History bezeichnet.

Reads-From-Beziehung.

$$T_i \leftarrow T_j$$

Eine Transaktion T_i liest von einer Transaktion T_j falls

1. T_i liest x, nachdem T_j x geschrieben hat;
2. T_j abortet nicht, bevor T_i x liest;
3. jede andere Transaktion, die x in der Zeit zwischen $w_j[x]$ und $r_i[x]$ schreibt, abortet vor $r_i[x]$;

Committed Projection. Die committed projection einer History $C(H)$ resultiert aus H durch Löschen aller Operationen, die nicht committed sind.

Konfliktrelation. Die Konfliktrelation einer History H ist die Menge der nach Ausführungsreihenfolge geordneten Paare von konfligierenden Operationen.

$$KR(H) = \{(o <_H p) : o, p \in H,\ o \not\parallel p\}$$

Konfliktäquivalenz. Die Histories H und H' sind konfliktäquivalent, falls sie sie gleichen Operationen enthalten und die Konfliktrelationen von $C(H)$ und $C(H')$ identisch sind.

Cascading Abort

4.2 Anomalien

Lost Update Update geht verloren, da es von einer anderen Transaktion überschrieben wird

$$r_1[x] < r_2[x] < w_2[x] < w_1[x]$$

Dirty Read Datenobjekt wird in einem inkonsistenten Zustand gelesen

$$r_1[x] < w_1[x] < r_2[x] < w_2[x] < c_2 < a_1$$

Non-Repeatable Read Leseergebnis nicht wiederholbar, weil andere Transaktion das Datenobjekt zwischenzeitlich geändert hat.

$$r_1[x] < r_2[x] < w_2[x] < r_1[x]$$

Phantom Read entspricht Non-Repeatable Reads auf Mengen statt Werten: Während einer Transaktion wiederholte gleiche Anfragen ergeben unterschiedliche Ergebnismengen, da andere Transaktion die Relation geändert haben

4.3 Eigenschaften von Histories

Prefix Commit-Closed Eine Eigenschaft α einer History $H = o_1 \dots o_n$ heißt prefix-commit closed, falls α auch für jedes Präfix $H' = o_1 \dots o_k, k < n$ von H gilt.

4.4 Konfliktserialisierbarkeit

Konfliktgraph. Zu einer History H, an der mehrere Transaktionen $\mathcal{T} = \{T_1, \dots, T_n\}$ beteiligt sind, gibt es einen Konfliktgraphen $G_K(H) = (V \subseteq \mathcal{T}, E \subseteq \mathcal{T} \times \mathcal{T})$. Zur Erstellung des Konfliktgraphen betrachtet man den Konfliktrelation von H, eingeschränkt auf diejenigen Konflikte, die zwischen Operationen aus verschiedenen Transaktionen bestehen.

$$KRT(H) = \{(o <_H p) \in KR(H) : o \in T_i, p \in T_j, i \neq j\}$$

Für jeden Konflikt $(o_i <_H p_j)$ mit $o \in T_i$ und $p \in T_j$ wird in den Konfliktgraph eine gerichtete Kante (T_j, T_i) eingefügt. Diese Kante kann als die Beziehung "T_j hängt ab von T_i" verstanden werden.

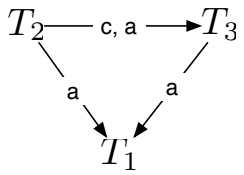

Figure 2: Beispiel eines Konfliktgraphen

konfliktserialisierbar ist eine History H

- wenn der Konfliktgraph $G_k(H)$ azyklisch ist *oder*
- wenn für eine serielle History H_s gilt: $C(H)$ und $C(H_s)$ sind konfliktäquivalent.

sichtserialisierbar wenn der zugehörige Konfliktgraph azyklisch ist

Recoverability. Um Recoverability zu gewährleisten darf eine Transaktion erst dann committet werden, wenn alle Transaktionen, von denen sie gelesen hat, bereits committet sind. Es muss gelten

$$T_i \leftarrow T_j \wedge c_i \in H \Rightarrow c_j <_H c_i$$

Wenn dies für eine History H zutrifft schreibt man $H \in RC$.

Cascadelessness / Avoids Cascading Aborts. Um Cascadelessness zu gewährleisten und Cascading Aborts zu vermeiden, darf jede Transaktion nur von zuvor committeten Transaktionen lesen. Damit $H \in ACA$ ist muss gelten

$$T_i \leftarrow T_j \Rightarrow c_j < r_i[x]$$

Cascadelessness ist eine Einschränkung von Recoverability:

$$ACA \subset RC$$

Strictness. Um Strictness zu gewährleisten dürfen geschriebene Daten einer noch laufenden Transaktion nicht geschrieben oder gelesen werden. Damit $H \in ST$ ist muss gelten

$$w_j[x] < o_i[x](i \neq j) \Rightarrow c_j < o_i[x] \wedge a_j < o_i[x]$$

Strictness ist eine Einschränkung von Cascadelessness:

$$ST \subset ACA$$

	$H \in RC$	$H \notin RC$
H konfliktserialisierbar	ja	nein
H nicht konfliktserialisierbar		nein

Table 1: Korrektheit

Korrektheit.

ACID-Eigenschaften

Atomicity

4.5 Locking

Serielle Ausfürhung.