Datenbanksysteme 2, 9. Übung
Transaktionsmanagement

Aufgabe 9.1: Serialisierbarkeit und Konfliktserialisierbarkeit

Betrachten Sie die folgenden Ausführungen S_1, S_2 und S_3 der beiden Transaktionen T_1 und T_2:

S_1		S_2		S_3	
T_1	T_2	T_1	T_2	T_1	T_2
R(a)		R(a)		R(a)	
a:=a-10			R(b)	a:=a-10	
W(a)		a:=a-10			R(b)
R(b)			b:=b*1.2	W(a)	
b:=b+10		W(a)			b:=b*1.2
W(b)			W(b)	R(b)	
	R(b)	R(b)			W(b)
	b:=b*1.2		R(c)	b:=b+10	
	W(b)	b:=b+10			R(c)
	R(c)		c:=c+20	W(b)	
	c:=c+20	W(b)			c:=c+20
	W(c)		W(c)		W(c)

Für welche der 3 Schedules treffen die Begriffe seriell, serialisierbar oder nicht serialisierbar zu? Gehen Sie davon aus, dass alle Datenwerte mit 0 initialisiert sind. Welche sind konfliktserialisierbar? Begründen Sie Ihre Entscheidung.

Aufgabe 9.2: Konfliktserialisierbarkeit

Betrachten Sie die beiden folgenden Schedules:

- $R_1(X); R_3(X); W_1(X); R_2(X); W_3(X)$
- $R_3(X); R_2(X); W_3(X); R_1(X); W_1(X)$

a) Schreiben Sie zu beiden Schedules die Konfliktpaare auf.
b) Zeichnen Sie zu beiden Schedules den Abhängigkeitsgraphen.
c) Falls ein Schedule konfliktserialisierbar ist, überführen Sie ihn durch konfliktäquivalente Umformungen in einen seriellen Schedule.

Aufgabe 9.3: Rücksetzbarkeit von Schedules

Betrachten Sie die Schedules aus Aufgabe 9.1 in Bezug auf Rücksetzbarkeit und kaskadierende Abbrüche. Untersuchen Sie dabei auch die Varianten, dass T_1 einen Commit direkt nach der letzten Operation der Transaktion absetzt, T_2 aber nicht, und umgekehrt.

Untersuchen Sie für alle 9 Schedules, ob sie rücksetzbar sind und ob kaskadierende Abbrüche auftreten könnten. Begründen Sie Ihre Antworten jeweils.

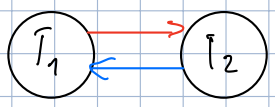
1.)

S_1: 

- seriell, jede Schritte folgen unmittelbar hinter einander

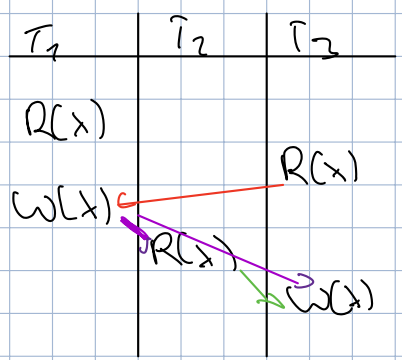
S_2: 

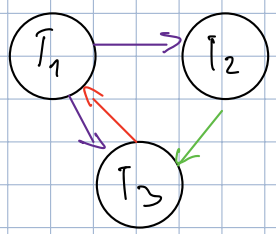
- serialisierbar, da dass Ergebnis gleich ist wie an dem seriellen

S_3: 

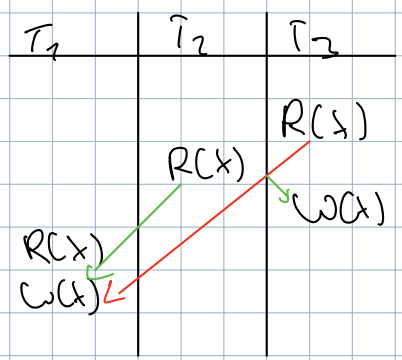
- es ist nicht konfliktserialisierbar, da Zyklen vorhanden sind
- jedoch ist es serialisierbar, da das Ergebnis trotz des "Lost Updates" gleich bleibt!?

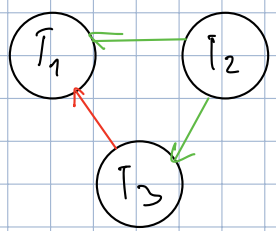
2.) S_1:





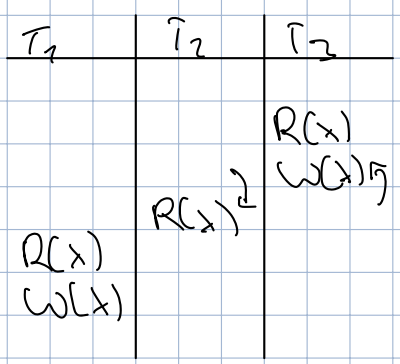
S_2:





c.) S1 geht nicht !

S2:



3)

S_1		S_2		S_3	
T_1	T_2	T_1	T_2	T_1	T_2
R(a)		R(a)		R(a)	
a:=a-10			R(b)	a:=a-10	
W(a)		a:=a-10			R(b)
R(b)			b:=b*1.2	W(a)	
b:=b+10		W(a)			b:=b*1.2
W(b)			W(b)	R(b)	
	R(b)	R(b)			W(b)
	b:=b*1.2		R(c)	b:=b+10	
	W(b)	b:=b+10			R(c)
	R(c)		c:=c+20	W(b)	
	c:=c+20	W(b)			c:=c+20
	W(c)		W(c)		W(c)

S_1 ohne Commit

- Wenn T_2 abgebrochen wird passiert nichts
- Falls T_1 abgebrochen wird muss T_2 auch
  => kaskadierende Abbrüche

S_1 mit Commit nach T_1

- rücksetzbar

S_1 mit Commit nach T_2

- nicht rücksetzbar

S_2 ohne Commit

- Wenn T_1 abgebrochen wird passiert nichts
- Falls T_2 abgebrochen wird muss T_2 auch
  => kaskadierende Abbrüche

S_2 mit Commit nach T_1

=nicht rücksetzbar

S_2 mit Commit nach T_2

- rücksetzbar

liefert keine Transaktion von einer anderen

=> rücksetzbar ✓

0 bleib 0