Serialisierbarkeit von Transaktionen

- Die nebenläufige Bearbeitung von Transaktionen geschieht für den Benutzer transparent, d.h. als ob die Transaktionen (in einer beliebigen Reihenfolge) hintereinander ausgeführt werden
- **Begriffe**
  - Ein *Schedule* für eine Menge $\{T_1, \ldots, T_n\}$ von Transaktionen ist eine Folge von Aktionen, die durch Mischen der Aktionen der T_is entsteht, wobei die Reihenfolge innerhalb der jeweiligen Transaktion beibehalten wird.
  - Ein *serieller Schedule* ist ein Schedule S von $\{T_1, \ldots, T_n\}$, in dem die Aktionen der einzelnen Transaktionen nicht untereinander verzahnt sondern in Blöcken hintereinander ausgeführt werden.
  - Ein Schedule S von $\{T_1, \ldots, T_n\}$ ist *serialisierbar*, wenn er dieselbe Wirkung hat wie ein beliebiger serieller Schedule von $\{T_1, \ldots, T_n\}$.

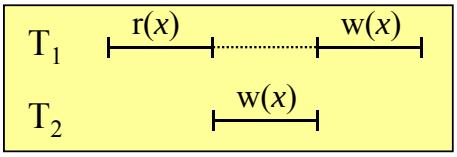
Nur serialisierbare Schedules dürfen zugelassen werden!

Serialisierbarkeit von Transaktionen

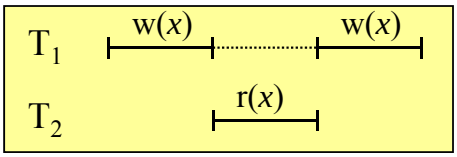
Datenbanksysteme I
Ergänzungsfolien zu Kapitel 8: Transaktionen

Beispiele für nicht-serialisierbare Schedules:

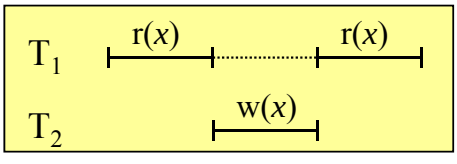
- Lost Update: $S=(r_1(x), w_2(x), w_1(x))$



- Dirty Read: $S=(w_1(x), r_2(x), w_1(x))$



- Non-repeatable Read: $S=(r_1(x), w_2(x), r_1(x))$



Kriterium für Serialisierbarkeit

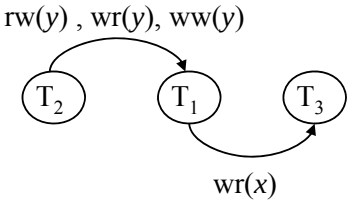
Mit Hilfe von Serialisierungsgraphen (Abhängigkeitsgraphen) kann man prüfen, ob ein Schedule $\{T_1, \ldots, T_n\}$ serialisierbar ist.

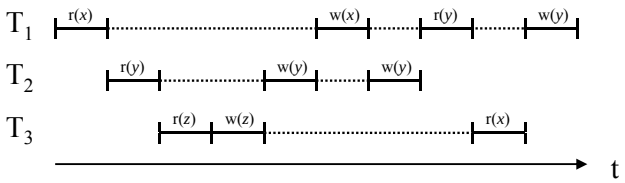
- Die beteiligten Transaktionen $\{T_1, \ldots, T_n\}$ sind die *Knoten* des Graphen.
- Die *Kanten* beschreiben die Abhängigkeiten der Transaktionen:
  Eine Kante $T_i \rightarrow T_j$ wird eingetragen, falls im Schedule
  - $w_i(x)$ vor $r_j(x)$ kommt: Schreib-Lese-Abhängigkeiten $wr(x)$
  - $r_i(x)$ vor $w_j(x)$ kommt: Lese-Schreib-Abhängigkeiten $rw(x)$
  - $w_i(x)$ vor $w_j(x)$ kommt: Schreib-Schreib-Abhängigkeiten $ww(x)$
- Ein Schedule ist serialisierbar, falls der Serialisierungsgraph *zyklenfrei* ist.
- Einen zugehörigen seriellen Schedule erhält man durch topologisches Sortieren des Graphen.

Kriterium für Serialisierbarkeit

Beispiele:

- $S = (r_1(x), r_2(y), r_3(z), w_3(z), w_2(y), w_1(x), w_2(y), r_1(y), r_3(x), w_1(y))$

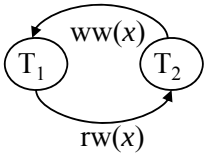




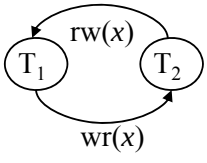
Serialisierungsreihenfolge: (T_2, T_1, T_3)

- Nicht-serialisierbare Schedules

$S=(r_1(x), w_2(x), w_1(x))$
Lost Update



$S=(w_1(x), r_2(x), w_1(x))$
Dirty Read



$S=(r_1(x), w_2(x), r_1(x))$
Non-Repeatable Read