

# The Serial Safety Net

## Efficient Concurrency Control on Modern Hardware

Tianzheng Wang,  
Ryan Johnson,  
Alan Fekete,  
Ippokratis Pandis

**Florian Lüdiger**  
**[florian.luediger@tu-dortmund.de](mailto:florian.luediger@tu-dortmund.de)**

Im Rahmen des Seminars  
Transaktionsverwaltung auf moderner Hardware  
von Prof. Dr. Jens Teubner  
04. Dezember 2017



# Agenda



**1 Motivation**

**4 Bewertung**

**2 Snapshot Isolation**

**5 Zusammenfassung**

**3 The Serial Safety Net**

**6 Persönliche Meinung**

# Agenda

1

**Motivation**

4

**Bewertung**

2

**Snapshot Isolation**

5

**Zusammenfassung**

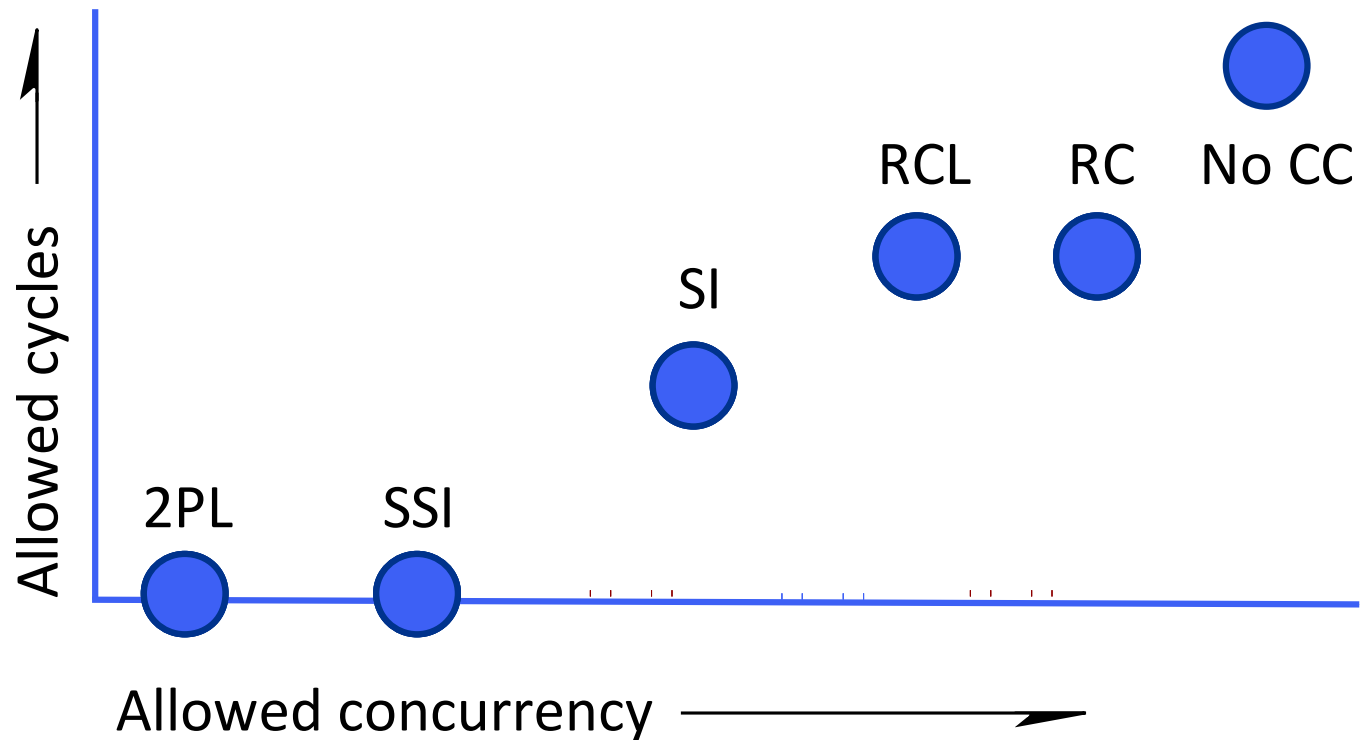
3

**The Serial Safety Net**

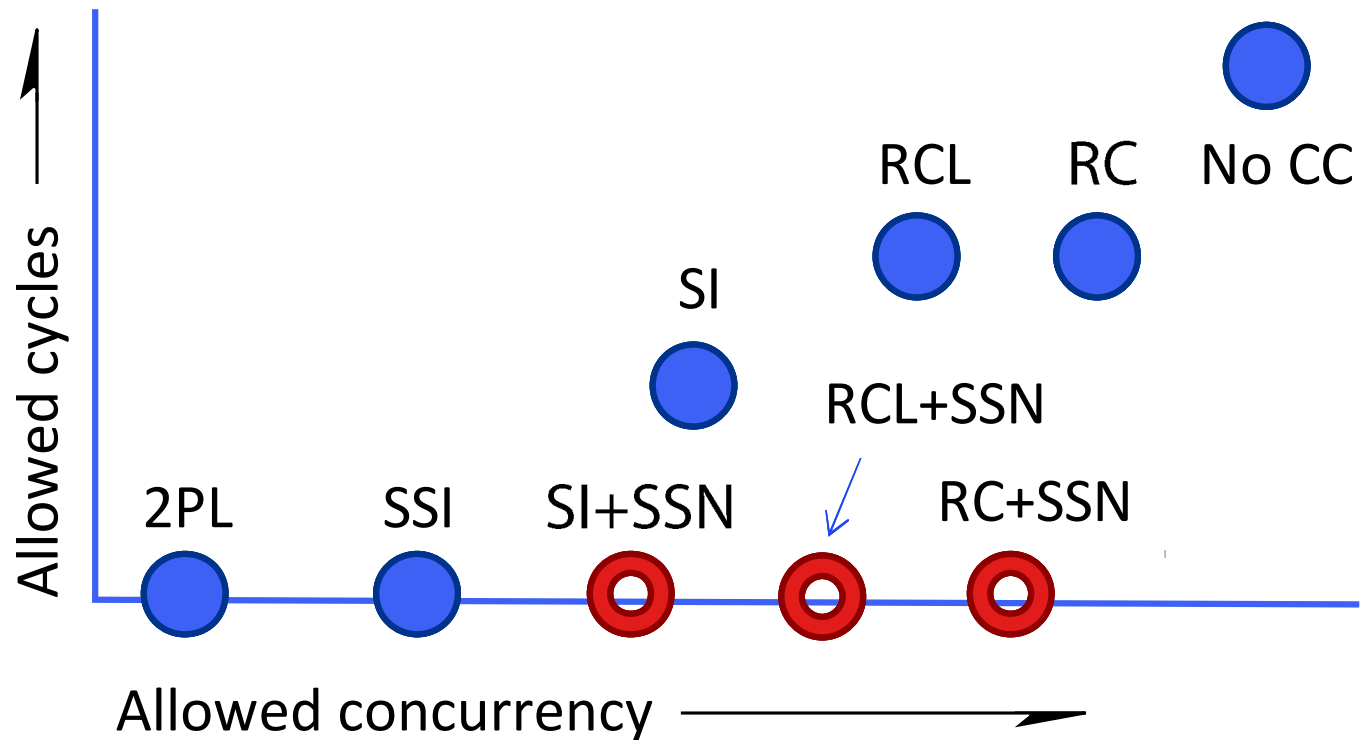
6

**Persönliche Meinung**

## Trade-Off: Striktheit gegen Nebenläufigkeit



# Trade-Off: Striktheit gegen Nebenläufigkeit



# Agenda

1 Motivation

4 Bewertung

2 Snapshot Isolation

5 Zusammenfassung

3 The Serial Safety Net

6 Persönliche Meinung

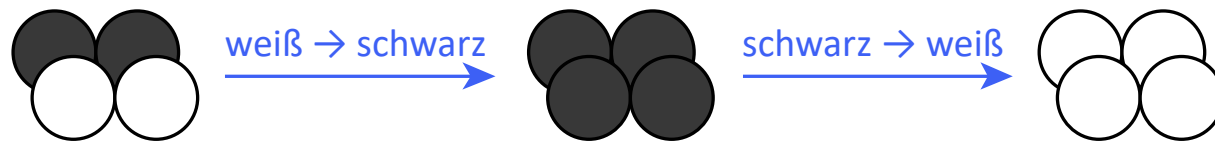


# Snapshot Isolation

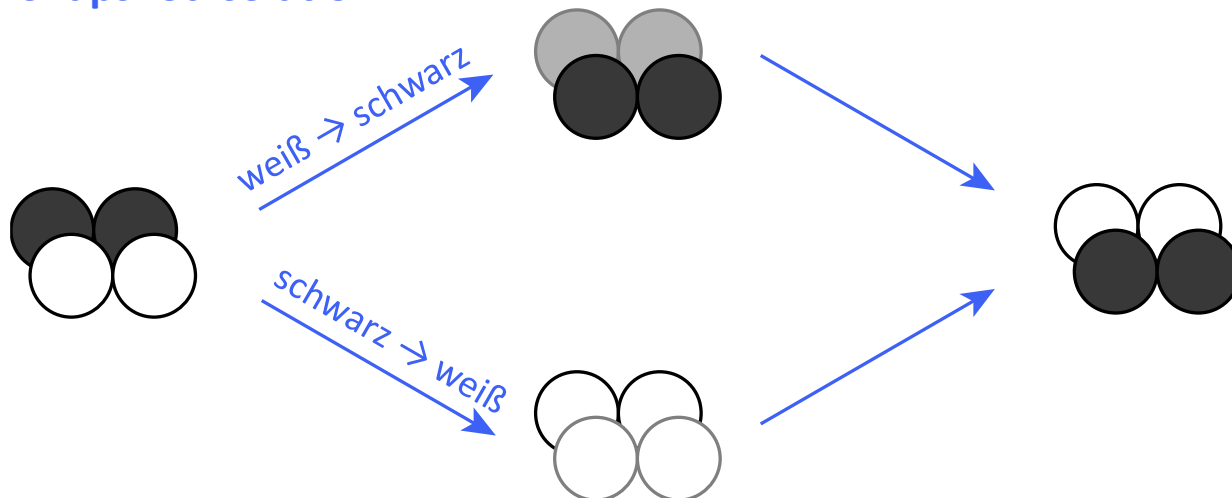
- Leseoperationen sehen Snapshot von Beginn der Transaktion
- Transaktion T1 mit Schreiboperationen darf nur committen, wenn es keine T2 gibt, sodass:
  - T2 zwischen Start- und Endzeitpunkt von T1 committet
  - T2 ein Datenobjekt manipuliert hat, welches T1 ebenfalls manipuliert hat
- Lässt sich nicht in standard Isolationslevel einordnen
- Serialisierbarkeit nicht gewährleistet → SSI

# Write Skew

## Serialisierbar



## Snapshot Isolation







# Agenda



**1 Motivation**

**4 Bewertung**



**2 Snapshot Isolation**

**5 Zusammenfassung**



**3 The Serial Safety Net**



**6 Persönliche Meinung**




# The Serial Safety Net

- Sichert kreisfreien Abhängigkeitsgraphen
- Baut auf CC Verfahren auf
  - Muss mindestens **Read Committed** unterstützen
- Untersuchung von **Abhängigkeiten von Transaktionen** in Verbindung mit **Commit-Reihenfolge**



# Abhängigkeitsgraph



$T \leftarrow U$  Abhängigkeit von  $T$  und  $U$  bei der  $U$  von  $T$  abhängig ist  
 $T$  ist direkter Vorgänger,  $U$  direkter Nachfolger

## Abhängigkeitsgraph

- Knoten: Committete Transaktionen
- Kanten: Abhängigkeiten zwischen Transaktionen
- Graph ohne Zyklen garantiert Serialisierbarkeit des Plans

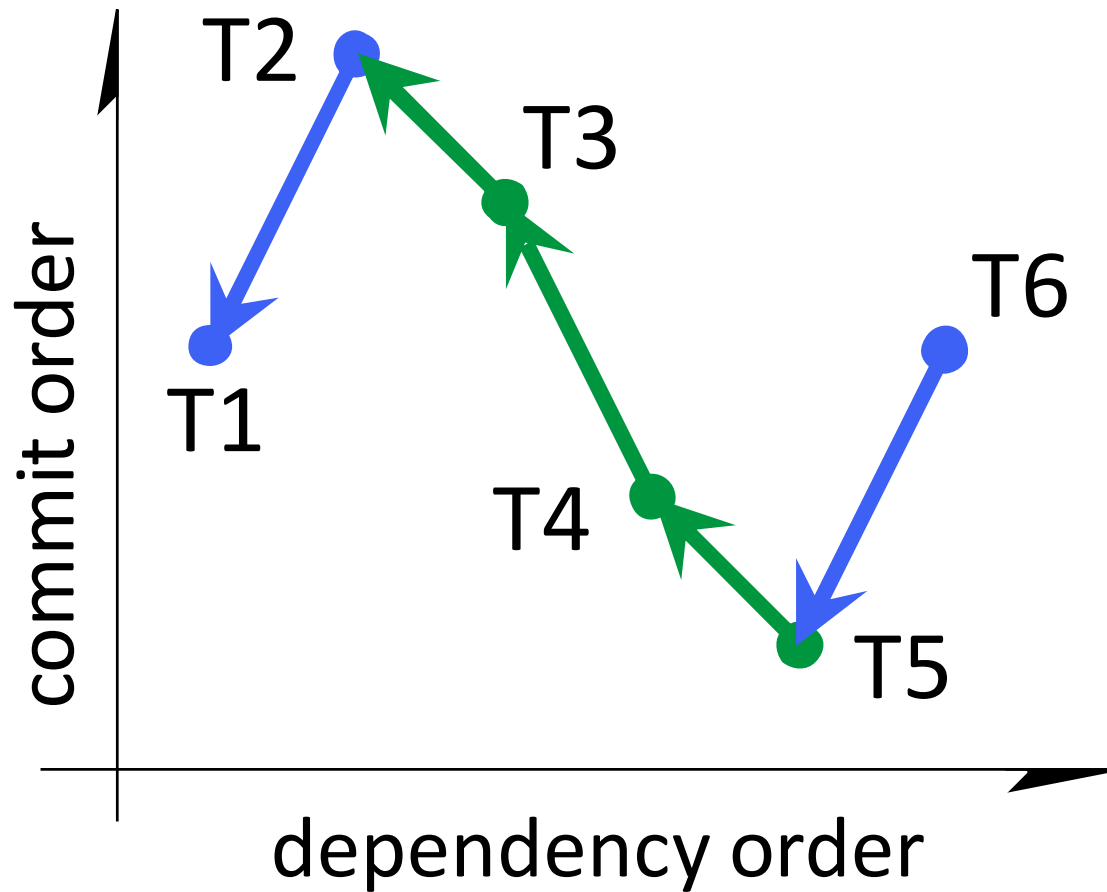
## Back- und Forward Edges

**Back-Edge**  $T \xleftarrow{b} U$ : Der Nachfolger  $U$  committet zu erst

**Forward-Edge**  $T \xleftarrow{f} U$ : Der Vorgänger  $T$  committet zu erst

**Reflexive, transitive Back-Edge**  $T \xleftarrow{b^*} U$ :  $T$  ist von  $U$  über ausschließlich Back-Edges erreichbar

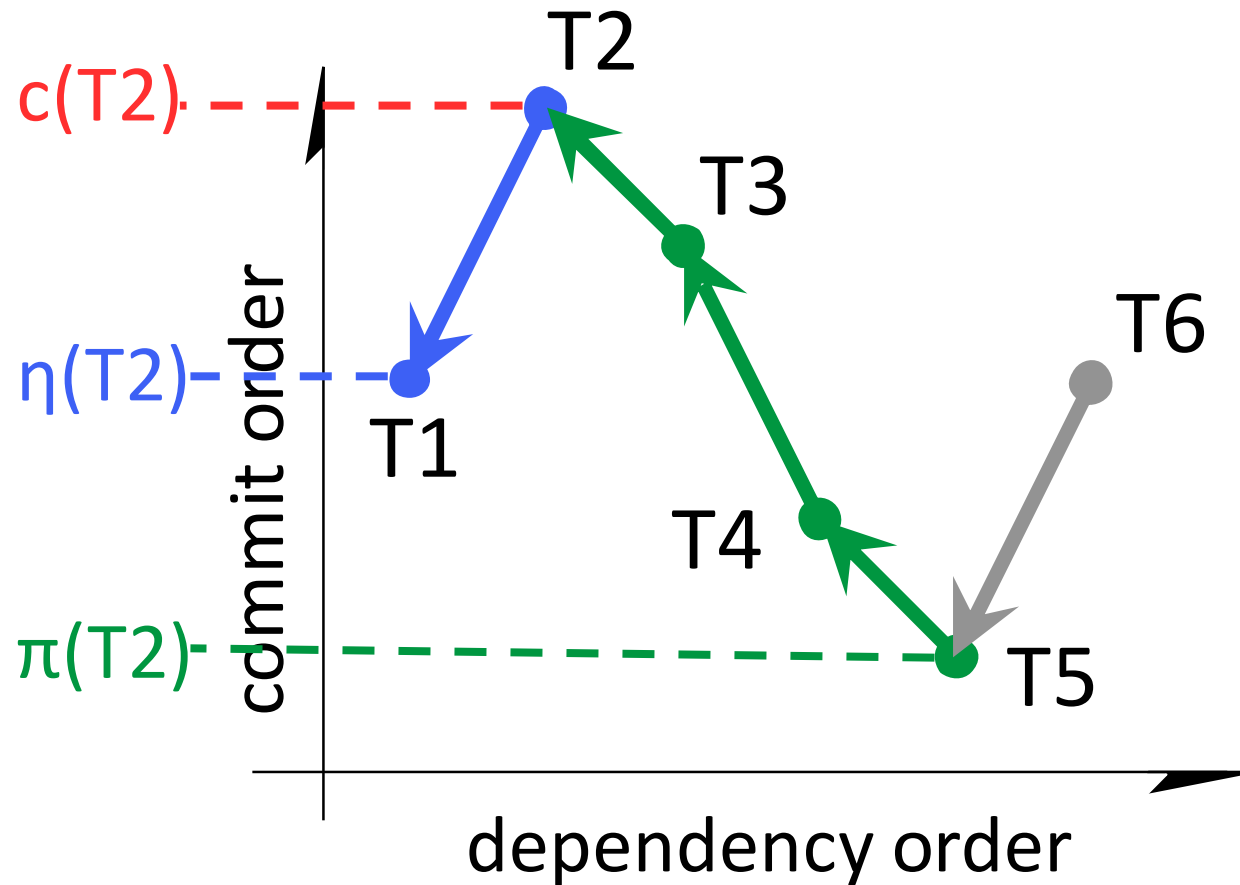
## Back- und Forward Edges



## Zu erfassende Timestamps

- $c(T)$ : Commit-Zeit der Transaktion  $T$
- $\pi(T)$ : Commit-Zeit des ältesten Nachfolgers, der durch Back-Edges erreichbar ist
- $\pi(T) = \min(c(U): T \xleftarrow{b^*} U) = \min(\{\pi(U): T \xleftarrow{b} U\} \cup \{c(T)\})$
- $\eta(T)$ : Commit-Zeit des zuletzt committeten Vorgängers von  $T$
- $\eta(T) = \max(\{c(U): U \xleftarrow{f} T\} \cup \{-\infty\})$

## Zu erfassende Timestamps



# Verletzung des Ausschlussfensters

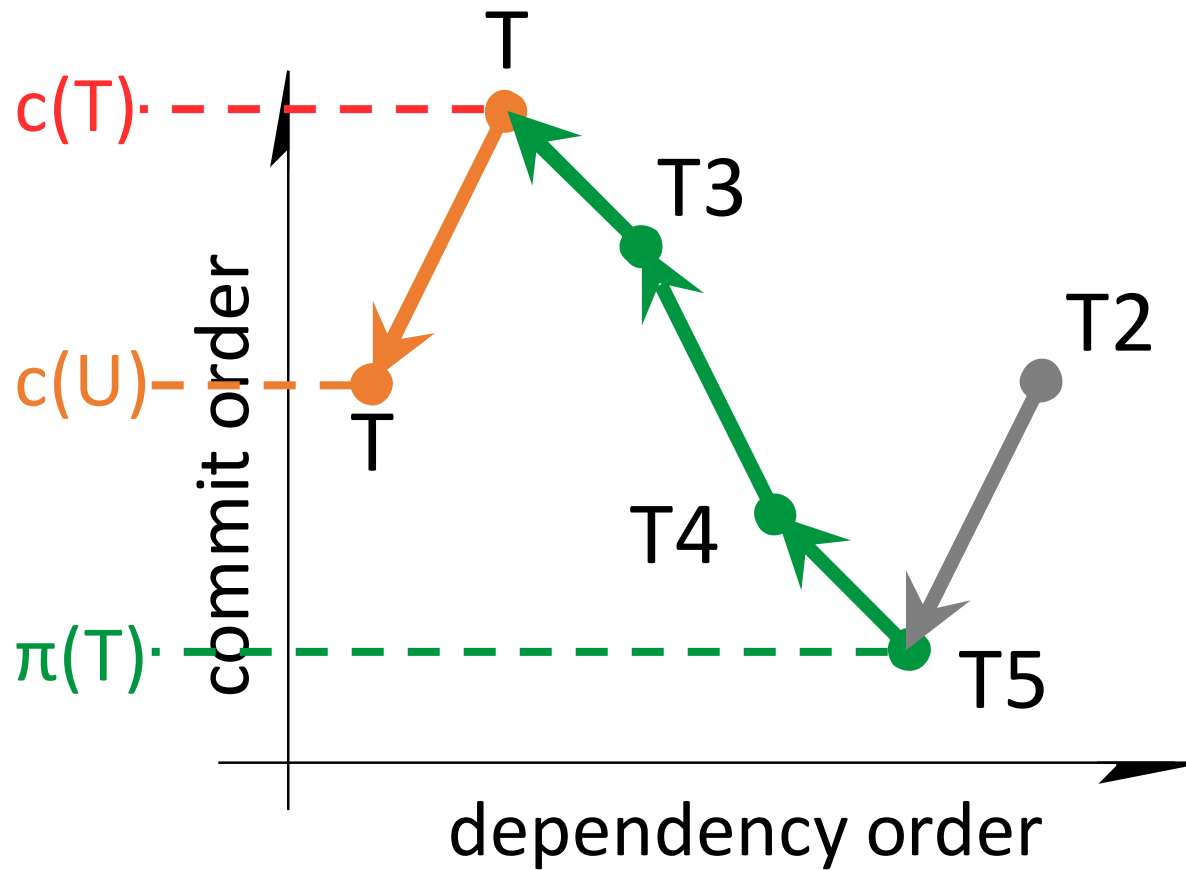
- Ausschlussfenster garantiert, dass ein Vorgänger von T nicht gleichzeitig ein Nachfolger sein kann → Abhängigkeitskreis
- Verletzung liegt vor, falls es einen Vorgänger U gibt, sodass:

$$\pi(T) \leq c(U) \leq c(T)$$



Verletzungsbedingung:  $\pi(T) \leq c(U) \leq c(T)$

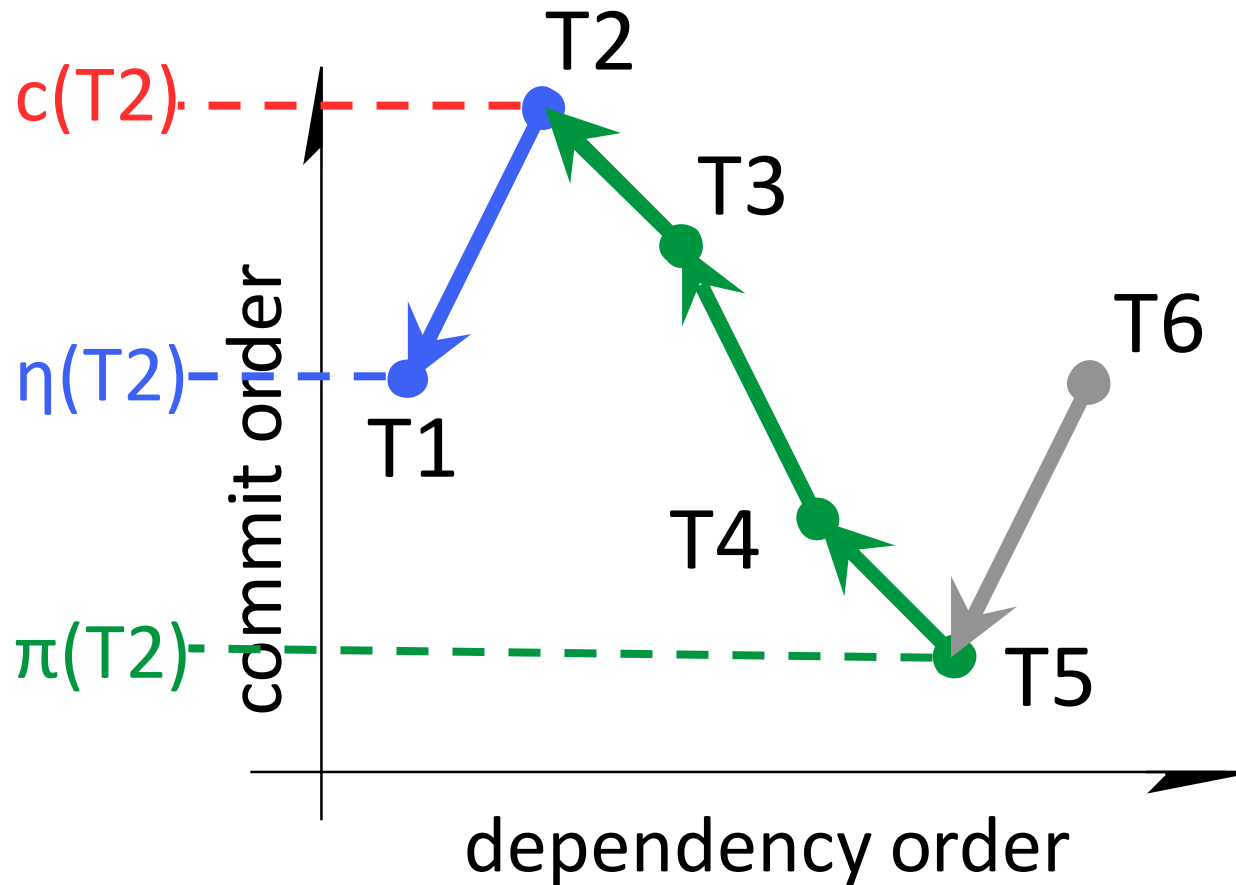
## Verletzung des Ausschlussfensters



## Verletzung des Ausschlussfensters

- Ausschlussfenster garantiert, dass ein Vorgänger von T nicht gleichzeitig ein Nachfolger sein kann → Abhängigkeitskreis
  - Verletzung liegt vor, falls es einen Vorgänger U gibt, sodass:  
 $\pi(T) \leq c(U) \leq c(T)$
  - Vereinfachung 1: Nur Vorgänger betrachten, die vor T committet sind
  - Vereinfachung 2: Nur zuletzt committeten Vorgänger von T betrachten
- Vereinfachung der Ungleichung zu:  $\pi(T) \leq \eta(T)$

## Verletzung des Ausschlussfensters



# Agenda

1

**Motivation**

4

**Bewertung**

2

**Snapshot Isolation**

5

**Zusammenfassung**

3


**The Serial Safety Net**

6

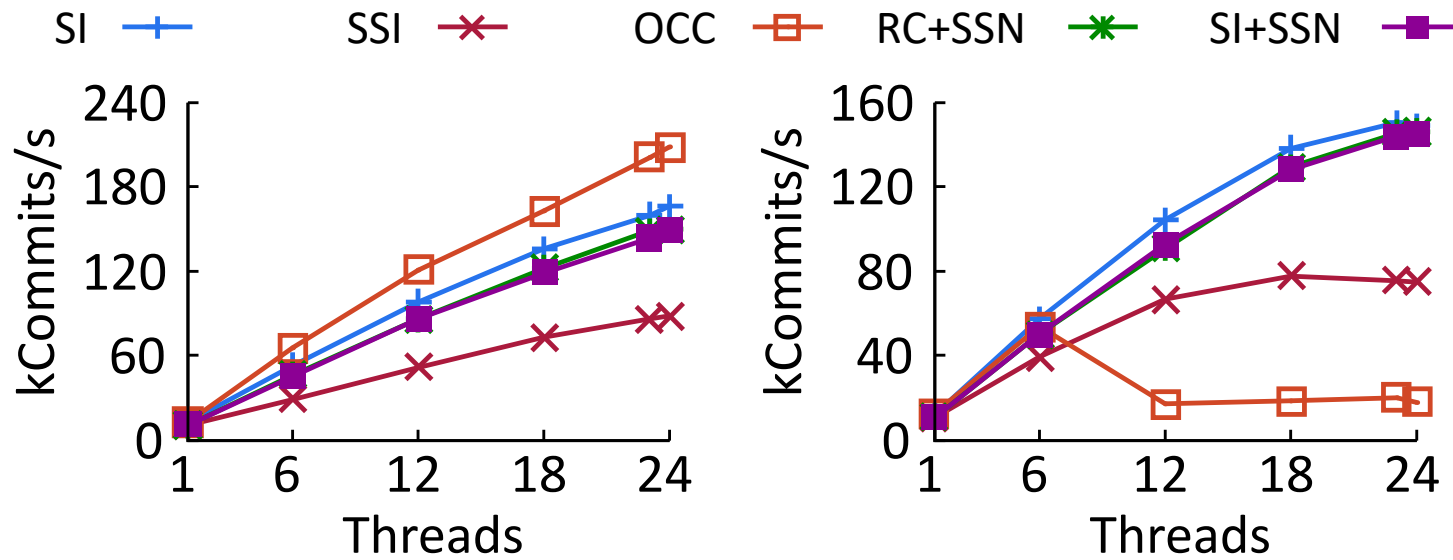
**Persönliche Meinung**



# TPC-C Benchmark

- 
- Online Transaction Processing (OLTP) Benchmark
  - Mehrere Transaktionstypen, komplexe Datenbank
  - Fünf nebenläufige Transaktionen
  - Modelliert Aktivitäten eines Großhandels

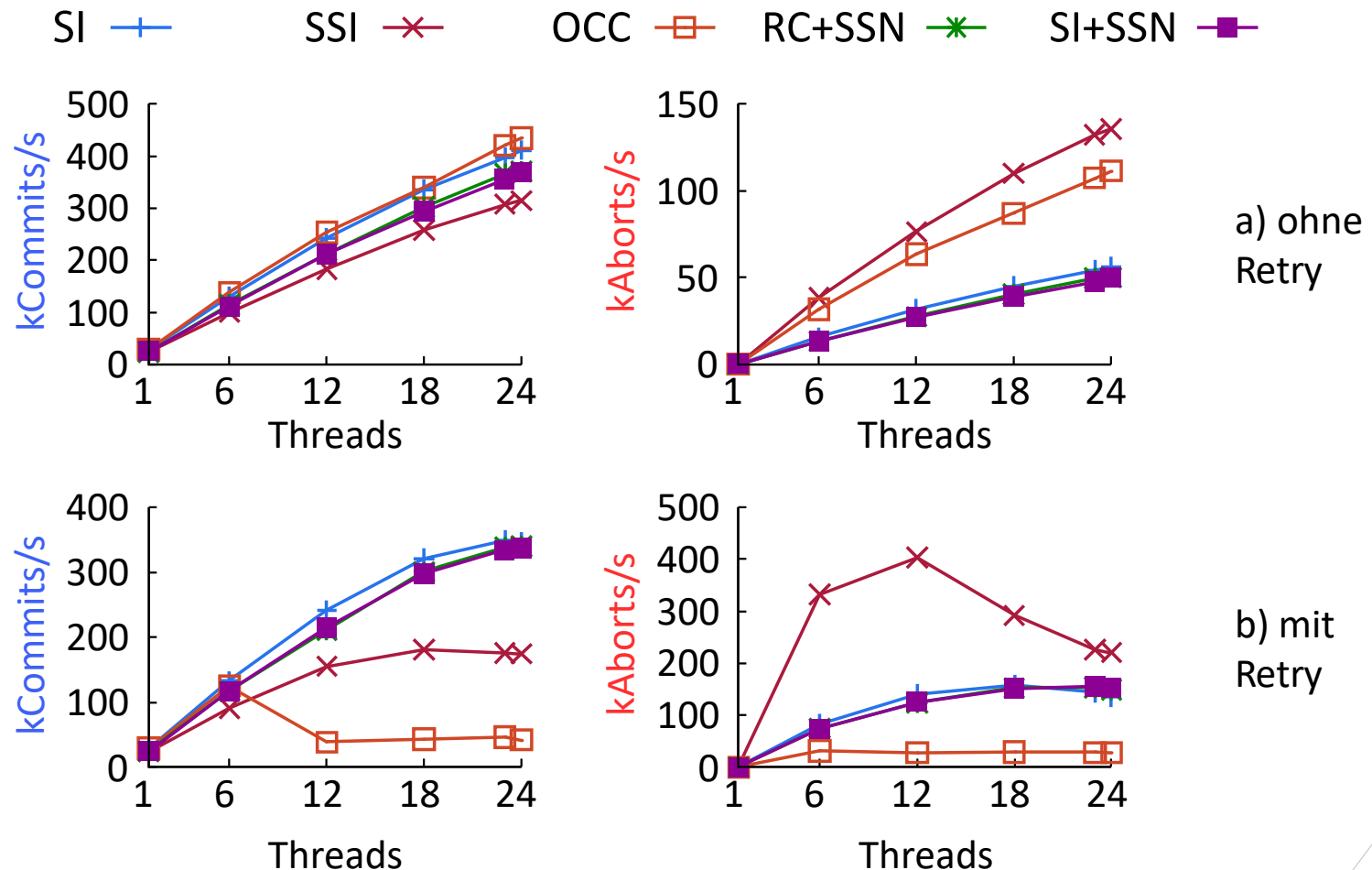
# Schreibintensive Transaktionen



a) ohneRetry

b) mitRetry

# Commit- und Abbruchraten



# Agenda

1 Motivation

4 Bewertung

2 Snapshot Isolation

5 Zusammenfassung


3 The Serial Safety Net

6 Persönliche Meinung





# Zusammenfassung

- 
- SSN verwendet bekannte hoch performante CC Verfahren und sichert Serialisierbarkeit für diese
  - Geringer Overhead, hohe Performanz und keinerlei Anomalien (Abgesehen von Phantomen)
  - Leicht zu implementieren und wenig anfällig für Fehler

# Agenda

1 Motivation

4 Bewertung

2 Snapshot Isolation

5 Zusammenfassung

3 The Serial Safety Net

6 Persönliche Meinung

# Meine Meinung zum Paper



Einsteigerfreundlich und leicht zu verstehen



Allgemein wissenschaftlich fundiert und bewiesen

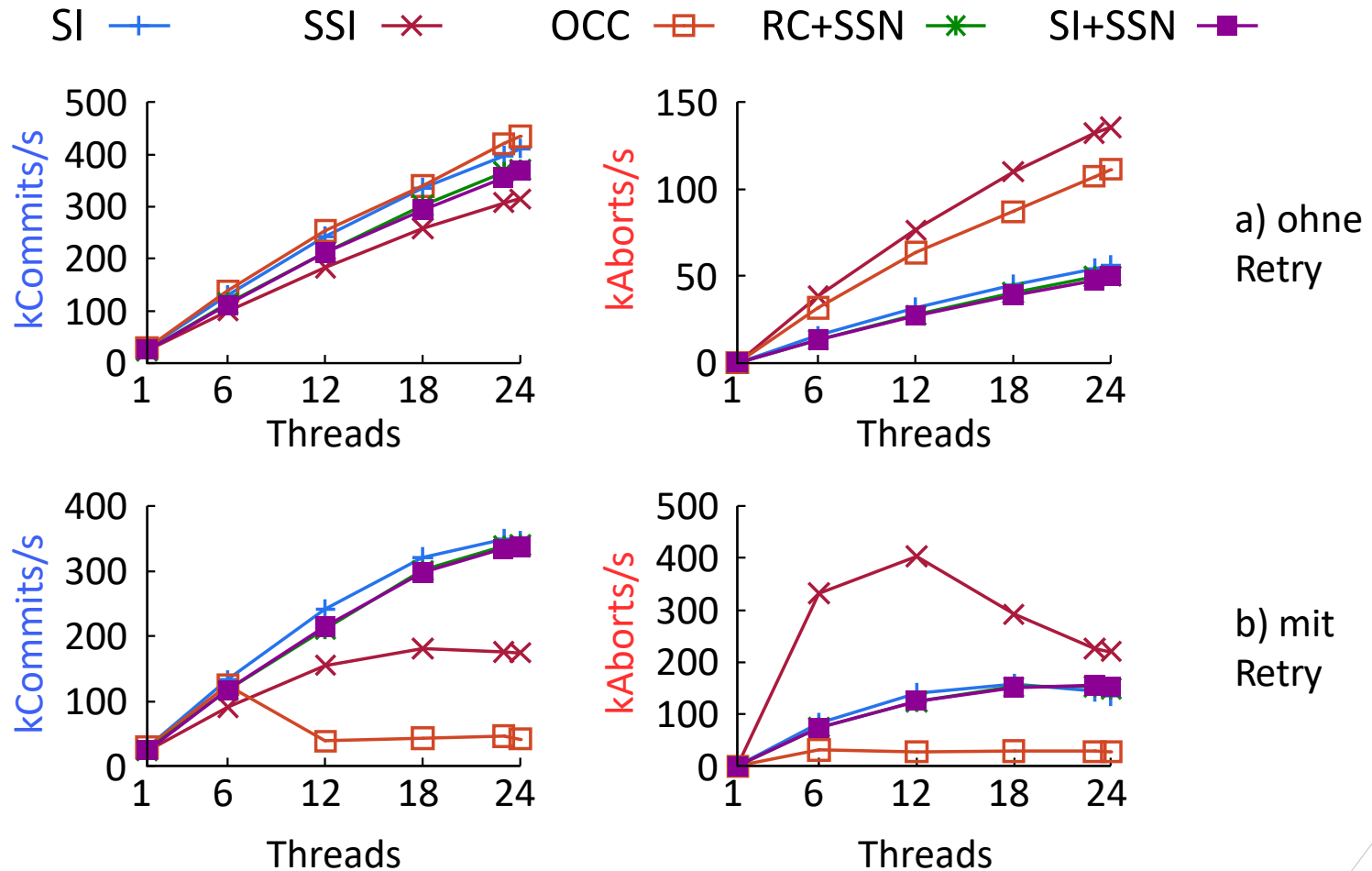


Auffälligkeiten bei Benchmarks werden teilweise überhaupt nicht erklärt



Keine klare Begründung für schlechte Performanz von Serializable Snapshot Isolation

# Commit- und Abbruchraten



# Quellen

- Tianzheng Wang, Ryan Johnson, Alan Fekete, and Ippokratis Pandis. The Serial Safety Net: Efficient Concurrency Control on Modern Hardware. *DaMoN 2015*, Juni 2015.  
DOI: [10.1145/2771937.2771949](https://doi.org/10.1145/2771937.2771949)
- Craig Freedman. Serializable vs. Snapshot Isolation Level. Mai 2007.  
<https://blogs.msdn.microsoft.com/craigfr/2007/05/16/serializable-vs-snapshot-isolation-level/>
- TPC-C Benchmark. <http://www.tpc.org/tpcc/>