# ERMIA: Fast Memory-Optimized Database System for Heterogeneous Workloads

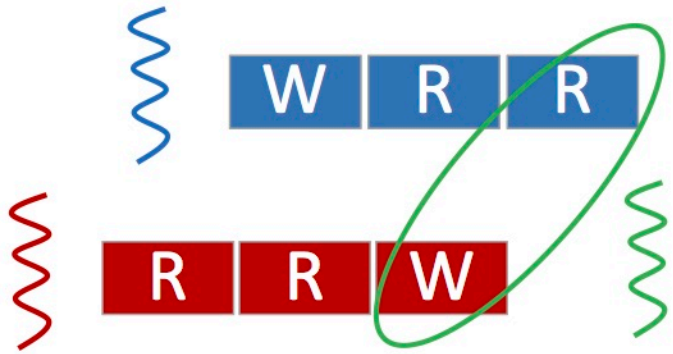
Authors: Kangnyeon Kim, Tianzheng Wang, Ryan Johnson, Ippokratis Pandis

Presented by: Fengjian Pan

Some slides contains material from the original authors' slides.

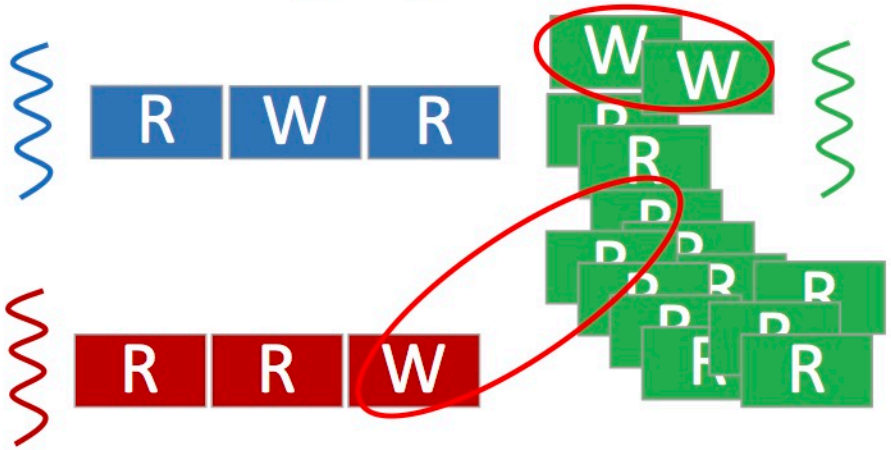
# Heterogeneous is the new OLTP

Convenient/traditional




- Short write-intensive
- Short read-only

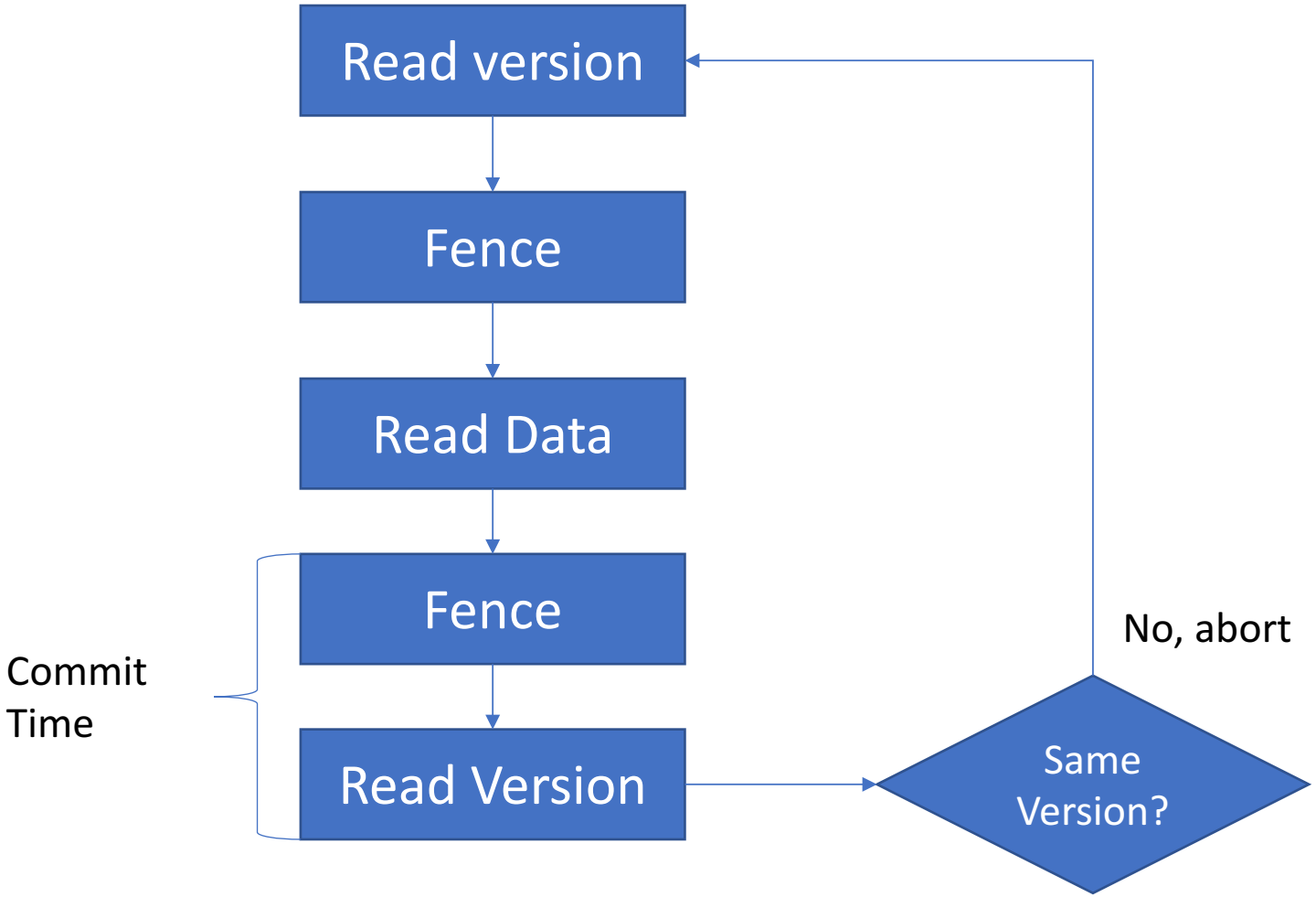
Emerging workloads




- Short write-intensive
- Longer read-***mostly***

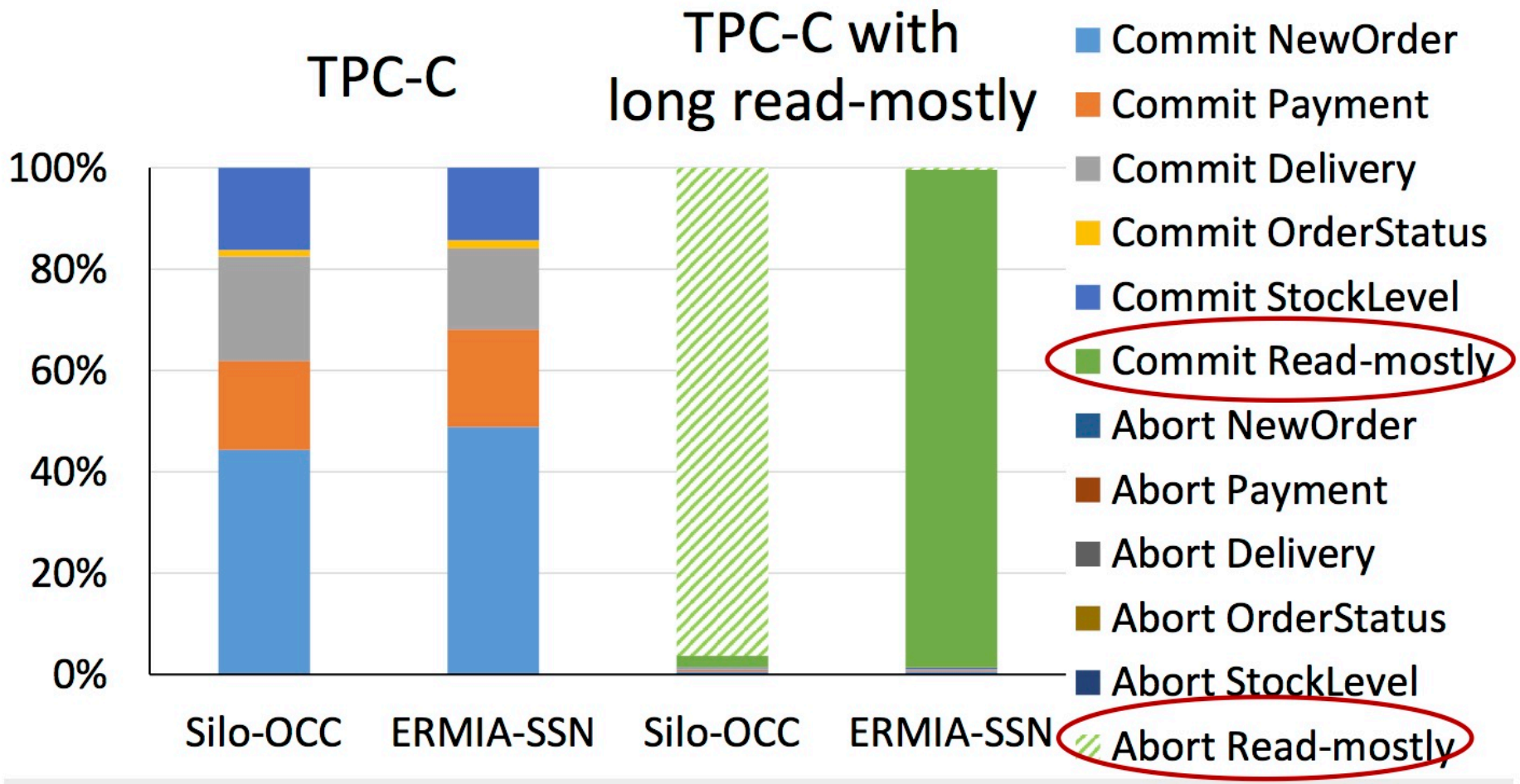
**Worst case for current systems, esp. CC**

# GAP SILO: Lightweight OCC




**Writes wins over read**

# OCC + Read-mostly = Wasted cycles

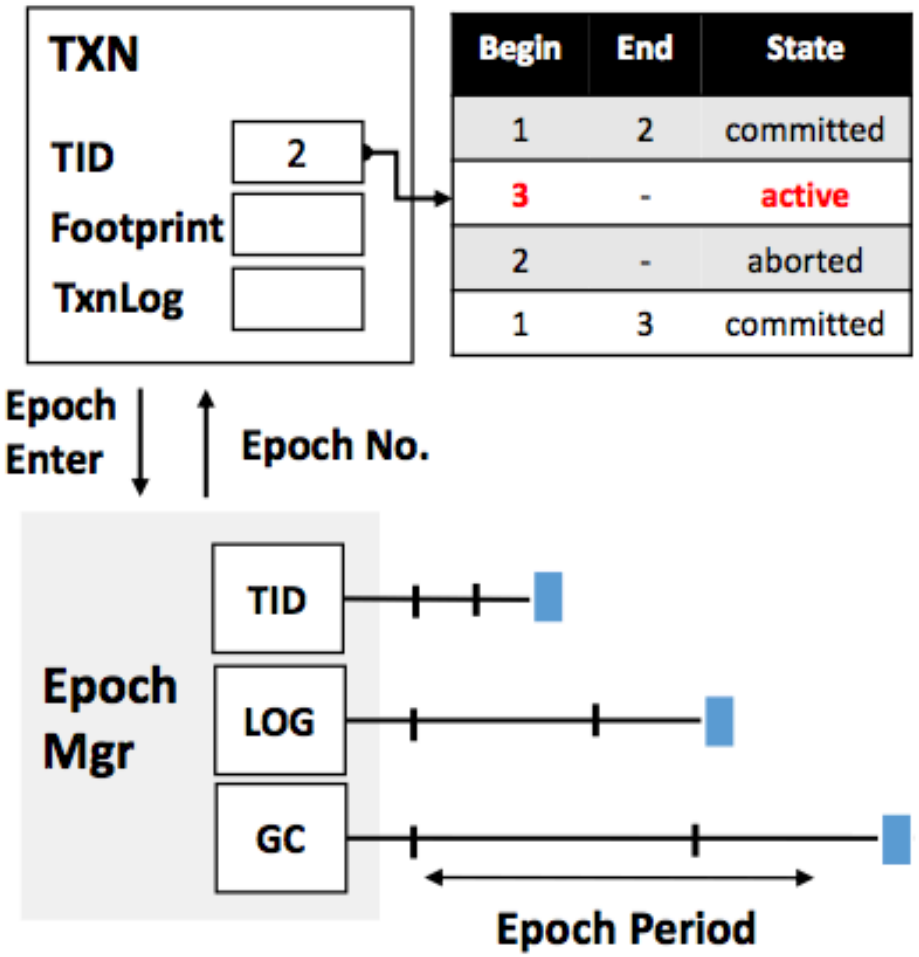



**Read-mostly tx: fair and robust CC needed**

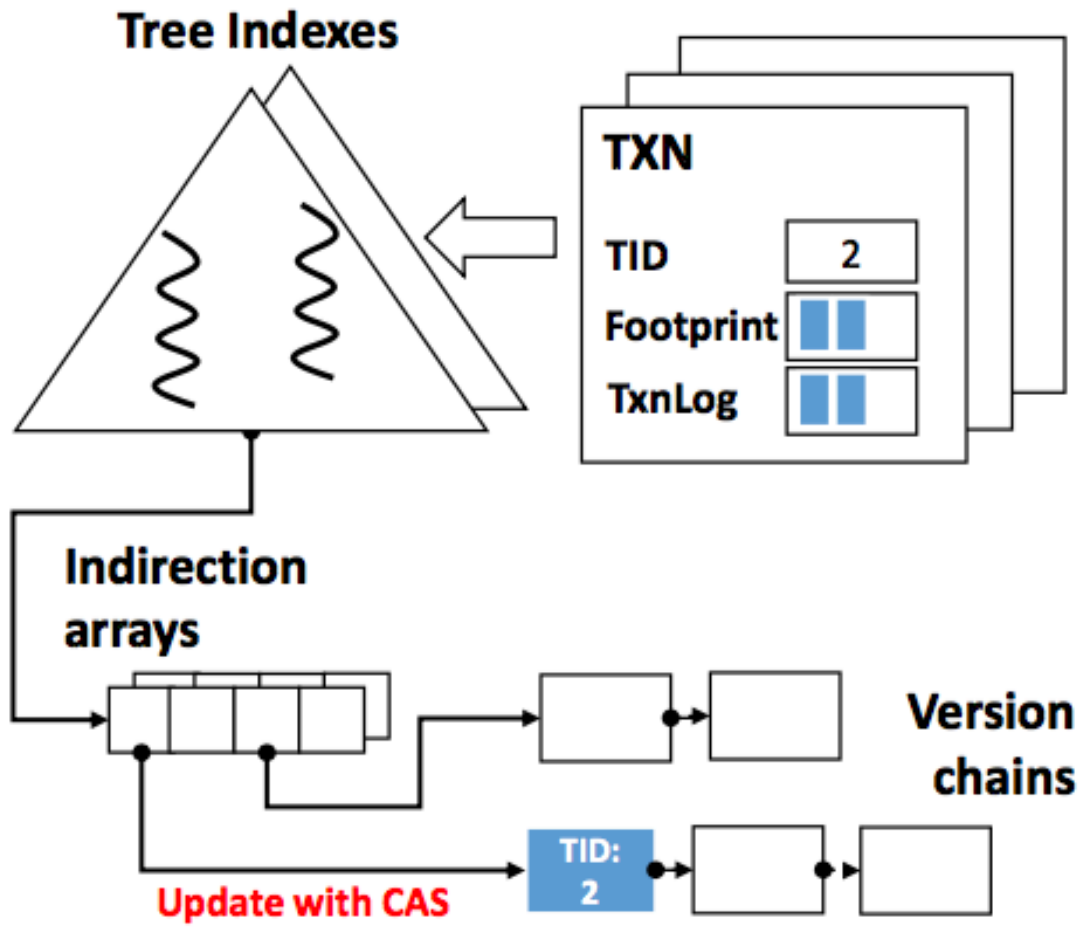
# ERMIA Key Principles

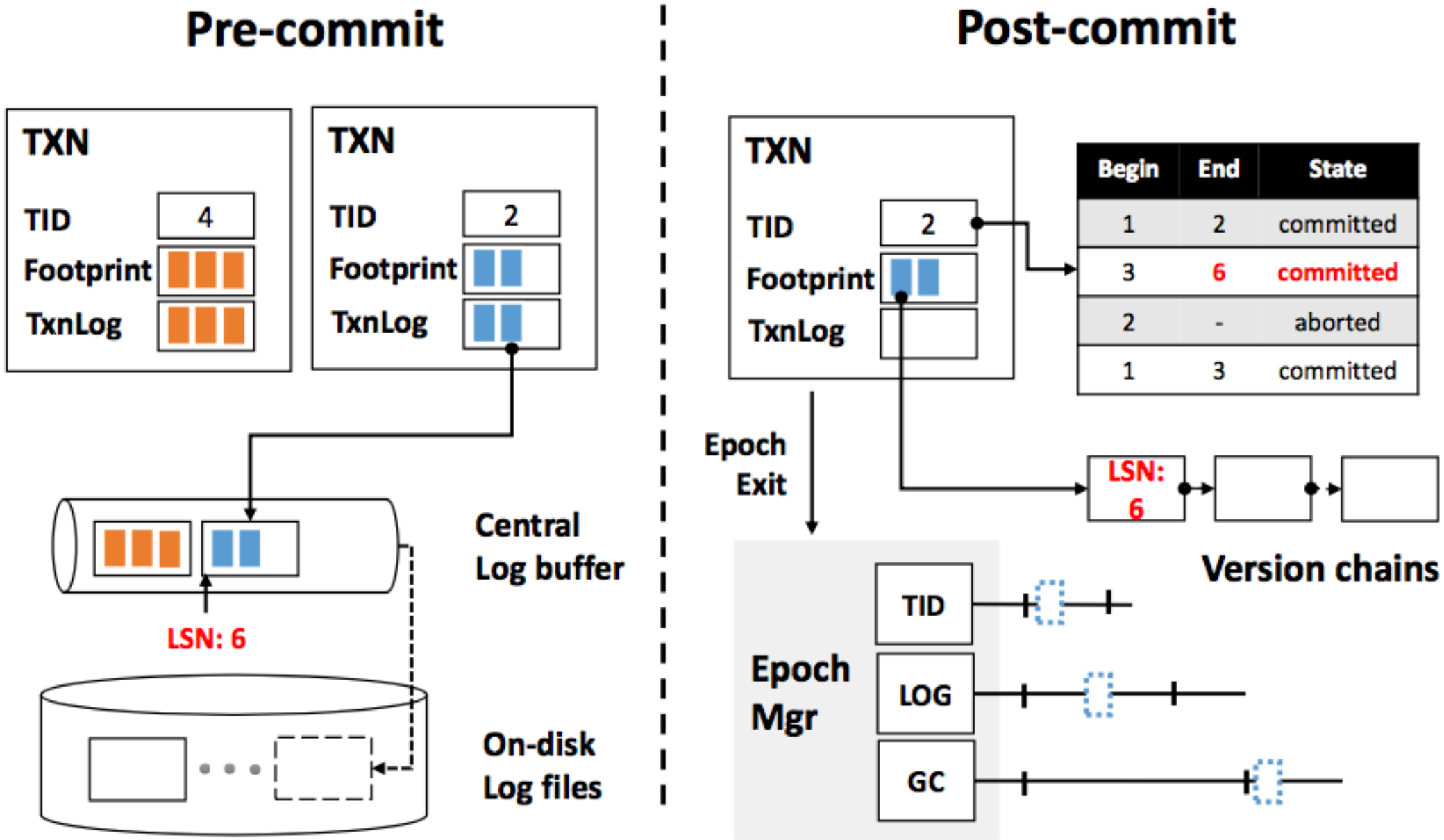
- Latch-free indirection arrays
  - Provide low overhead multi-versioning.
- Snapshot-isolation(SI)
  - Fairness between read and write
  - Early detection of doomed transaction
- Serial Safety Net(SSN)
  - Provide serializability on top of SI.

## Initialization



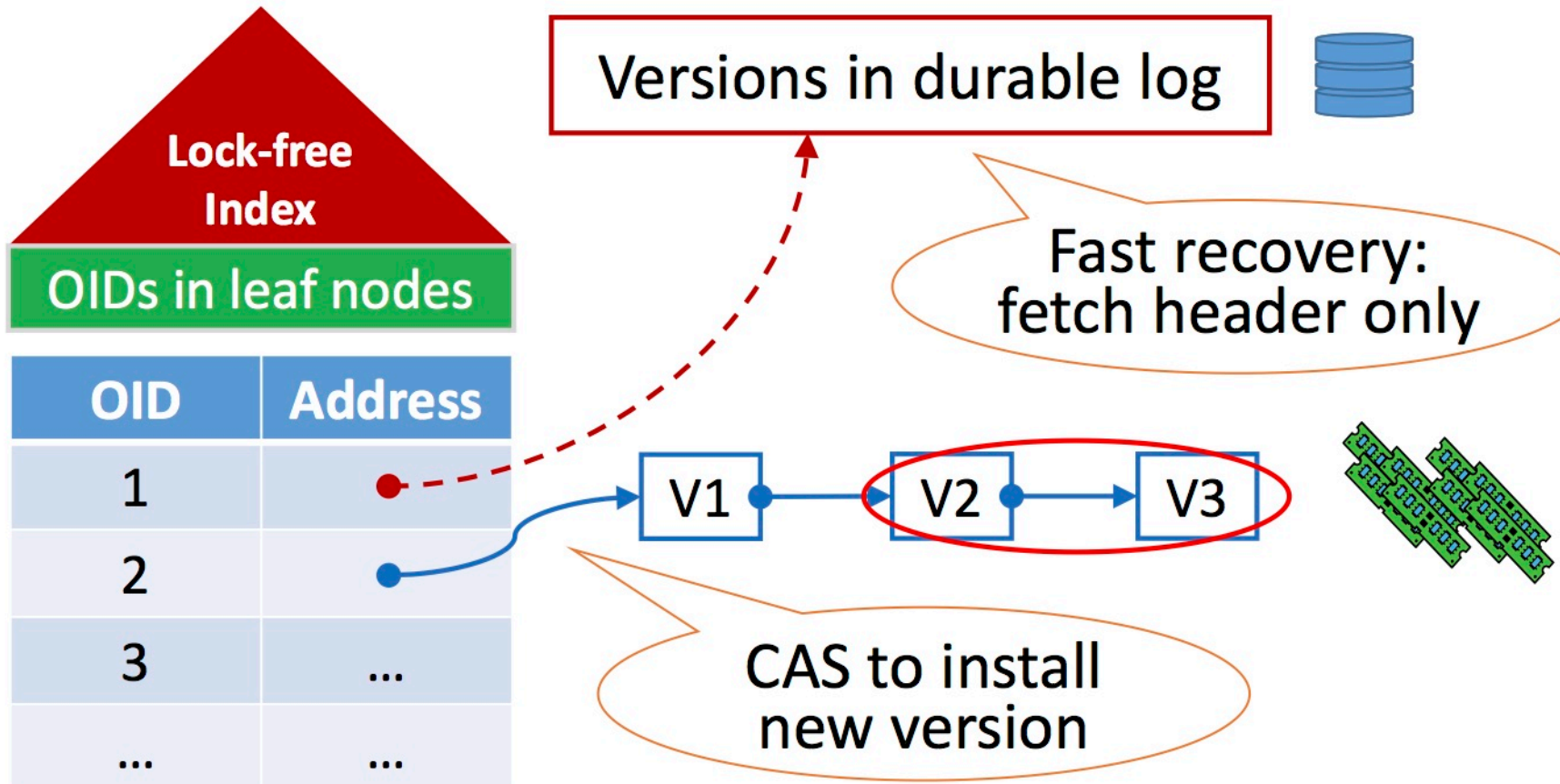

## Forward Processing




Pre-commit
Post-commit
TXN
TID
4
Footprint
TxnLog
TXN
TID
2
Footprint
TxnLog
LSN: 6
Central
Log buffer
On-disk
Log files
TXN
TID
2
Footprint
TxnLog
Begin
End
State
1
2
committed
3
6
committed
2
-
aborted
1
3
committed
Epoch
Exit
LSN:
6
Version chains
Epoch
Mgr
TID
LOG
GC

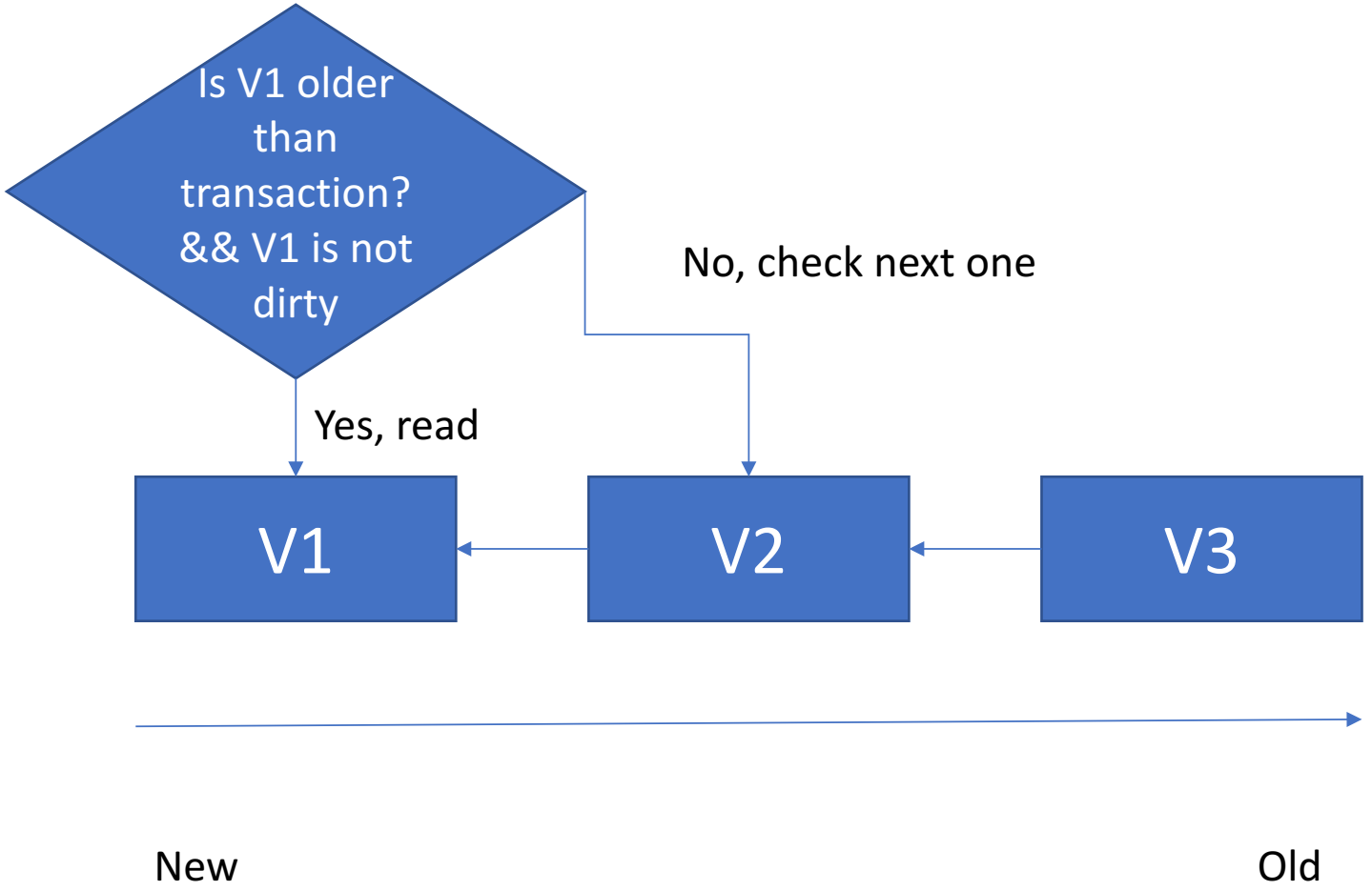
# Latch-free Indirection Array

- Object IDs rather than pointers in leaf nodes
- No update propagation to secondary indexes



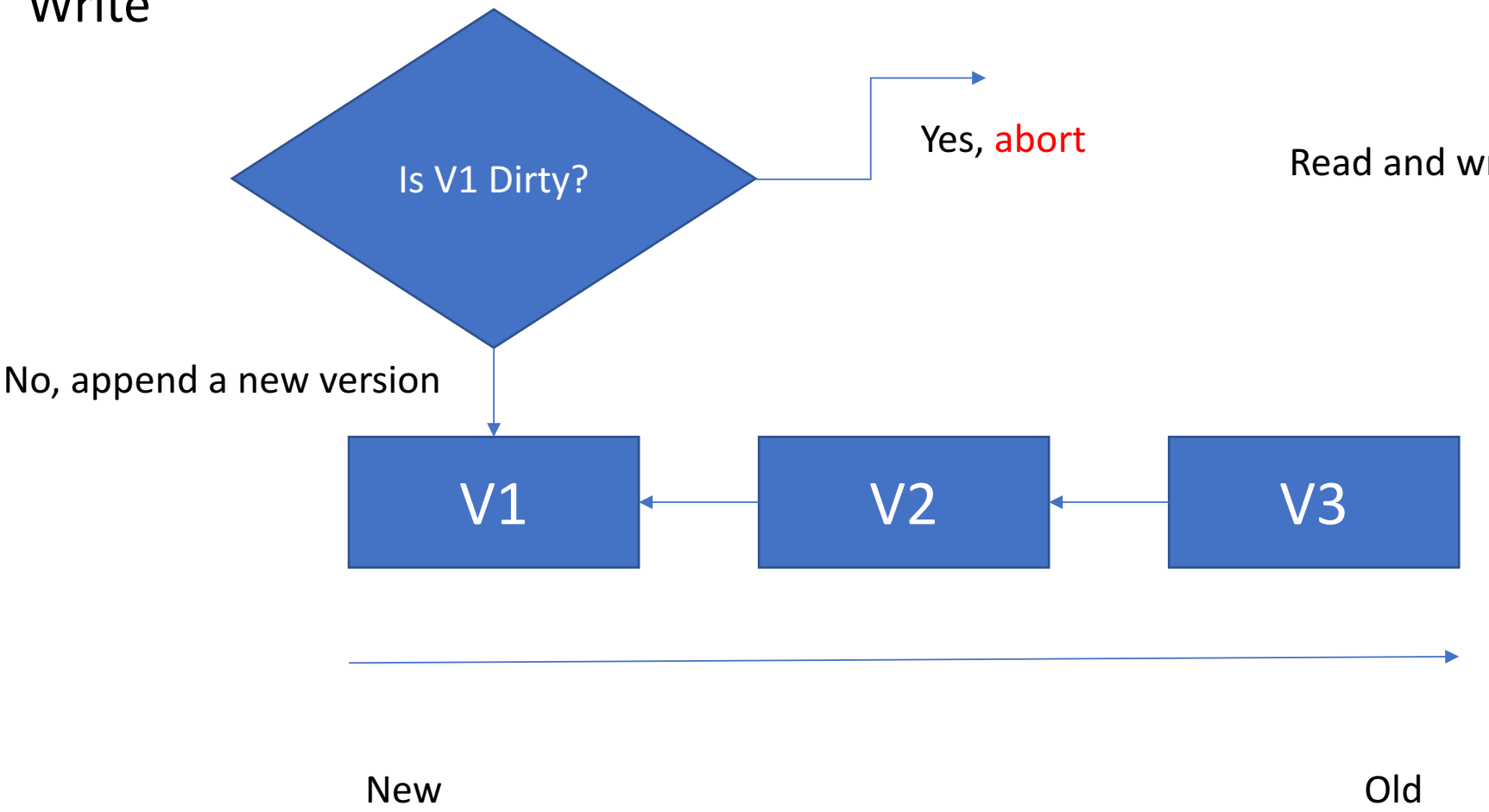
# Snapshot-Isolation(SI)

Read



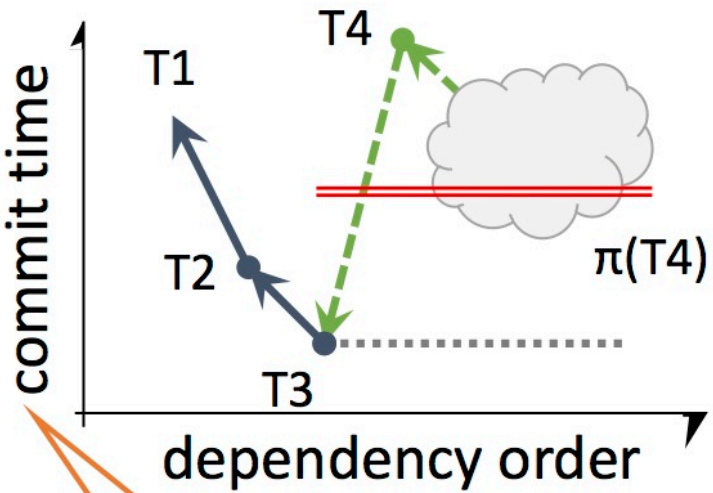
# Snapshot-Isolation(SI)

Write




Read and write never block each other

# The Serial Safety Net




T1, T2, T3:
***predecessors*** of T4

T4's ***earliest***
successor

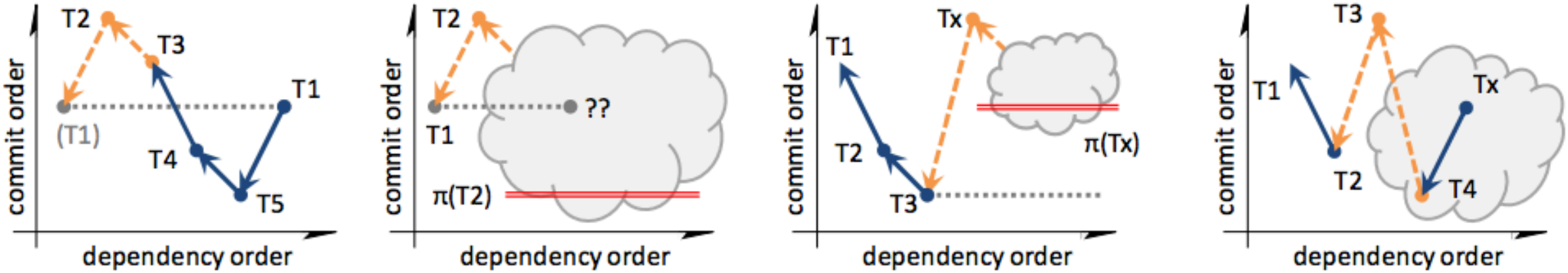
c(T3):
commit time

Forbid $\exists P \leftarrow T$ : $\pi(T) \leq c(P) \leq c(T)$

"exclusion window" of T

**The Serial Safety Net: Efficient Concurrency Control on Modern Hardware**
T. Wang, R. Johnson, A. Fekete, I. Pandis, *DaMoN '15*

# Serial Safety Net(SSN)

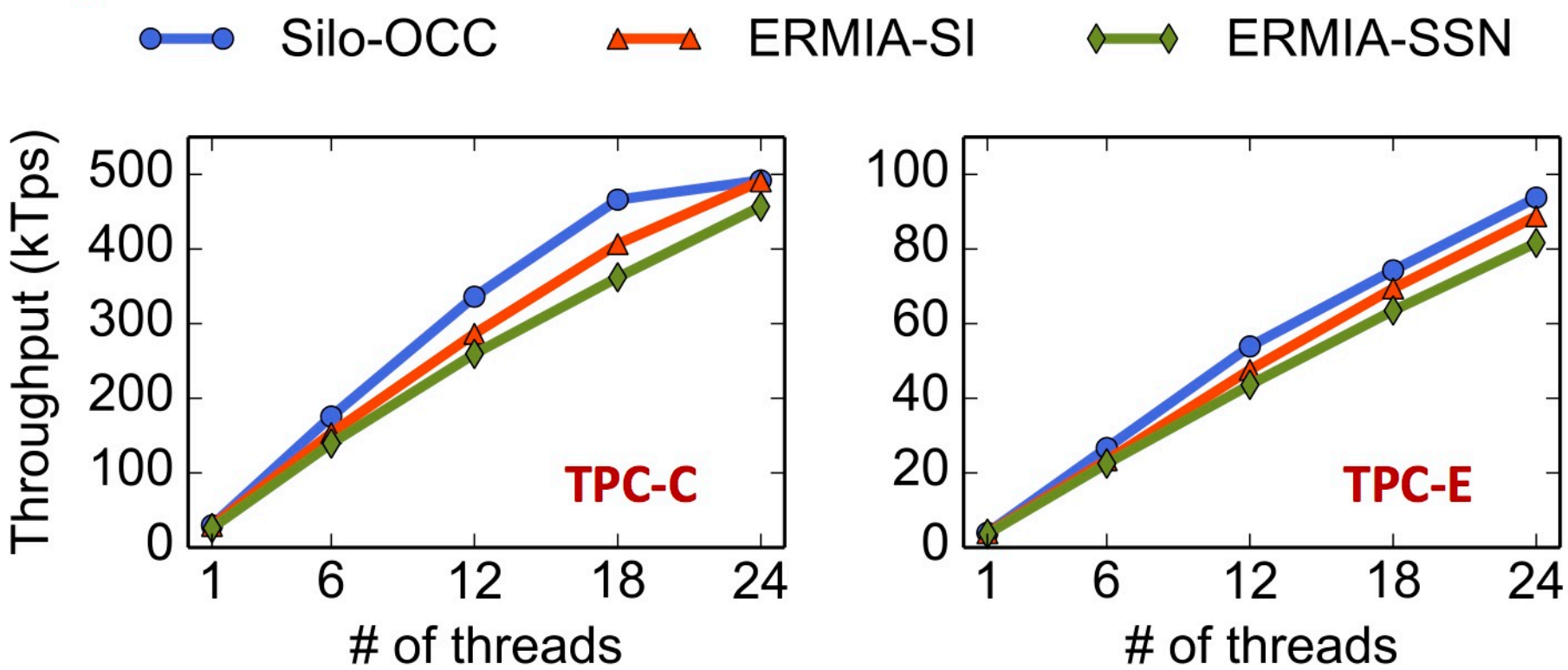


# Benchmarks

- 24-core, quad-socket Xeon E7-4807
- Everything in main memory
- OCC vs. ERMIA-SI vs. ERMIA-SSN
- "Convenient"/traditional OLTP
  - Original TPC-C
  - Original TPC-E
- Heterogeneous OLTP
  - TPC-C with read-mostly transaction (*TPC-C-hybrid*)
  - TPC-E with read-mostly transaction (*TPC-E-hybrid*)

# “Convenient”

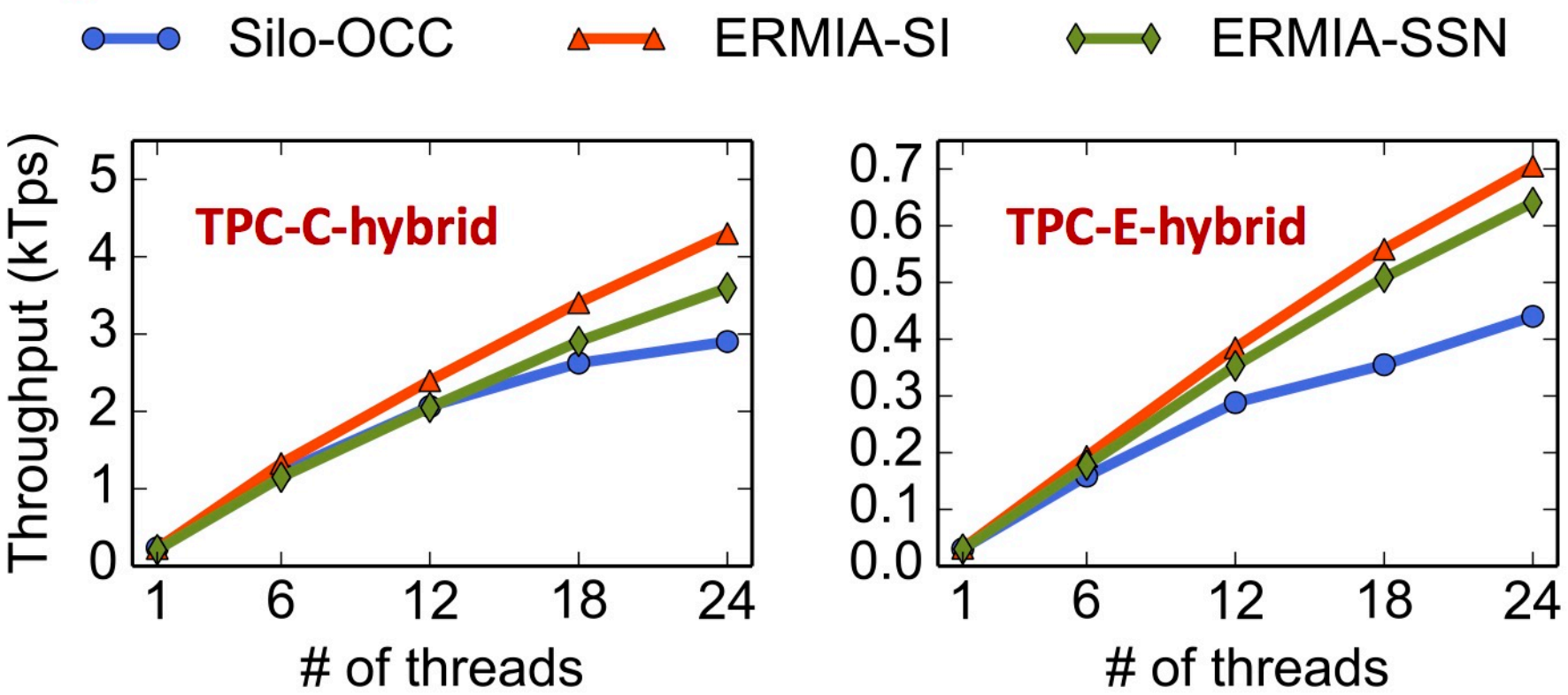
***Higher is better***




**Comparable performance to OCC**

# Robust heterogeneous performance

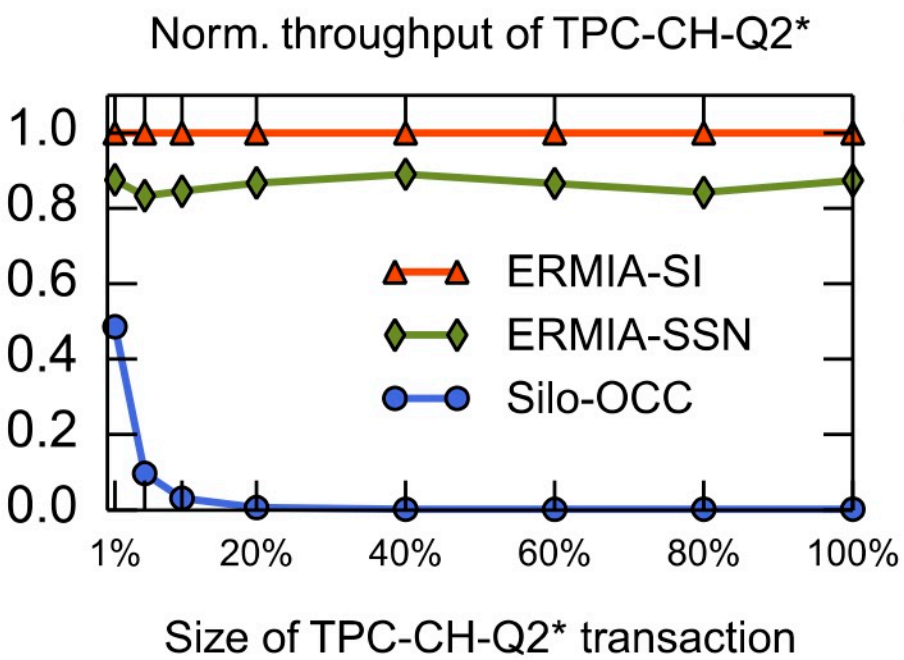
***Higher is better***



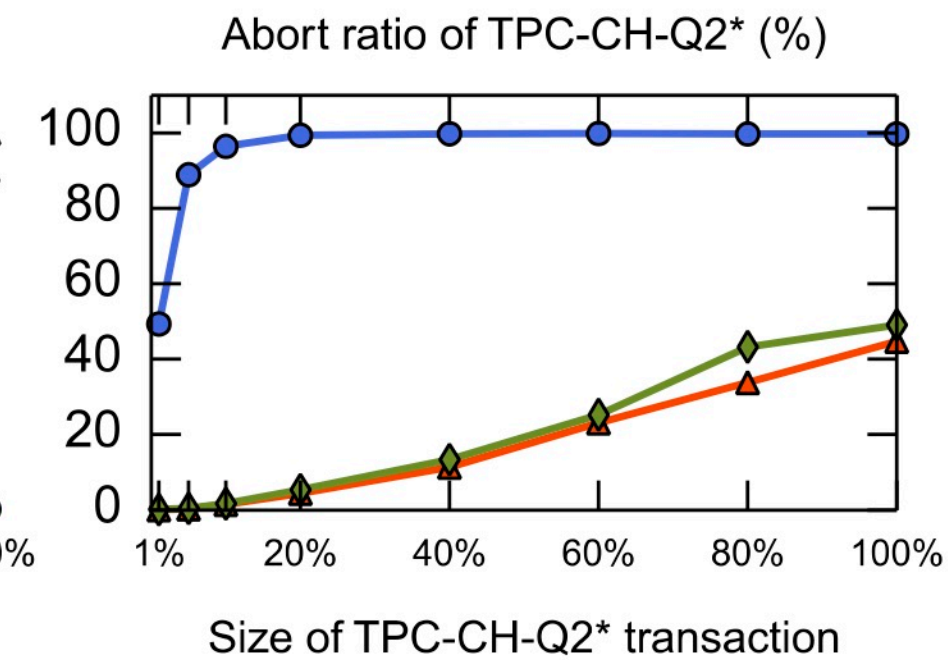

**(Much) better overall throughput**

# Robust heterogeneous performance

***Higher is better***




***Lower is better***




**(Much) lower abort rate for read-mostly tx**

# Conclusion

- The ERMIA provides
- Reasonable performance on traditional workflow.
- Better performance on read-most workflow compare to using lightweight OCC.