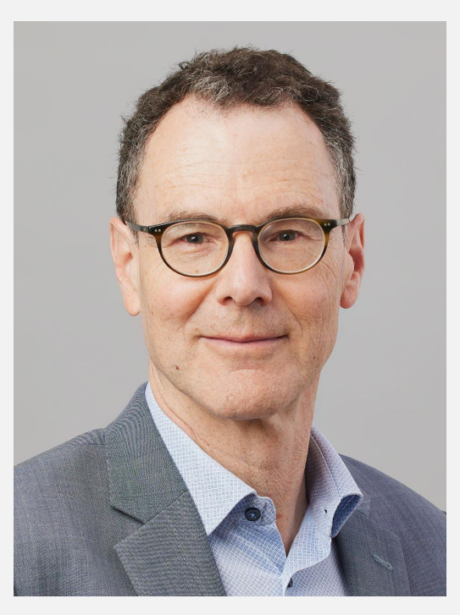
ETH zürich

Department of Computer Science
Information Security Group

Pushing the Limit: Verified Performance-Optimal Causally-Consistent Database Transactions



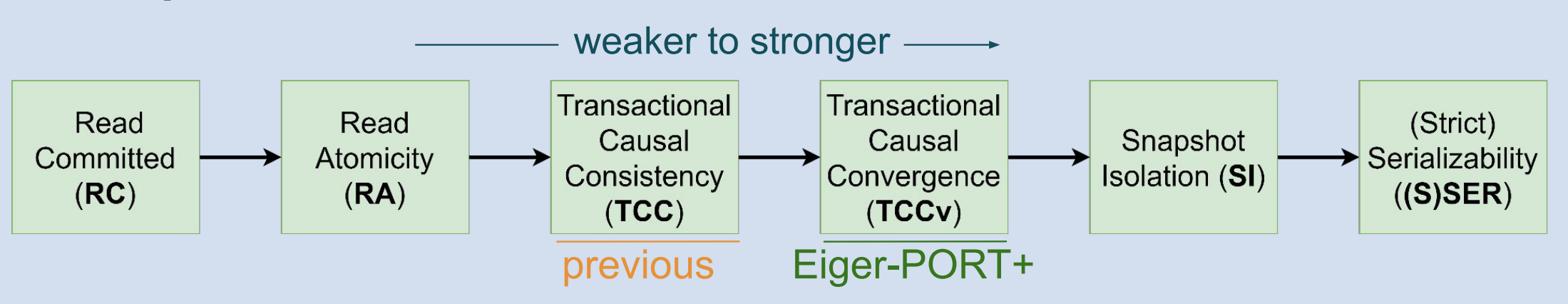
Shabnam Ghasemirad · Dr. Christoph Sprenger · Dr. Si Liu · Luca Multazzu · Prof. Dr. David Basin

shabnam.ghasemirad@inf.ethz.ch

Project repository:

1 Motivation

- Distributed databases (key-value stores)
- **Isolation** of concurrent transactions, realized by concurrency control protocols.
- Spectrum of isolation levels:


weaker to stronger
Read Committed (RC)
Read Atomicity (RA)
Transactional Causal Consistency (TCC)
Transactional Causal Convergence (TCCv)
Snapshot Isolation (SI)
(Strict) Serializability ((S)SER)
previous
Eiger-PORT+

- Trade-off: isolation vs performance
- TCC previously conjectured to be the strongest achievable isolation level for performance-optimal reads in the presence of transactional writes.
- We **refute** the **conjecture** and push the limit to **TCCv** with our novel protocol **Eiger-PORT+.**
- Concurrency control protocols are highly complex and prone to design errors and isolation bugs. → Deductive **verification**

Contributions

- ❖ **Eiger-PORT+**
  - ➢ *Stronger isolation* guarantee
  - ➢ *Superior performance*
- ❖ **Protocol verification** in Isabelle/HOL
  - ➢ Refinement
  - ➢ Reduction

2 Abstract Model: Isolation Level

We specify isolation as an abstract model parameterized by an Isolation Level (IL).

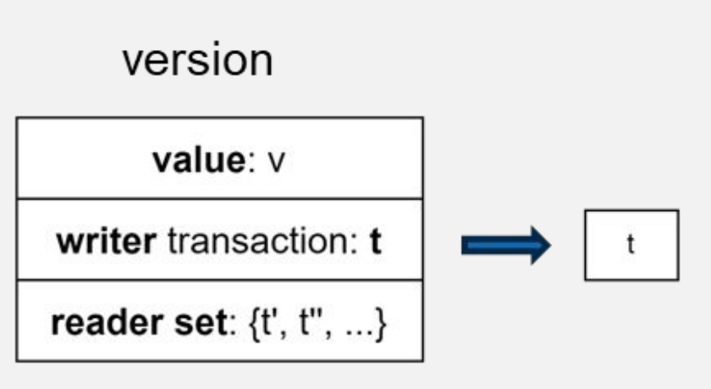
The abstract model's event system:

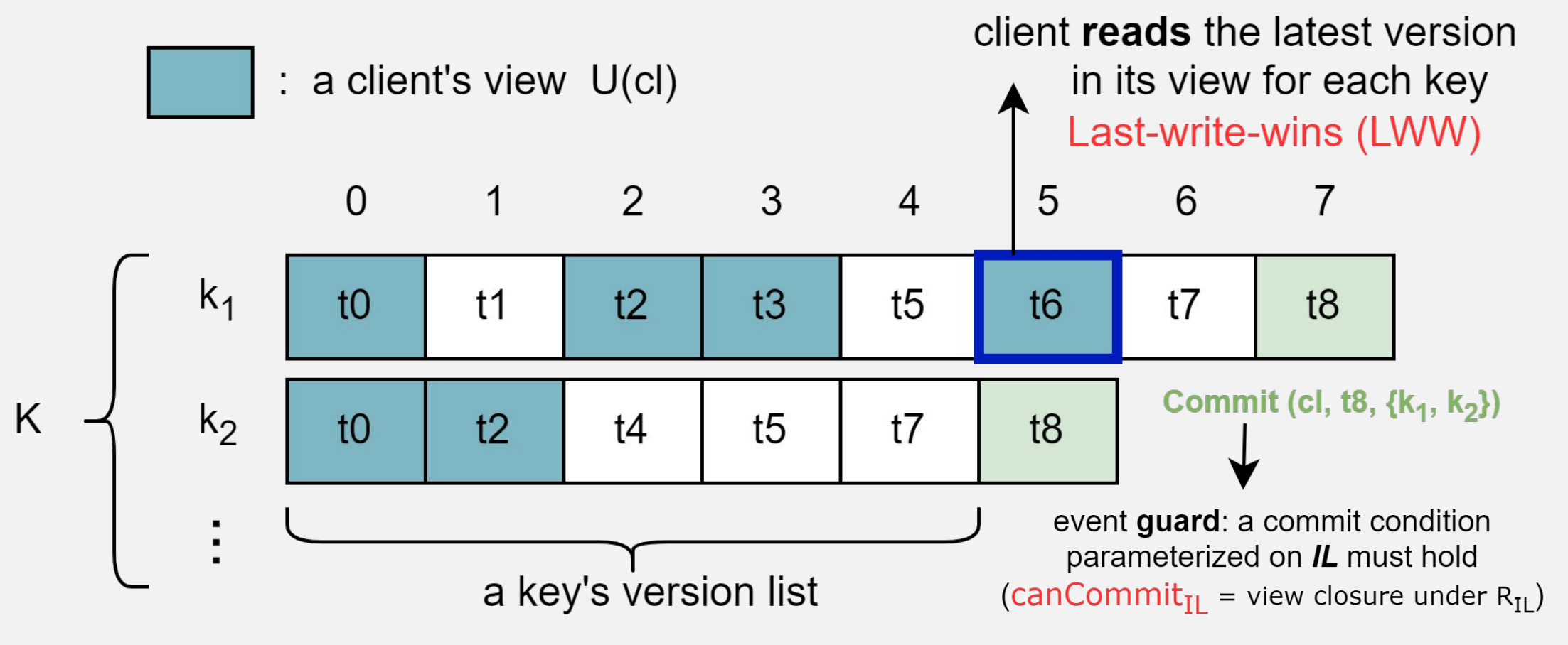
➔ States: (K, U) { **K**: centralized multi-versioned key-value store; **U**: client views, capturing the distributed aspect

➔ Events:

1) Atomic Commit
$(K, U) \xrightarrow{\text{commit}}_{IL} (K', U')$

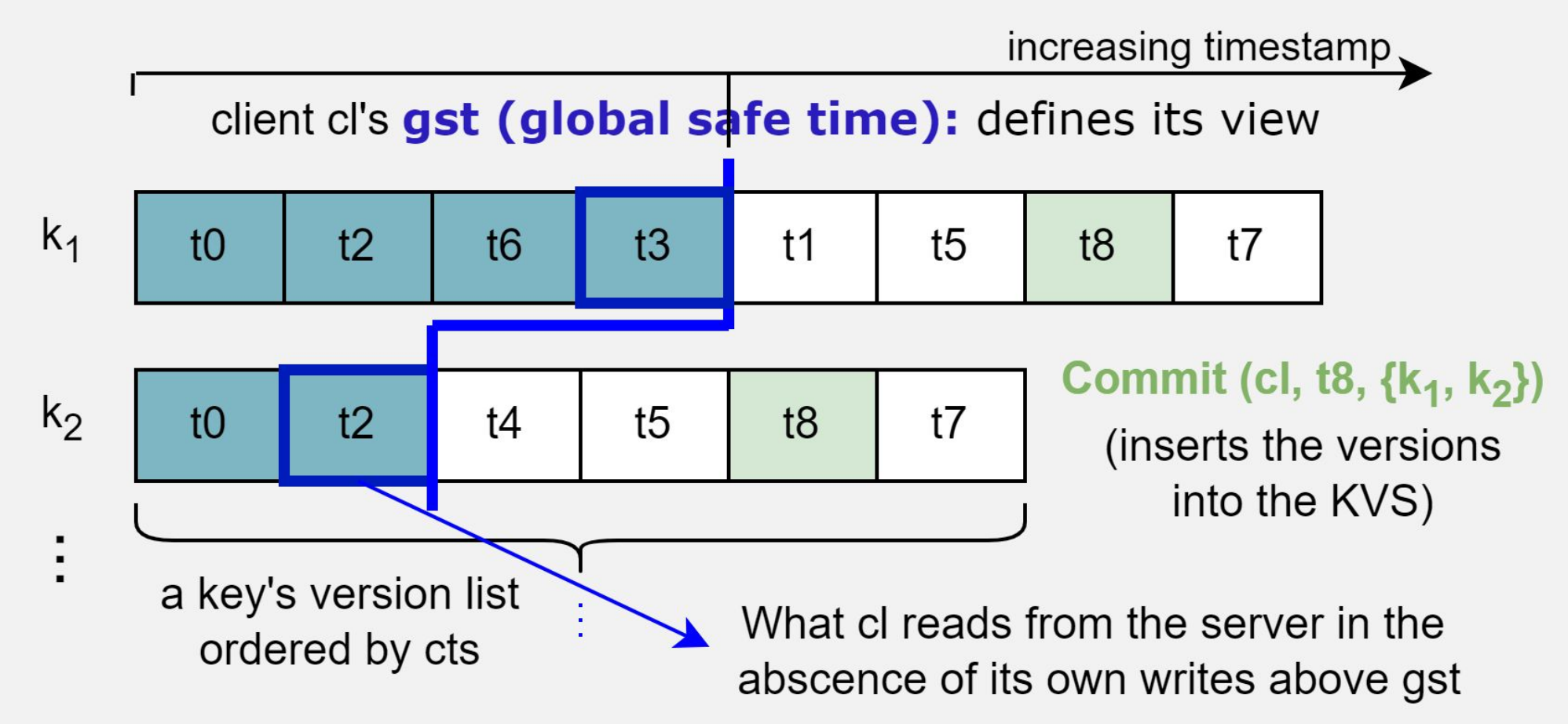
2) View Extension
$(K, U) \xrightarrow{\text{view ext}}_{IL} (K, U')$




: a client's view U(cl)
client reads the latest version in its view for each key
Last-write-wins (LWW)
0 1 2 3 4 5 6 7
K
k1
t0 t1 t2 t3 t5 t6 t7 t8
k2
t0 t2 t4 t5 t7 t8
Commit (cl, t8, {k1, k2})
a key's version list
event guard: a commit condition parameterized on IL must hold
(canCommitIL = view closure under RIL)

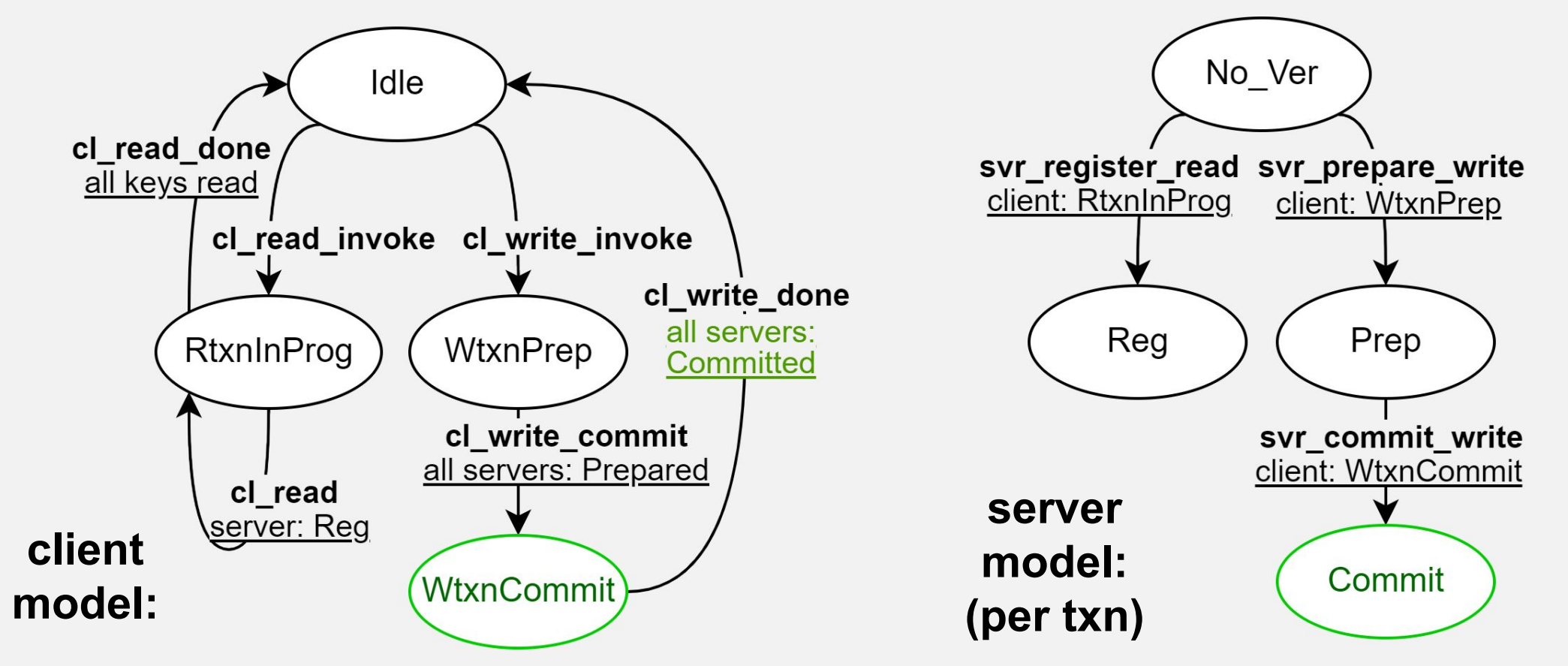
3 Concrete Model: Eiger-PORT+

Eiger-PORT+ protocol:
Timestamp-based (uses Lamport clocks)


increasing timestamp
client cl's gst (global safe time): defines its view
k1
t0 t2 t6 t3 t1 t5 t8 t7
k2
t0 t2 t4 t5 t8 t7
Commit (cl, t8, {k1, k2})
(inserts the versions into the KVS)
a key's version list ordered by cts
What cl reads from the server in the abscence of its own writes above gst

- read-only transactions
  → own write above gst
  → or last write below gst

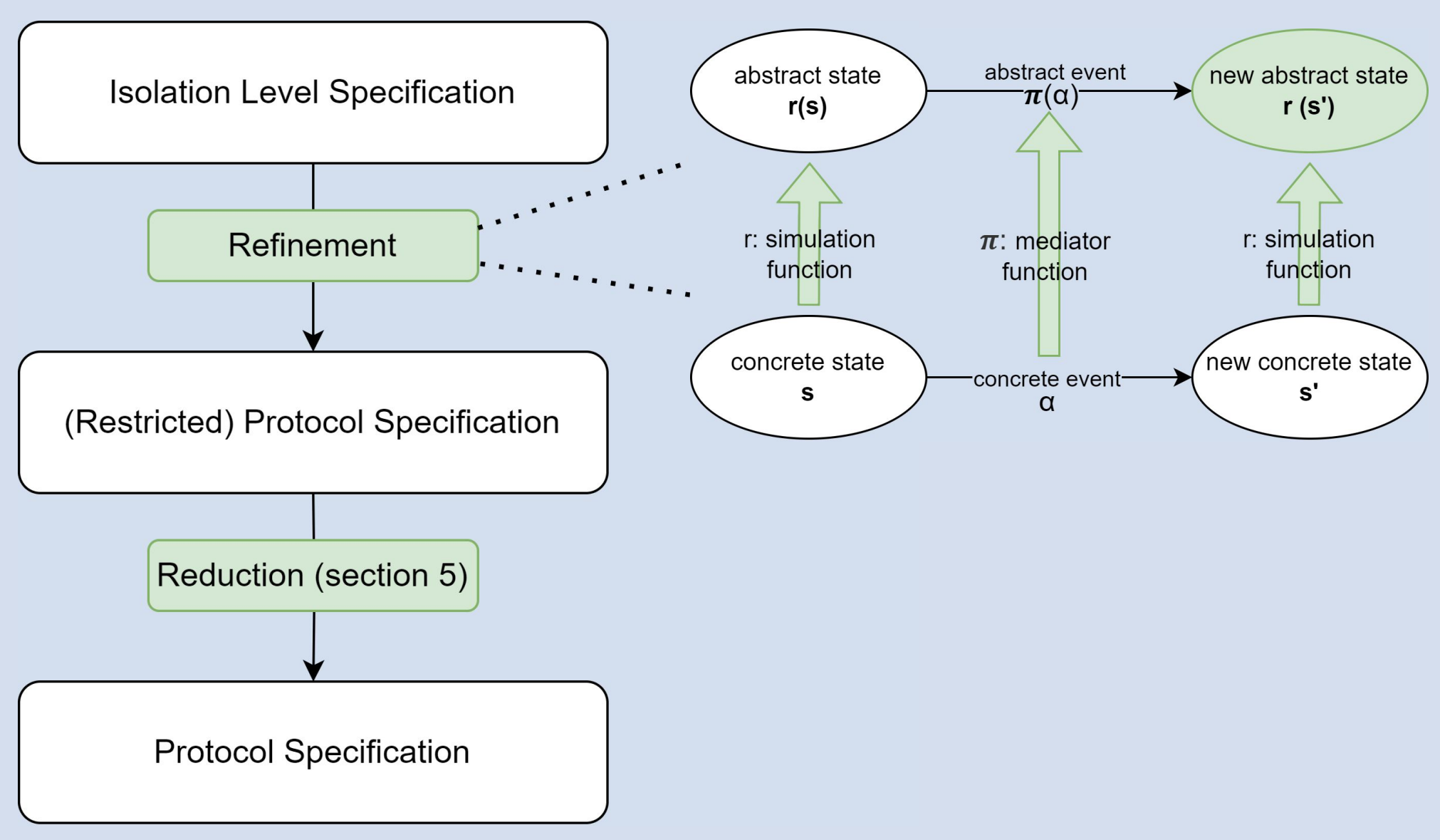
- write-only transactions: two-phase commit



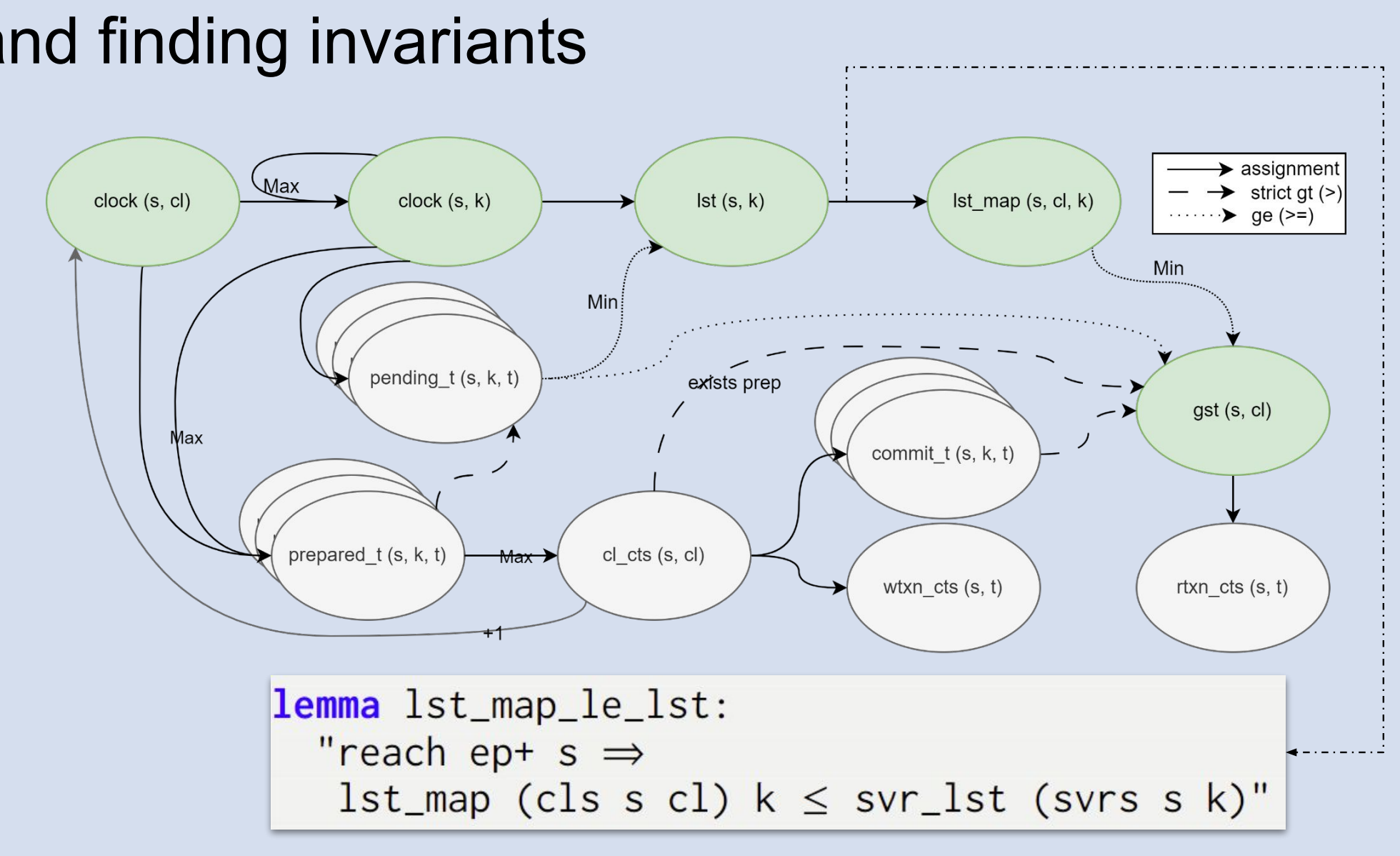
4 Correctness Proof & Invariants

Proof guarantee:
refinement (reach (protocol)) ⊆ reach (IL)

- refinement mapping:
  r : K and U reconstructed as shown above
  π: client_write_commit and client_read_done mapped to *Atomic commit*
- proof obligations:
  canCommit_{IL}: needs invariants (below)
  LWW: needs reduction



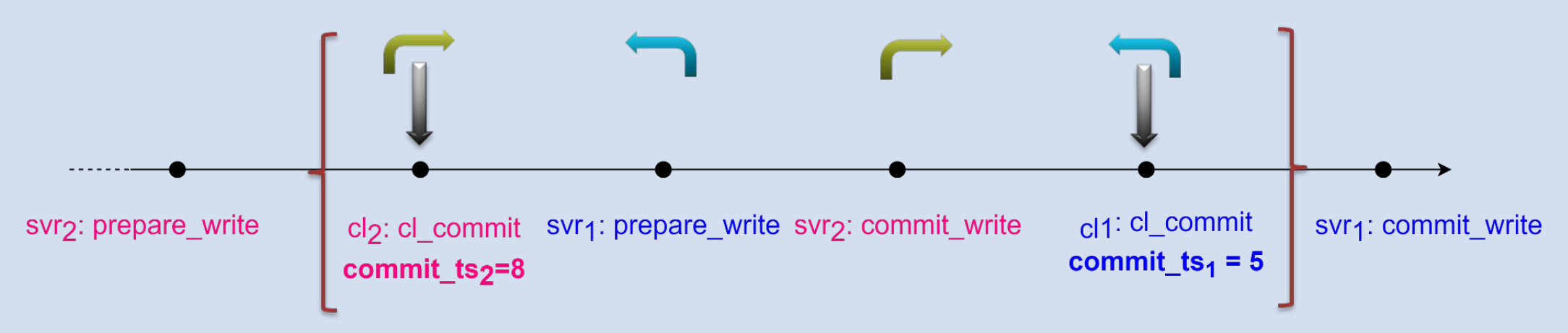
- The relation of different timestamps in the model and finding invariants



```
lemma lst_map_le_lst:
  "reach ep+ s ⟹
   lst_map (cls s cl) k ≤ svr_lst (svrs s k)"
```

5 Inverted Commits & Reduction

Inverted commits: pairs of client commits in protocol executions not ordered by **commit timestamps**.



Can occur for causally independent concurrent transactions.

Problem: Inverted commits would require ***inserting*** rather than ***appending*** a transaction's version to the version list.

Can not be simulated by the abstract model.

Hence, refinement alone is not enough for verifying the protocol.

Solution:

- We introduce a restricted protocol model that doesn't produce inverted commits.

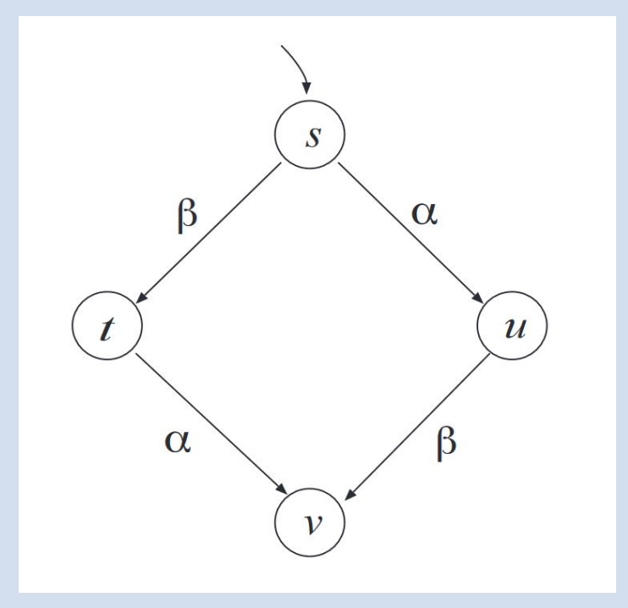
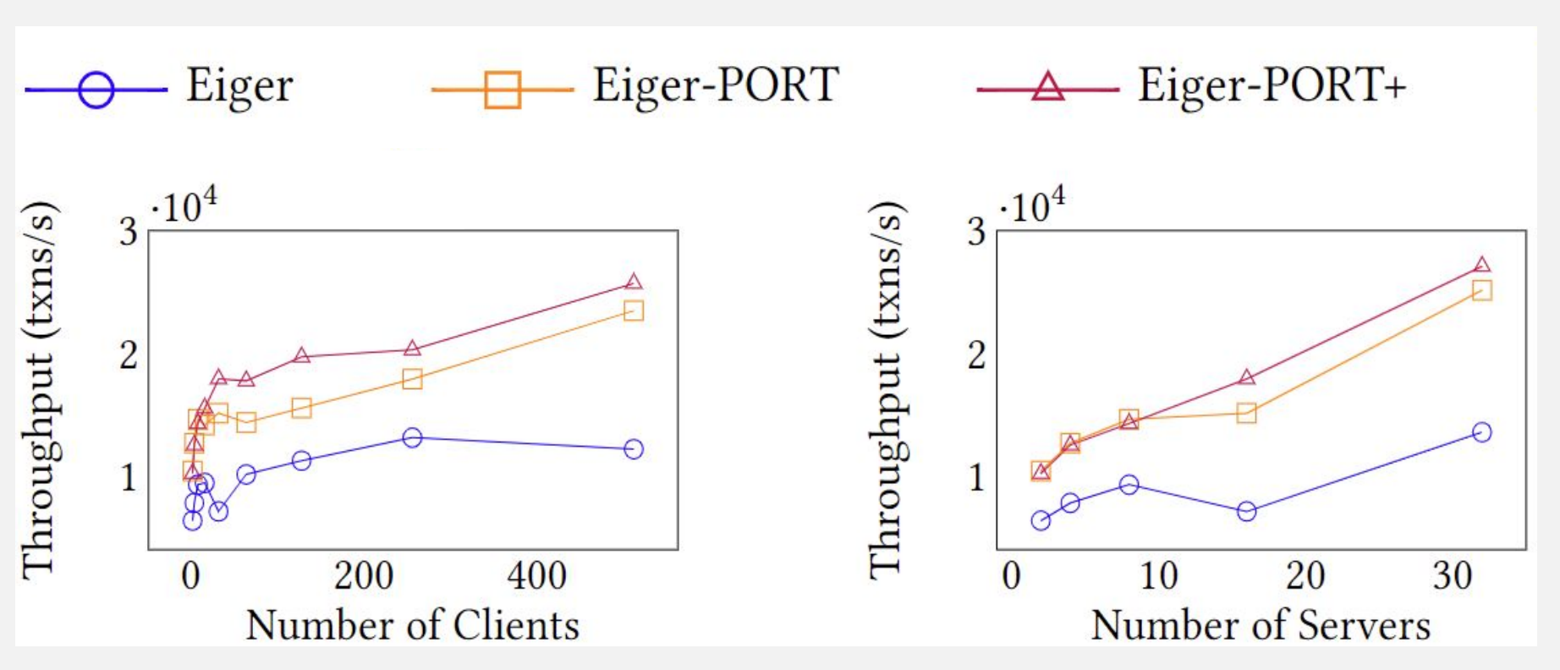


Image source: C. Baier, J. Katoen. 2008. Principles of model checking, p. 595, Figure 8.1.

- We use **reduction** to transform any protocol execution into one of the restricted model such that:
  reach (protocol) = reach (restricted protocol).
- This is achieved by commuting independent concurrent events to eliminate inverted commits. (see arrows on the execution above)

6 Conclusions and Discussion

- Our Eiger-PORT+ protocol provides TCCv, thus refuting an open conjecture.
- Eiger-PORT+ outperforms state-of-the-art



- Refinement is not always enough
- We deductively verify that Eiger-PORT+ satisfies TCCv, using a combination of refinement and reduction.