

# CSE 444: Database Internals

## Lecture 25 Replication

# References

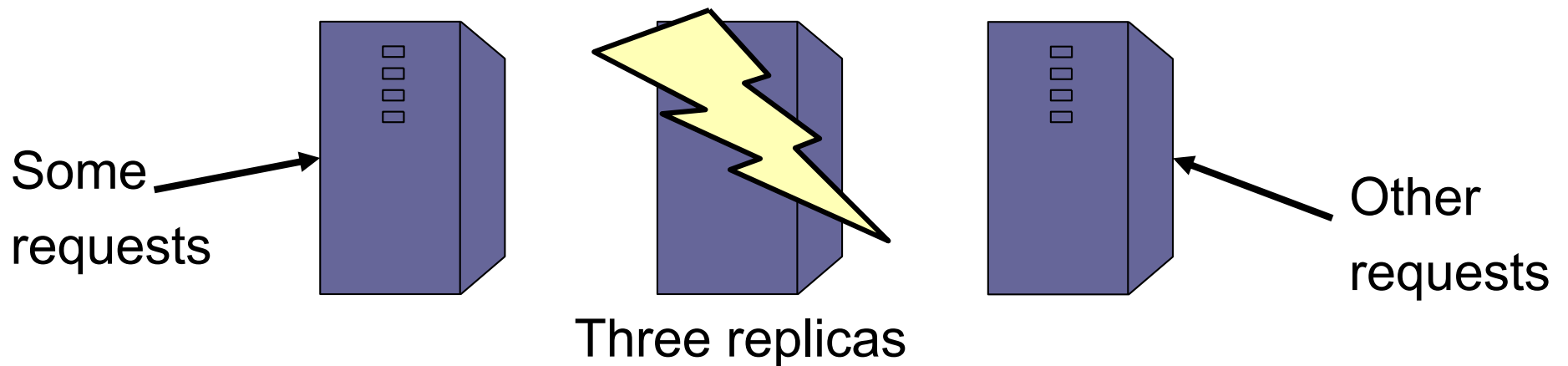
- Ullman Book Chapter 20.6
- **Database management systems.**  
Ramakrishnan and Gehrke.  
Third Ed. **Chapter 22.11**

# Outline

- Goals of replication
- Three types of replication
  - Eager replication
  - Lazy replication
  - Two-tier replication

# Goals of Replication

- Goal 1: availability
- Goal 2: performance



- But, it's easy to build a replicated system that reduces performance and availability

# Synchronous Replication

- Also called **eager replication**
- All updates are applied to all replicas (or to a majority) as part of a single transaction (need two phase commit)
- Main goal: as if there was only one copy
  - Maintain **consistency**
  - Maintain **one-copy serializability**
  - I.e., execution of transactions has same effect as an execution on a non-replicated db
- Transactions must acquire **global locks**

# Synchronous Master Replication

- One master for each object holds primary copy
  - The “Master” is also called “Primary”
  - To update object, transaction must acquire a lock at the master
  - Lock at the master is global lock
- Master propagates updates to replicas synchronously
  - Updates propagate as part of the same distributed transaction
    - Need to run 2PC at the end
  - For example, using triggers

# Crash Failures

- What happens when a secondary crashes?
  - Nothing happens
  - When secondary recovers, it catches up
- What happens when the master/primary fails?
  - Blocking would hurt availability
  - Must choose a new primary: run election

# Network Failures

- Network failures can cause trouble...
  - Secondaries think that primary failed
  - Secondaries elect a new primary
  - But primary can still be running
  - Now have two primaries!



# Majority Consensus

- To avoid problem, only majority partition can continue processing at any time
- In general,
  - Whenever a replica fails or recovers...
  - a set of communicating replicas must determine...
  - whether they have a majority before they can continue

# Synchronous Group Replication

- **With  $n$  copies**
  - Exclusive lock on  $x$  copies is global exclusive lock
  - Shared lock on  $s$  copies is global shared lock
  - Must have:  $2x > n$  and  $s + x > n$
- **Majority locking**
  - $s = x = \lceil (n+1)/2 \rceil$
  - No need to run any reconfiguration algorithms
- **Read-locks-one, write-locks-all**
  - $s=1$  and  $x = n$ , high read performance
  - Need to make sure algo runs on quorum of computers

# Synchronous Replication Properties

- Favours **consistency** over availability
  - Only majority partition can process requests
  - There appears to be a single copy of the db
- **High runtime overhead**
  - Must lock and update at least majority of replicas
  - Two-phase commit
  - Runs at pace of slowest replica in quorum
  - So overall system is now slower
  - Higher deadlock rate (transactions take longer)

# Asynchronous Replication

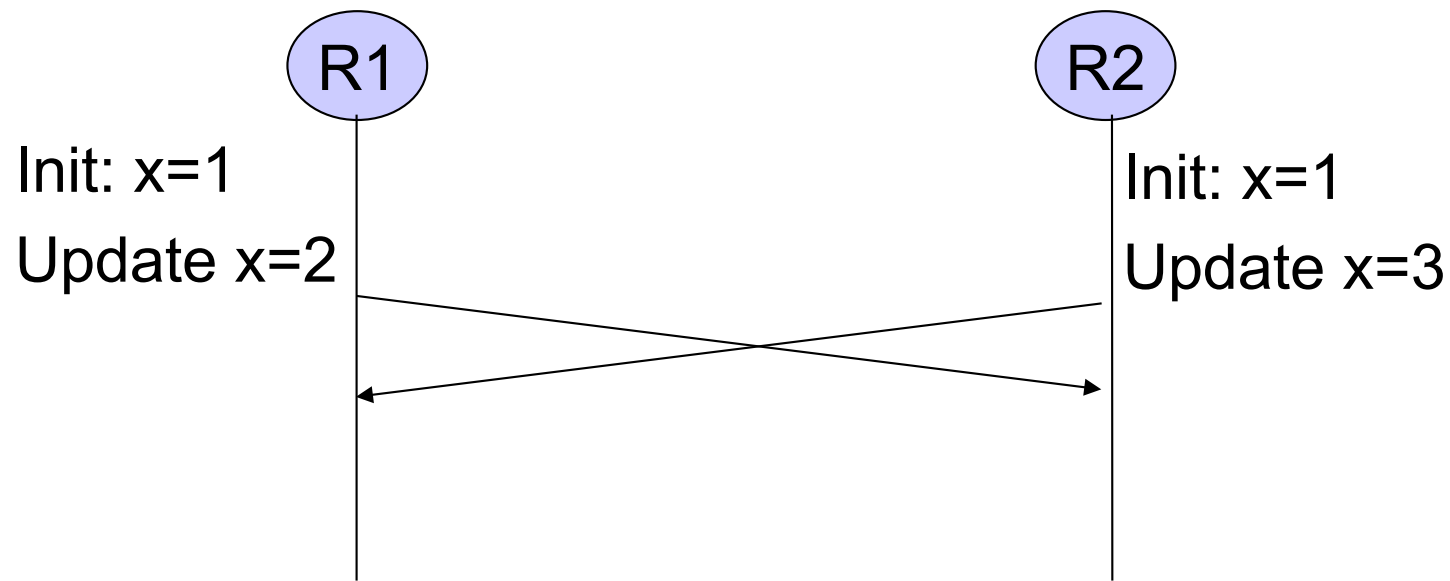
- Also called **lazy replication**
- Also called **optimistic replication**
- Main goals: availability and performance
- Approach
  - One replica updated by original transaction
  - Updates propagate asynchronously to other replicas

# Asynchronous Master Replication

- **One master holds primary copy**
  - Transactions update primary copy
  - Master asynchronously propagates updates to replicas, which process them in same order (e.g. through log shipping)
  - Ensures single-copy serializability
- **What happens when master/primary fails?**
  - Can lose most recent transactions when primary fails!
  - After electing a new primary, secondaries must agree who is most up-to-date

# Asynchronous Group Replication

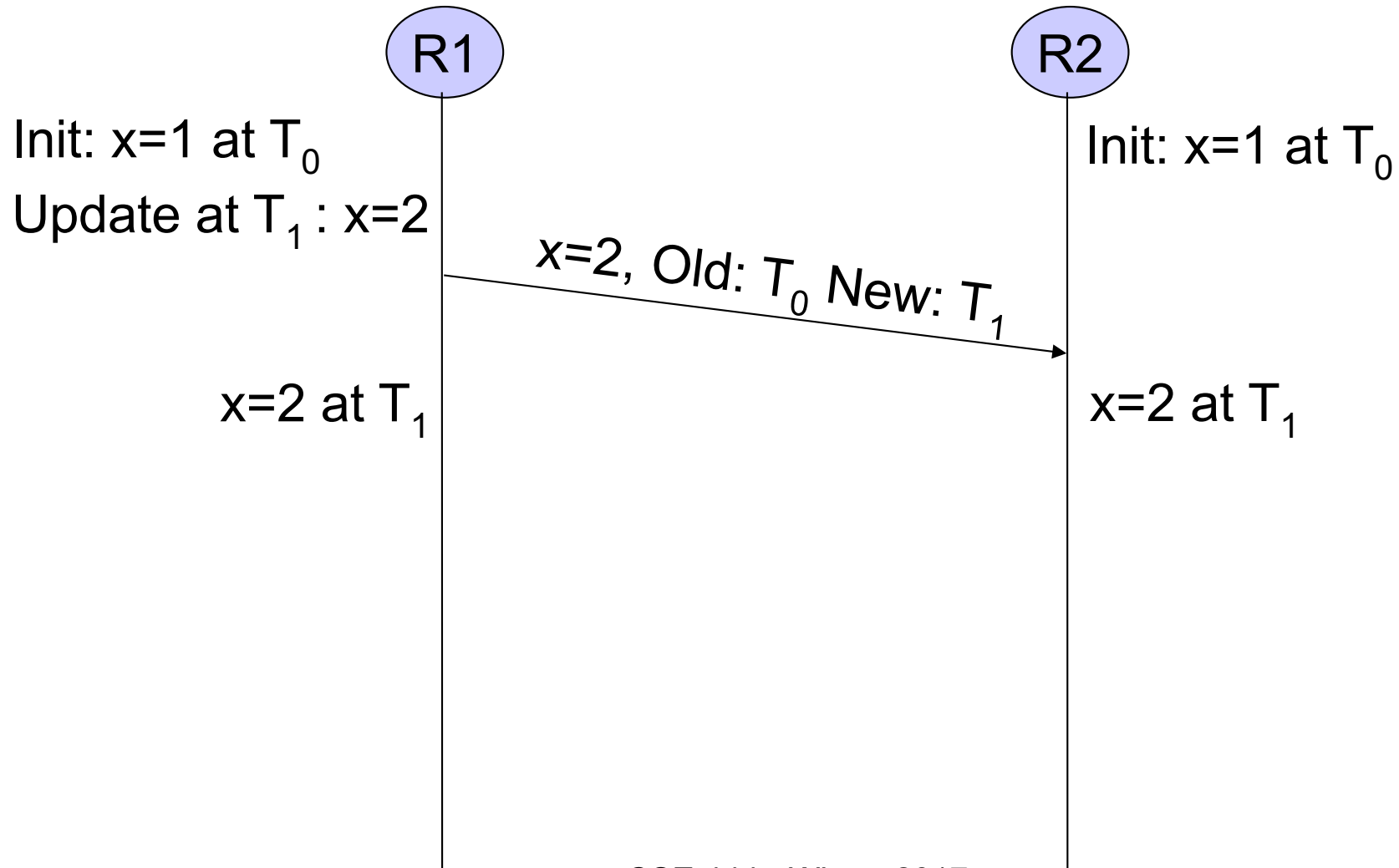
- Also called **multi-master**
- Best scheme for availability
- **Cannot guarantee one-copy serializability!**



# Asynchronous Group Replication

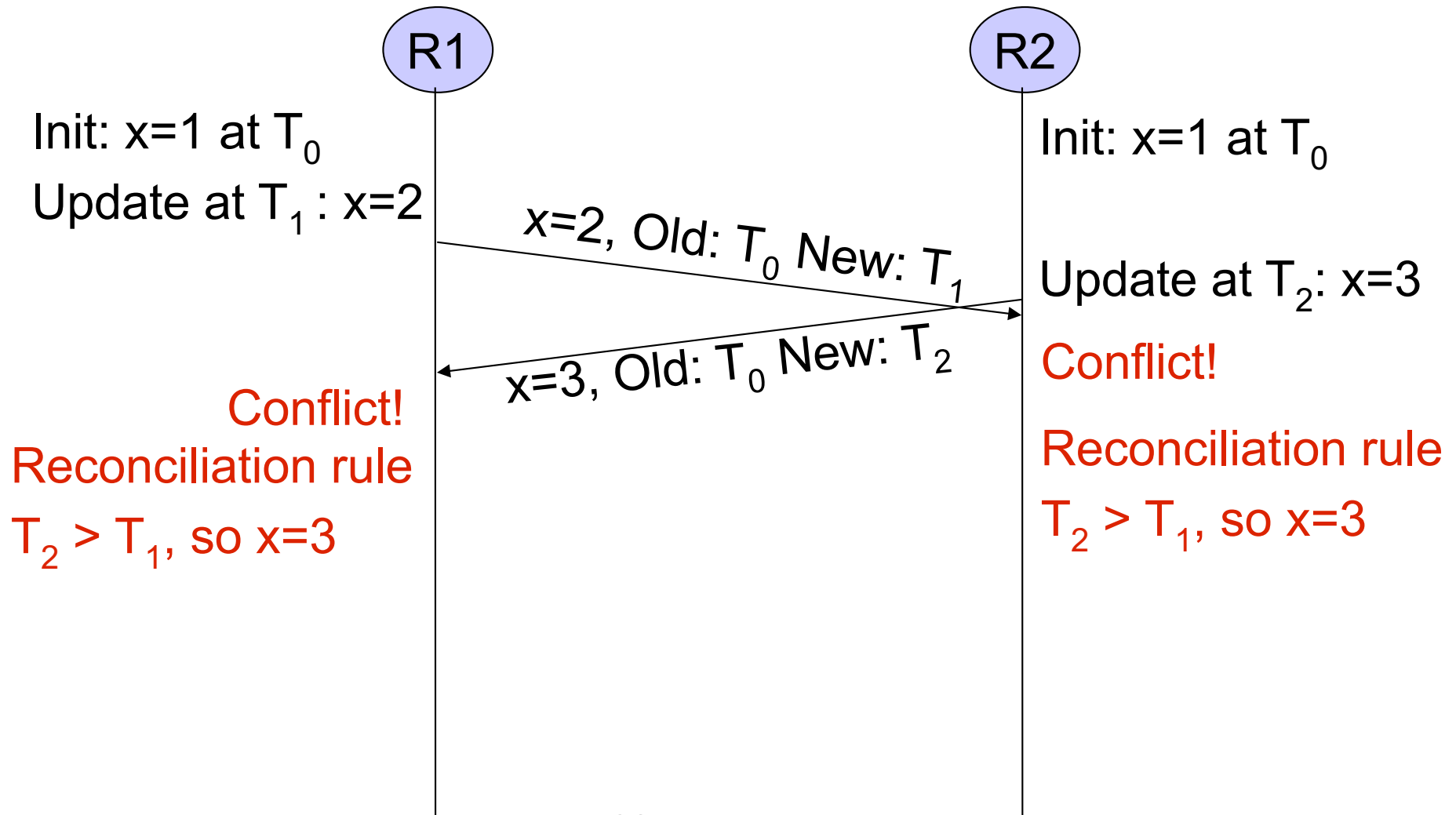
- **Cannot guarantee one-copy serializability!**
- **Instead guarantee convergence**
  - Db state does not reflect any serial execution
  - But all replicas have the same state
- **Detect conflicts and reconcile replica states**
- **Different reconciliation techniques are possible**
  - Manual
  - Most recent timestamp wins
  - Site A wins over site B
  - User-defined rules, etc.

# Detecting Conflicts Using Timestamps





# Detecting Conflicts Using Timestamps



# Asynchronous Group Replication Properties

- Favours **availability** over consistency
  - Can read and update any replica
  - High runtime performance
- **Weak consistency**
  - Conflicts and reconciliation

# Two-Tier Replication

- Benefits of lazy master and lazy group
- Each object has a master with primary copy
- When disconnected from master
  - Secondary can only run **tentative transactions**
- When reconnects to master
  - Master reprocesses all tentative transactions
  - Checks an acceptance criterion
  - If passes, we now have **final commit order**
  - Secondary **undoes tentative and redoes committed**

# Conclusion

- Replication is a very important problem
  - Fault-tolerance (various forms of replication)
  - Caching (lazy master)
  - Warehousing (lazy master)
  - Mobility (two-tier techniques)
- Replication is complex, but basic techniques and trade-offs are **very well known**
  - Synchronous or asynchronous replication
  - Master or quorum