MDCC: multi-data center consistency

Authors: Tim Kraska, Gene Pang, Michael J. Franklin, Samuel Madden, Alan Fekete

Presenters: Darshan Acharya, Goodness Ayinmode, Haneul Lee, Viral Bhadeshiya

Data Center Failures

- Examples
  - June 29, 2009:
    Rackspace power outage of “approximately 40 minutes”
  - April 21, 2011:
    AWS East outage of over 2 hours, with data loss

- Unreliable

Geo-Replication

Problems with Geo-Replication

High Latency

Unreliable Data Centers

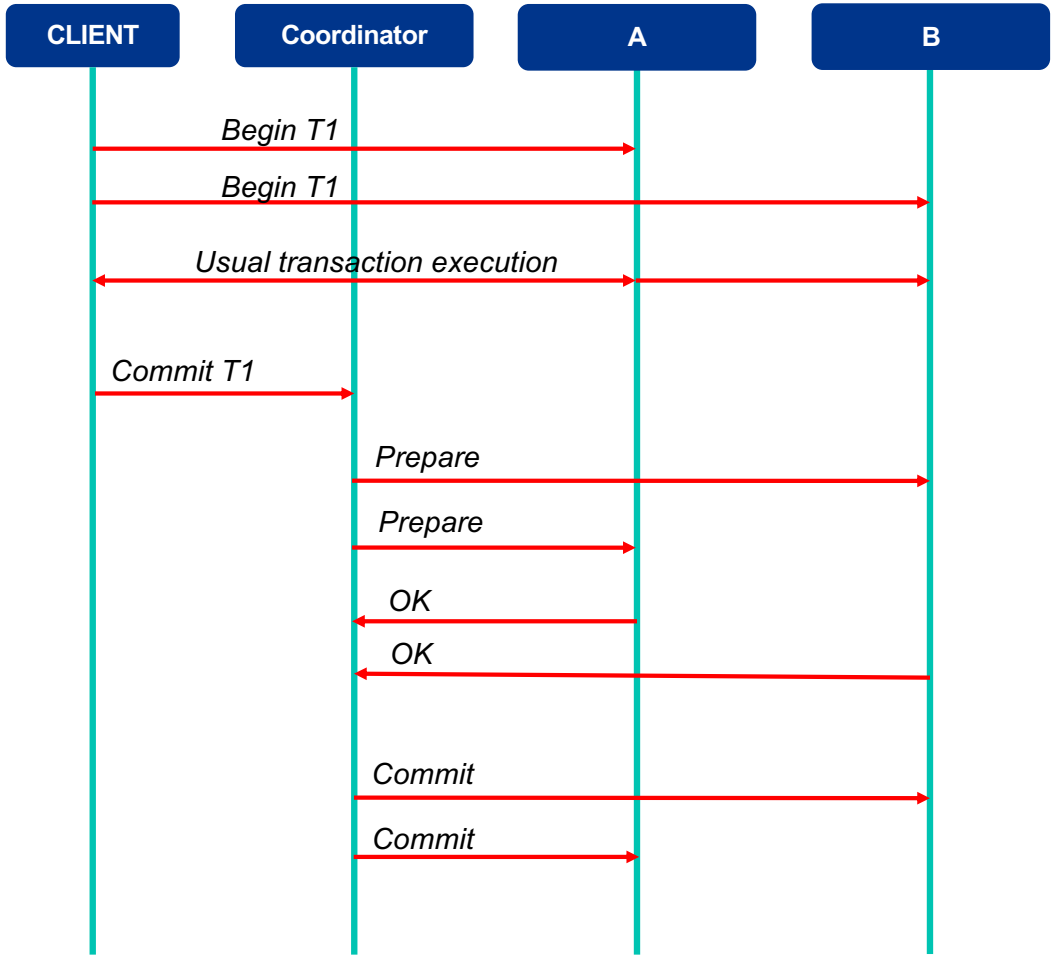
Multi-Data Center Consistency (MDCC)

Optimistic Distributed Commit Protocol

Low Latency

Strong Consistency

Two-Phase Commit



Two-Phase Commit(2PC)

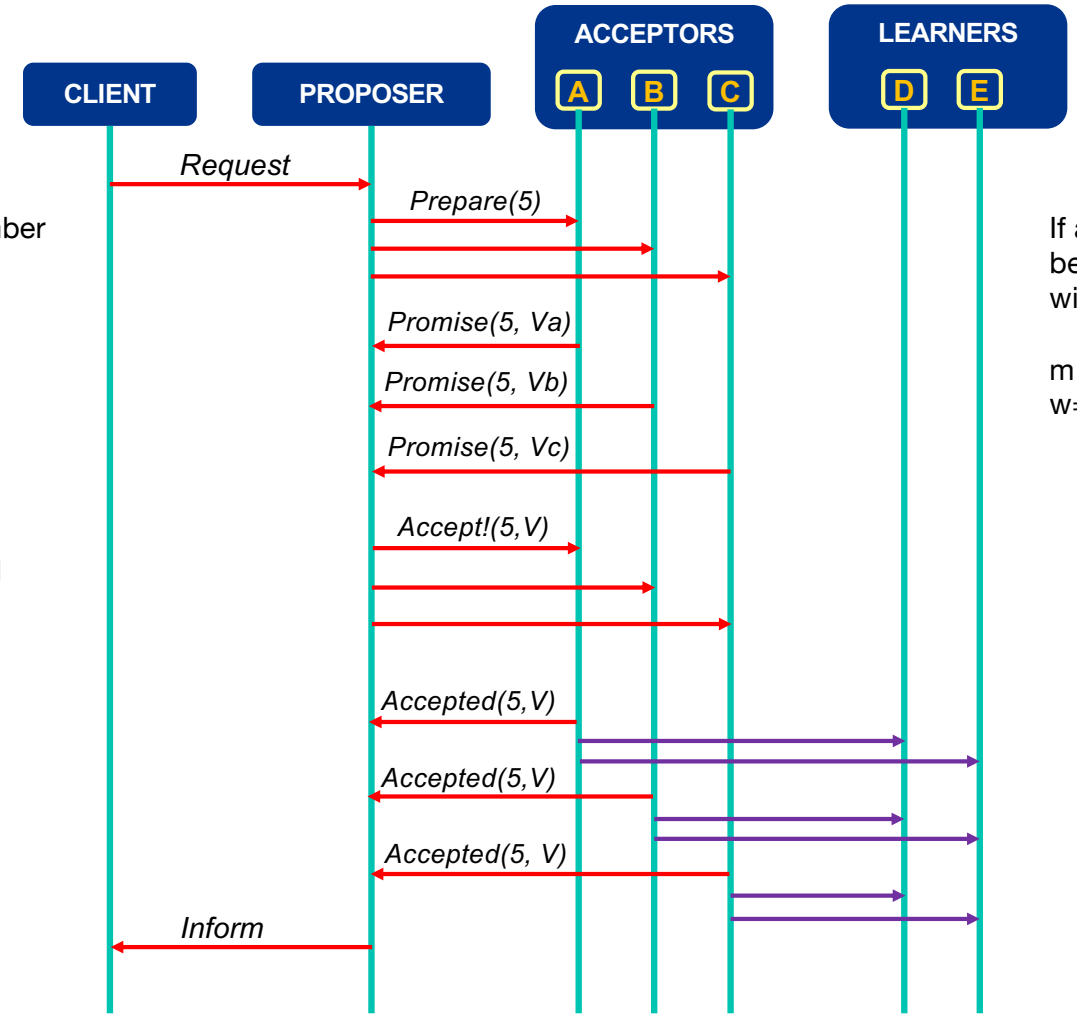
- Why 2PC doesn't work well in geo-replicated networks?
  - Coordinator waits for responses from all the participants
  - Coordinator becomes a Single Point of Failure

Paxos

Paxos

- MDCC optimizes paxos
  - Classic Paxos
  - Multi Paxos
  - Fast Paxos
  - Generalized Paxos

Classic Paxos



Proposer Selects 5 as Ballot Number

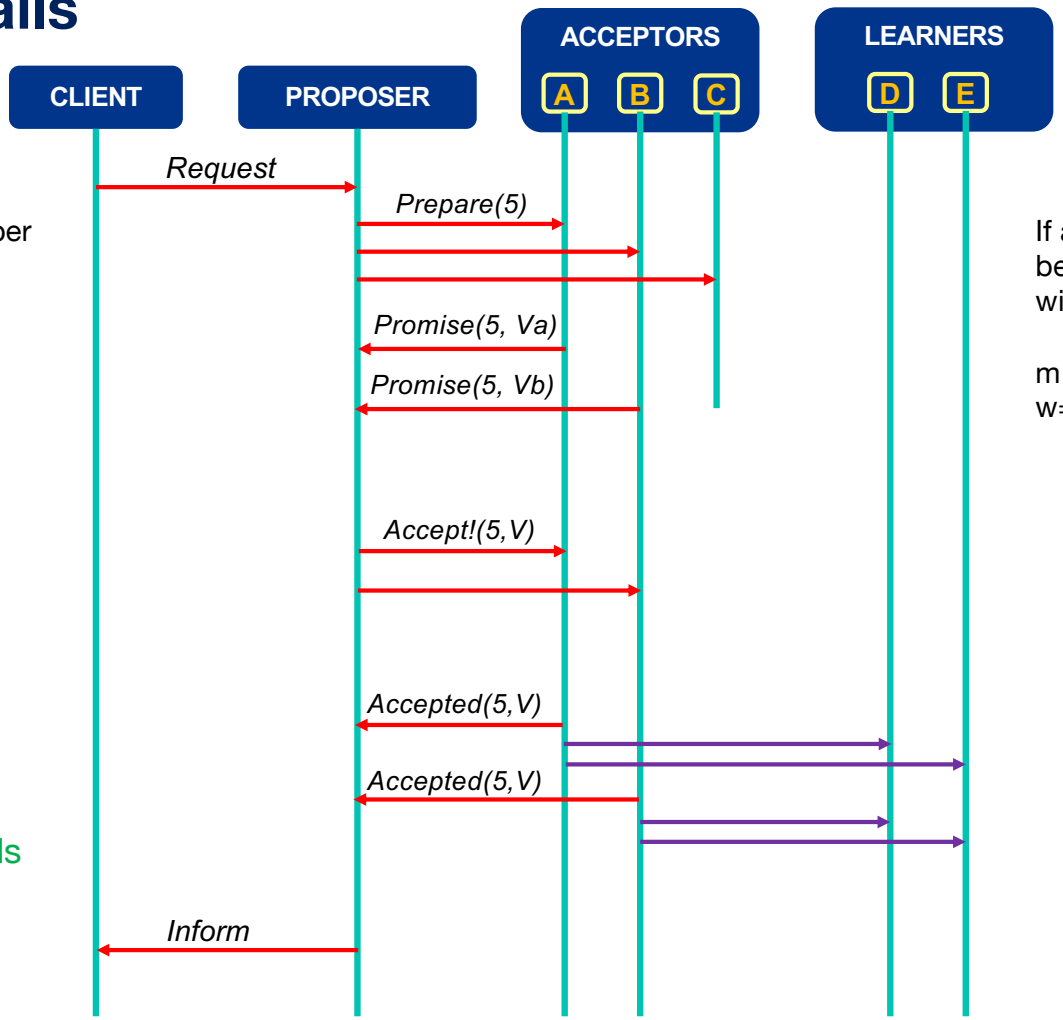
Va, Vb, Vc = null, if acceptors accept ballot number 5

Accept!(5, w) or Accept!(5, x)
x = original value to be proposed

If a previous proposal has been accepted, they reply with Promise(m,w)

m = Prev Ballot Number
w= Accepted Value

When Acceptor Fails



Proposer Selects 5 as Ballot Number

Va, Vb = null, if acceptors accept ballot number 5

Accept!(5, w) or Accept!(5, x)
x = original value to be proposed

Paxos Still Succeeds

If a previous proposal has been accepted, they reply with Promise(m,w)

m = Prev Ballot Number
w= Accepted Value

Problems of Classic Paxos

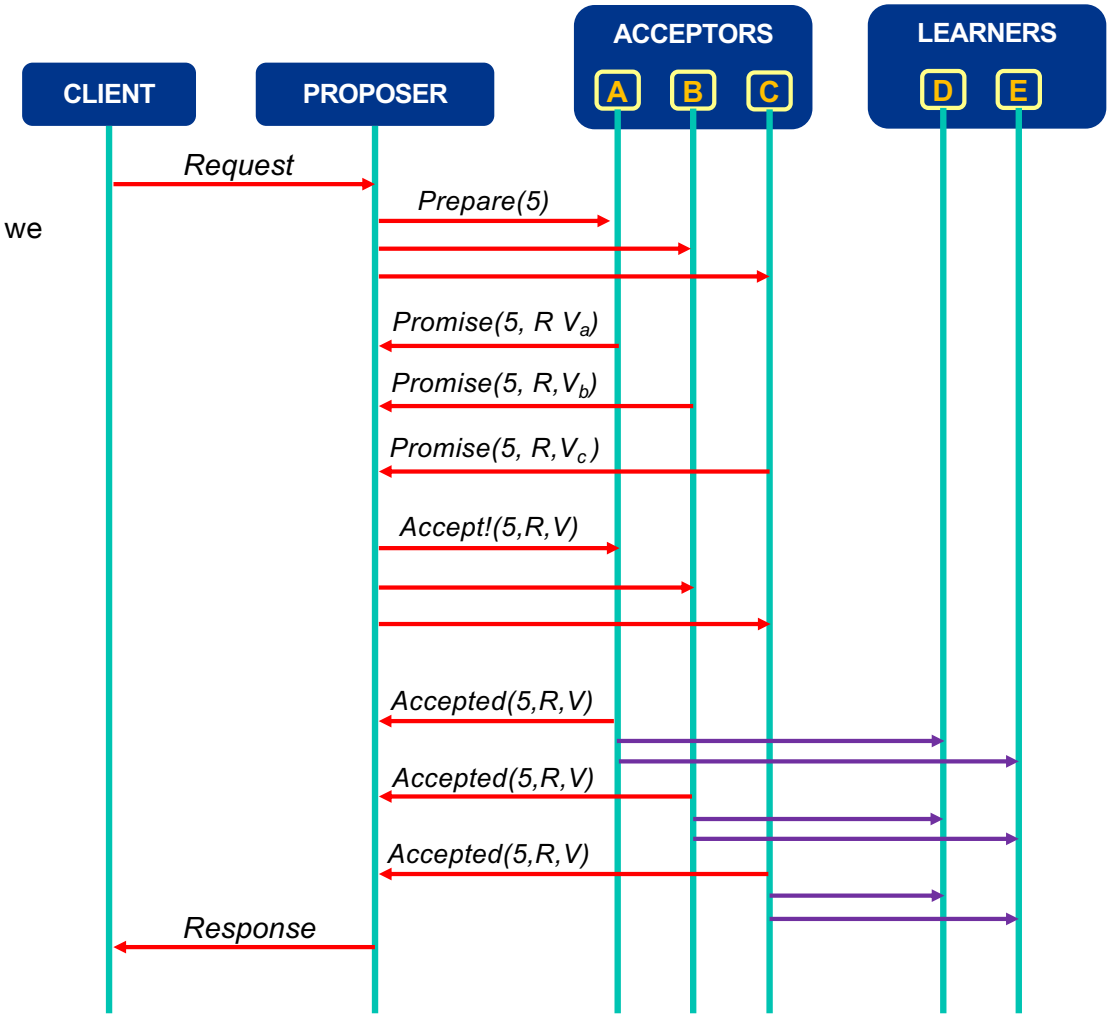
Requires two phases to reach a consensus on a single output value

Latency in the event of a leader failure

Multi Paxos

Same Idea as Classic but now we are keeping track of instances Consensus Rounds (R)

Va, Vb, Vc = null, if acceptors accept ballot number 5



UCDAVIS

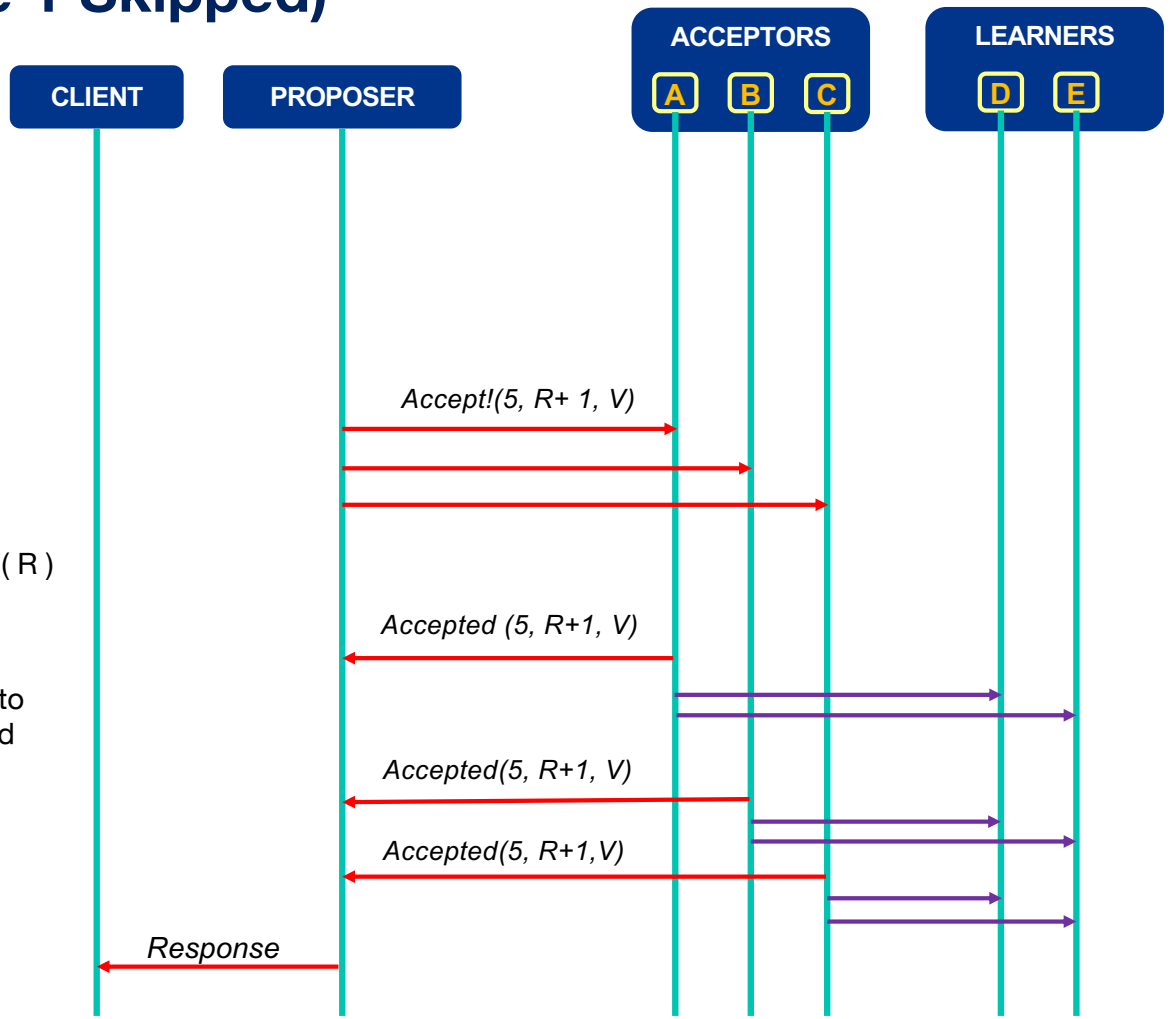
Multi Paxos (Phase 1 Skipped)

Proposer can be selected to lead several consensus rounds, no need for phase 1.

Round numbers are used to identify and track each round of the protocol

Leader increments round number (R) for each proposal

Acceptors use the round number to differentiate between proposal and determine order



Problems of Multi Paxos

- Dependency on master

- Does not support transactions with multiple updates in one round-trip

Write-Write Conflict

Transaction 1	A	Transaction 2	A
Read(A)	10		
A = A+5	15		
		Read(A)	10
		A = A-5	5
		Write(A)	5
Write(A)	15		

Write-Write Conflict

Transaction 1	A	Transaction 2	A
Lock(A)			
Read(A)	10		
A = A+5	15		
		~~Read(A)~~	~~10~~
		~~A = A-5~~ **Wait**	~~5~~
		~~Write(A)~~	~~5~~
Write(A)	15		
Unlock(A)			

Deadlock

Transaction 1	Transaction 2
Lock(A)	
Read(A)	
	Lock(B)
	Read(A)
Write(B)	
Wait unlock(B)	
	Write(A)
	Wait unlock(A)

Transaction Support

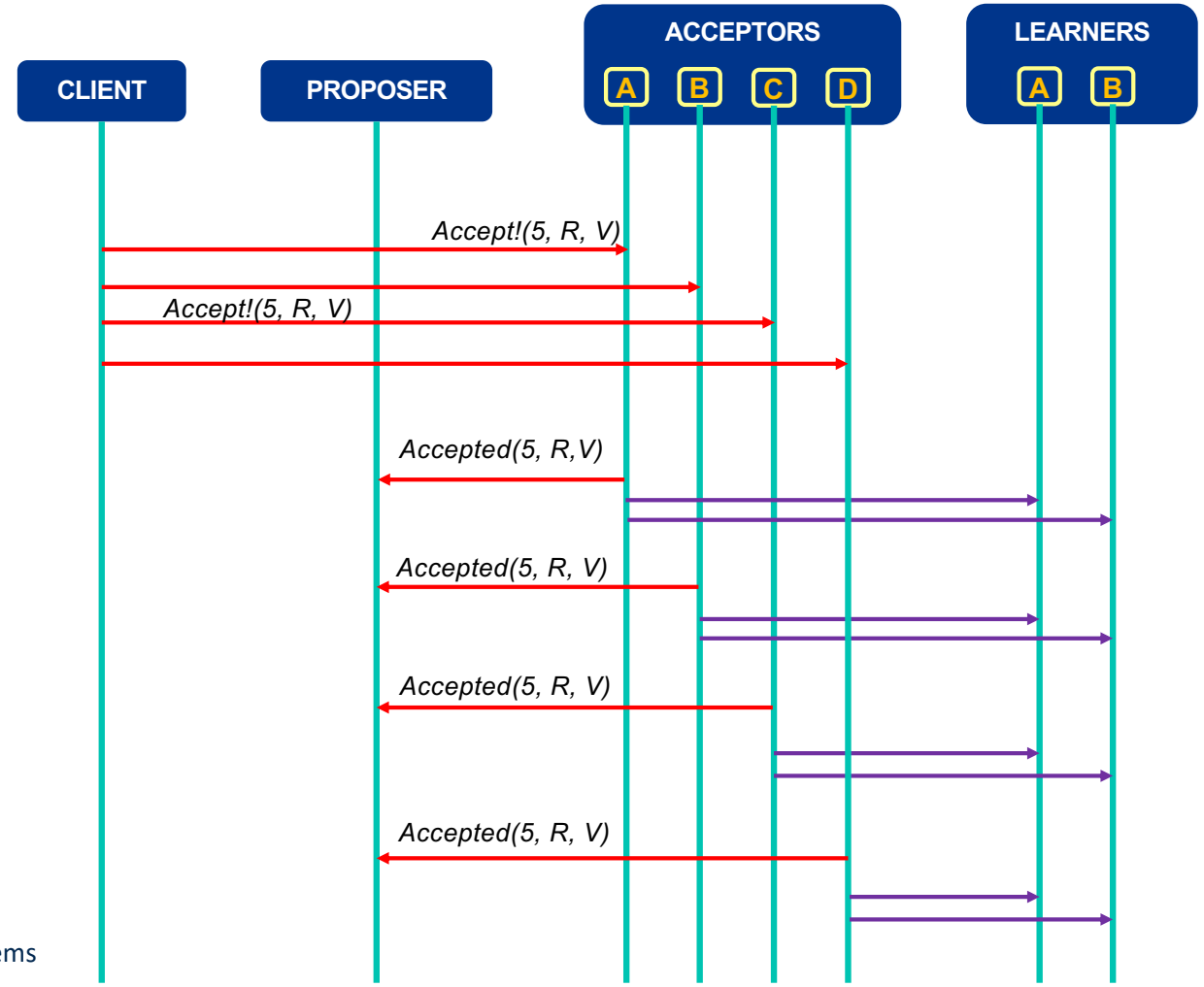
- Extension of Multi-Paxos supports multi-record transactions
  - Ensure atomic durability
  - Detect write-write conflict

- To guarantee consistency
  - Accept an option not writing value directly
  - After committing transaction, notify storage nodes to execute options

Fast paxos

- Fast Paxos bypasses the Leader
- MDCC uses fast paxos
- Because MDCC assumes conflicts are rare
- It would achieve consensus in one round-trip
  - If it does not have conflicts during the process

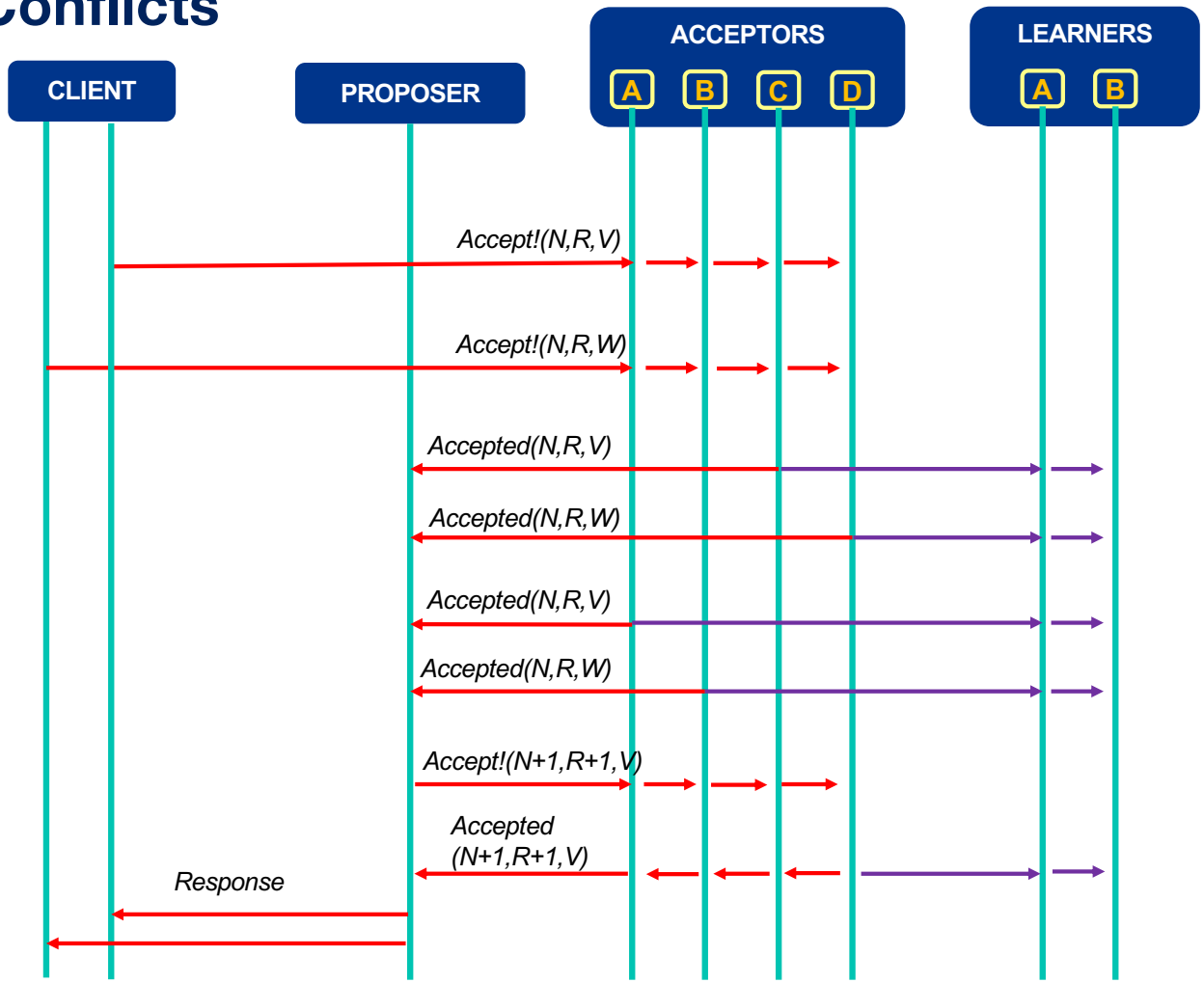
FAST Paxos


CLIENT
PROPOSER
ACCEPTORS
A
B
C
D
LEARNERS
A
B
Accept!(5, R, V)
Accept!(5, R, V)
Accepted(5, R,V)
Accepted(5, R, V)
Accepted(5, R, V)
Accepted(5, R, V)

FAST Paxos: Conflicts

Concurrent conflict proposals received in different order by the acceptors (within the same round)

If leader detects conflict sends accept message for a new round



UCDAVIS

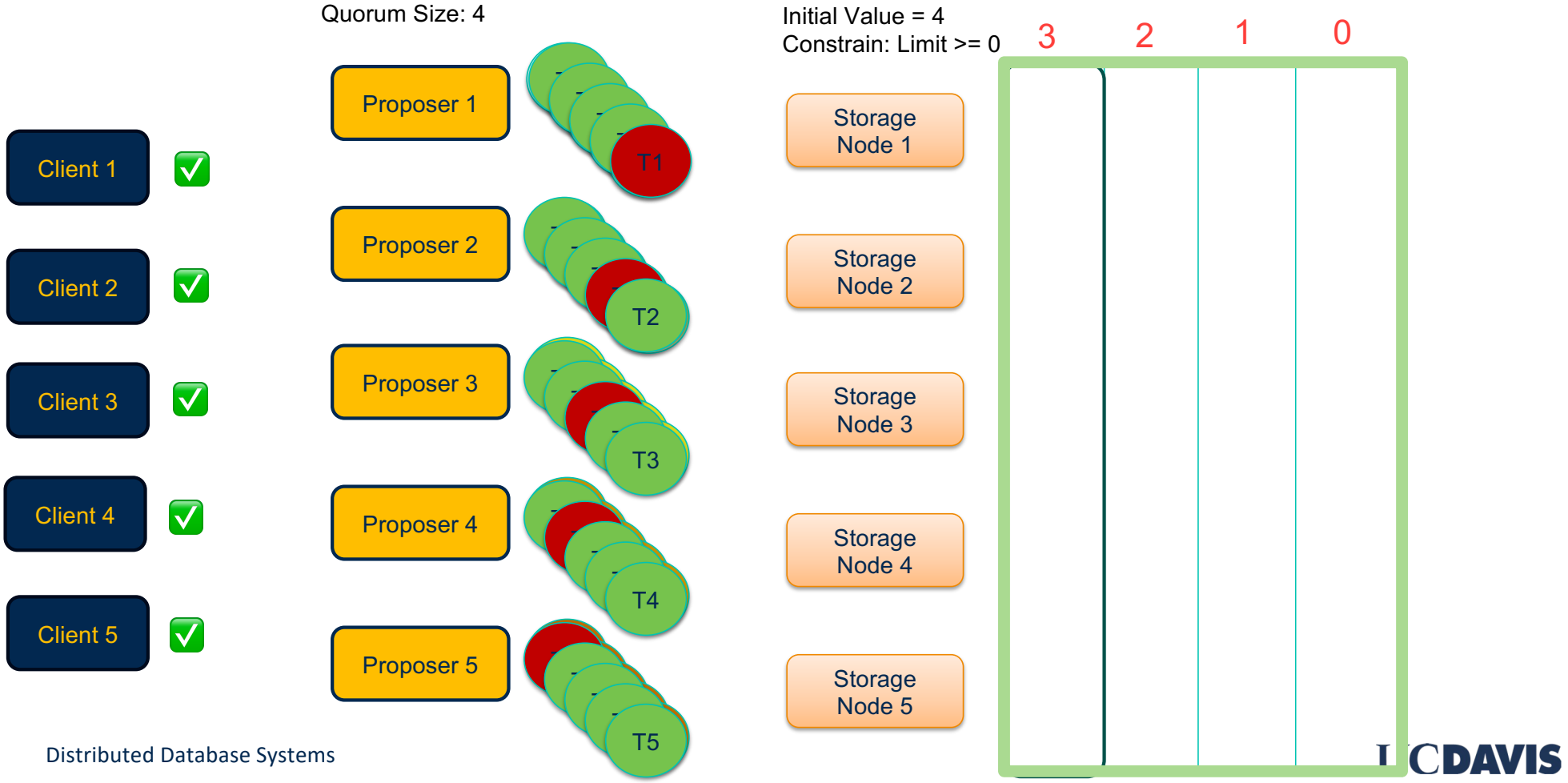
Conflicts case in Fast Paxos

- Concurrent Updates might cause conflicts
- Need Leader to resolve conflicts
- 2 additional rounds required to resolve conflicts
- As a result, it needs **3 rounds** to reach consensus

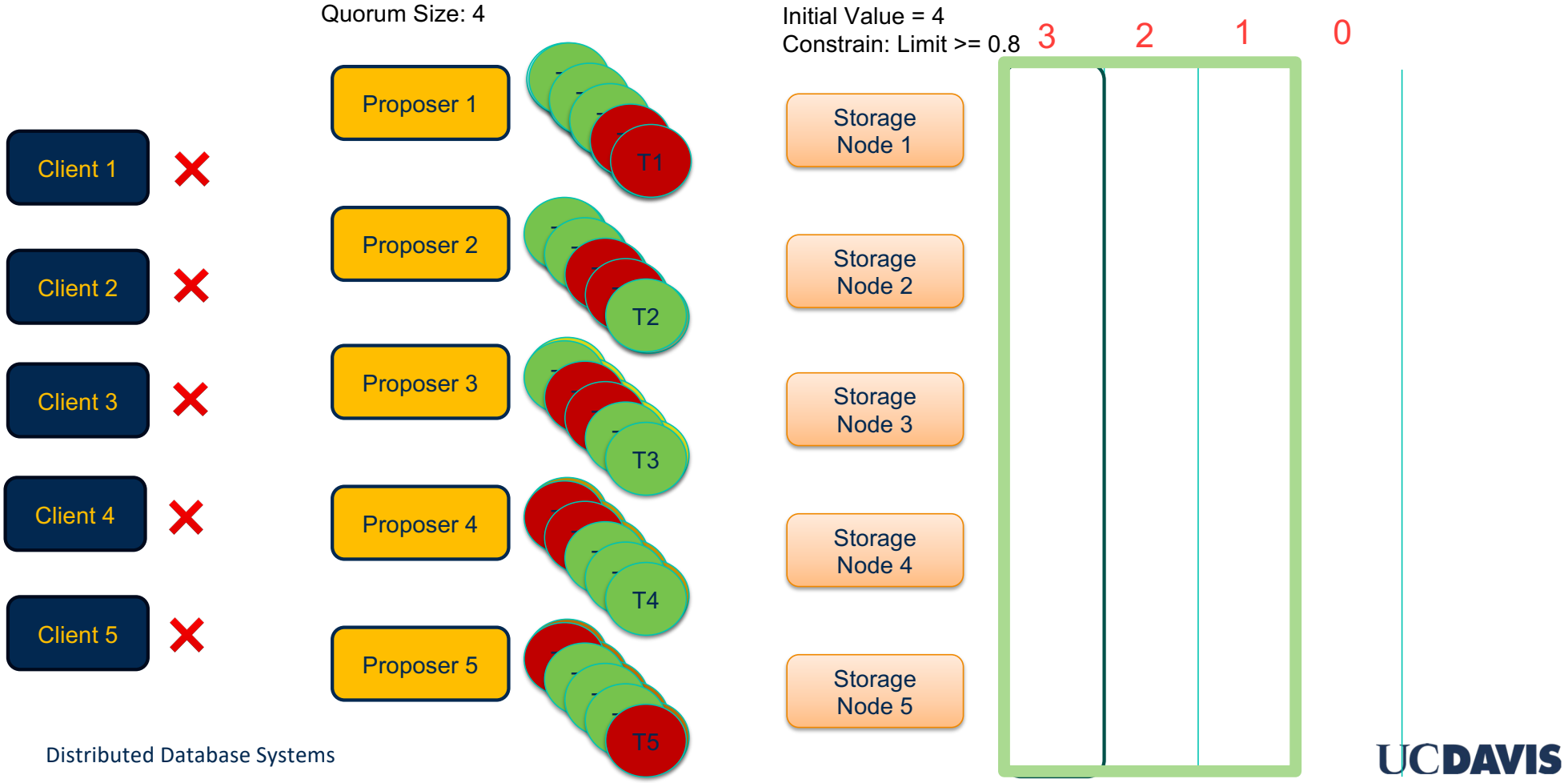
Generalized Paxos

- Generalization of Fast Paxos
- Conflicts commute
- The order of updates does not matter
  - No need to switch back to classic paxos to resolve the conflicts
- But, the integrity constraints does matter

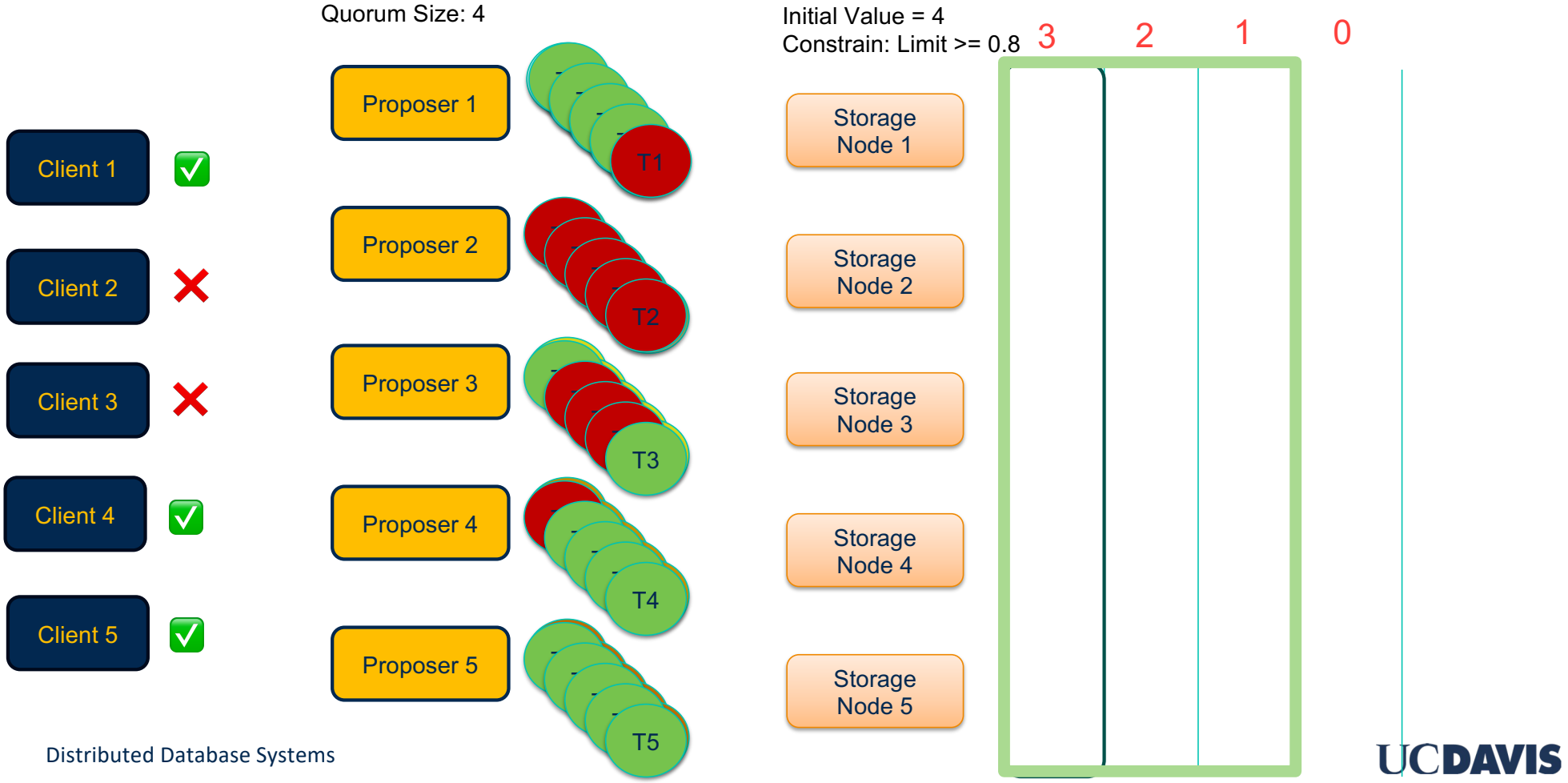
Generalized Protocol (Worst case)



MDCC Quorum Demarcation (Worst case)

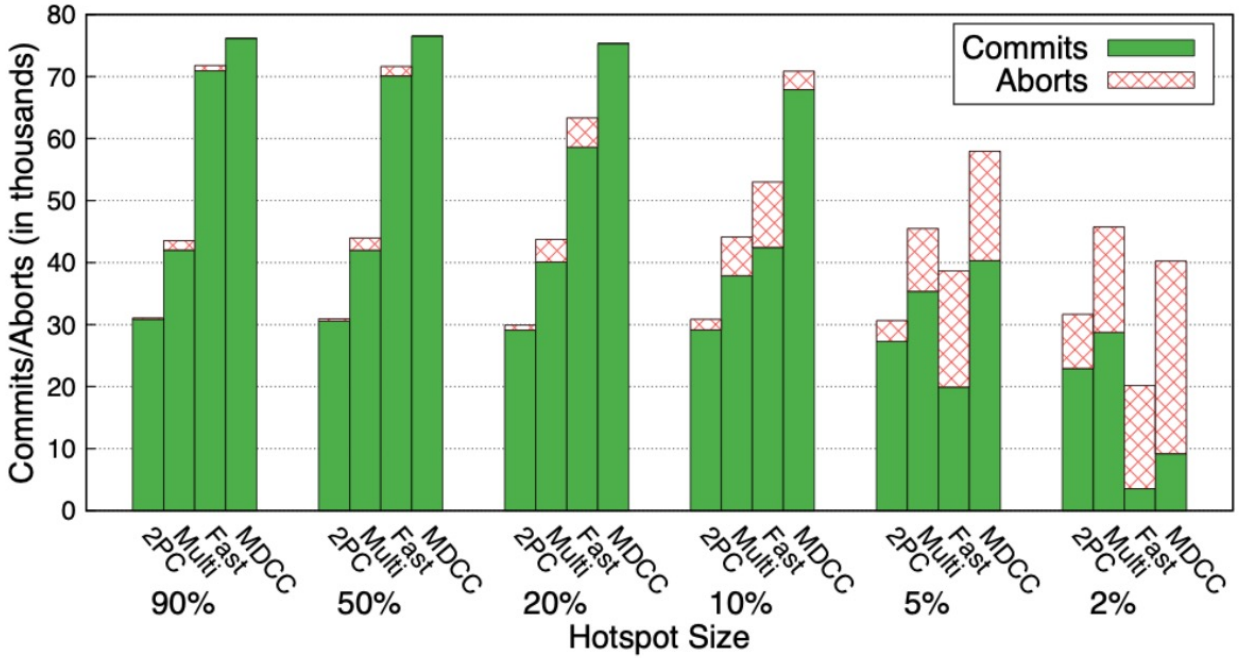


MDCC Quorum Demarcation (Normal Case)



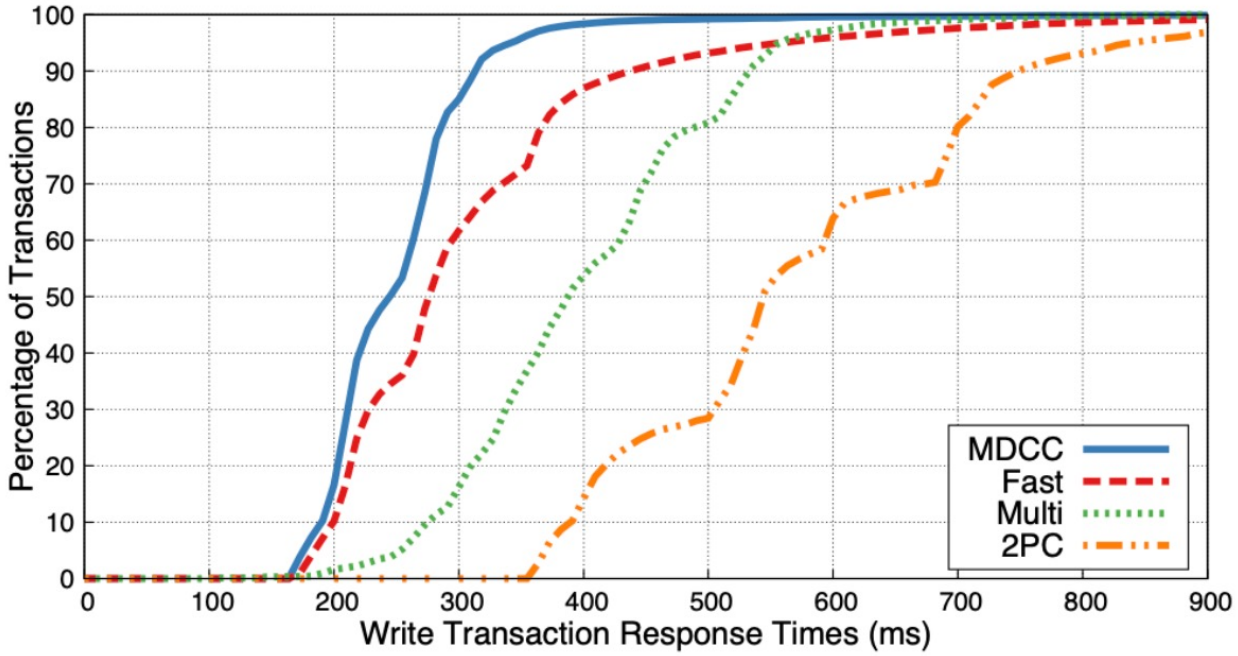
Evaluation

Micro benchmark
- varying conflict rate

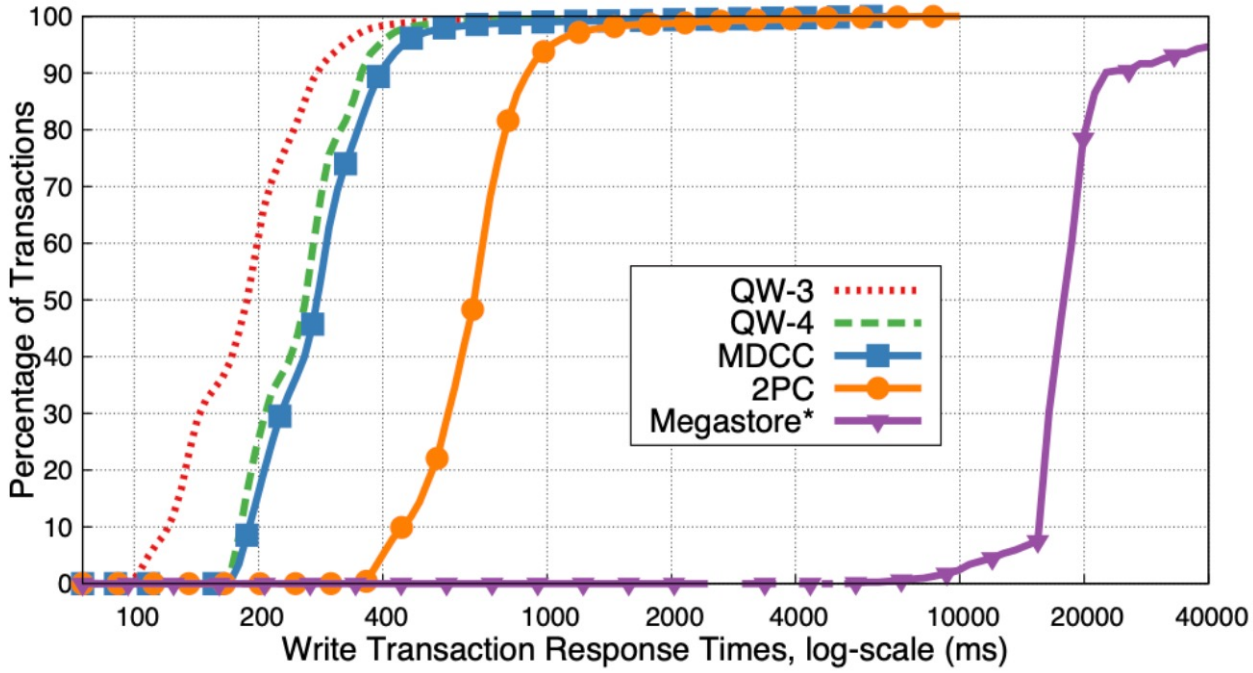


UCDAVIS

Micro benchmark
- write response times CDF



TPC-W benchmark



Conclusion

- MDCC - Multi-Data Center Consistency
  - Optimization protocol
    - Conflicts are rare
    - Updates are commutative
  - Low latency
  - Strong consistency
  - Requires only 1 message round