Consistency in key-value stores

Monika Moser

Consistency guarantees explain a system's behavior

Where is my data I just updated?

The consistency guarantees of a system influence the behavior perceived by a client.

An ideal world

The result of every write-operation is reflected by subsequent read-operations.

1-copy-serializability

Replication

Internet-scale storage systems replicate data

Performance
Availability
Persistence

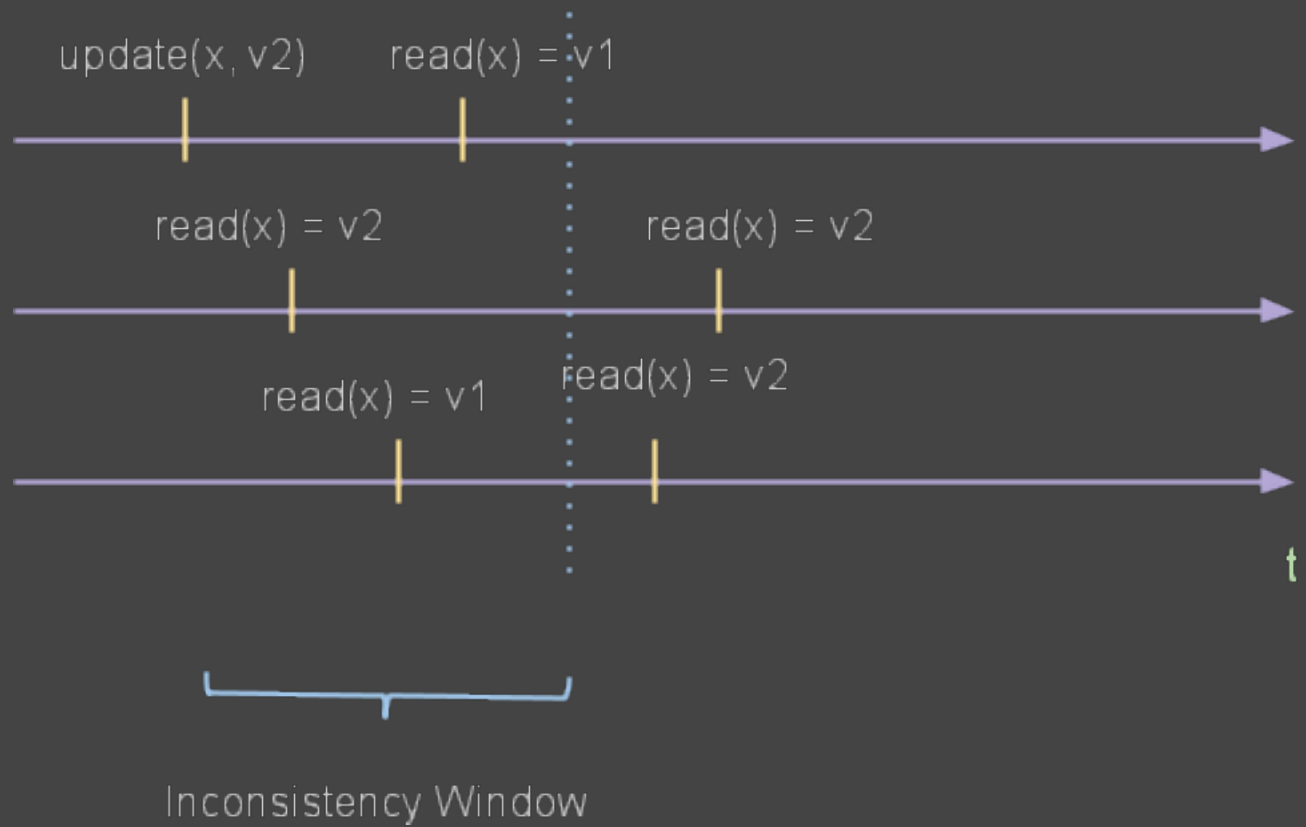
Different kinds of consistency guarantees for the client

Consistency from a client's point of view

Eventual

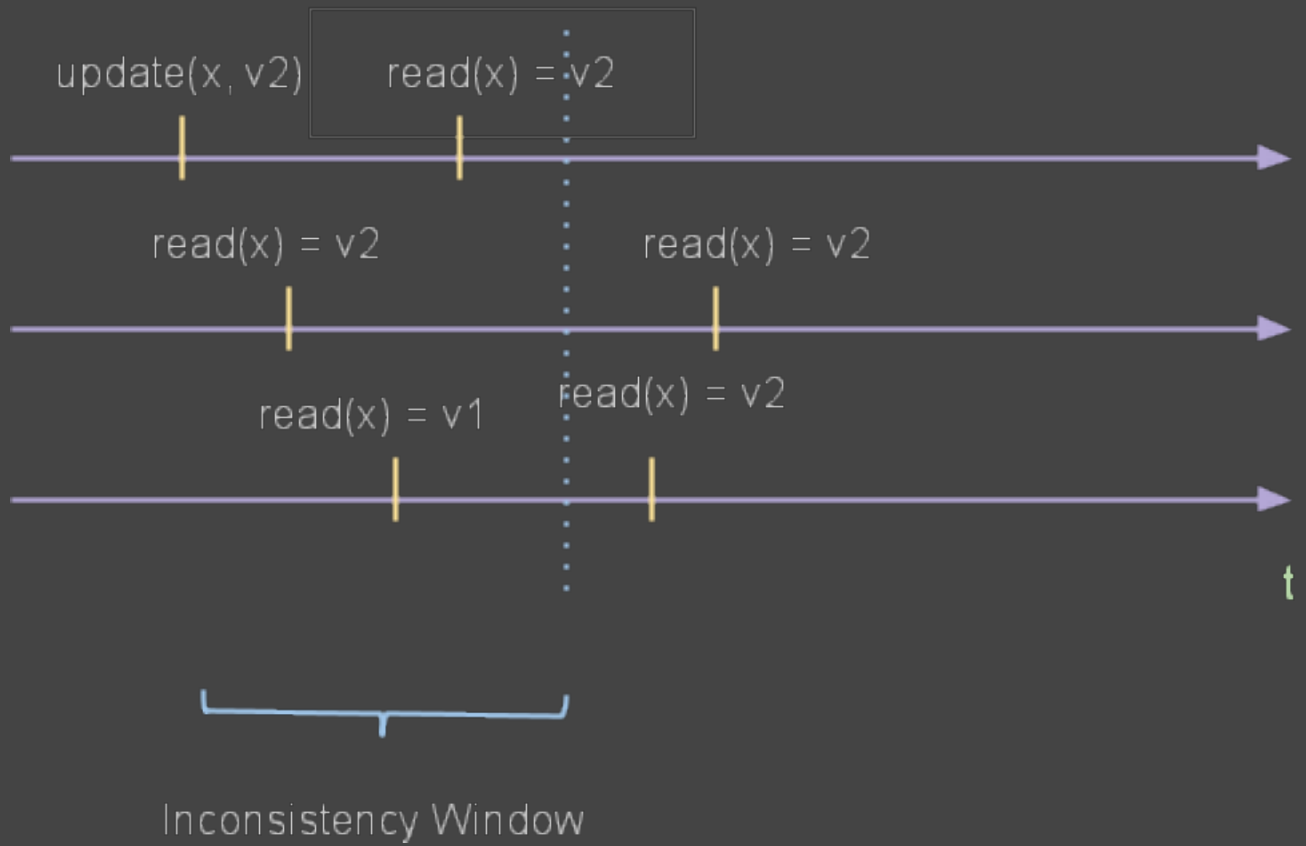
Strong (1-copy serializability)

Eventual consistency



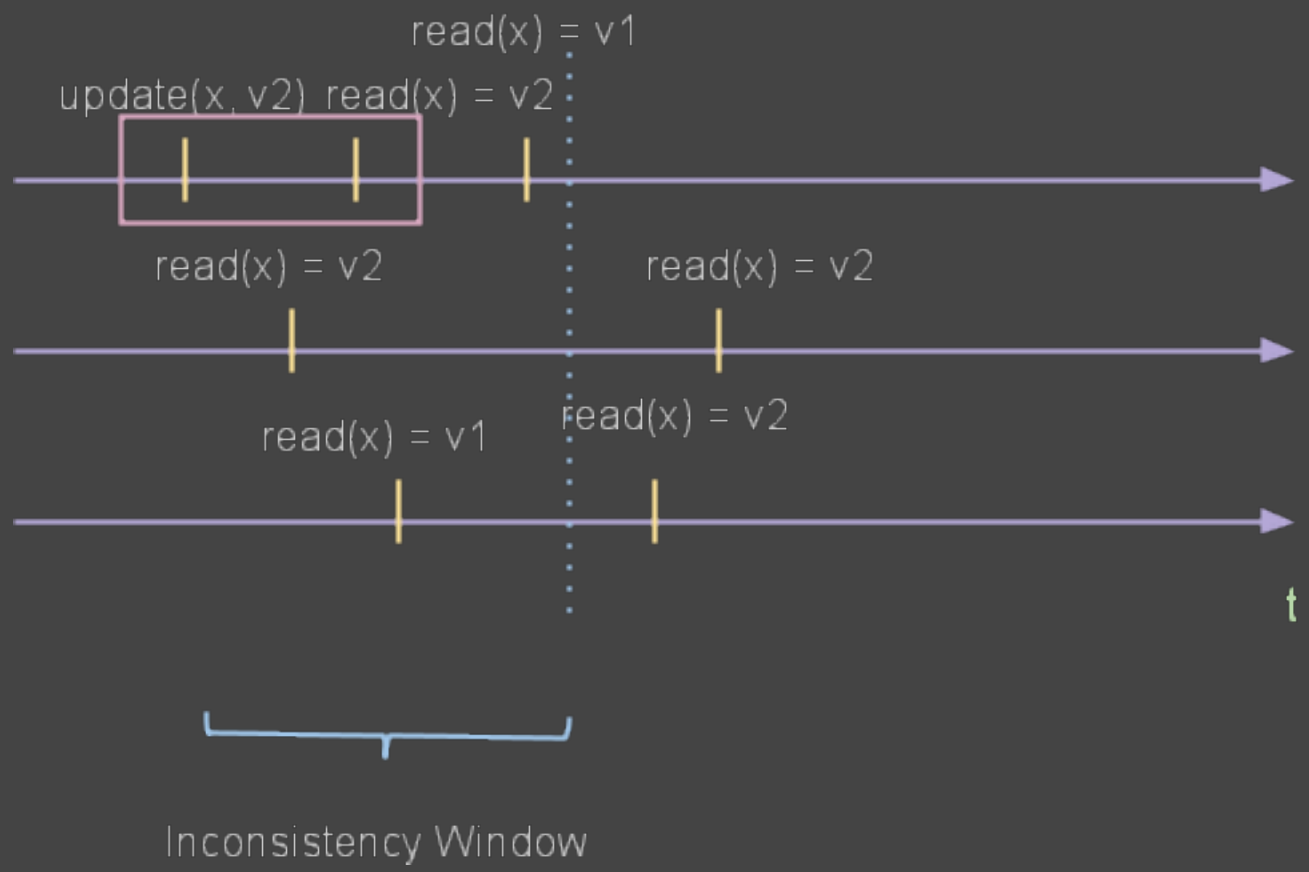
Eventual consistency flavors

Read your writes consistency



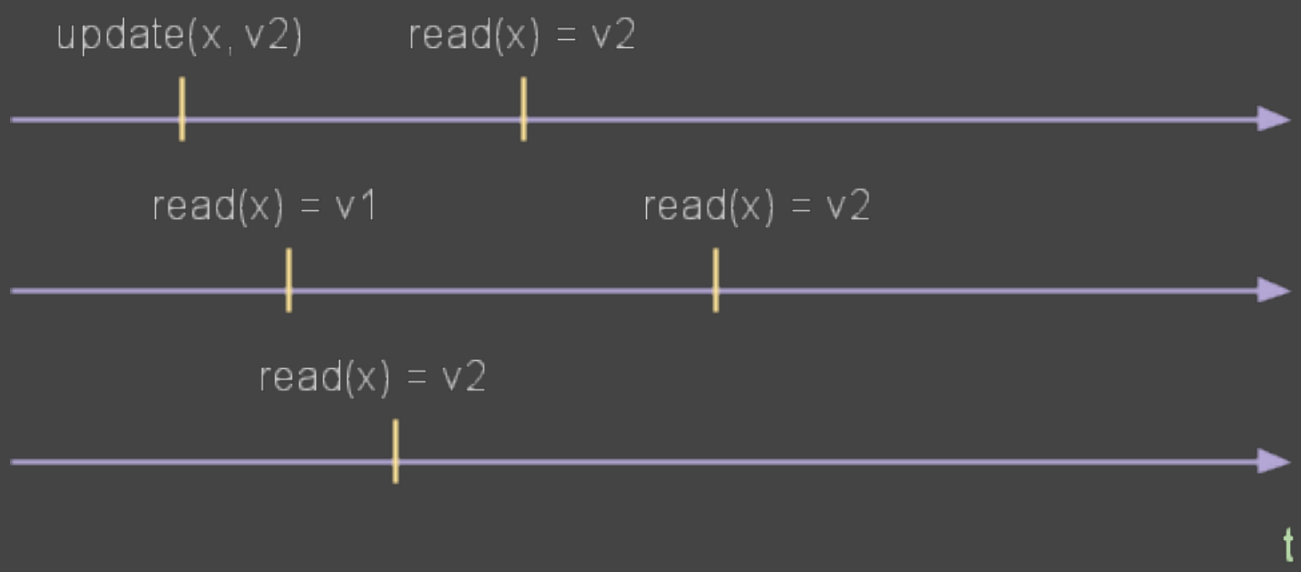
Eventual consistency flavors

Session consistency



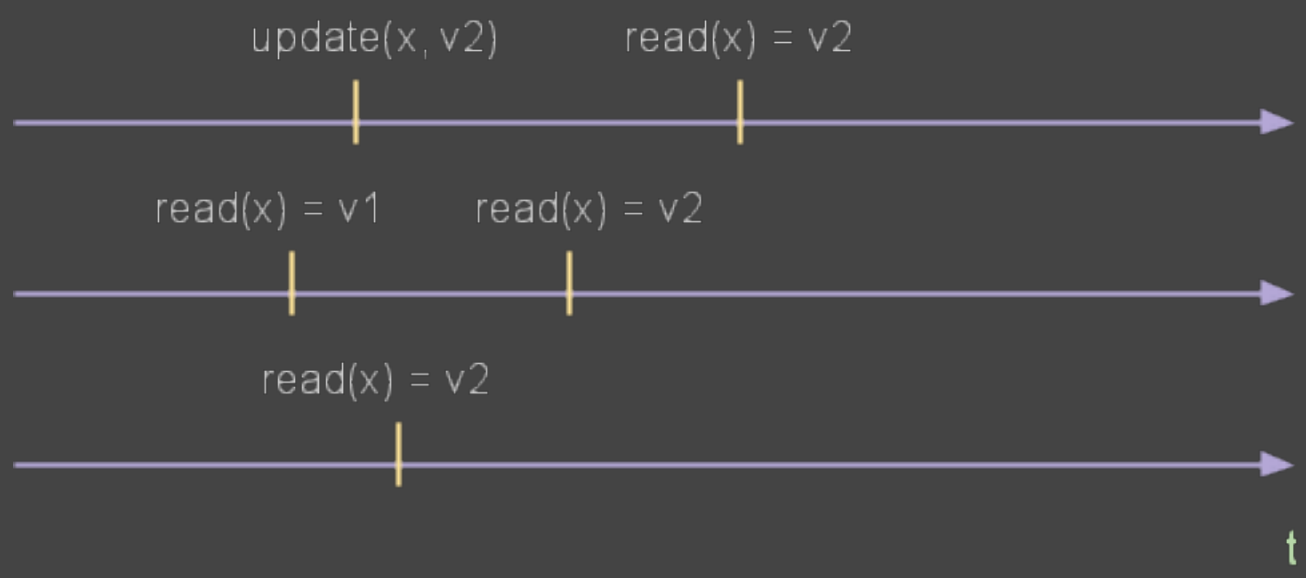
Eventual consistency flavors

Monotonic read consistency



Strong consistency

1 copy serializability



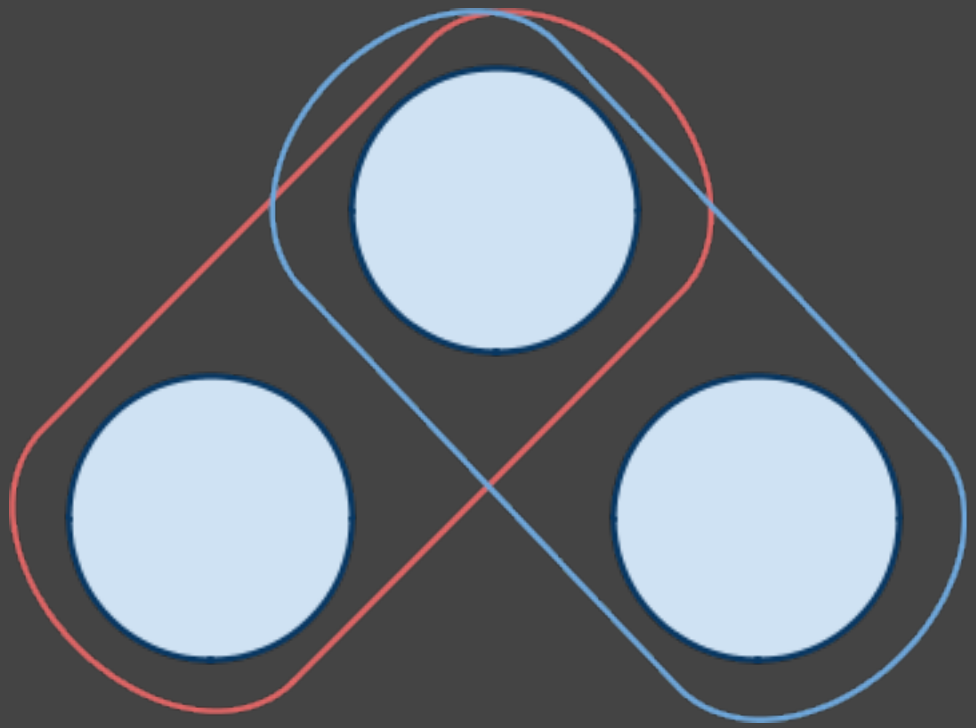
Consistency from a system's point of view

Read and write sets

N Replicas
W Replicas in the write set
R Replicas in the read set

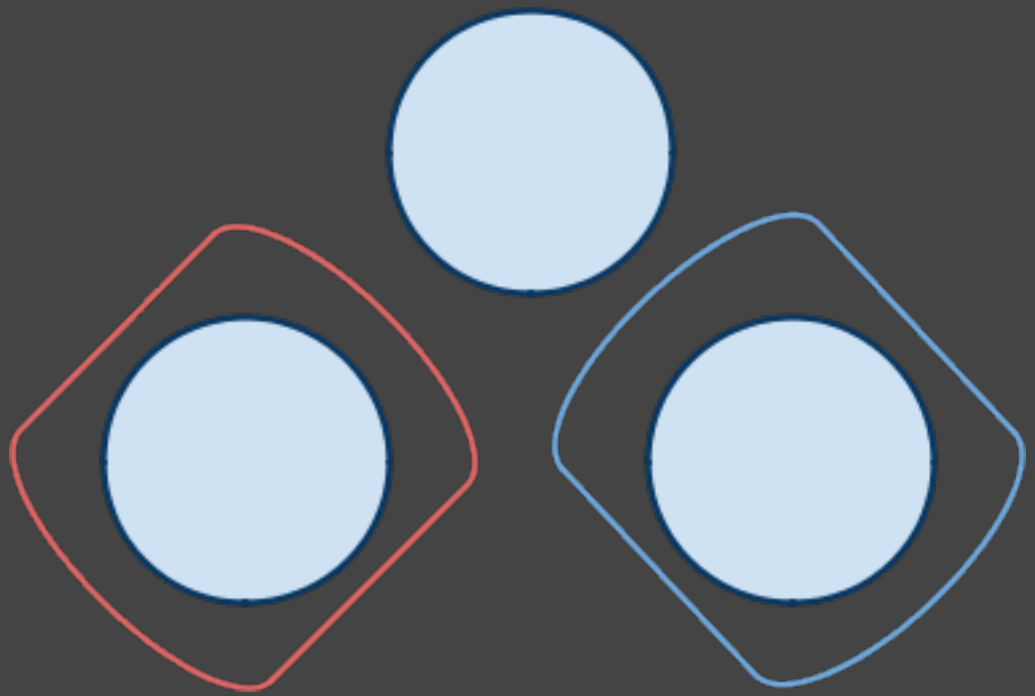
Strong consistency

W + R > N



Weak or eventual consistency

W + R <= N



You have to choose!

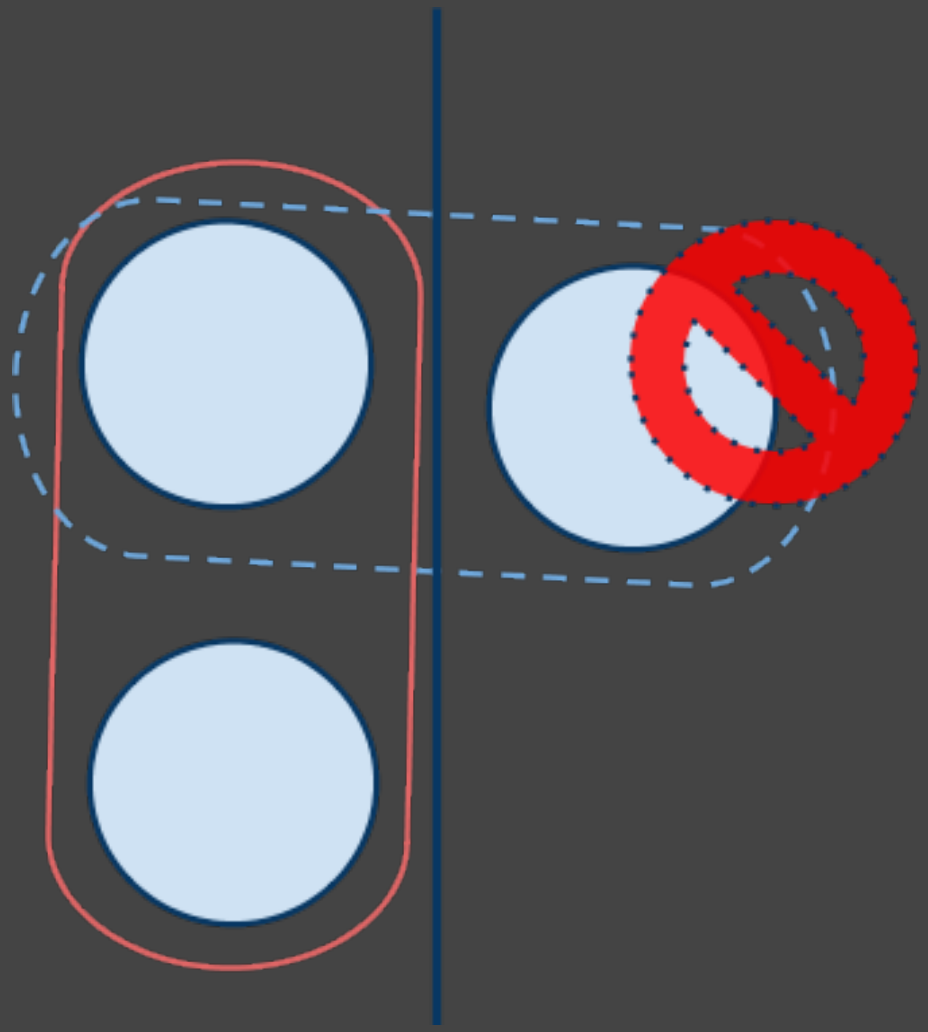
CAP-Theorem

2 out of 3

Partition tolerance
Availability
Consistency

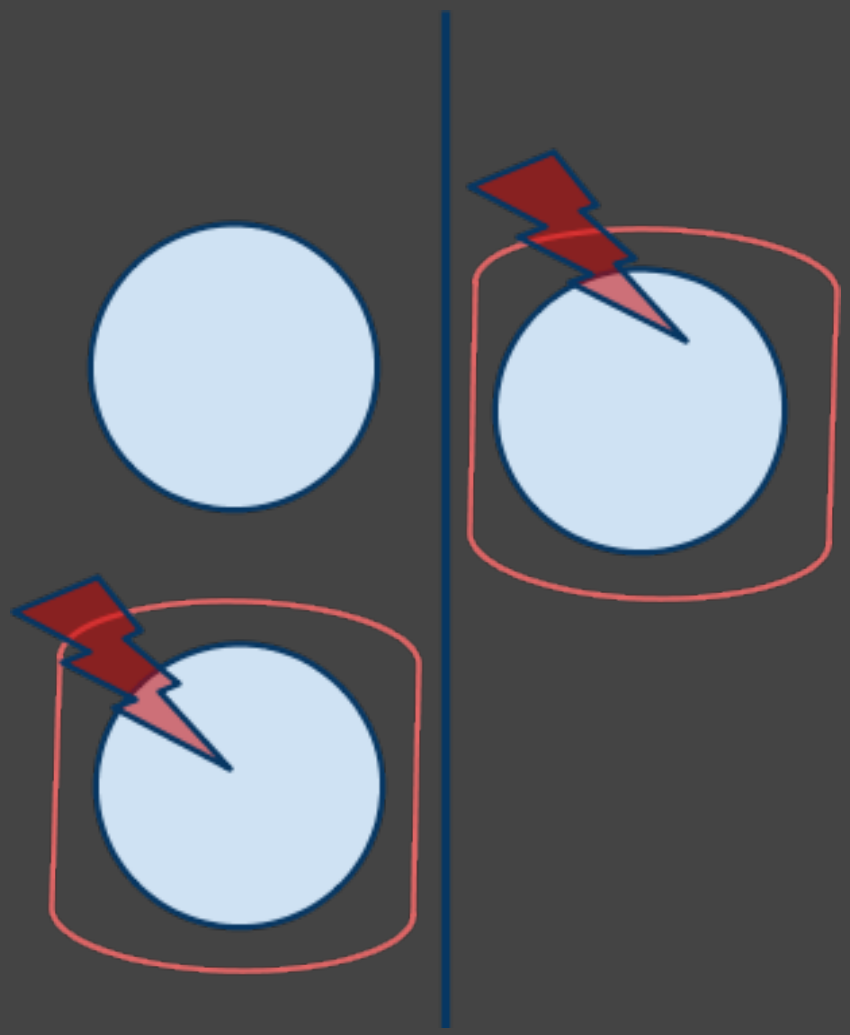
Consistency, Availability, Partition Tolerance

Availability of data cannot be guaranteed



Consistency, Availability, Partition Tolerance

Consistency of data cannot be guaranteed



Strong consistency.
What makes it expensive?

Strong consistency

Write all, read one

- *No failures tolerated*
- *Good read performance*

Strong consistency

Quorum, e.g Paxos

- *Read and writes from a majority*
- *Tolerates the failure of a minority*

If you need agreement you're lost

“Each node in a system should be able to make decisions purely based on local state. ***If you need to do something under high load with failures occurring and you need to reach agreement, you’re lost…*** *If you’re concerned about scalability, any algorithm that forces you to run agreement will eventually become your bottleneck. Take that as a given.”*

— Werner Vogels, Amazon CTO and Vice President

Paxos

A fault tolerant quorum-based consensus algorithm

Paxos properties

- Replicas may crash and later recover
- Messages may be lost
- If a majority of replicas runs without crashing, all replicas eventually agree

Paxos

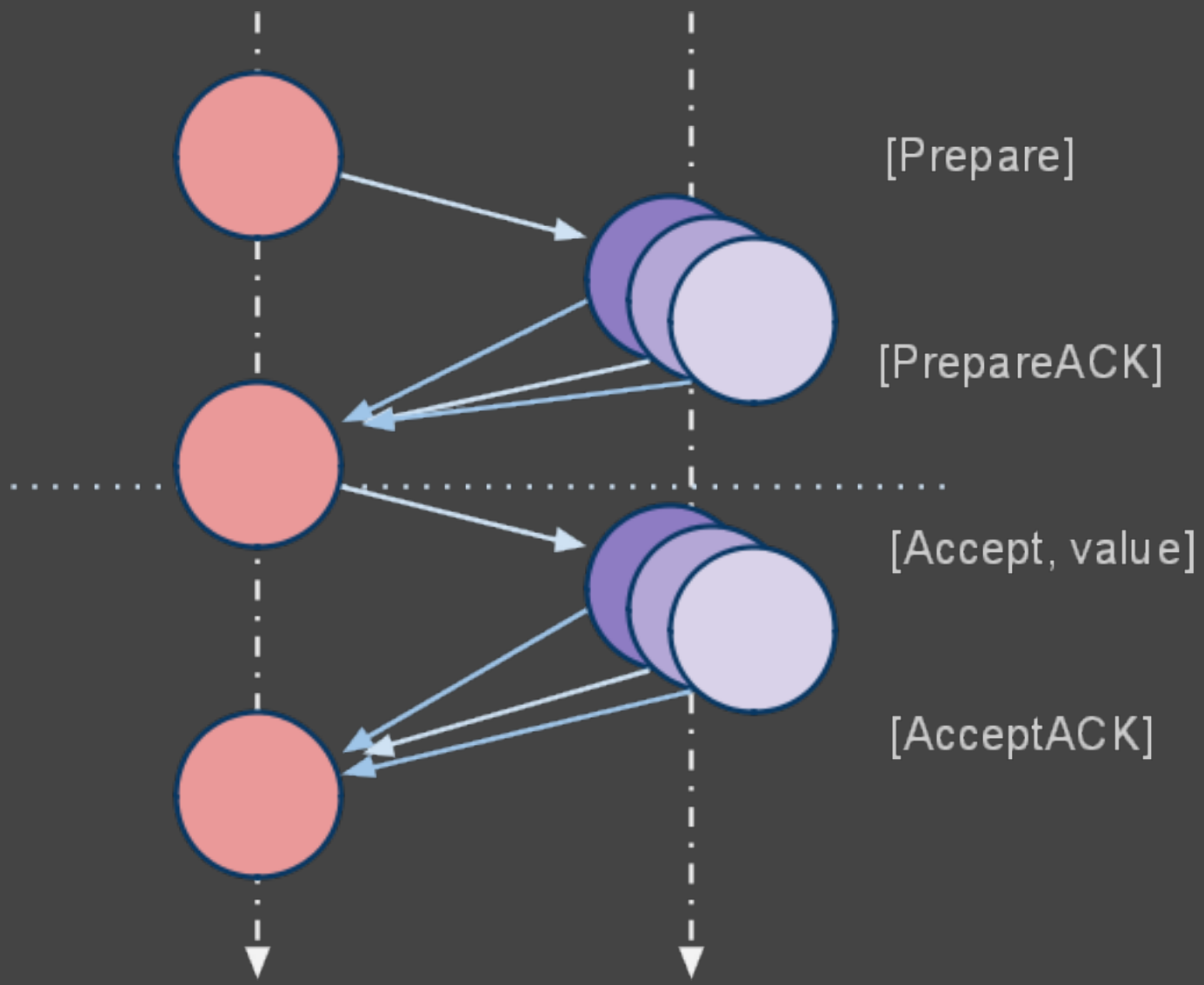
2 Phases

- *Read phase*
  Get a promise that your update will be executed by a quorum
- *Write phase*
  If the read phase was acknowledged, start the write phase

Paxos - communication steps

Writes:

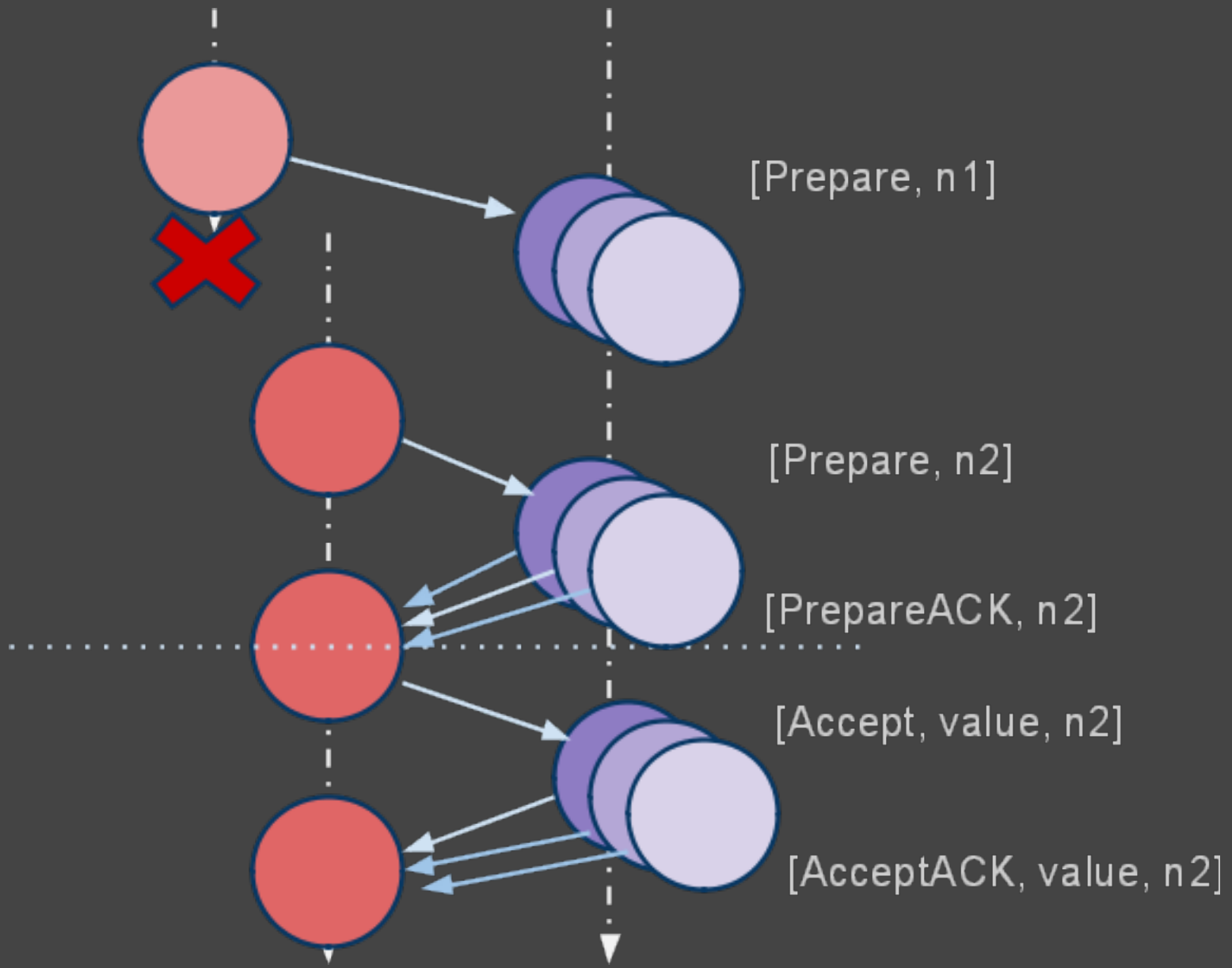
Leader



Paxos - leader fails

Writes:
New
Leader

Ordering
among
leaders

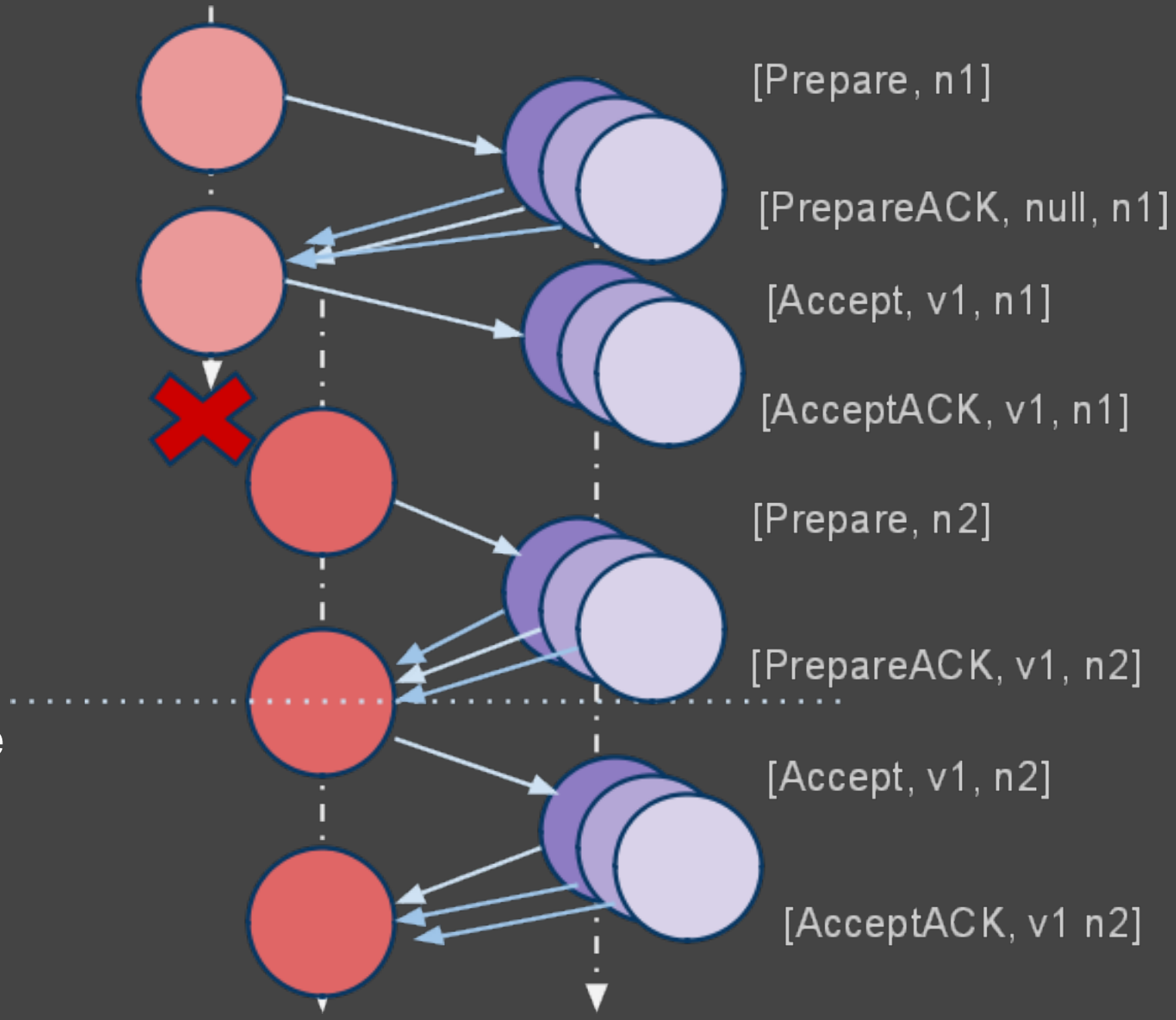


Paxos - leader fails

Writes:
New
Leader

Ordering
among
leaders

Choice of value
restricted



Tradeoffs

- Availability vs Consistency
- Read vs Write Performance

- Read patterns of the system?
- Write patterns? Probability of a conflict?
- Can the system deal with the conflict resolution techniques?

Questions?

Thanks!

monika.m.moser@googlemail.com