CRDTs: Consistency without concurrency control

Mihai Letia, Nuno Preguiça, Marc Shapiro

INRIA Research Report #6956
June 2009

Presented by Johann Schleier-Smith
Berkeley CS294-110
October 26, 2015

Outline

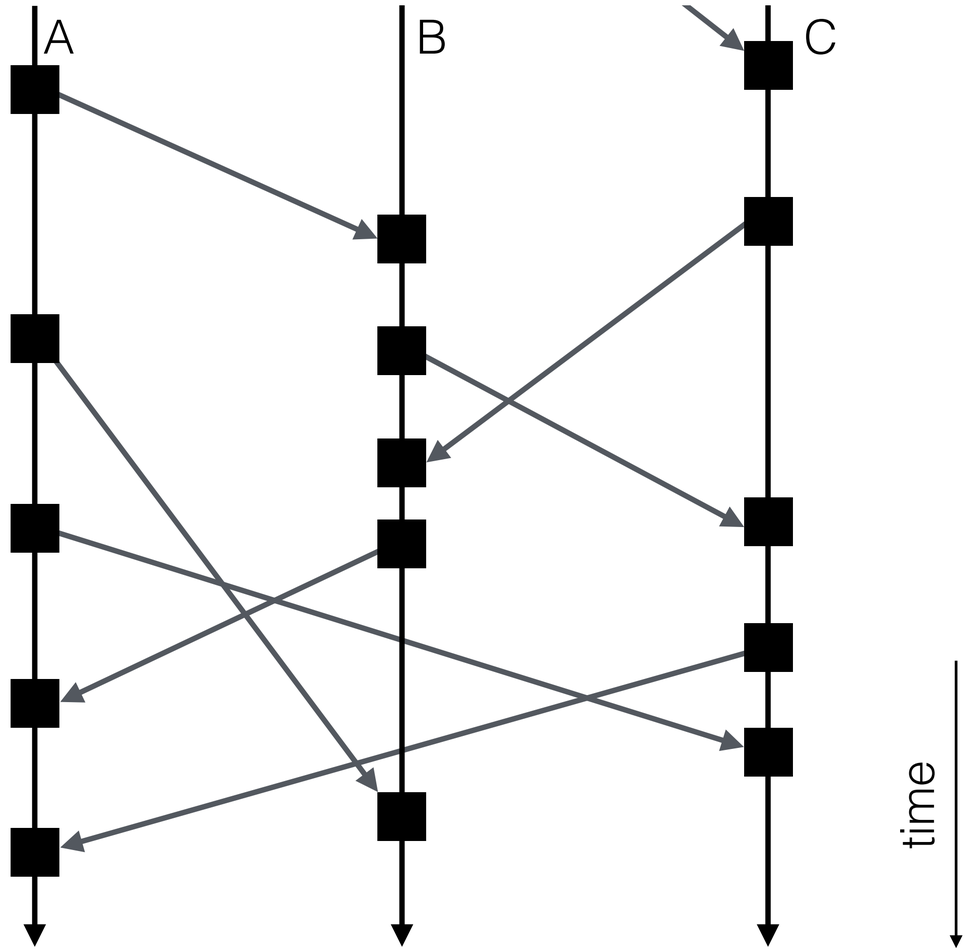
- Sample application: collaborative and concurrent editing
- Review assumptions
- CAP theorem
- CRDT Treedoc solution
- Discussion questions

Assumptions

- Replicated data
- Multiple nodes
- Concurrent use

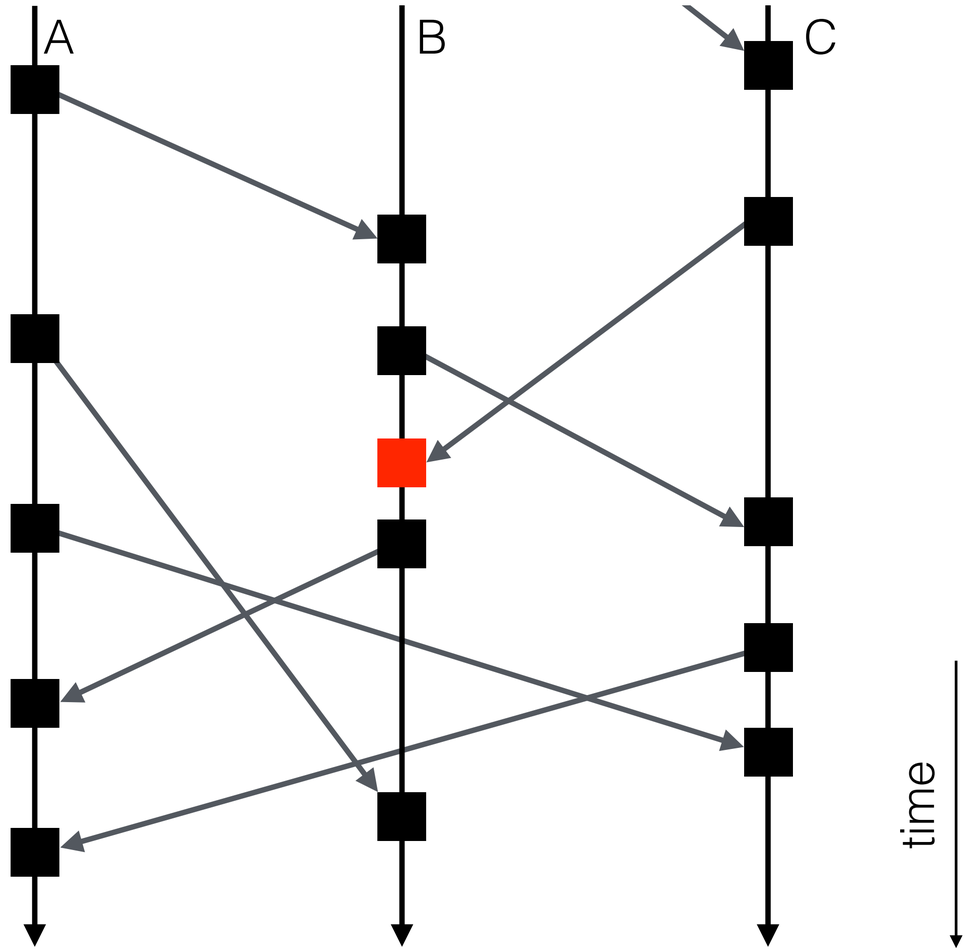
Assumptions

- Replicated data
- Multiple nodes
- Concurrent use



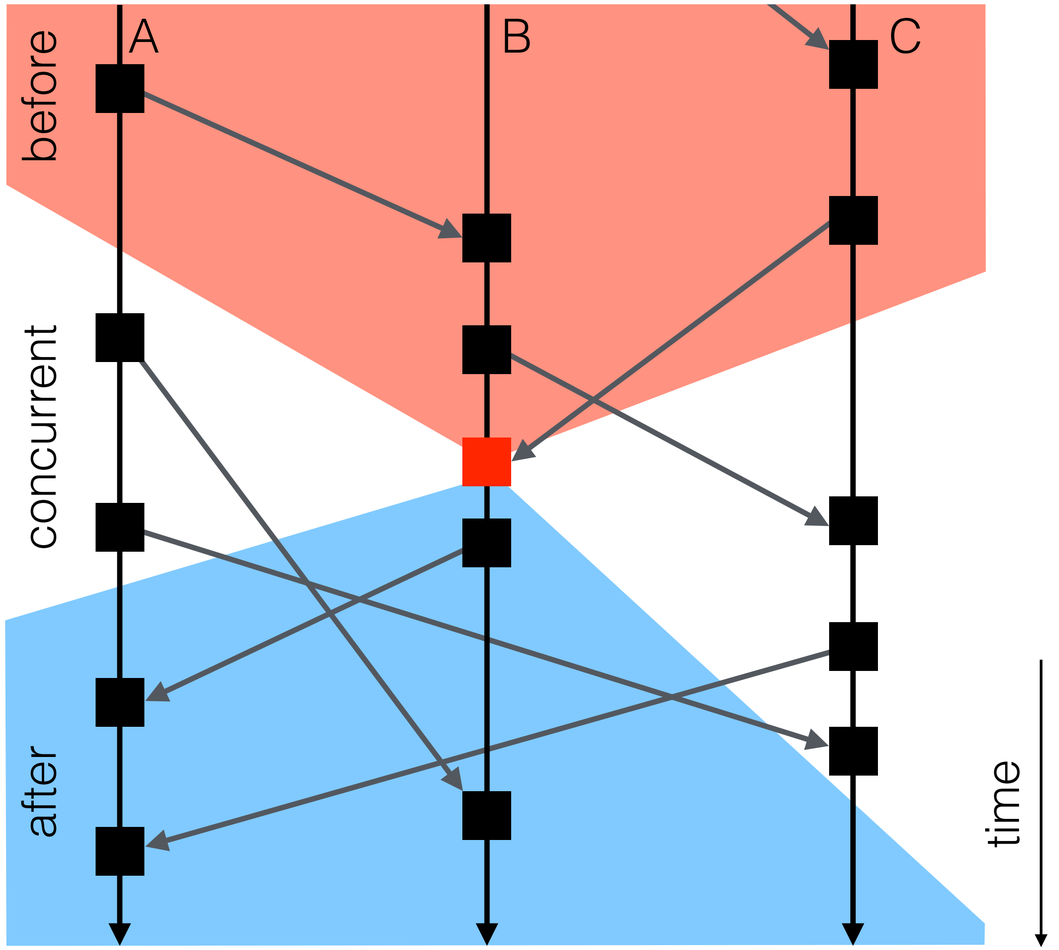
Assumptions

- Replicated data
- Multiple nodes
- Concurrent use

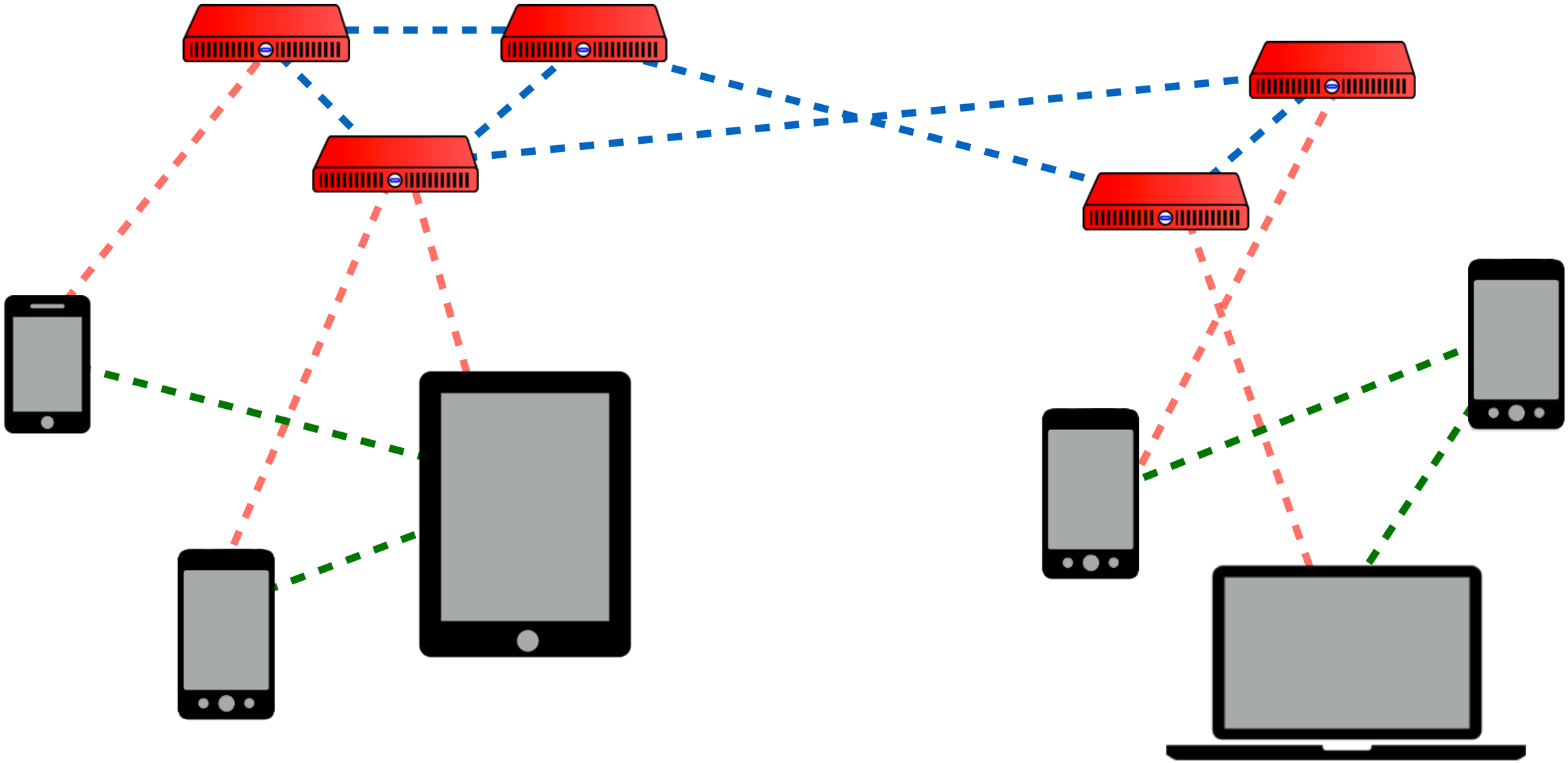


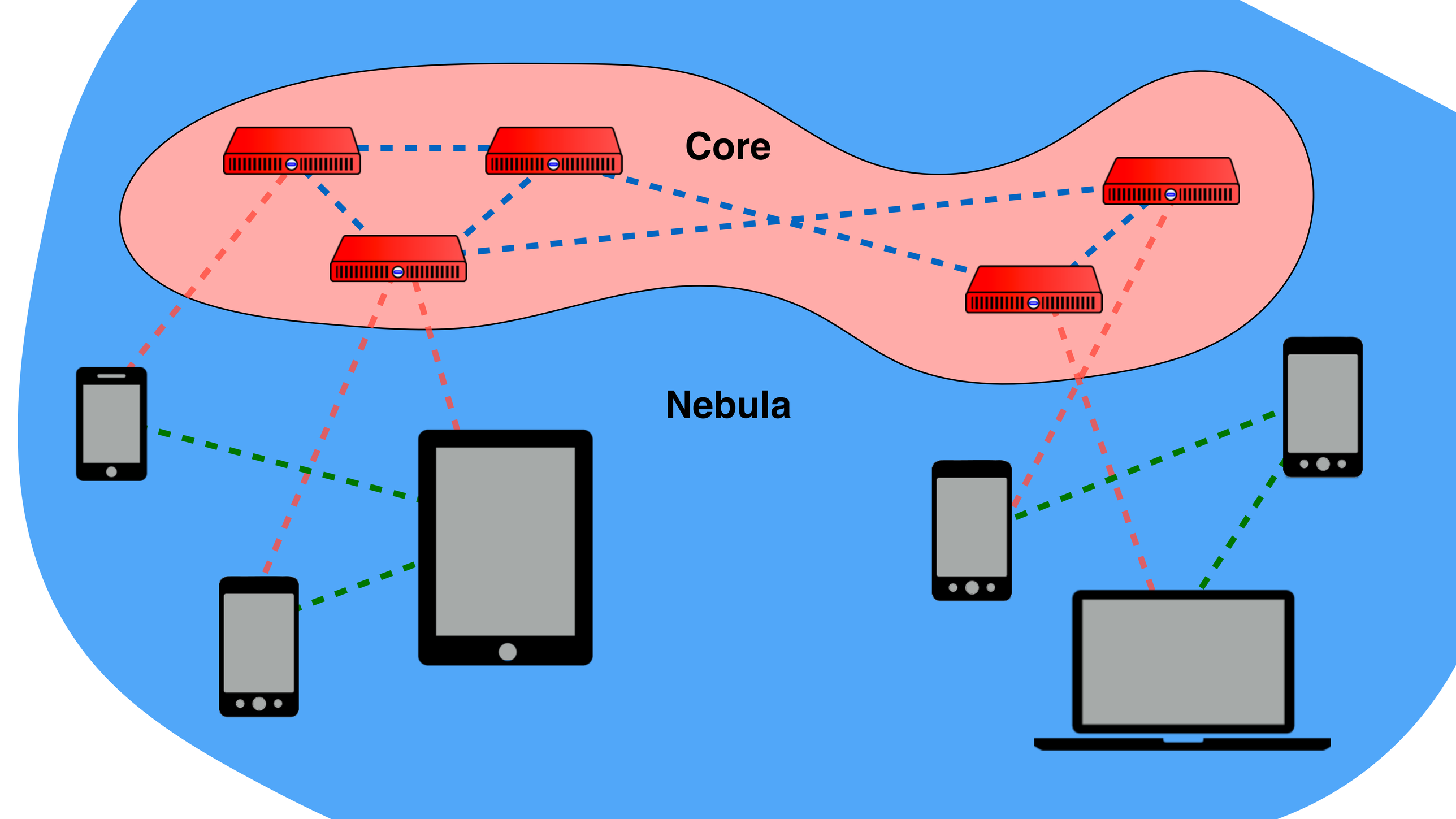
Assumptions

- Replicated data
- Multiple nodes
- Concurrent use



source

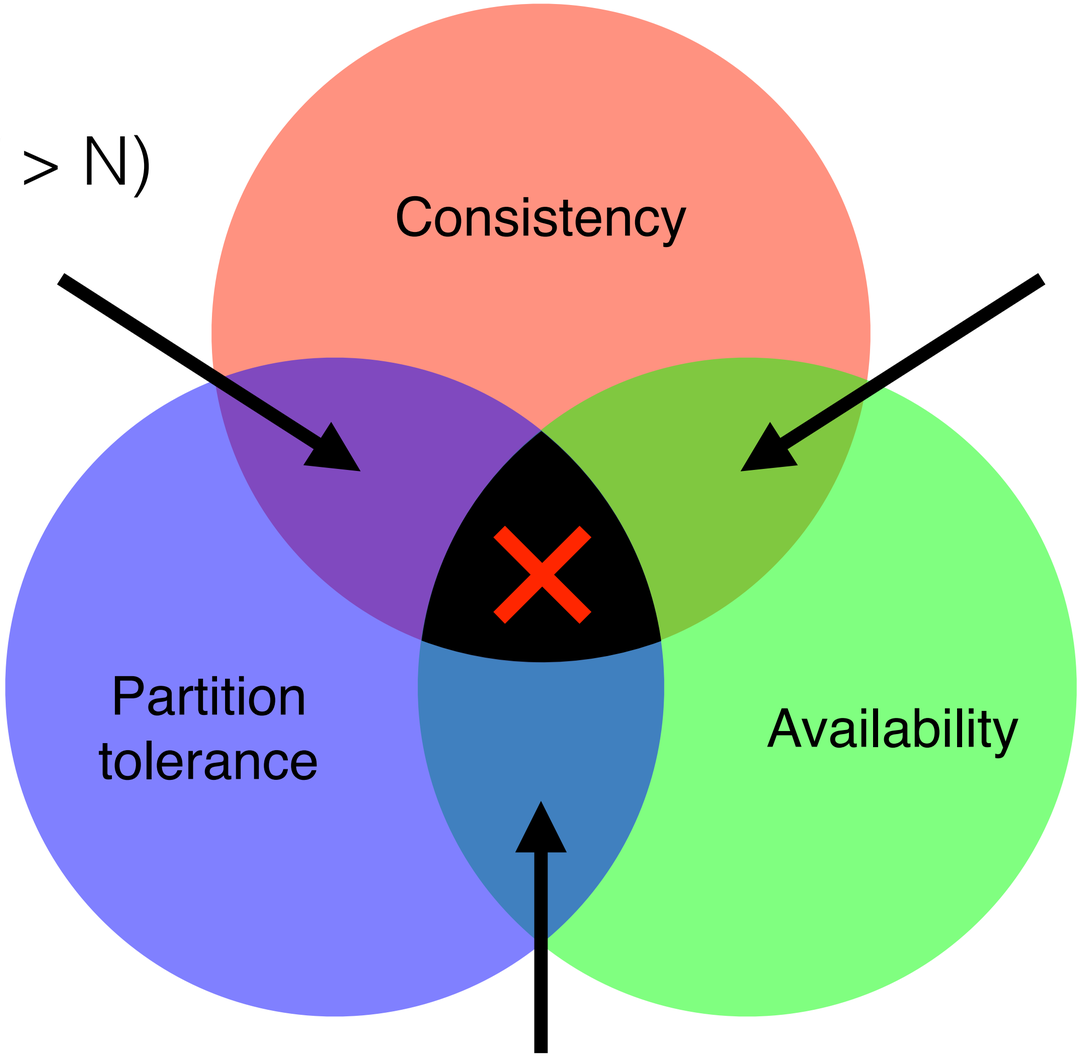



Core
Nebula

CAP Expectations

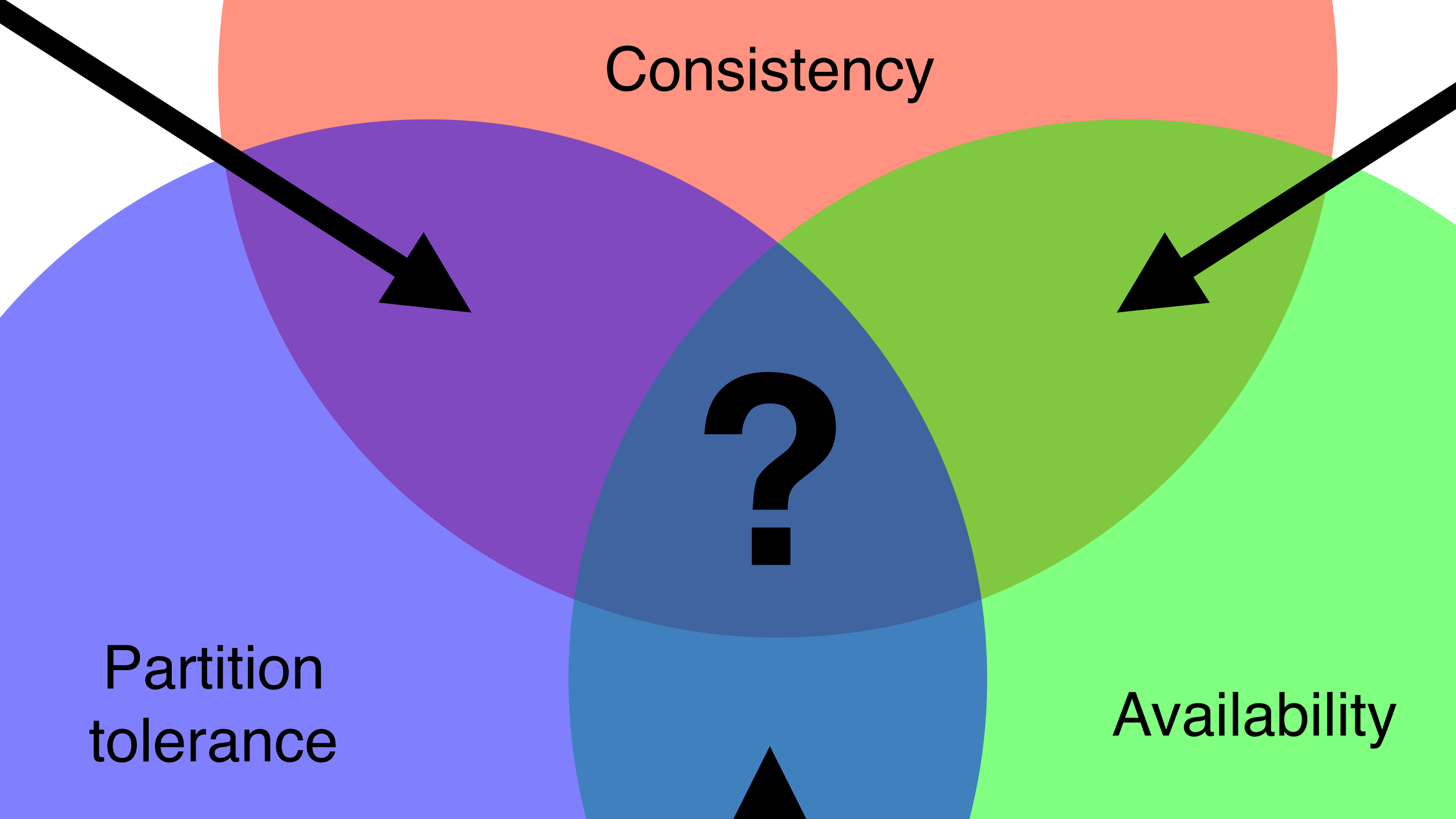
- **Consistency** - people see documents that make sense
- **Availability** - keep working so long as local node is serviceable
- **Partition tolerance** - network graph not always fully connected

- Cassandra (R+W > N)
- Spanner / F1
- BigTable



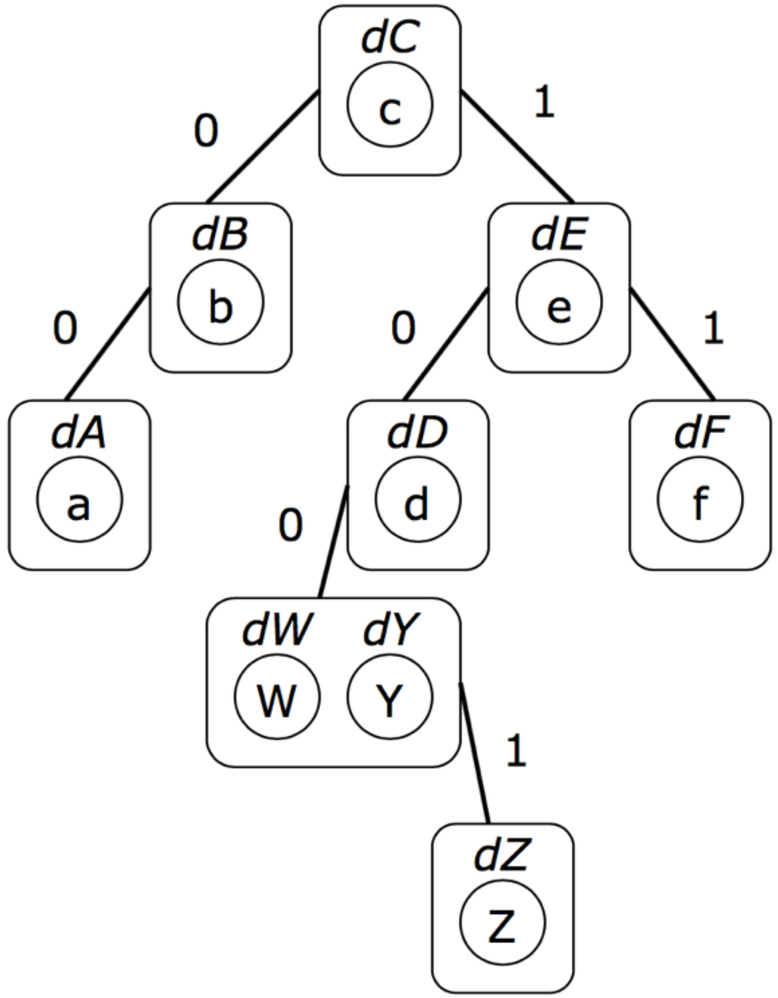
- Traditional RDBMS
- Impala

- Cassandra (R+W ≤ N)
- Dynamo
- Riak


Consistency
?
Partition
tolerance
Availability

Treedoc CRDT

- Broadcast all operations in messages
- Unique naming of insertion points
- Concurrent operations commute
- Coordinate for garbage collection



Questions

- Is messaging model realistic (guaranteed delivery, causal order)?
- Could we implement garbage collection without coordination?
- Do CRDTs really “beat CAP”?
- Do CRDTs describe the entire design space that “beats CAP”?
- Are CRDTs useful in real applications?