Causal+ Consistency (review)

1. Writes that are potentially causally related must be seen by all processes in same order.

2. Concurrent writes may be seen in a different order on different processes.

Concurrent: Ops not causally related

Causal+ Consistency (review)

• Partially orders all operations, does not totally order them
  • Does not look like a single machine

• Guarantees
  • For each process, ∃ an order of all writes + that process’s reads
  • Order respects the happens-before (→) ordering of operations
  • + replicas converge to the same state
    • Skip details, makes it stronger than eventual consistency
Causal consistency within replicated systems

Consistency vs Scalability

**Scalability:** Adding more machines allows more data to be stored and more operations to be handled!

<table>
<thead>
<tr>
<th>System</th>
<th>Consistency</th>
<th>Scalable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paxos/RAFT</td>
<td>Linearizable</td>
<td>No</td>
</tr>
<tr>
<td>Bayou</td>
<td>Causal</td>
<td>No</td>
</tr>
<tr>
<td>Dynamo</td>
<td>Eventual</td>
<td>Yes</td>
</tr>
</tbody>
</table>

It’s time to think about scalability!

Implications of laziness on consistency

- Linearizability / sequential: Eager replication
- Trades off low-latency for consistency
- Maintain local ordering when replicating
- Operations may be lost if failure before replication

Consistency vs Scalability

**Scalability:** Adding more machines allows more data to be stored and more operations to be handled!

<table>
<thead>
<tr>
<th>System</th>
<th>Consistency</th>
<th>Scalable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamo</td>
<td>Eventual</td>
<td>Yes</td>
</tr>
<tr>
<td>Bayou</td>
<td>Causal</td>
<td>No</td>
</tr>
<tr>
<td>COPS</td>
<td>Causal</td>
<td>Yes</td>
</tr>
<tr>
<td>Paxos/RAFT</td>
<td>Linearizable</td>
<td>No</td>
</tr>
</tbody>
</table>

Next Time!
COPS:
Scalable Causal Consistency for Geo-Replicated Storage

Geo-Replicated Storage serves requests quickly

Inside the Datacenter

Scalability through Sharding
Causality By Example

Remove boss from friends group

Post to friends: “Time for a new job!”

Friend reads post

Causality ( )

Same process

Reads-From

(message receipt)

Transitivity

Bayou’s Causal Consistency

• Log-exchange based

• Log is single serialization point within DC

✓ Implicitly captures & enforces causal order

Sharded Log Exchange

• What happens if we use a separate log per shard?

• What happens if we use a single log?

Scalability Key Idea

• Capture causality with explicit dependency metadata

• Enforce with distributed verifications

– Delay exposing replicated writes until all dependencies are satisfied in the datacenter
COPS Architecture

All Ops Local = Available and Low Latency

Client Library

Read

write after = write + ordering metadata

Write

Replication
**Basic Architecture Summary**

- All ops local, replicate in background
  - Availability and low latency

- Shard data across many nodes
  - Scalability

- Control replication with dependencies
  - Causal consistency

**Challenge: Many Dependencies**

Dependencies grow with client lifetime

**Nearest Dependencies**

- Transitively capture ordering constraints
Nearest Dependencies

- Transitivity capture ordering constraints
- Need extra server-side state to calculate

One-Hop Dependencies

- Small superset of nearest dependencies
- Simple to track:
  - Last write
  - Subsequent reads

Scalable Causal+

From fully distributed operation

- Checking them suffices for causality
  - Competitive to eventually-consistent system
- Never store dependencies on the server
  - Transitive Closure
- Simplifies client-side dep tracking
  - Clear on every write
Scalability

• Shard data for scalable storage

• New distributed protocol for scalably applying writes across shards

• Also need a new distributed protocol for consistently reading data across shards...

Reads Aren’t Enough

Asynchronous requests + distributed data = ??

COPS Scaling Evaluation

Read-Only Transactions

Consistent up-to-date view of data, across many servers

More on transactions next time!
COPS

• Scalable causal consistency
  – Shard for scalable storage
  – Distributed protocols for coordinating writes and reads
    • Evaluation confirms scalability

• All operations handled in local datacenter
  – Availability + low latency

• Next time: scalable strong consistency