Dynamo / Bayou

10/11/19

[Adapted from Andrew Or's]

A Note on Assignment 2

- You will need to handle concurrent snapshots.
  - Concurrent snapshots should not interfere with each other.
  - Capture state (state = number of tokens) at the time of marker reception.

- Start snapshot at server by calling StartSnapshot on the server object.

- Always update your local state (state = number of tokens).
  - Snapshotting process should not affect operation of system.

Some context...

Dynamo and Bayou both offer high availability and weak consistency

Most traditional databases offer strong consistency and low availability

Not suitable for modern applications with super high demands

What are some example applications of each?

Flight ticket booking (HA)

Amazon shopping carts (HA)

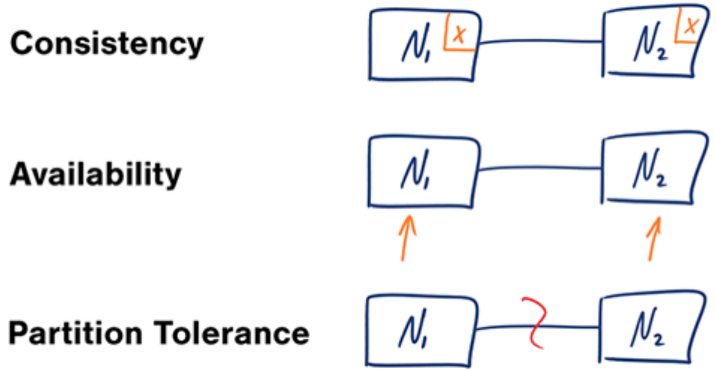
Offline edits (HA)

Billing services (SC)

Bank accounts (SC)

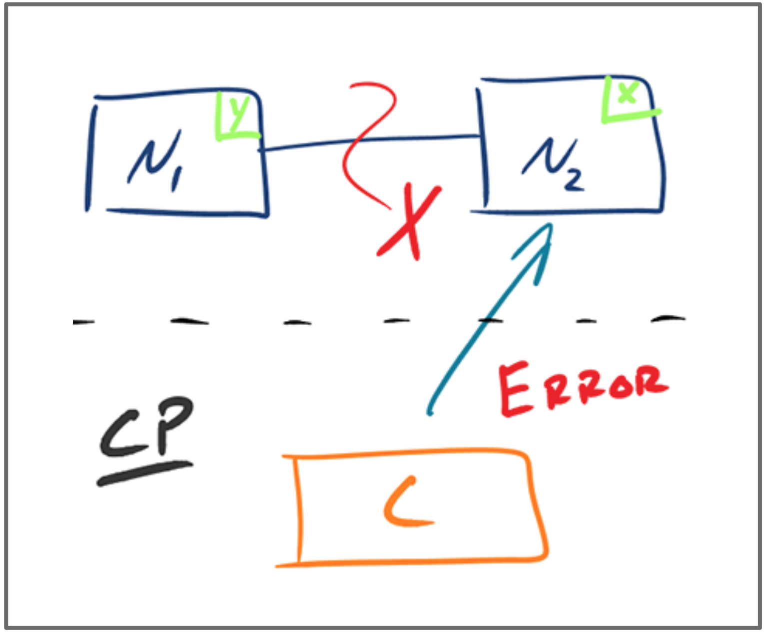
Both are desirable properties, but why can't we achieve both in a system?

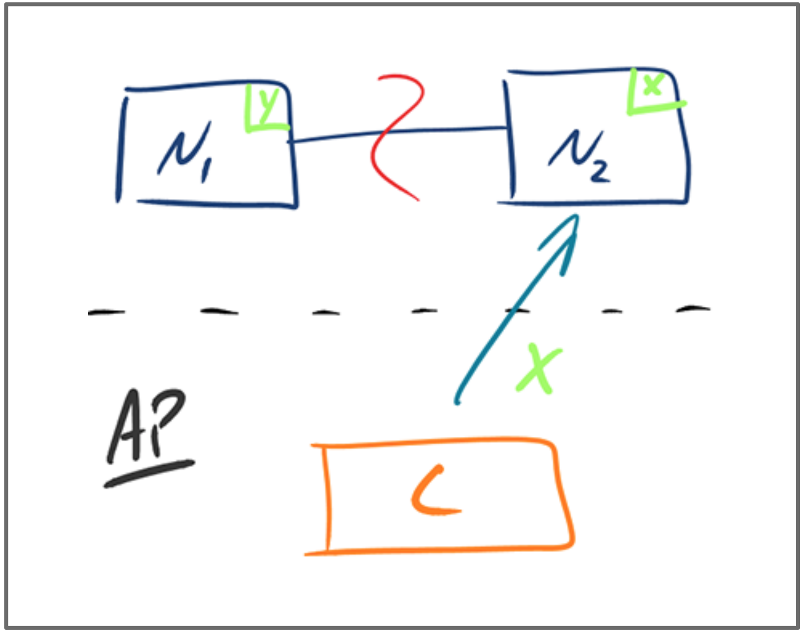
CAP Theorem



During network partitions, impossible to achieve both C and A

CAP Theorem





Availability is important

Tens of millions of customers at peak times

Tens of millions of shopping cart requests, 3 million checkouts per day

Hundreds of thousands of concurrently active sessions

Strict Service-Level Agreements (SLAs) translate to business value

Dynamo

Fully decentralized, highly available key-value store

Always writeable, resolve conflicts during reads

API for clients to specify requirements (99.9th percentile)

Departure from RDBMS: simpler functionality, fewer guarantees, runs on commodity hardware (low-end, broadly compatible, non-specialized machines)

Techniques for achieving availability

Consistent hashing for partitioning key space

Vector clocks for reconciling conflicts during reads

Sloppy quorums for handling temporary failures

Anti-entropy using Merkle trees for syncing key-value pairs

Gossip-based protocol for membership notifications

Techniques for achieving availability

Consistent hashing for partitioning key space

Vector clocks for reconciling conflicts during reads

Sloppy quorums for handling temporary failures

Anti-entropy using Merkle trees for syncing key-value pairs

Gossip-based protocol for membership notifications

Consistent Hashing

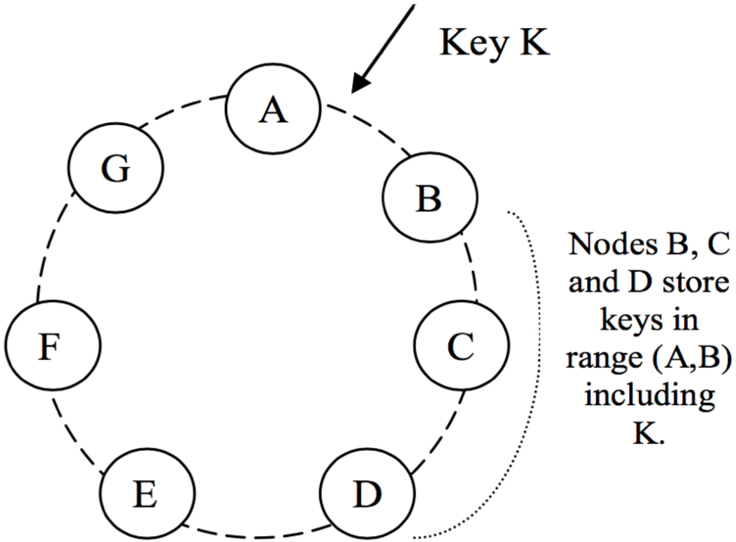
Assign each node a random position on the ring

Node owns the preceding key range

For fault tolerance, replicate each key at N successor nodes in the ring

Virtual nodes: each physical node gets assigned multiple nodes on the ring (e.g. B, D, F)

Virtual Nodes



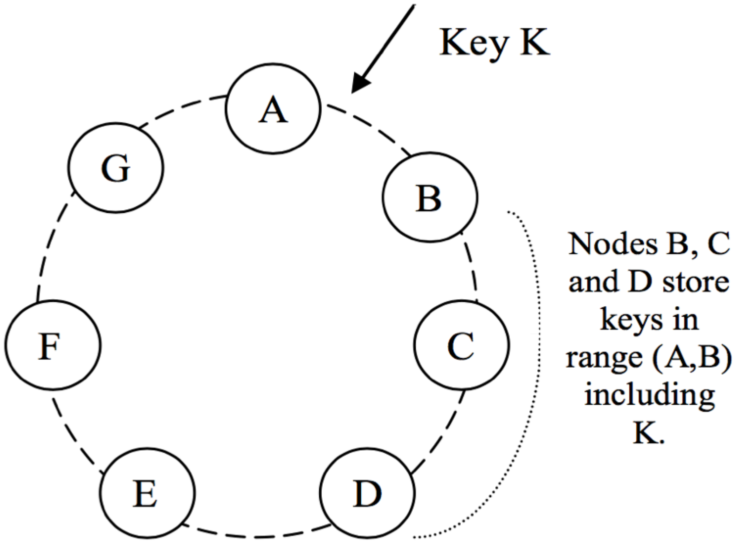
Consistent Hashing

Desirable properties?

Uniform distribution of load

Minimum object movements when nodes join or leave the ring

Number of virtual nodes can be adjusted for device heterogeneity



Techniques for achieving availability

Consistent hashing for partitioning key space

Vector clocks for reconciling conflicts during reads

Sloppy quorums for handling temporary failures

Anti-entropy using Merkle trees for syncing key-value pairs

Gossip-based protocol for membership notifications

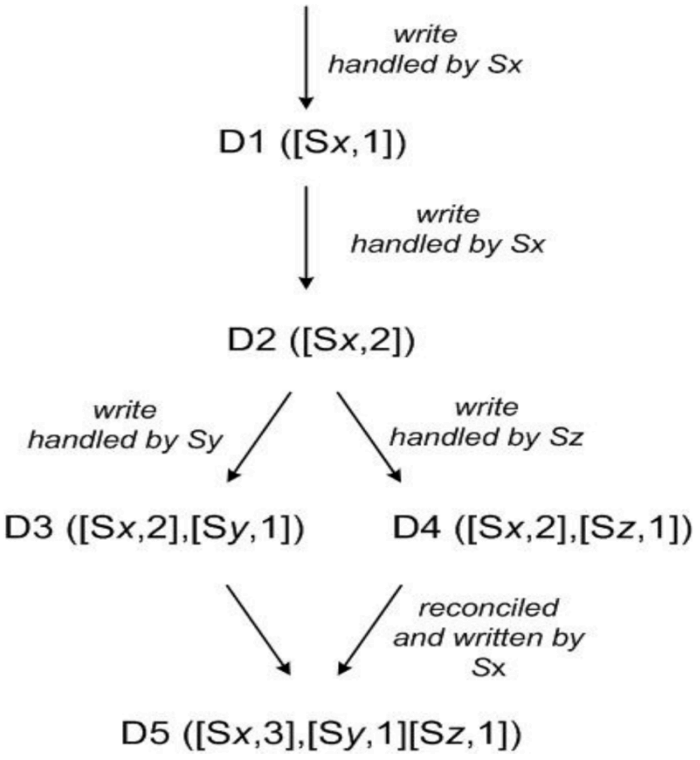
Conflict resolution

Two machines write different values to the same key

Vector clocks: list of (node, count) pairs where count is incremented on write

If one vector clock subsumes another, discard older value

Else, return all conflicting values to client



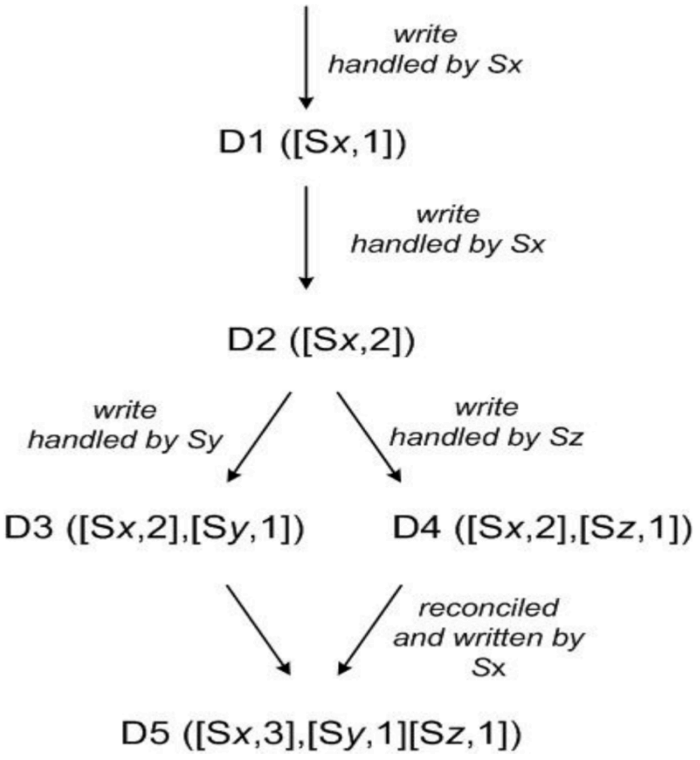
Context contains vector clocks

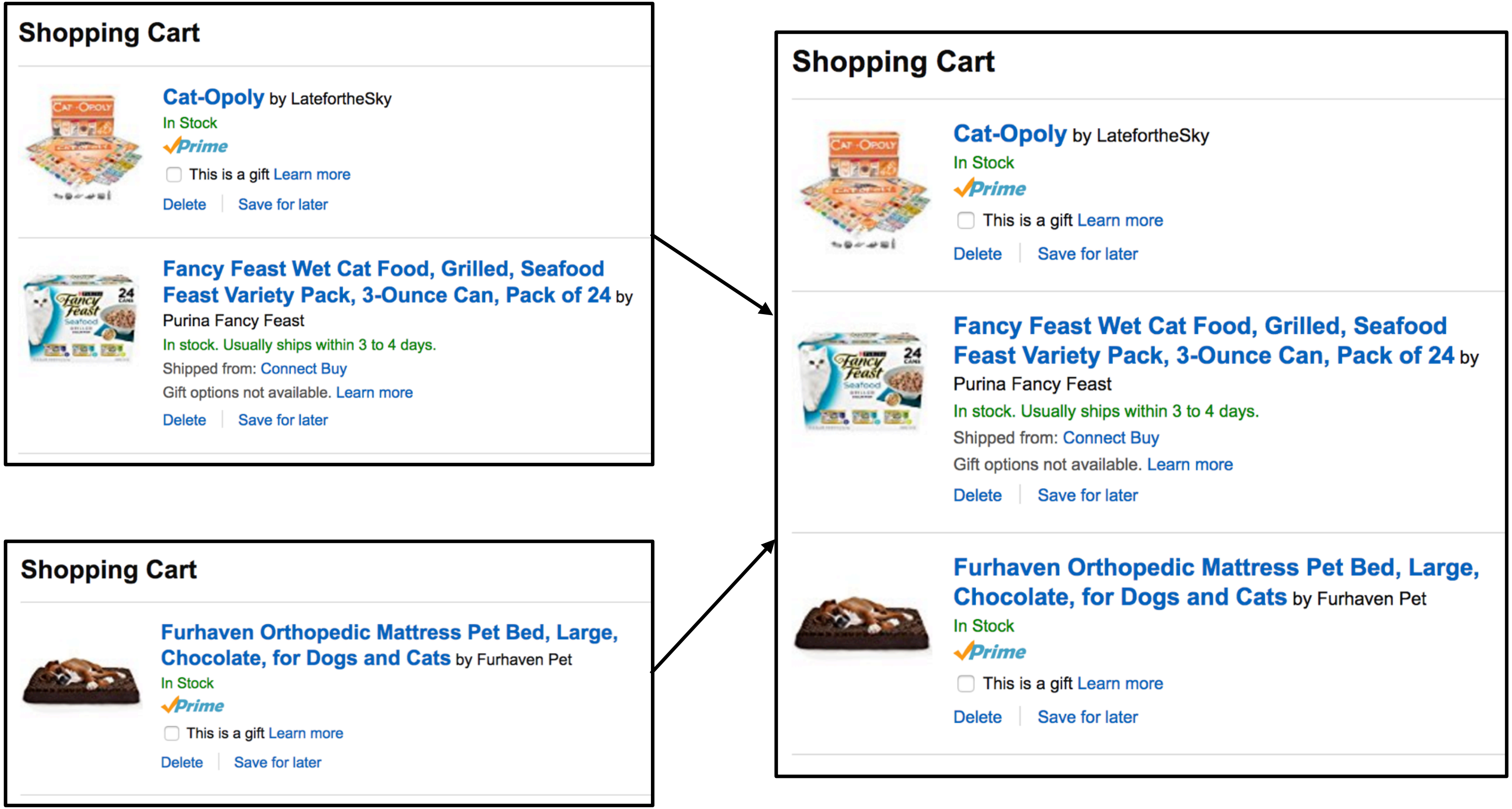
Dynamo client API is simple:

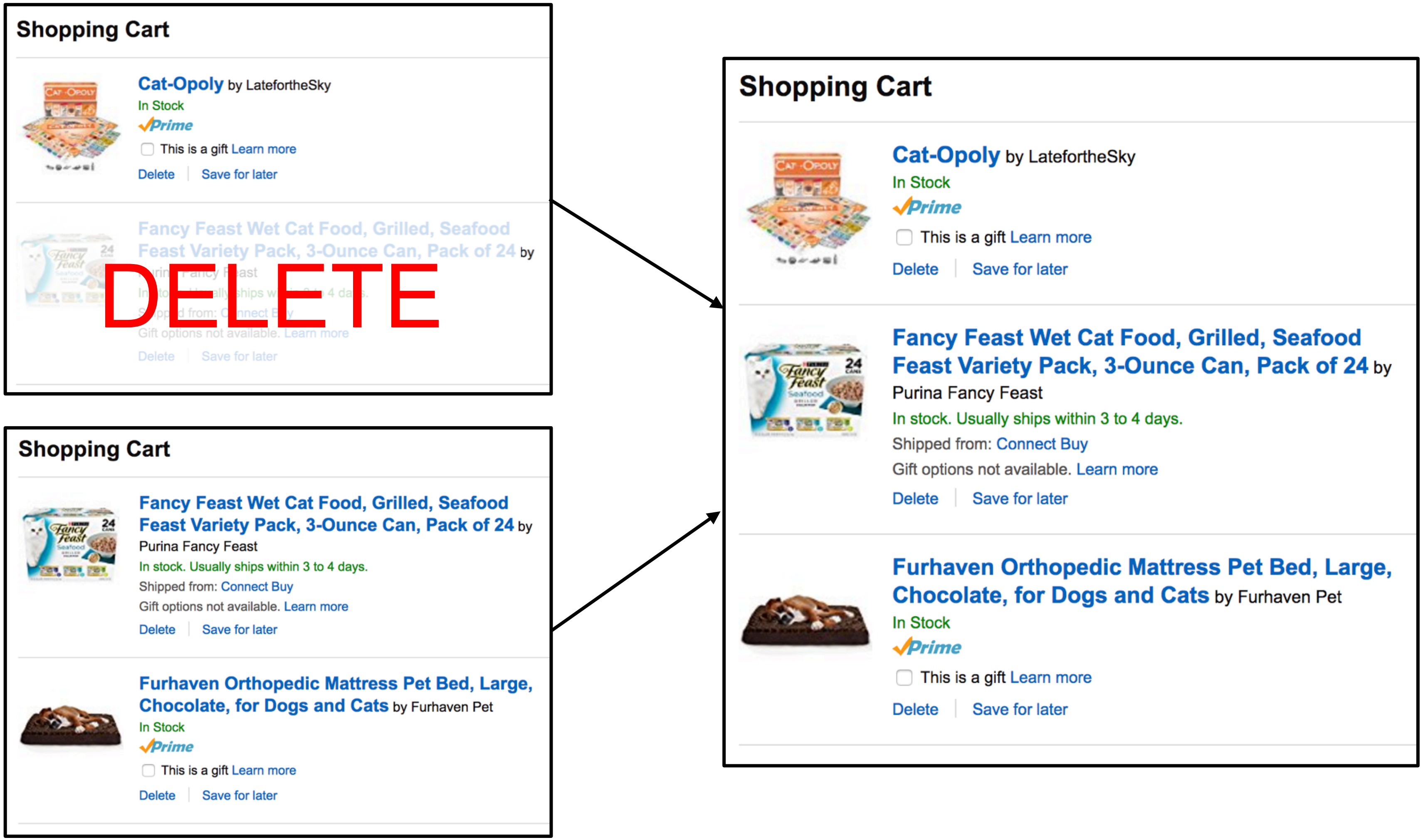
```
get(key) (value, context)

put(key, value, context)
```

Common pattern: put after get




Shopping Cart
Cat-Opoly by LatefortheSky
In Stock
Prime
This is a gift Learn more
Delete
Save for later
Fancy Feast Wet Cat Food, Grilled, Seafood Feast Variety Pack, 3-Ounce Can, Pack of 24 by Purina Fancy Feast
In stock. Usually ships within 3 to 4 days.
Shipped from: Connect Buy
Gift options not available. Learn more
Delete
Save for later
Shopping Cart
Furhaven Orthopedic Mattress Pet Bed, Large, Chocolate, for Dogs and Cats by Furhaven Pet
In Stock
Prime
This is a gift Learn more
Delete
Save for later
Shopping Cart
Cat-Opoly by LatefortheSky
In Stock
Prime
This is a gift Learn more
Delete
Save for later
Fancy Feast Wet Cat Food, Grilled, Seafood Feast Variety Pack, 3-Ounce Can, Pack of 24 by Purina Fancy Feast
In stock. Usually ships within 3 to 4 days.
Shipped from: Connect Buy
Gift options not available. Learn more
Delete
Save for later
Furhaven Orthopedic Mattress Pet Bed, Large, Chocolate, for Dogs and Cats by Furhaven Pet
In Stock
Prime
This is a gift Learn more
Delete
Save for later


Shopping Cart
Cat-Opoly by LatefortheSky
In Stock
Prime
This is a gift Learn more
Delete
Save for later
Fancy Feast Wet Cat Food, Grilled, Seafood Feast Variety Pack, 3-Ounce Can, Pack of 24 by
DELETE
Gift options not available. Learn more
Delete
Save for later
Shopping Cart
Fancy Feast Wet Cat Food, Grilled, Seafood Feast Variety Pack, 3-Ounce Can, Pack of 24 by Purina Fancy Feast
In stock. Usually ships within 3 to 4 days.
Shipped from: Connect Buy
Gift options not available. Learn more
Delete
Save for later
Furhaven Orthopedic Mattress Pet Bed, Large, Chocolate, for Dogs and Cats by Furhaven Pet
In Stock
Prime
This is a gift Learn more
Delete
Save for later
Shopping Cart
Cat-Opoly by LatefortheSky
In Stock
Prime
This is a gift Learn more
Delete
Save for later
Fancy Feast Wet Cat Food, Grilled, Seafood Feast Variety Pack, 3-Ounce Can, Pack of 24 by Purina Fancy Feast
In stock. Usually ships within 3 to 4 days.
Shipped from: Connect Buy
Gift options not available. Learn more
Delete
Save for later
Furhaven Orthopedic Mattress Pet Bed, Large, Chocolate, for Dogs and Cats by Furhaven Pet
In Stock
Prime
This is a gift Learn more
Delete
Save for later

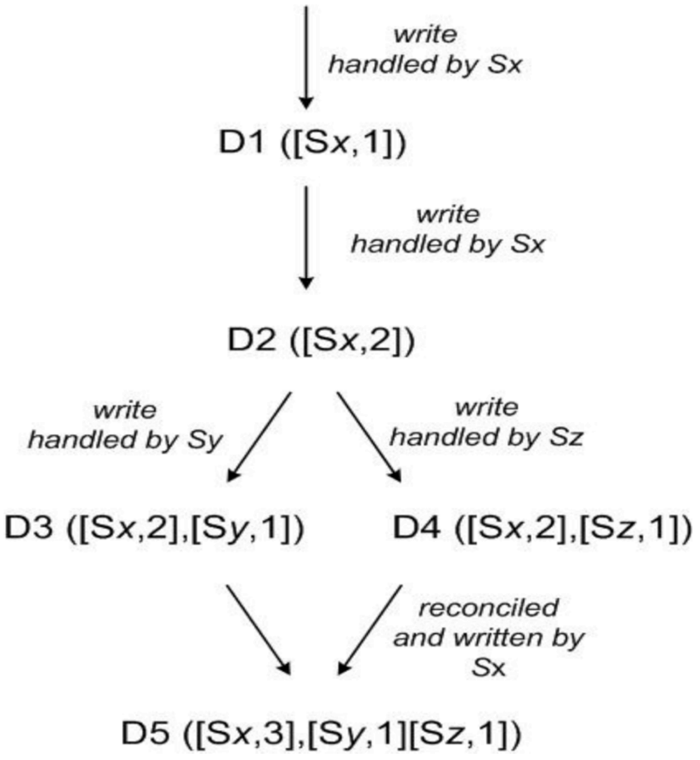
Conflict resolution

Two machines write different values to the same key

Vector clocks: list of (node, count) pairs where count is incremented on write

If one vector clock subsumes another, discard older value

Else, return all conflicting values to client



Techniques for achieving availability

Consistent hashing for partitioning key space

Vector clocks for reconciling conflicts during reads

Sloppy quorums for handling temporary failures

Anti-entropy using Merkle trees for syncing key-value pairs

Gossip-based protocol for membership notifications

Sloppy Quorums

Write to N nodes, return success when W < N nodes respond

Read from N nodes, return value(s) from R < N nodes

Typically, W+R > N means at least one writer and one reader overlap, so values are consistent

Sloppy here means skip nodes that have failed, such that even if W+R > N, the readers and writers may not overlap = not consistent!

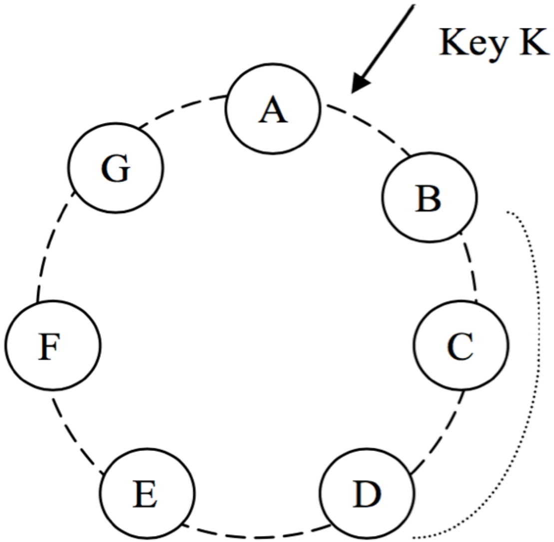
Sloppy Quorums

Example:

Typical values are N = 3, W = R = 2

Nodes C and D have failed, so key *k* is written to E and F instead

Nodes C and D recover, and now client tries to read from C and D = stale value



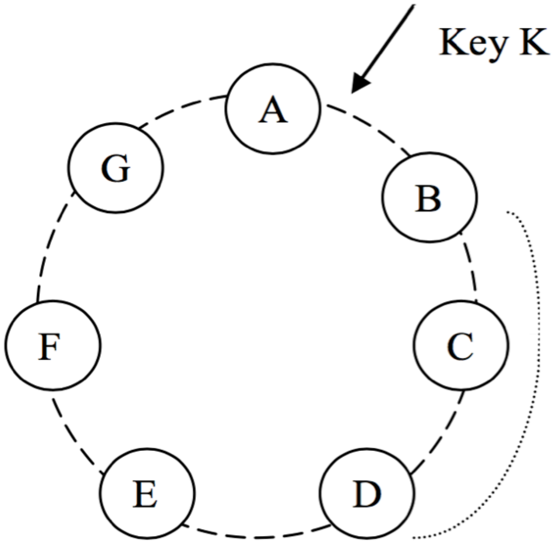
Hinted Handoff

"Hint" refers to the node the data originally belongs to

Example:

Nodes E and F remember they are writing on behalf of C and D

As soon as C and D recovers, E and F transfer their values for *k* to C and D



Sloppy Quorums

Write to N nodes, return success when W < N nodes respond

Read from N nodes, return value(s) from R < N nodes

Typically, W+R > N means at least one writer and one reader overlap, so values are consistent

Sloppy here means skip nodes that have failed, such that even if W+R > N, the readers and writers may not overlap = not consistent!

Techniques for achieving availability

Consistent hashing for partitioning key space

Vector clocks for reconciling conflicts during reads

Sloppy quorums for handling temporary failures

Anti-entropy using Merkle trees for syncing key-value pairs

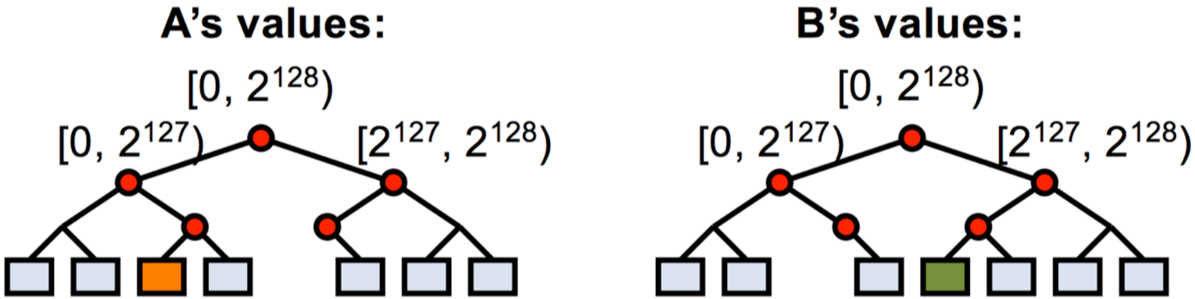
Gossip-based protocol for membership notifications

Anti-entropy using Merkle trees

Goal: minimize durability loss from above techniques

Nodes responsible for the same key spaces exchange Merkle trees

Find differences quickly while exchanging little information



Techniques for achieving availability

Consistent hashing for partitioning key space

Vector clocks for reconciling conflicts during reads

Sloppy quorums for handling temporary failures

Anti-entropy using Merkle trees for syncing key-value pairs

Gossip-based protocol for membership notifications

Membership notification

Gossip-based protocol to propagate membership changes

Each node learns the key spaces handled by all other nodes

Result: zero-hop distributed hash table (DHT)

Clearly not infinitely scalable, but storage requirement not a problem in practice

Bayou

What is it?

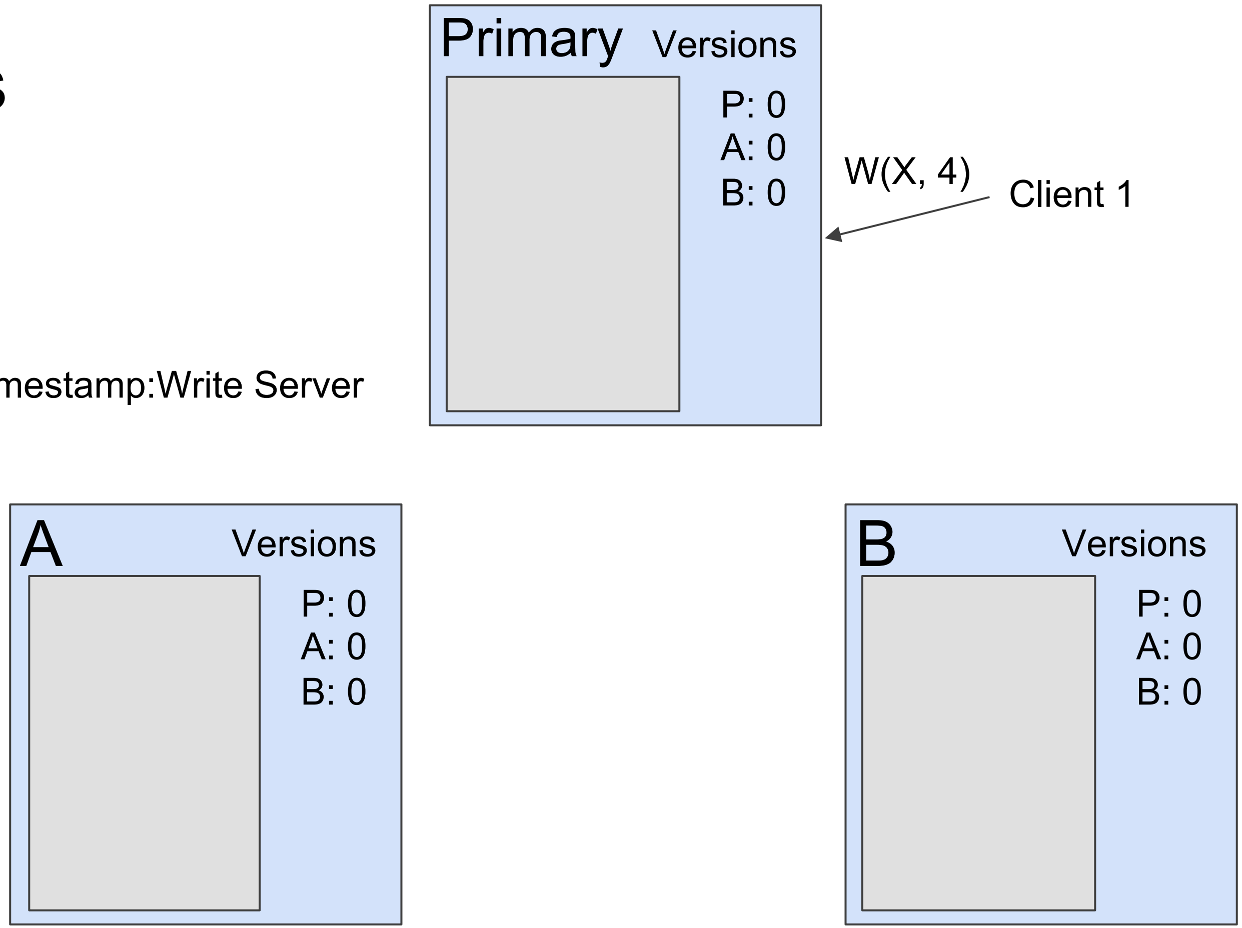
- Weakly consistent, replicated storage system

Goals:

- Maximize availability, *support offline collaboration*
- Minimize network communication
- Agree on all values (eventually)

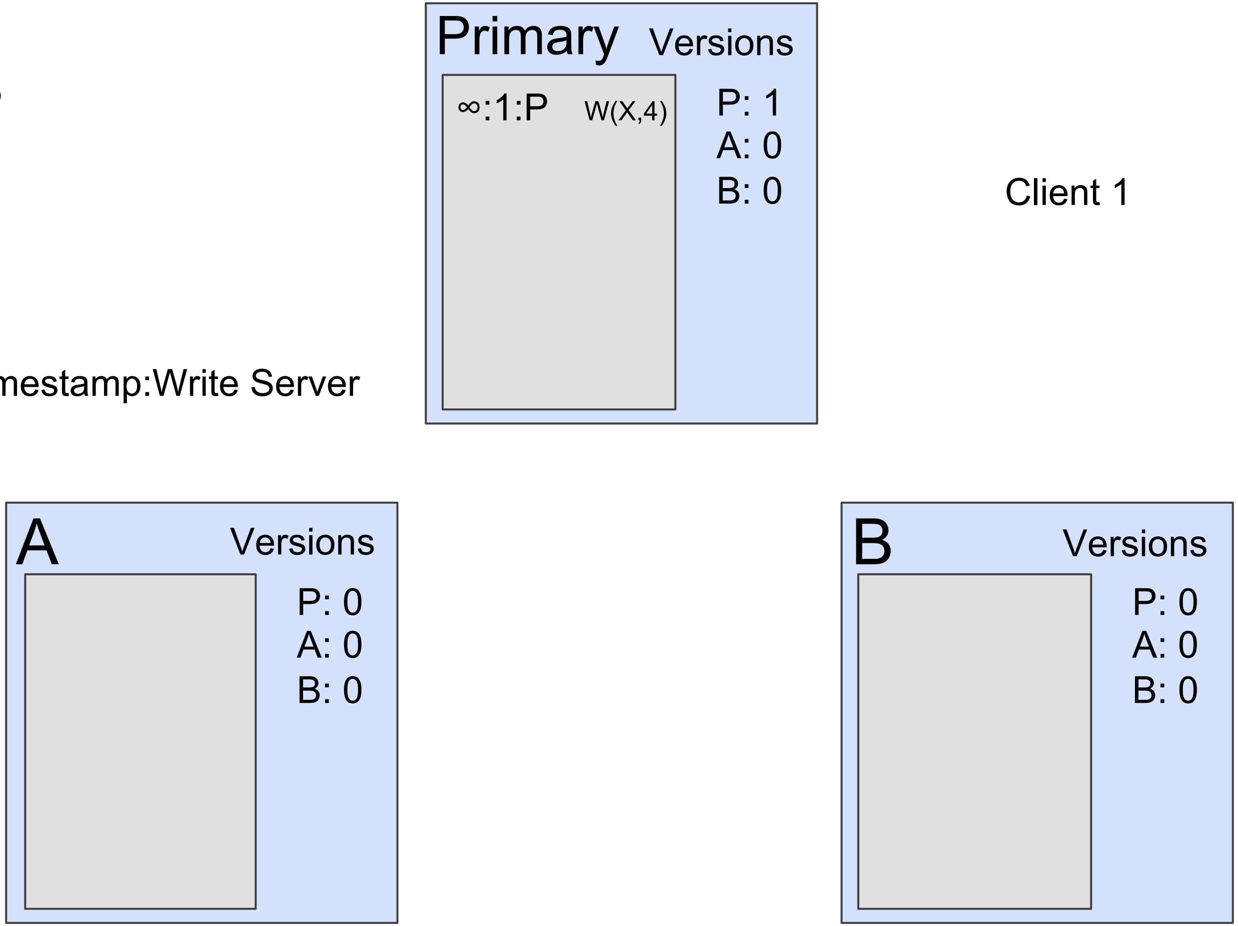
Bayou Writes

Legend
Commit Timestamp:Write Timestamp:Write Server



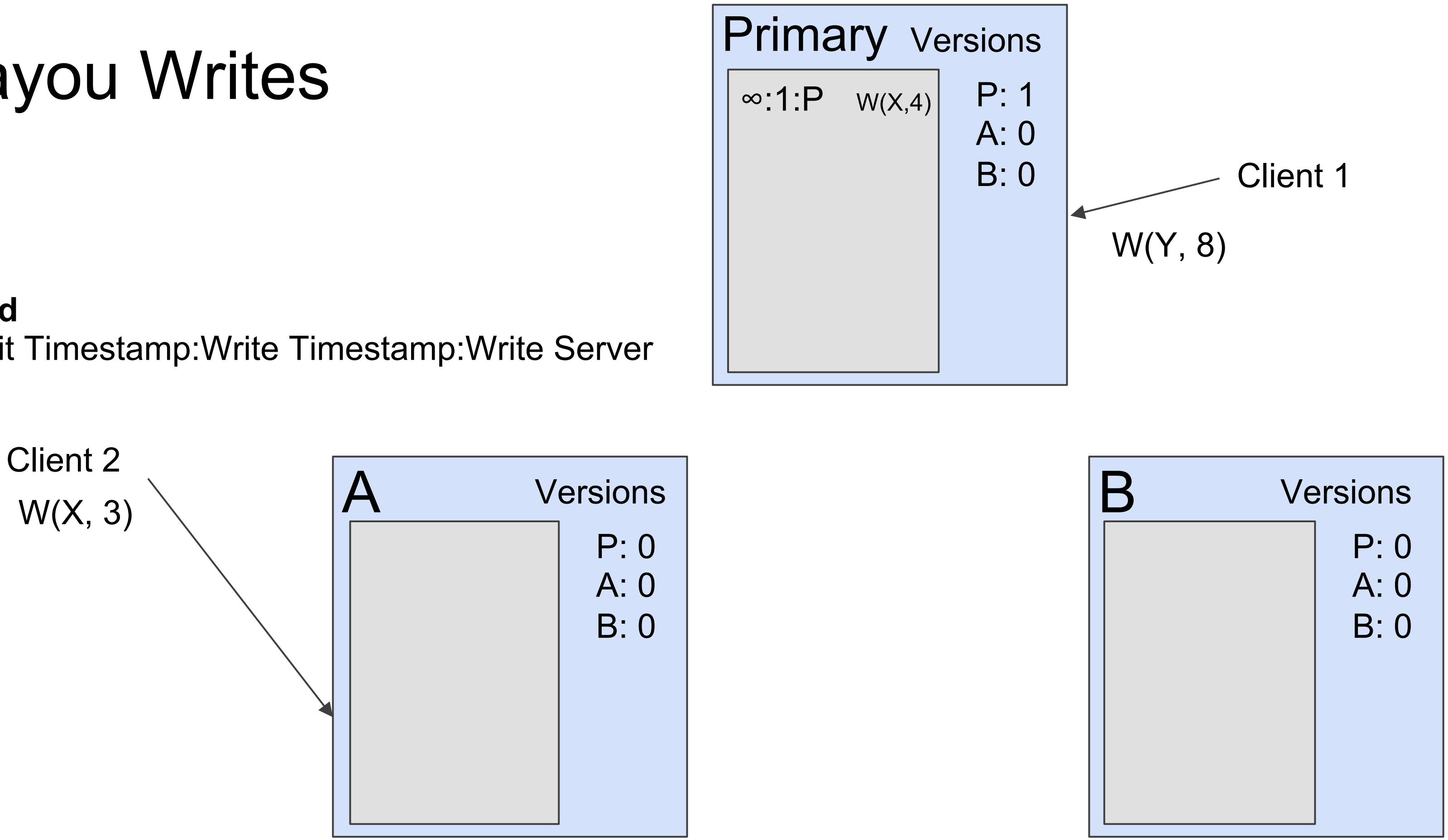
Bayou Writes

Legend
Commit Timestamp:Write Timestamp:Write Server



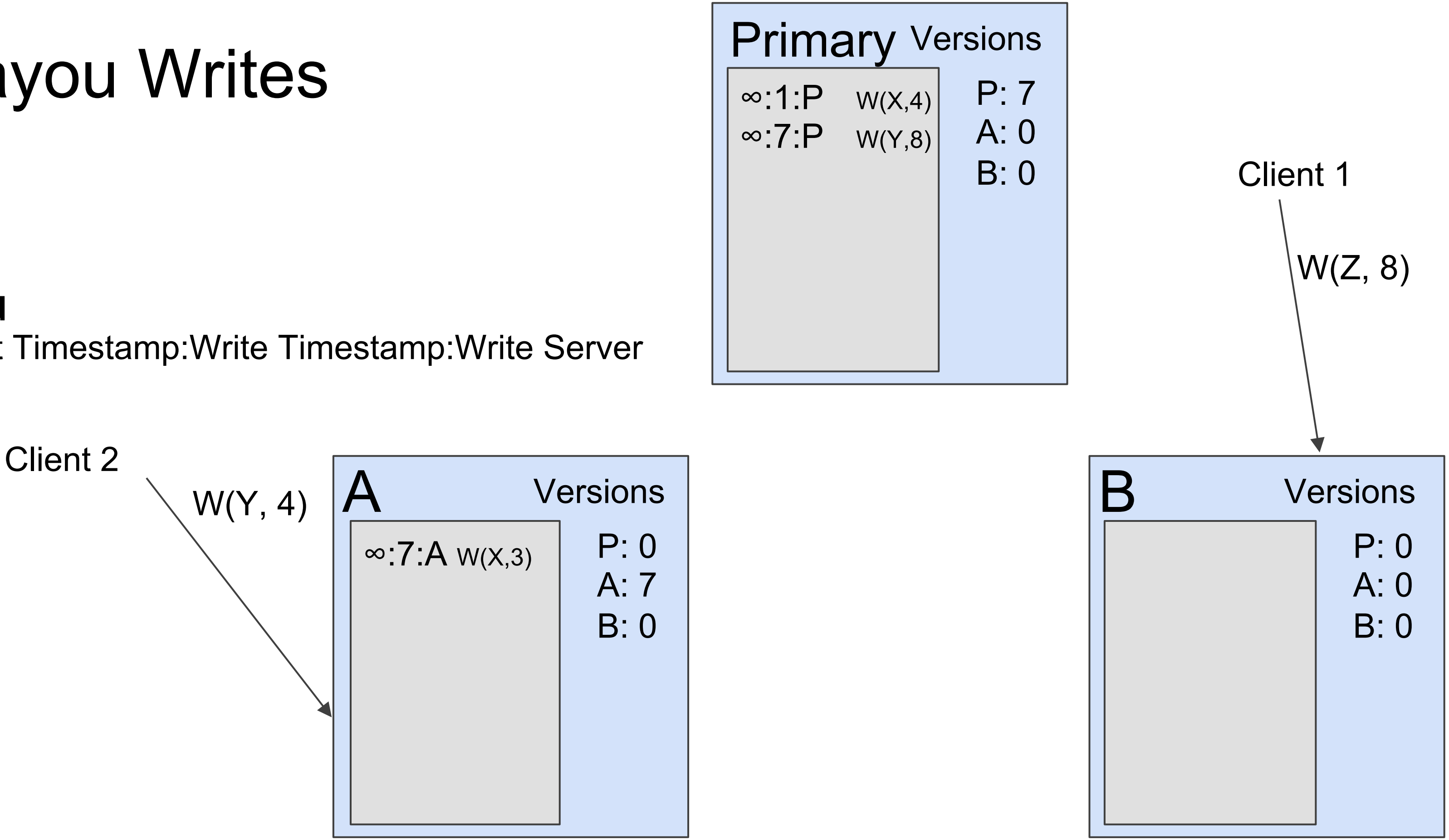
Bayou Writes

Legend
Commit Timestamp:Write Timestamp:Write Server



Bayou Writes

Legend
Commit Timestamp:Write Timestamp:Write Server



Bayou Writes

Primary Versions

∞:1:P W(X,4)
∞:7:P W(Y,8)

P: 7
A: 0
B: 0

Legend
Commit Timestamp:Write Timestamp:Write Server

A Versions

∞:7:A W(X,3)
∞:12:A W(Y,4)

P: 0
A: 12
B: 0

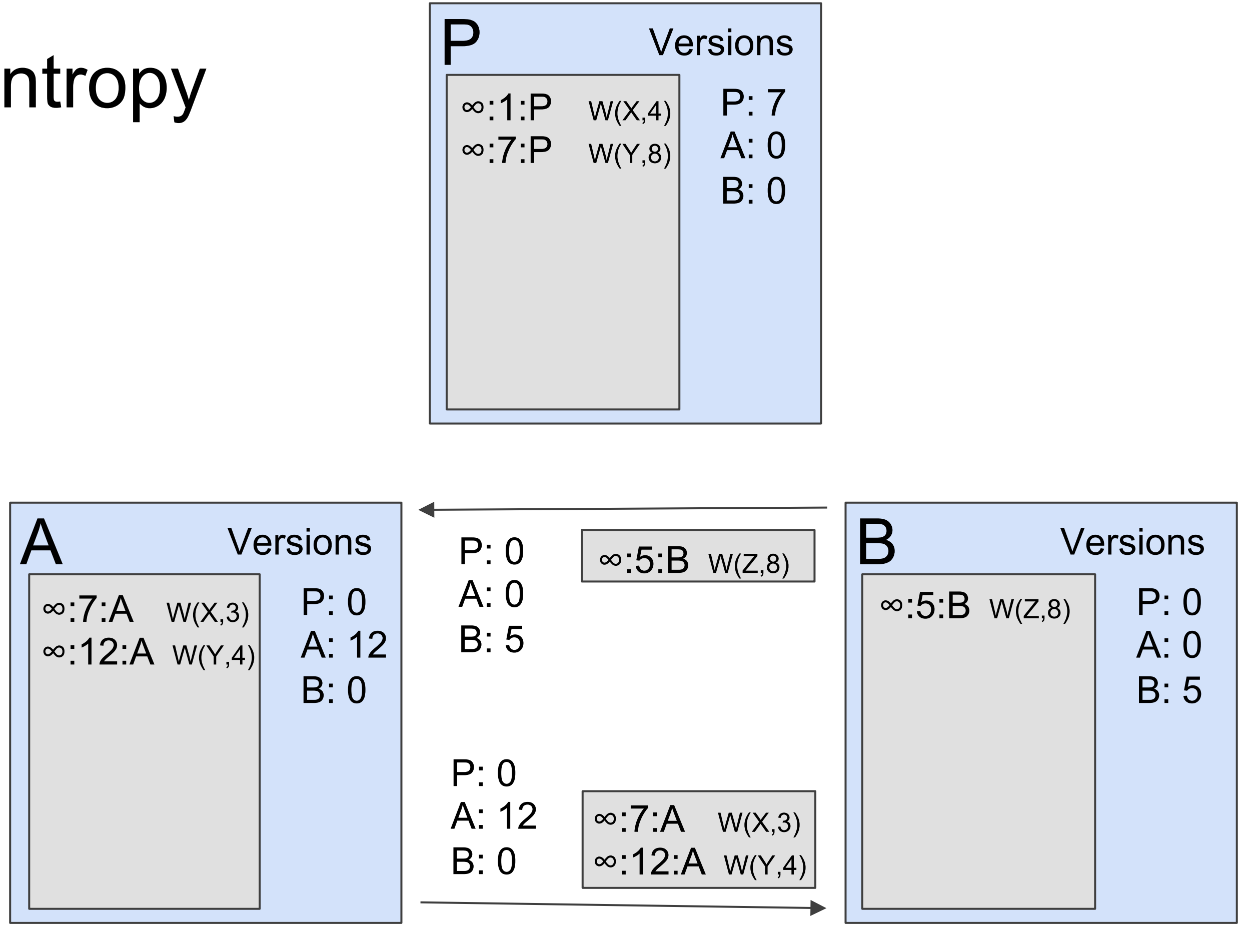
B Versions

∞:5:B W(Z,8)

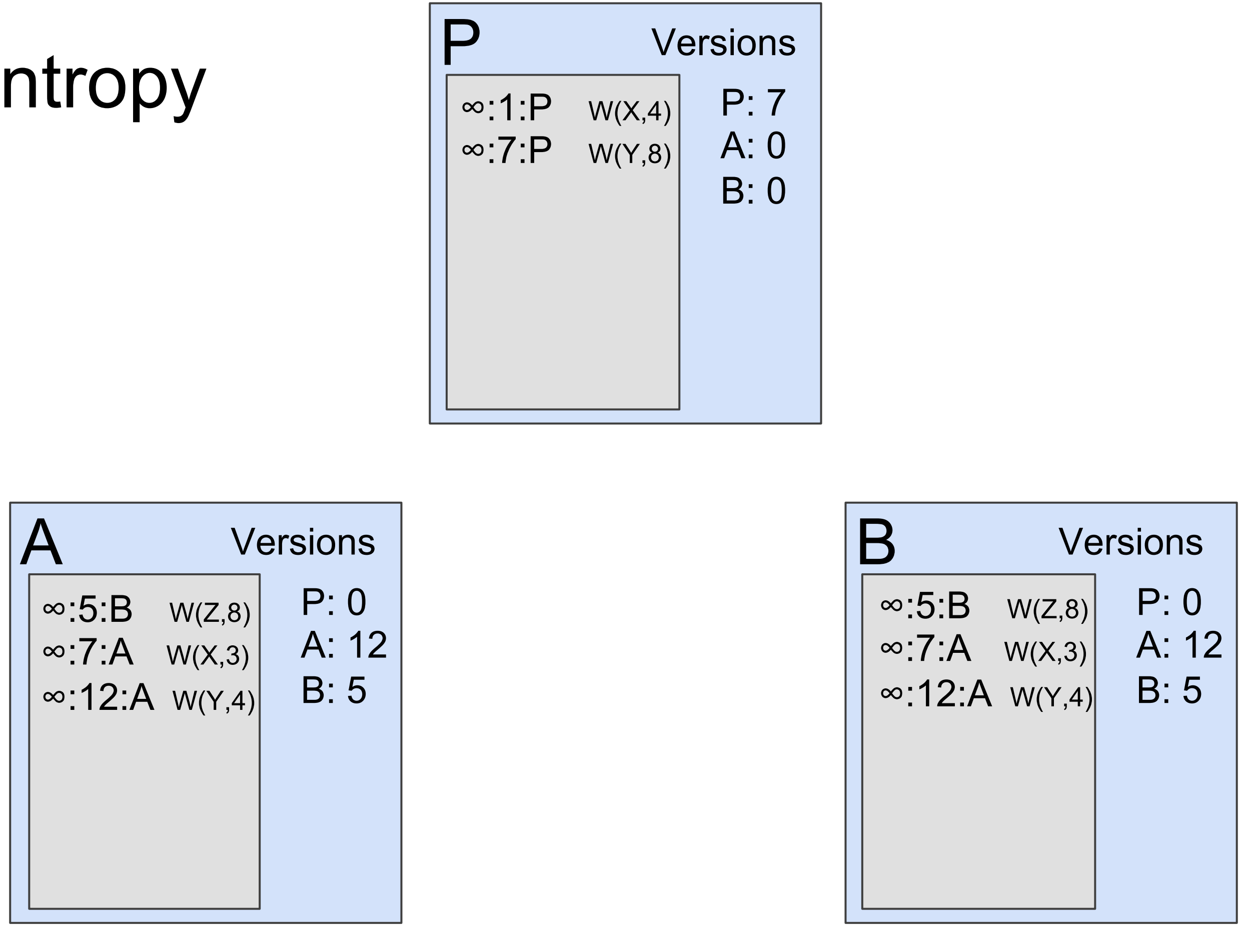
P: 0
A: 0
B: 5

Bayou Anti-Entropy

Anti-Entropy Session
A & B

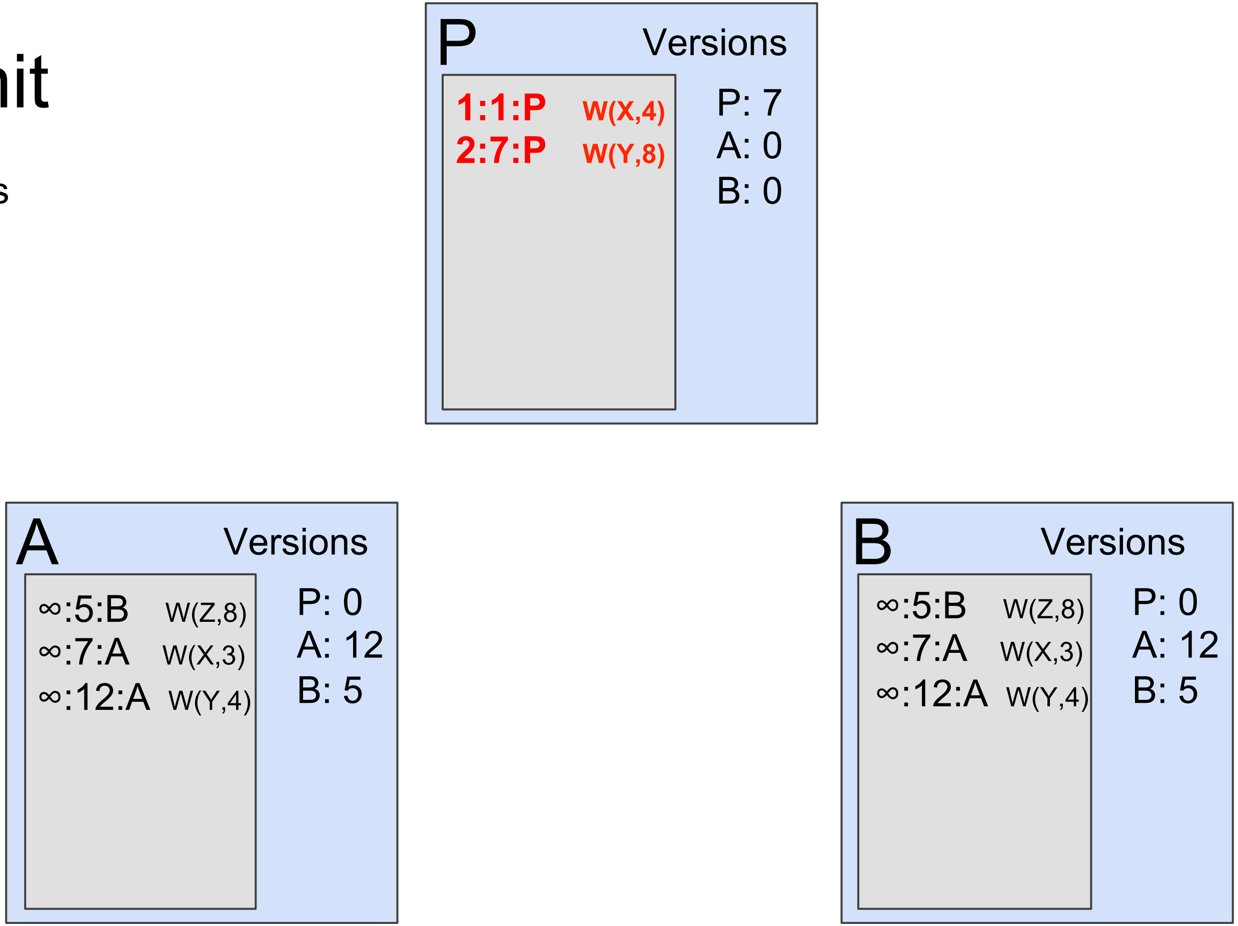


Bayou Anti-Entropy



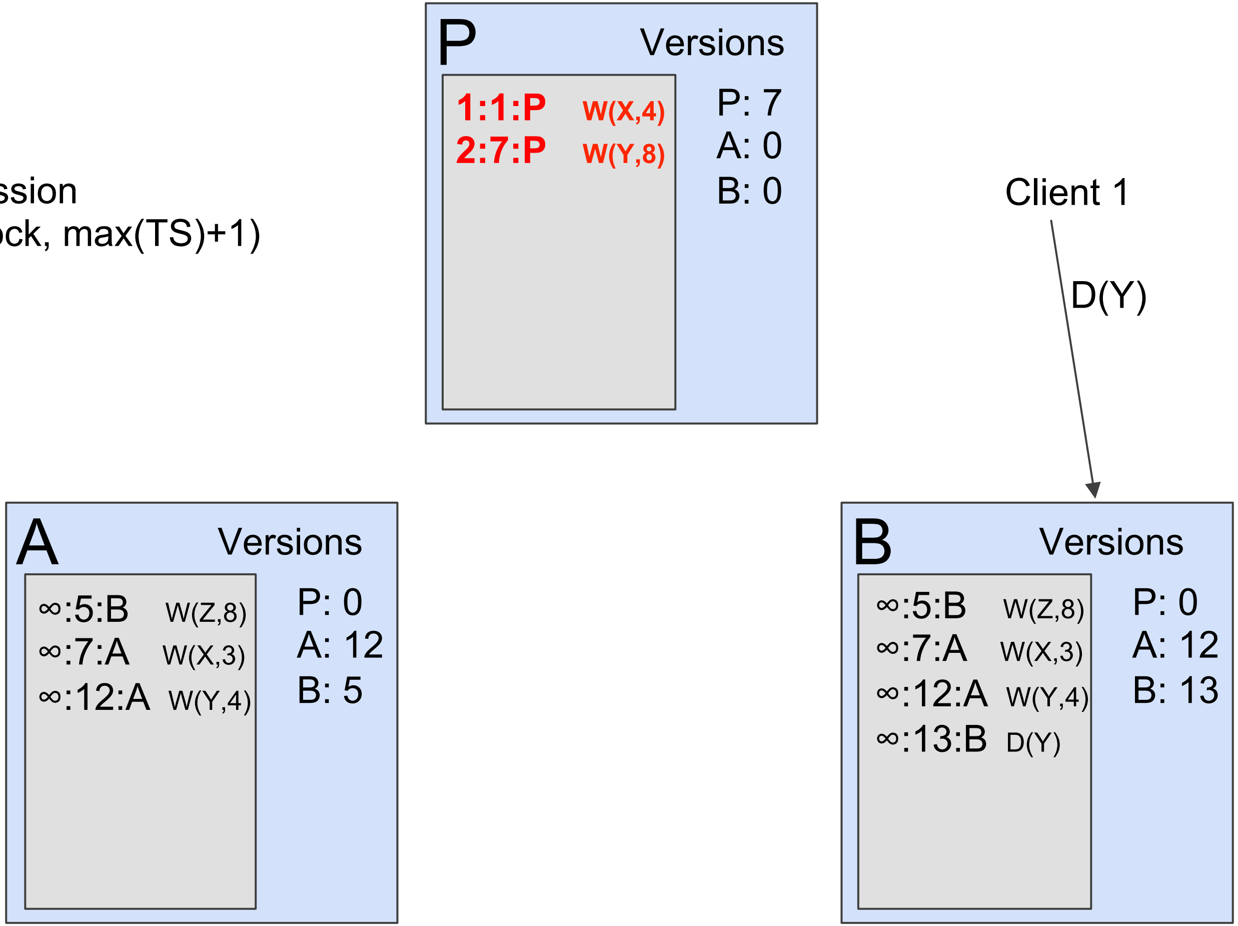
Bayou Commit

Primary commits its entries



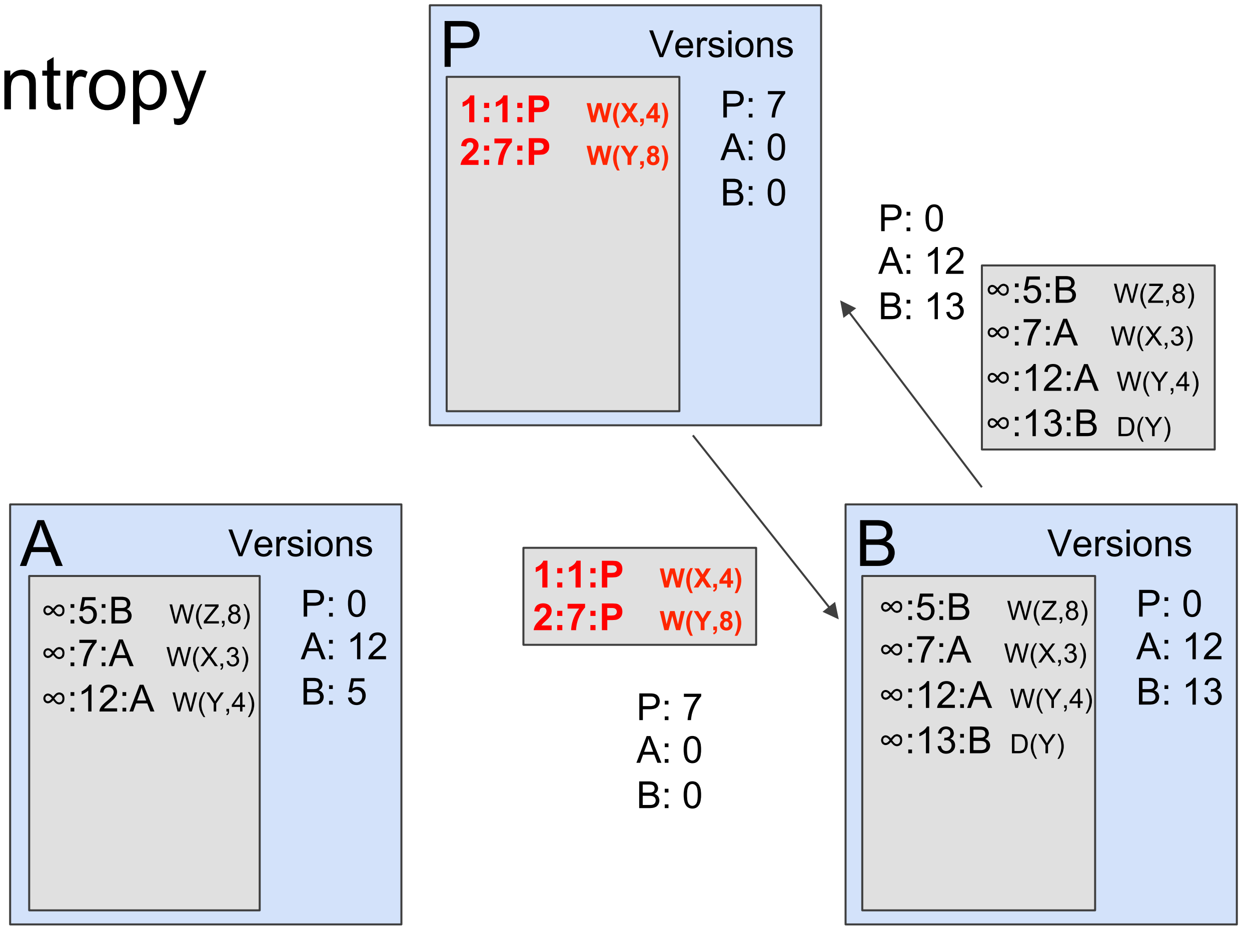
Bayou Write

Write after anti-entropy session
Write timestamp = max(clock, max(TS)+1)



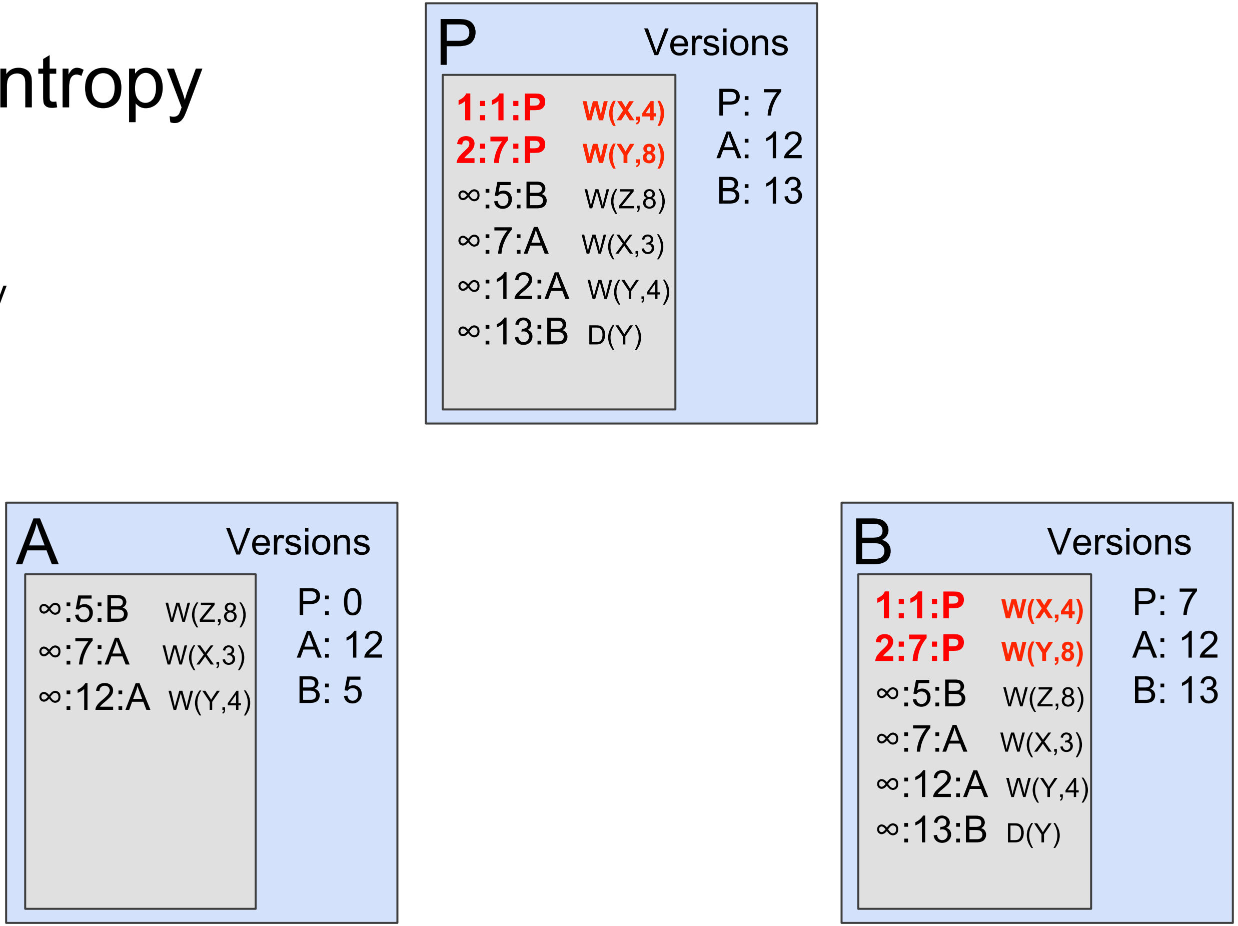
Bayou Anti-Entropy

Anti-Entropy Session
P & B



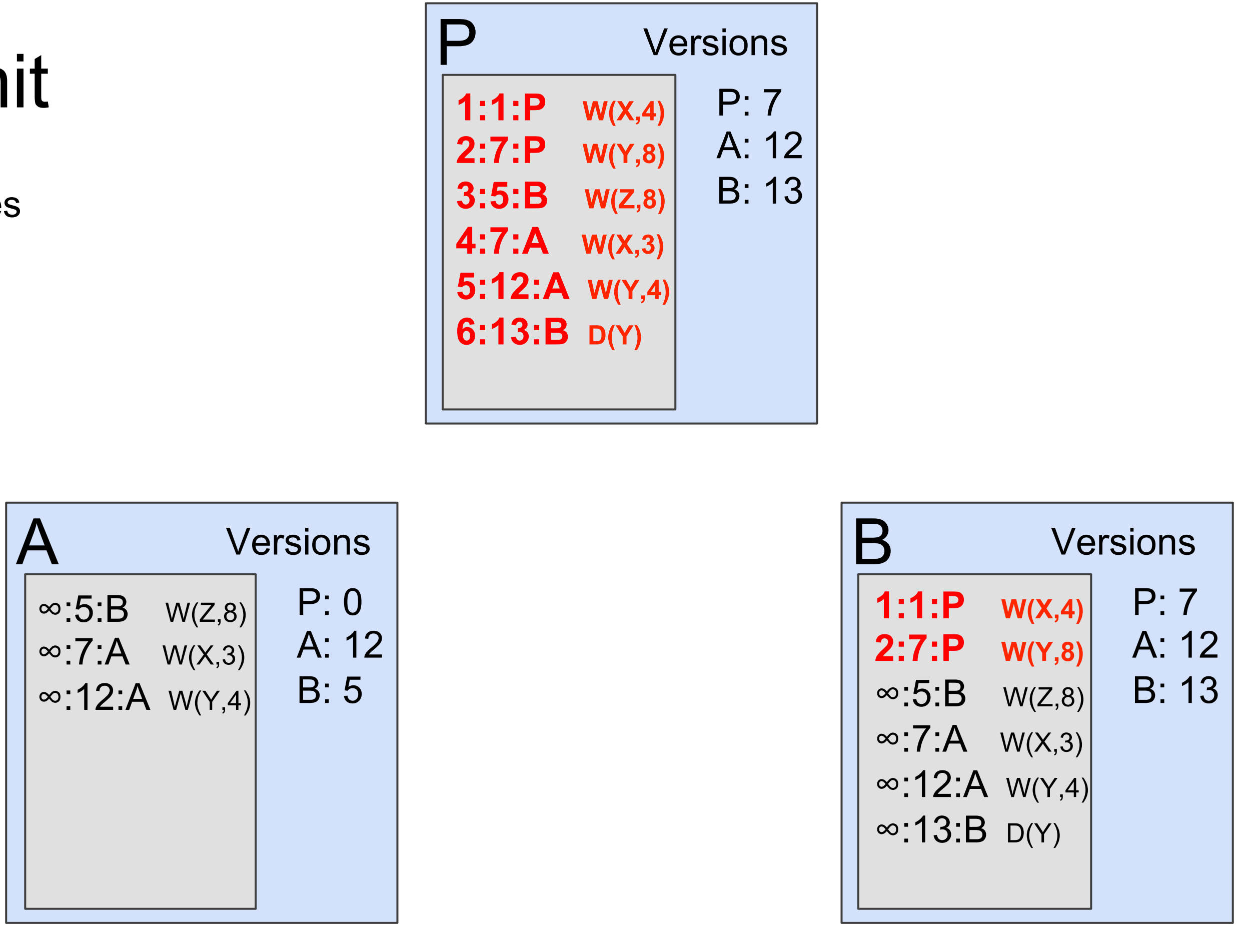
Bayou Anti-Entropy

Anti-Entropy Session
P & B
Primary respects causality



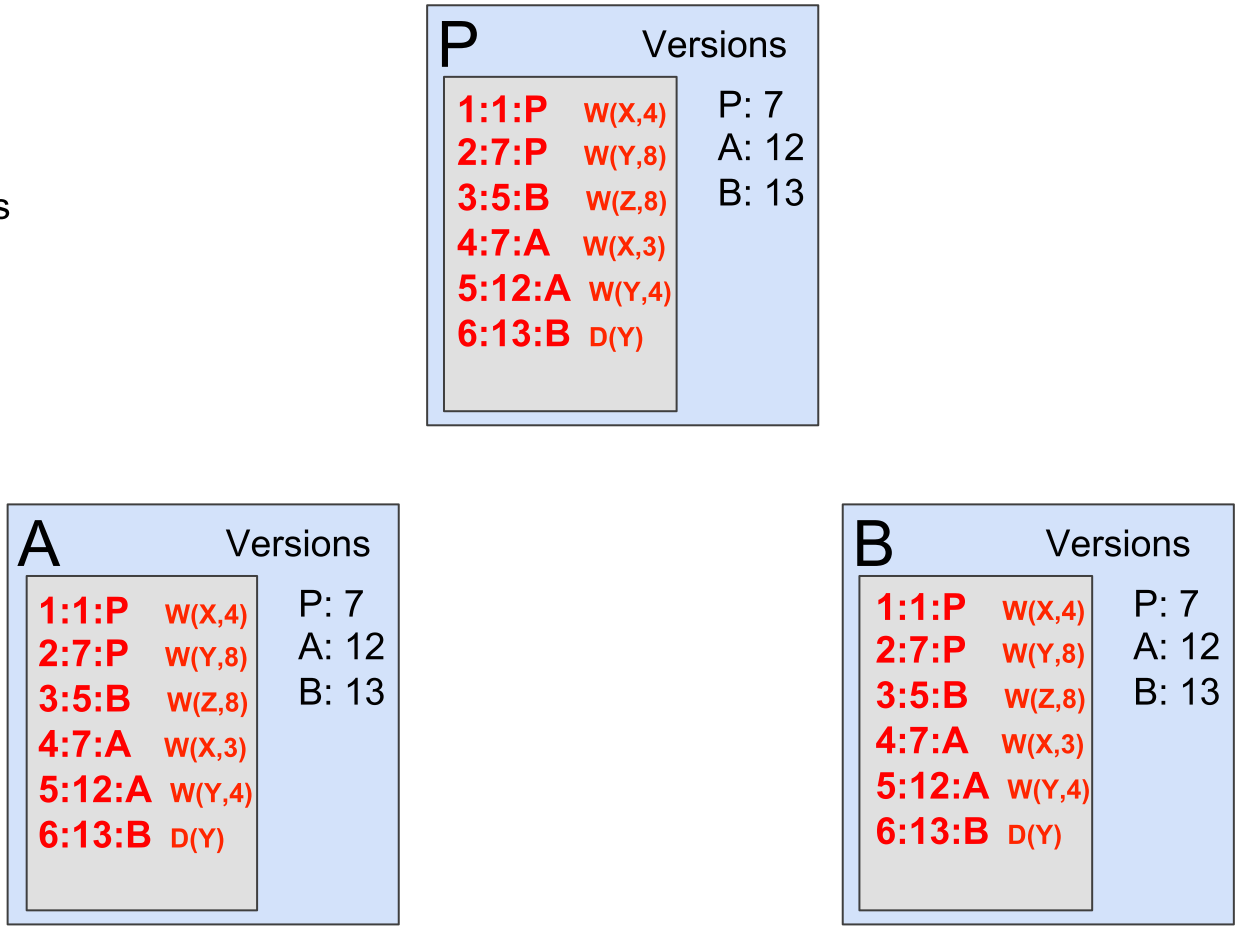
Bayou Commit

Primary commits Its entries



Bayou

After a number of commits
and anti-entropy sessions
(without further writes)



Bayou and Dynamo similarities

Anti-entropy to achieve eventual consistency

Exchange vector clocks to determine order of operations

Expose conflict resolution to application

High availability!