RSM + PNUTS Notes:

Replicated State Machines:

goal: - Reliable System from unreliable components
- Consistent Replicas of a server

Single-copy consistency: System behaves as if only one copy of server

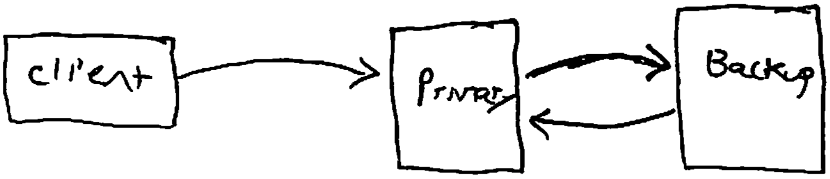
Eventual consistency: - Replicas eventually agree on state
- perform conflict resolution
- Might read stale Data.

How to achieve single-copy consistency?

Idea: - replicas start in same state
- Apply same operations in deterministic order

Key Issue: How to agree on order

one method : Primary Backup



- Primary determines order, forwards operations to Backup

- Backup Ignores client requests

- when primary dies, Backup is promoted to primary.

Problem: How does backup know primary dead and not just partitioned? How do clients learn backup is new primary?

want to avoid situation where 2 servers act as primary (split brain problem)

Soln: view server that monitors primary + Backup and performs promotion on failure.

Problem: - view server vulnerable

- want distributed concensus

Advanced Topic, take 6.824 to learn more

PNUTS:

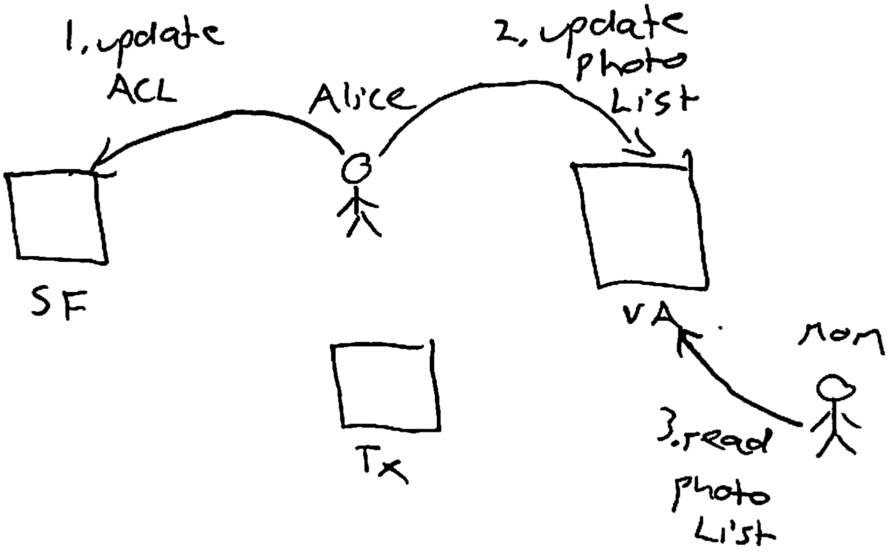
- Replicated Storage System across Wide Area

- Full replica in each region

- Not Single-Copy consistent (Too slow)
- Not Eventually consistent

Why not?

Spring Break problem:



Eventual consistency causes this short-term problem

Solution: Per-Record Timeline Consistency

- updates to single record happen in same order on all replicas

How?

- writes go to master Region for record

- Yahoo Message Broker at master region chooses order and guarantees same order applied at other regions

- If master not local, writes are slow. Therefore master changes to region w/ Majority of writes.

API:

- read-any(key): read local copy
  Fast, but might be stale

- read-critical(key, ver_n): read local copy if local version # ≥ ver_n
  useful for read-your-writes, can be Fast.

- read-latest (key): reads from master region

- test-and-set-write(key, value, ver_n):

  writes only if latest version # = ver_n

  can be used to increment a counter.

Practice problem on next page.

1 PNUTS Question

(From 6.824 Spring 2011 Quiz II)

You're running a PNUTS system. Records X and Y both start with value zero. Here are two functions that use the API described in Section 2.2 of the PNUTS paper:

```
fn1:
  x1 = read-any(X)
  x1 = x1 + 1
  write(X, x1) // X = x1
  write(Y, x1) // Y = x1

fn2:
  x1 = read-any(X)
  x2 = raed-latest(X)
  y1 = read-any(Y)
  print x1, x2, y1
```

You execute two calls to fn1, at different sites, at the same time. After both calls to fn1 have returned, you execute fn2 at a third site. There is no activity in the system other than described here, and no crashes or network failures.

What is it possible to see from fn2, given the design of PNUTS and the above scenario?

(A) 2, 2, 1

(B) 1, 2, 2

(C) 1, 1, 2

(D) 2, 1, 1

(E) 0, 0, 0

2 Answer

(A) **Yes:** fn1s happen in serial order. x1 and x2 represent latest X, y1 is slightly stale Y

(B) **Yes:** fn1s happen in serial order. x1 is slightly stale X, x2 is latest X, y1 is latest Y

(C) **No:** Y must be $\leq$ X, therefore y1 must be $\leq$ x2

(D) **No:** If latest X $= 1$, then stale X (x1) must be ≤ 1

(E) **No:** If both fn1s complete, latest X must be ≥ 1