Recitation 17: Consistency Models

MIT - 6.033

Spring 2021

Henry Corrigan-Gibbs

Plan

Intro: Consistency

Recitation Questions

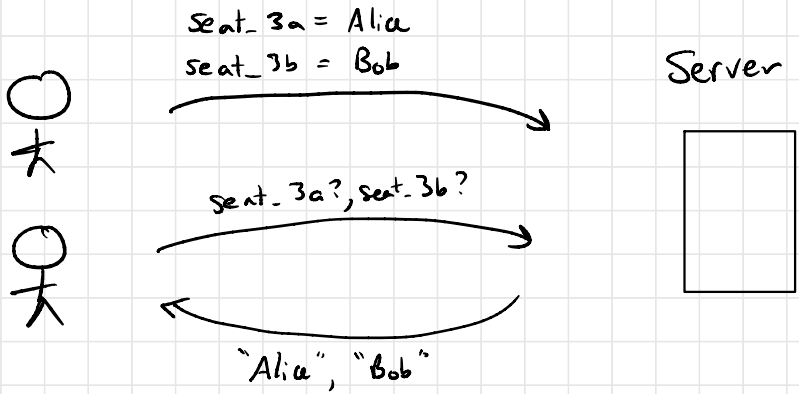
Types of consistency

Consistency Game

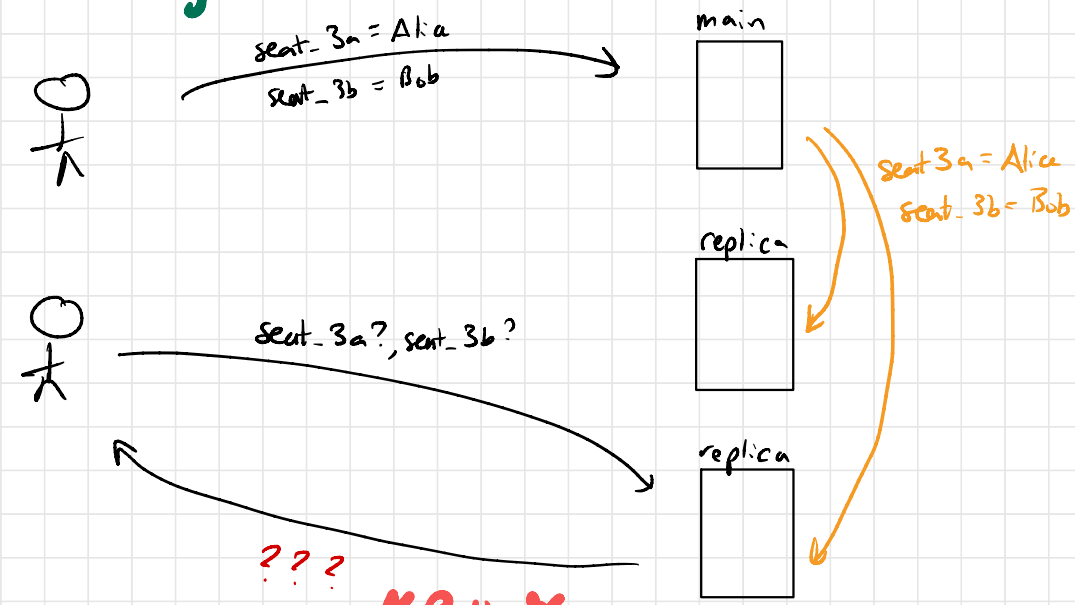
Logistics

* Welcome Back!
* Amir leading recitation on Thursday
* Reminder: S^3 is available for you!

In simpler times...



Nowadays...



✱Poll✱

↳ Which guarantees can lead to which answers?

Recitation Questions

1. What is a consistency guarantee? What aspects of a system does it affect?

   * The promise/contract that a storage system gives its user
     * Applies to multicore CPUs as well
   * Affects how app interacts w/ system

2. How does a system designer choose an appropriate consistency guarantee?

   * What options storage system offers { DynamoDB - strong - eventual $$$$
   * What app needs from storage system.

3. Why does the choice of a consistency guarantee matter?

   * Cost/performance !
   * Correctness
   * Ease of programming (don't underestimate)

Types of consistency

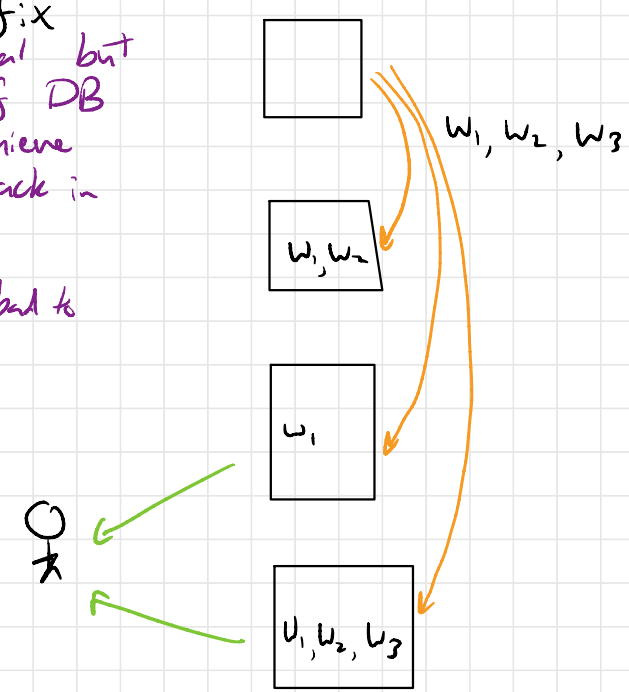
(N.B. These are fuzzy / debatable defns.)

Strong consistency

- See all writes
- Like one machine
- Easy to program against?
- Expensive

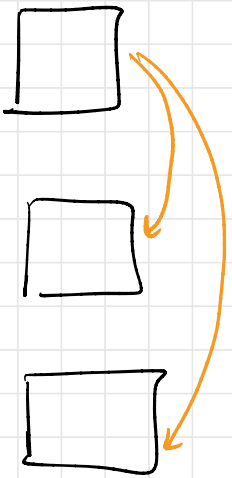
Consistent prefix

- See a sensical but stale view of DB
- Easier to achieve
- May travel back in time
- Perhaps not so bad to program against.



Bounded staleness

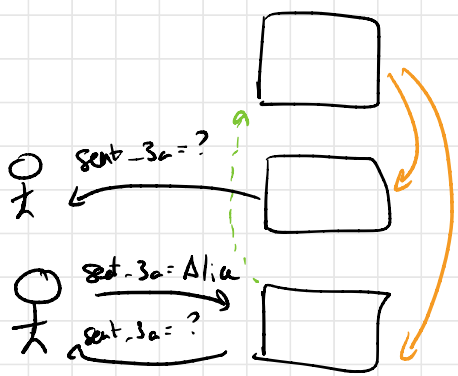
- Reads see oldish writes
- What guarantees do you get on recent writes?
  - ↳ unclear
- Maybe inexpensive to implement for certain time bounds
  - "good enough"



Some bound on latency = 1 sec

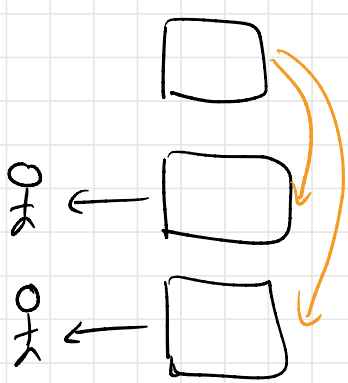
Read My Writes

- "Local consistency"
- If there's only a single writer, then this is good enough for writer
- Maybe not so bad to implement



Eventual consistency

- You get what you get!
- Maximum flexibility for storage service
- Not easy to program against in worst case... might get all sorts of inconsistent data



Amazon Dynamo DB

- Strong consistency or eventual
- Why only two options?

  Problems with giving developer many options

  * pushes burden on to app developer
  * If app changes, might need to change guarantees too (e.g. add a second score-keeper)

- Why eventual consistency if guarantee is so weak?

  ↳ Usually doesn't matter.

  e.g. # views on article, # clicks on ad, ...

  ↳ Cost of providing strong consistency >> Cost of stale data sometimes

  ↳ Usually, network is good, staleness is not worst case.

Consistency Game

AMBITIOUS! Will it work?! I can't stand the suspense!

We will run a system with

- eventual consistency
- strong consistency

and will measure (unscientifically) the perf diff.

WRITERS – Will write into DB by sending Amir Zoom DM in chat

s3 = _____ s4 = _____

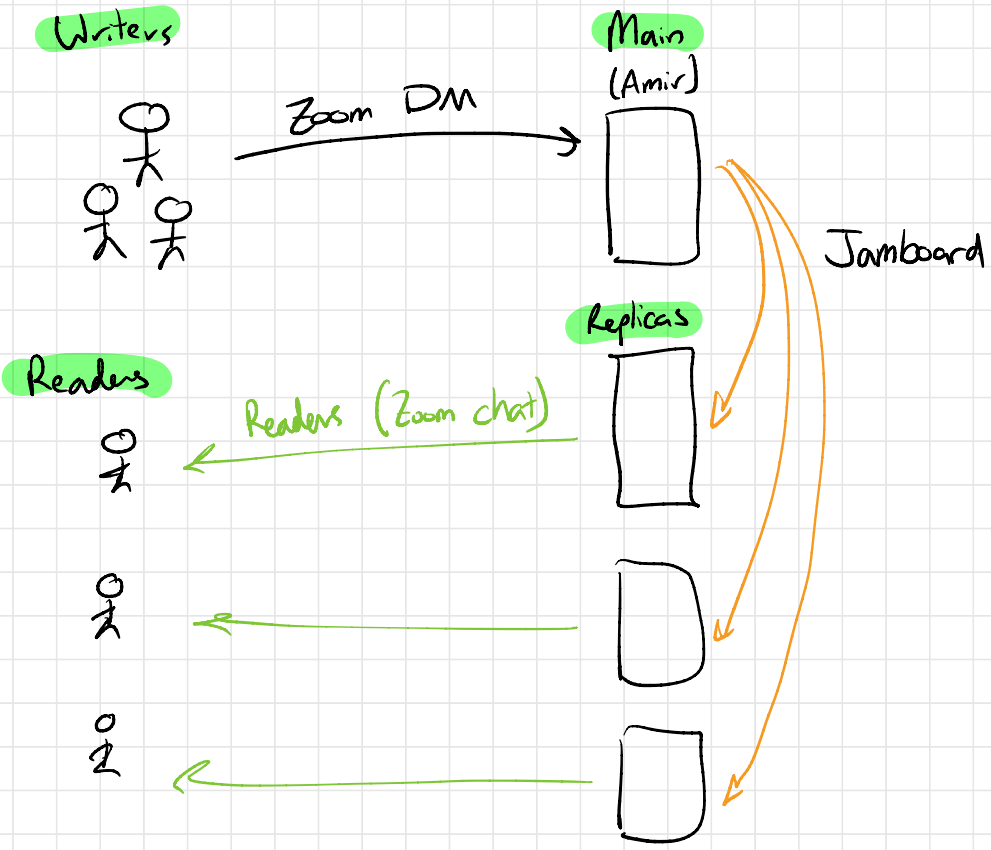
MAIN (Amir) – Will process writes by copying them out to the Jamboards of the replicas

READERS – Will make reads to DB by sending Zoom chat to everyone

r0 s3 = ?

REPLICAS – Will answer queries in public Zoom chat by reading off their jamboards

System Diagram (Figure 1)



1. Eventual consistency?
   - How to implement?
   - What QPS can we get?

2. Strong consistency?
   - How to implement?
     ↳ What can go wrong?
     - Use one replica
     - Shard data across replicas (txns?)
     - Use 2PC
   - What QPS can we get?