Grove

Reconfigure(newServers []Address)

each epoch is 1 configuration
- activates after each server is set up
  - 1st server becomes primary

invoked when there's a failure but can be called @ any time

ex/ if a replica crashes, puts will never complete

1. seal old server & copy state
2. send state to the other servers
3. set primary
4. activate epoch

switch 3 & 4

Problem: (b/c server is primary)
client sends put & it completes
reconfigure is called, an old server state is copied & sent
put is lost during 2nd reconfigure

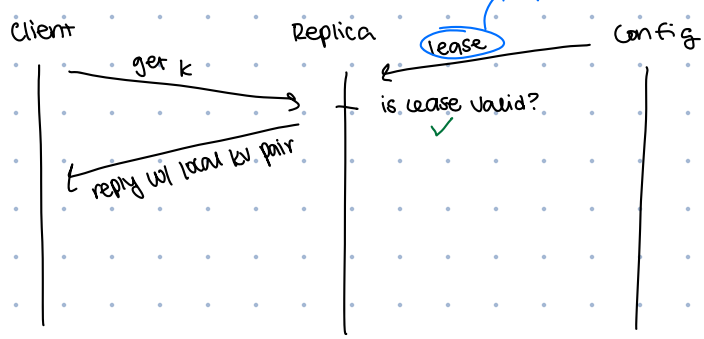
Reads

How do I know if a server is up to date?

1. can ask every other server
   - bad
2. Lease
   - something is true for x amount of time

   Q: when does x time end? clocks are not in sync

   A: GetTime() → (lower, upper) range for time



(happens periodically)

Problem: stale state

Primary can be in the middle of replication

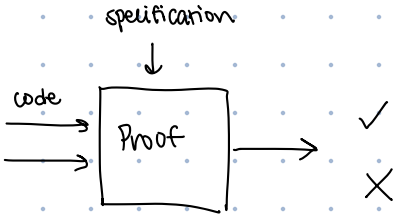
reply only goes through if lease is valid (no other server w/ a more updated config)

A: replica.nextIndex (# of writes so far)
- check that server has # commits

commit info is forwarded

Correctness of System

Formal verification
- won't check deadlock



Generalized Resources

currEpoch ↦ e : resource that owns epoch
- config service

currEpoch ≥ e :
- owned by all replicas

Proof:

accepted ↦ l resource that tracks accepted ops for each server

committed ↦ l globally committed list of ops that clients can access
- after calling apply, committed should contain that op
- used in proof specification

{ pre-condition }
Apply(op)
{ committed ≥ l + op }

also: committed ≥ l b/c append-only

part of spec for linearizability

ex/ sealed server has
accepted ↦ □l
read-only list

Lease Correctness

- proving no new epochs

Reconfigure must wait for all leases to expire b4 incrementing epoch

Prove Get

rep