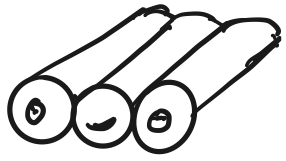
Raft: An RSM Protocol



Final Project Proposals

- Group Size
- Topics
- Proposal Length

Midterm Observations

- Succintness:

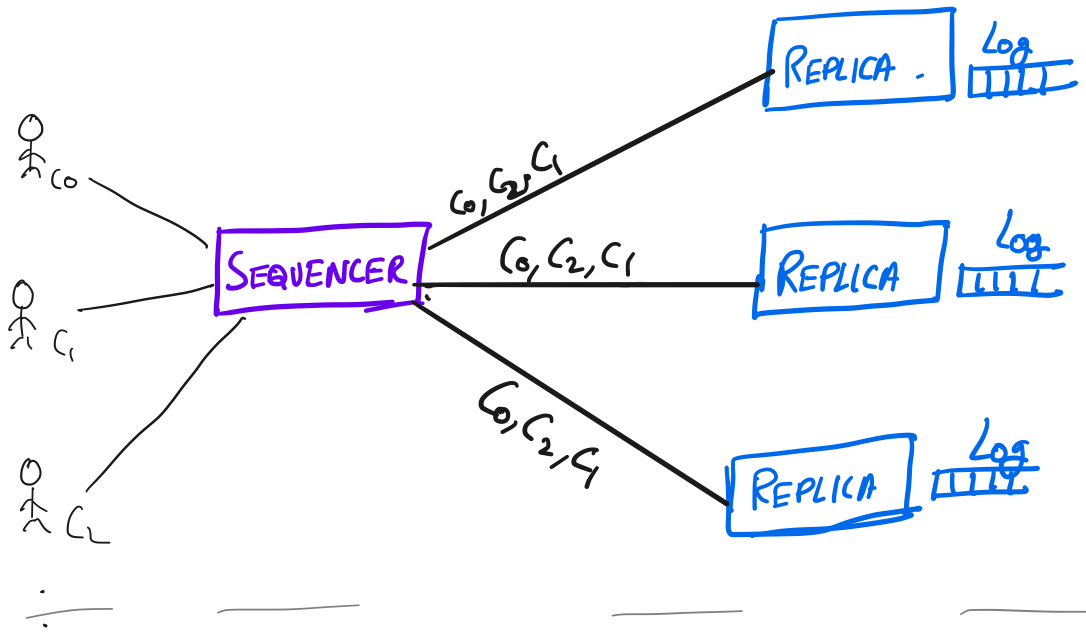
  Don't Tell Me About Safety & Liveness
  In A Question About Fairness

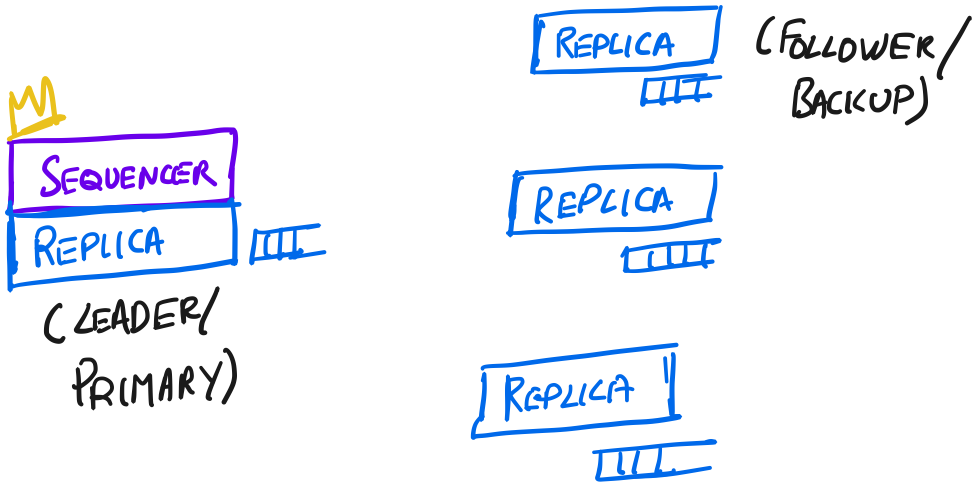
- On Proofs
  - ↳ Can Assume Anything Covered In Class
    Don't Need To Prove FLP
  - → But Must Show Why
    Why Is WFD Consensus?
  - → Must Be Logically Consistent

Where We Last Left Off

- State Machines
- Replicated State Machines
- Consensus
  - ↳ Impossibility & Workarounds

Today: An RSM Protocol





INVARIANTS

① LEADER'S LOG IS 'AUTHORITATIVE'

PROTOCOL MAKES SURE REPLICA LOGS == LEADER LOGS

② IF A MAJORITY OF REPLICAS HAVE LOG ENTRY E AT INDEX I, THEN ALL FUTURE LEADERS WILL HAVE LOG ENTRE E @ I

⇒ SAFE TO EXECUTE COMMAND AT INDEX I ONCE

(a) I-1 EXECUTED

(b) I REPLICATED BY MAJORITY

(i) LEADER ELECTION PROTOCOL ENSURES STABILITY OF A SUFFICIENTLY REPLICATED ENTRY

(ii) LEADER COUNTS # OF TIMES AN ENTRY IS REPLICATED & DECIDES WHEN COMMANDS ARE EXECUTED

(3) (RAFT SPECIFIC: LOG COMPLETENESS) IF INDEX I IS COMMITTED THEN INDEX I-1 MUST BE COMMITTED

SPLIT PROTOCOL INTO FOUR PORTIONS

(1) LOG SYNCHRONIZATION
HOW THE LEADER ADDS TO THE LOG

(2) FAILURE DETECTION
HOW FOLLOWERS DETECT LEADER FAILURES

(3) FAILURE RECOVERY
HOW A NEW LEADER IS CHOSEN

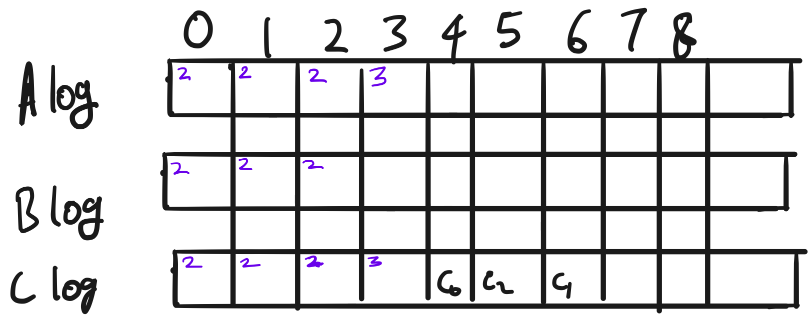
(4) [illegible]

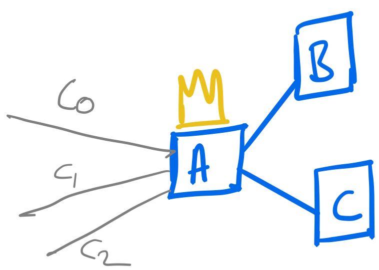
④ RECONFIGURATION

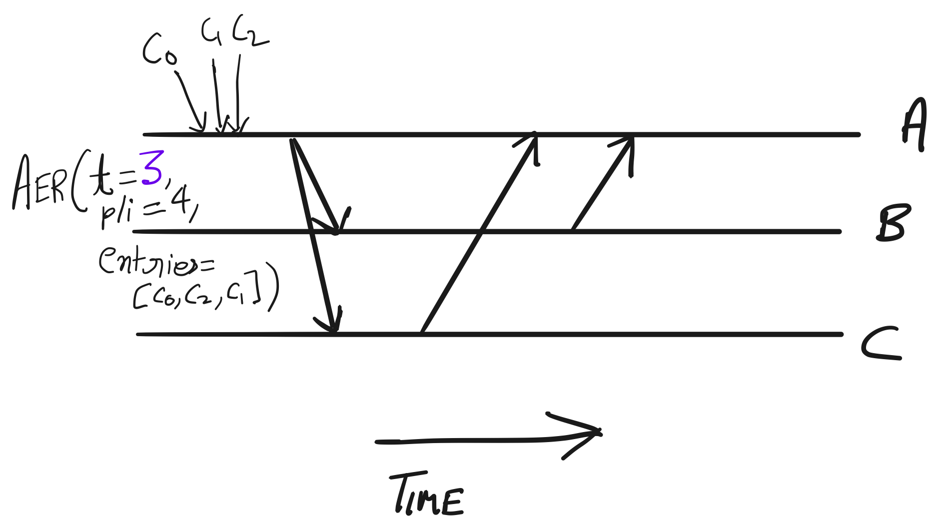
ADDING NEW MACHINES & REMOVING EXISTING ONES

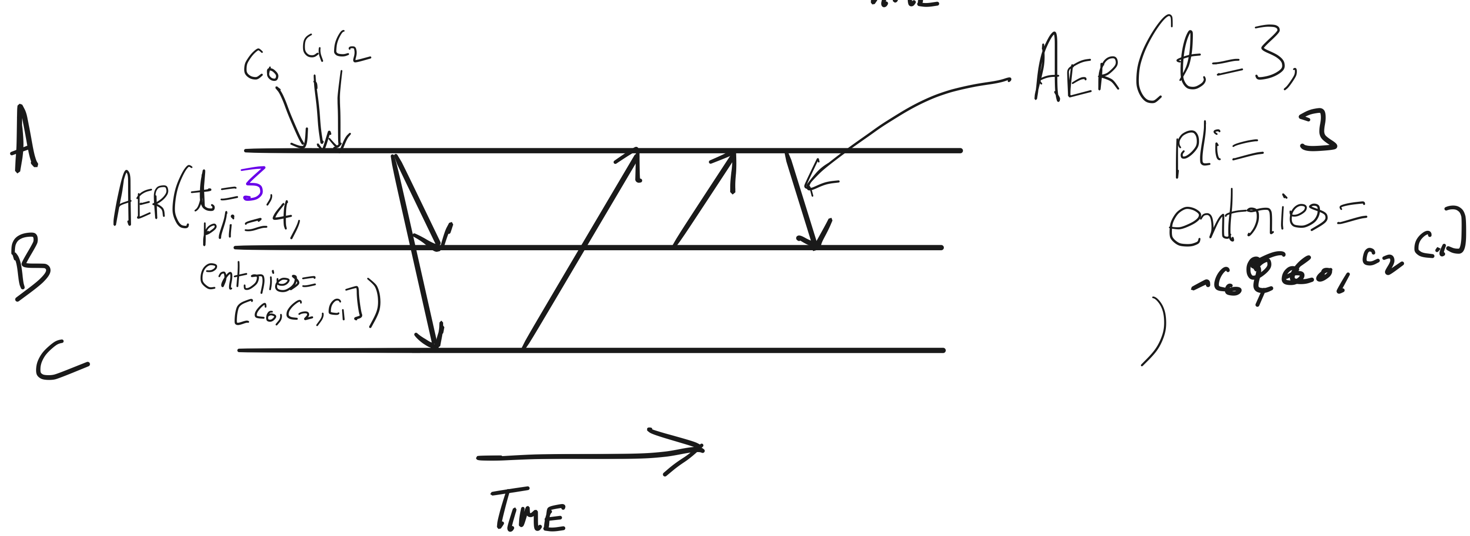
LOG REPLICATION

LOG ENTRY
- INDEX
- TERM
- COMMAND
- ...









So ... A execute C_0?

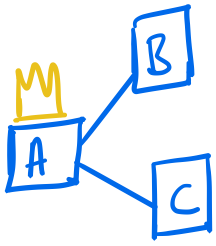
Q. When can A execute ... ?

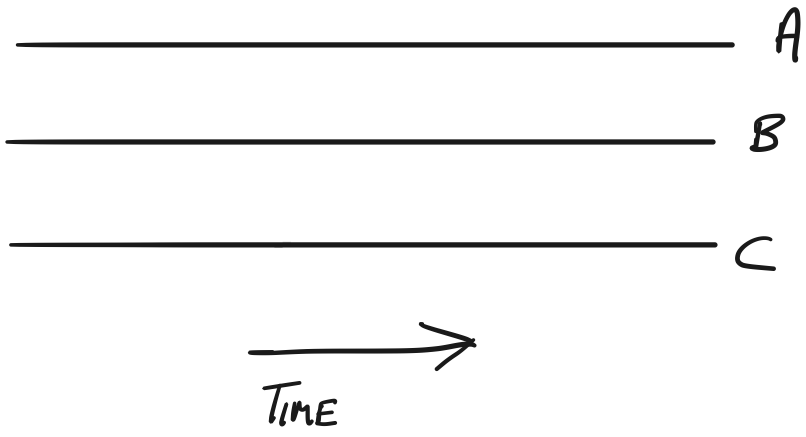
After C

Q. When can A respond to C?

After replication / after C

② FAILURE DETECTION





Problem: Leaders initiate all communication.

Silence → Failure

Silence → Failure
OR
→ No requests

Solution: Don't let leaders be silent for too long ⇒ Heartbeats.

Q: Heartbeat frequency?

Q: How long should a follower wait before assuming leader has failed? (ELECTION TIMEOUT)

Randomized Timeouts

Q: How accurate is this process: can a follower incorrectly suspect a live leader?

Yes

Q: Leader stability?

Failure Recovery / Leader Election

* If a majority of replicas have log entry E at index I, then all future leaders will have log entre E @ I

Requirements

(R1) Any follower can initiate leader election
- Randomized election timeout

(R2) * Must hold for any follower that becomes a leader
- Candidate must compare log with a majority of replicas

(R3) At most 1 leader at a time.

R3: Term Numbers.

- Replicas remember highest term number they have sent or received
- Reject any messages with lower term
- Observation: Replica's log can only be changed by leader with correct term.
- Must ensure at most one leader per term

R1+R2 + At most one leader per term

Any follower can become candidate (on election timeout)

1. Increments term number (term += 1)

Implications?

2. Broadcasts RequestVote message to all replicas

RequestVote(term, id, lastLogIndex,

LastLogTerm)

CLAIM: (INDEX, TERM) COMPLETELY IDENTIFIES A LOG ENTRY. WHY?

③ WHEN A REPLICA RECEIVES REQ VOTE

① UPDATE TERM IF NECESSARY

② check if already voted for this term (why)

If Voted → DO NOT GRANT VOTE

③ Log Matching

if (LastLogTerm > NODE.LAST LOG TERM) OR

(LastLogTerm == NODE.LASTLOGTERM

AND LastLog Index >= Node.Last Log INDEX)

B. term = 11
candidate
last Index = 2
last Term = 10

Vote yes

Crep. term 11
last index = 50
last Term = 5

else vote no

③ Wait for response from a

majority. If a majority vote yes ⇒ candidate becomes leader.

Requirements

1. At most one leader per term.

Why?

2. If a majority of replicas have log entry E at index I, then all future leaders will have log entre E @ I

Claim: (LastLogTerm > Node.Last Log Term) OR

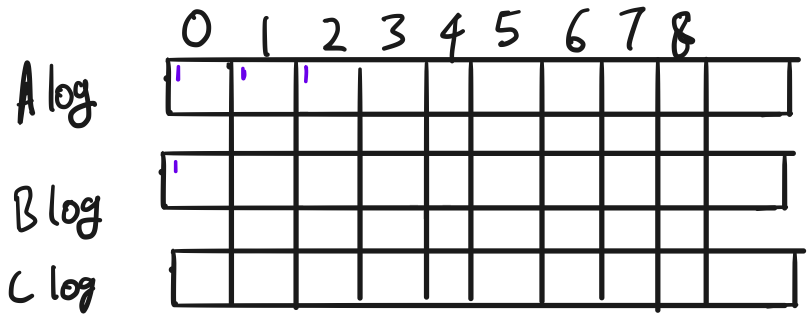
(LastLogTerm == Node.LastLogTerm

AND LastLog Index >= Node.Last Log Index)

⇒

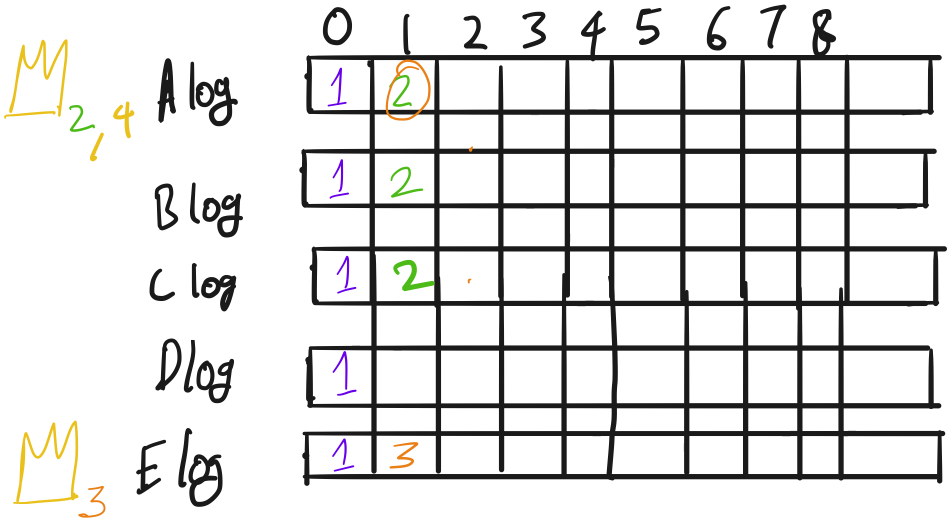
IF A MAJORITY OF REPLICAS HAVE LOG ENTRY E AT INDEX I, THEN ALL FUTURE LEADERS WILL HAVE LOG ENTRE E @ I

Why?



Other issues with leader election

- Entries from previous terms.



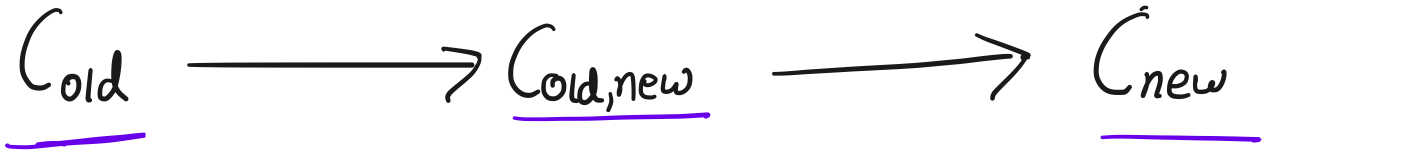
Problem: Larger term dominates.

④ Reconfiguration

- Add or remove nodes.

Problem: What constitutes a majority?

Solution: Require majority in both old & new configuration



System as described so far

System where any entry must be committed in both C_{old} & C_{new} before being executed

System as described so far

Commit configuration changes using Raft