# Distributed Systems } Paxos

BUT FIRST... ADMINISTRIVIA

- MIDTERM: NEXT WEEK, IN CLASS
  - COVERS EVERYTHING UP TO TODAY
  - OPEN BOOK
    - ↳ PAPERS, NOTES, ETC.
    - → ELECTRONIC DEVICES
      - ↳ CAN USE IPAD ETC.

        BUT
        - MUST DISCONNECT FROM INTERNET: DOWNLOAD AHEAD OF TIME
        - NO CHATBOTS, ETC.

- I uploaded last year's midterm to the website if you wanted an idea of what to expect.

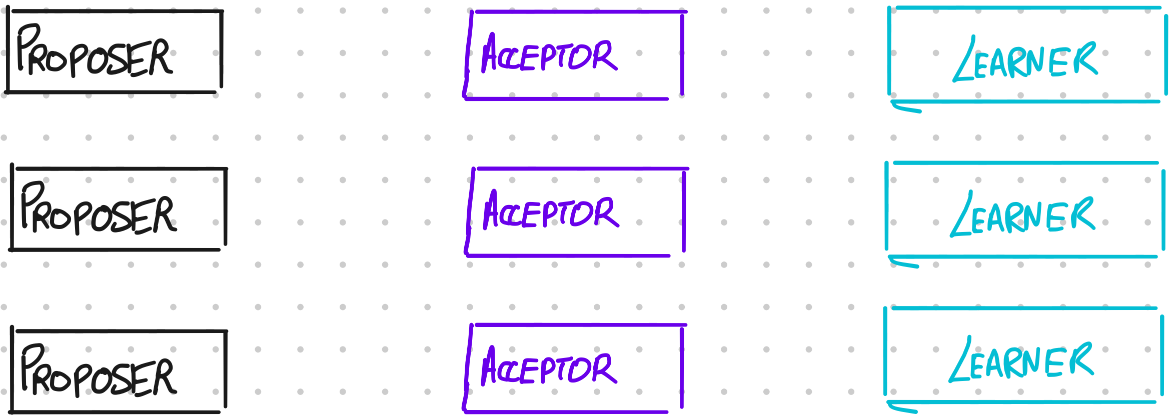
A brief digression ⟶

—

## Paxos

- Why? Reinforce ideas around correctness

- General form: 3 types of nodes.
  SYNOD: Log with 1 slot/index



① PROPOSE VALUES: Suggest command to set index to

$f+1$

② ACCEPT VALUES: CHOOSE WHAT VALUE TO ACCEPT & COMMIT

$2f+1$

③ LEARN VALUES: EXECUTE COMMAND

$f+1$

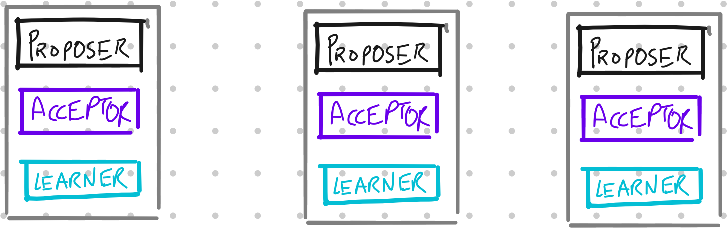
WHY SEPARATE THIS OUT?

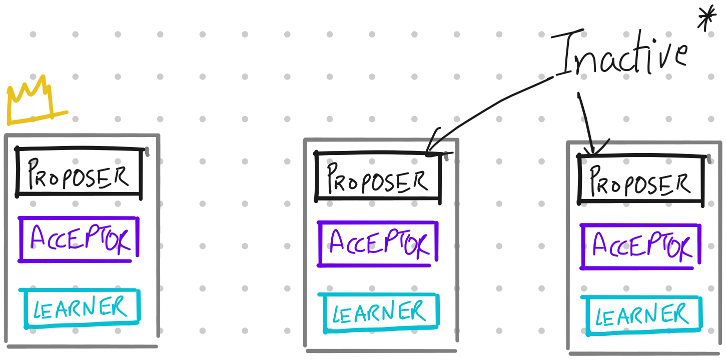
Recipe For Constructing RSM Protocols

↳ Different Types Of Proposers/Acceptors/
Learners

Return To This At The End Of
Class.

MultiPaxos /Chubby: Our Setting





Synod: The Algorithm For A Single Slot
(We Looked At This Last Class)

Goes In Rounds, Each Round Denoted By A "Proposal #"

Requirements

Commit

- Choose Some Value, Eventually (Avoids trivial solutions)

P1 ↳ Acceptor Accepts First Value

↳ Not the same as choosing!

P2 - Once A Value V Is Chosen, No Other Value

WILL BE CHOSEN [C.F., STATE MACHINE SAFETY FROM LAST CLASS]

↳ P2a. IF VALUE V IS CHOSEN AS PROPOSAL N, THEN ANY PROPOSAL WITH $\# > N$ ACCEPTED BY AN ACCEPTOR MUST PROPOSE V.

[NOTE, INCLUDES VALUES ACCEPTED BY ACCEPTORS WHO HAVE MISSED ALL PROPOSALS/MESSAGES SO FAR]

P2b. IF VALUE V IS CHOSEN AS PROPOSAL N, ~~THEN ANY PROPOSAL WITH # > N ACCEPTED BY AN ACCEPTOR MUST HAVE VALUE V~~ ANY HIGHER # PROPOSAL ISSUED BY A PROPOSER HAS VALUE V.

[Note, must be true even for proposals issued by a different proposer in the future.]

Observe - P2b ⇒ P2a

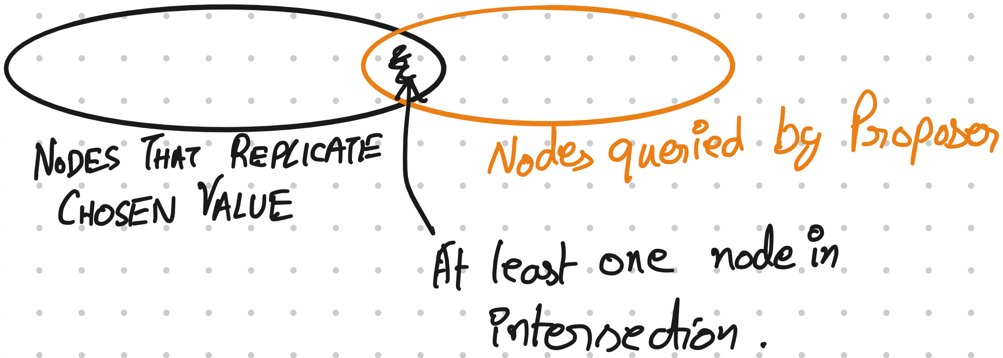
- For P2b to hold proposers need to be able to find any chosen (committed value V

- CAN USE QUORUM INTERSECTION FOR THIS (P2C)

COMMITTED

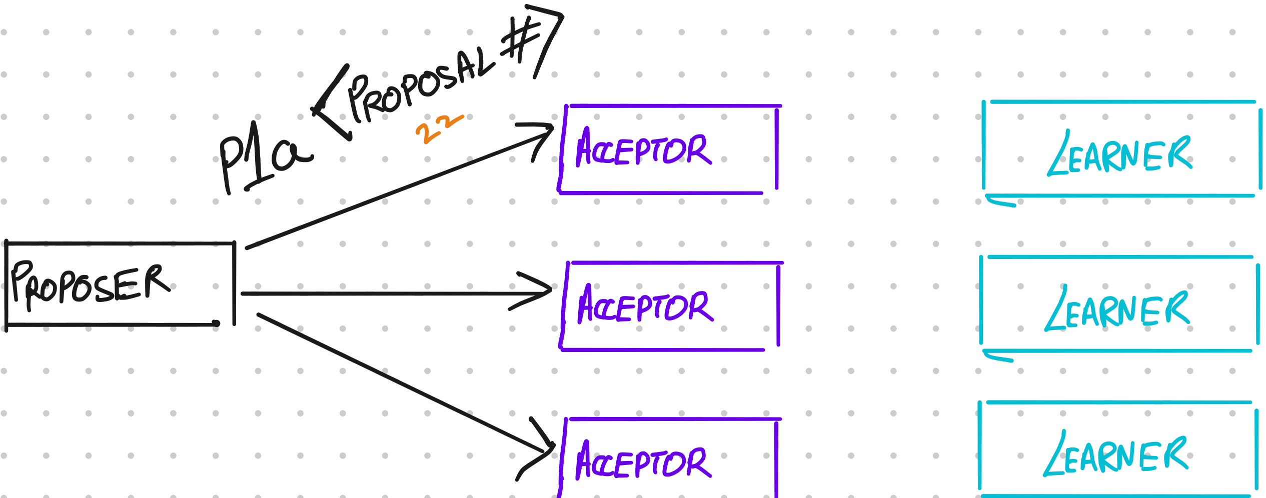
REQ: ① CHOSEN VALUES MUST BE ACCEPTED BY A QUORUM OF ACCEPTORS

② PROPOSER MUST CHECK WITH A QUORUM BEFORE PROPOSING A VALUE


NODES THAT REPLICATE CHOSEN VALUE
Nodes queried by Proposer
At least one node in intersection.

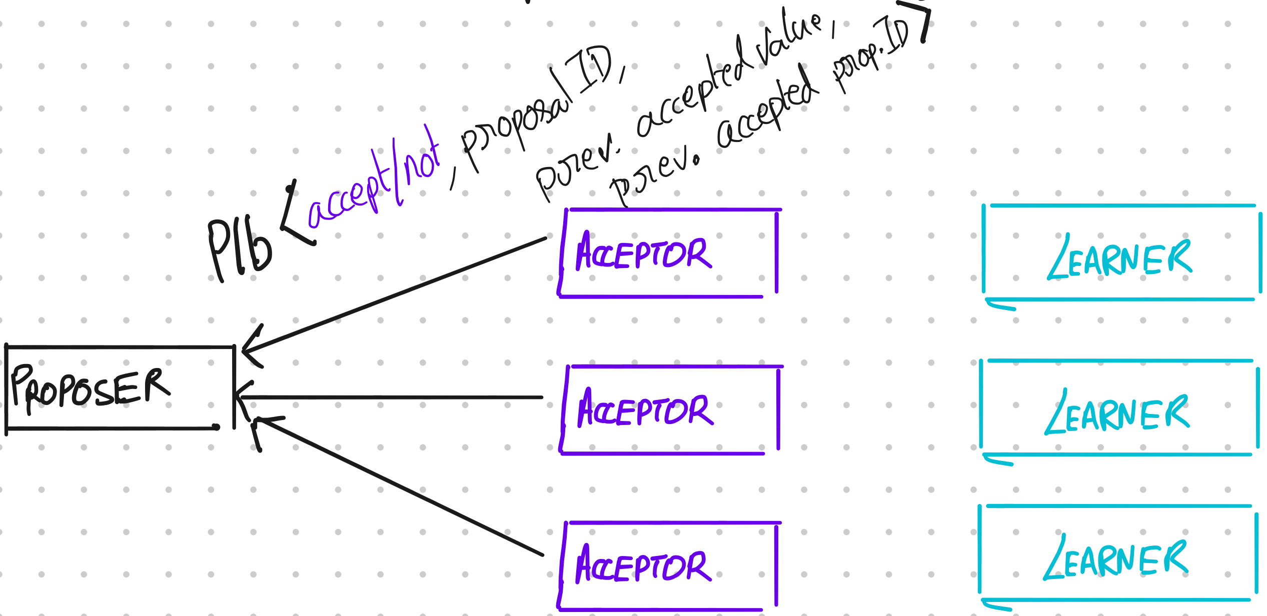
P2a/P2b ⇒ If a value V was committed w/ proposal # p, all accepted proposals w/ prop# p+1 or larger contain V

↳ Value associated with largest proposal # is the one likely committed



Goals ⓐ Query to find any previously accepted values.

ⓑ Prevent any pending proposals (with lower ID) from making progress


P1b [accept/not, proposal ID, prev. accepted value, prev. accepted prop. ID]
PROPOSER
ACCEPTOR
ACCEPTOR
ACCEPTOR
LEARNER
LEARNER
LEARNER

- FALSE, 24, . . . ← 22 is too low

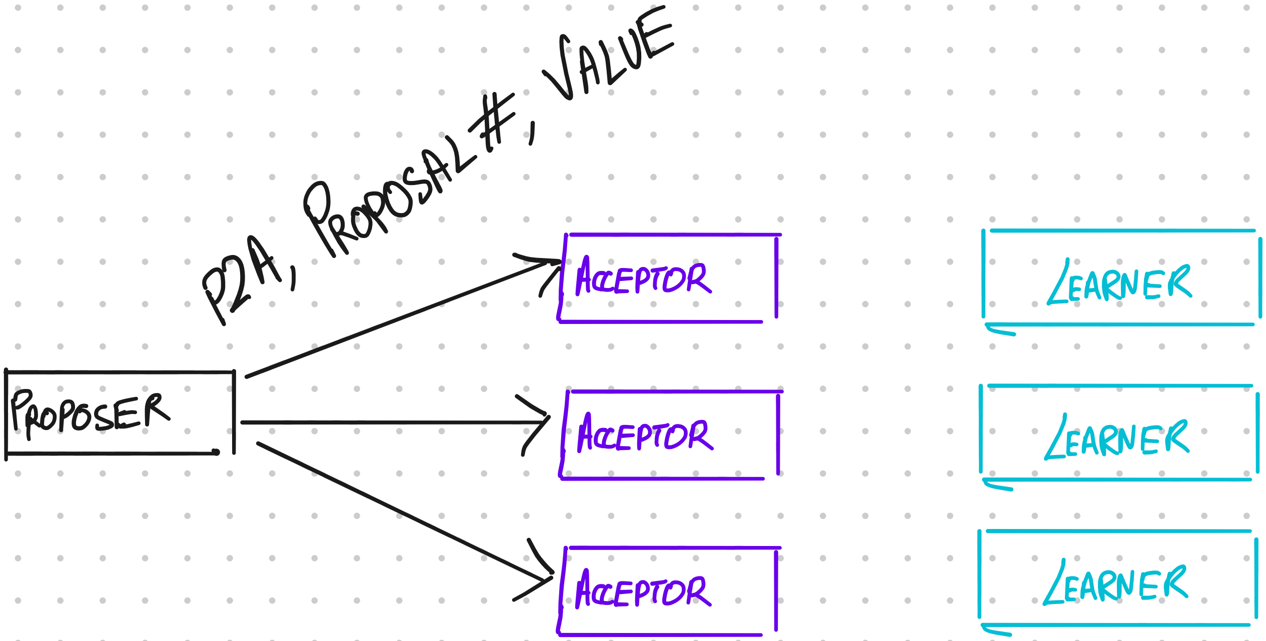
- True, 22, ⊥, – ← Nothing accepted

- True, 22, X, 2 ← x accepted as prop. ID 2

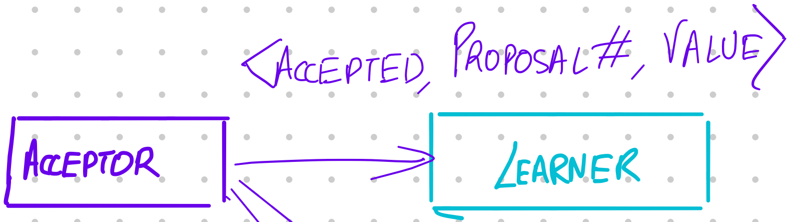
n=3

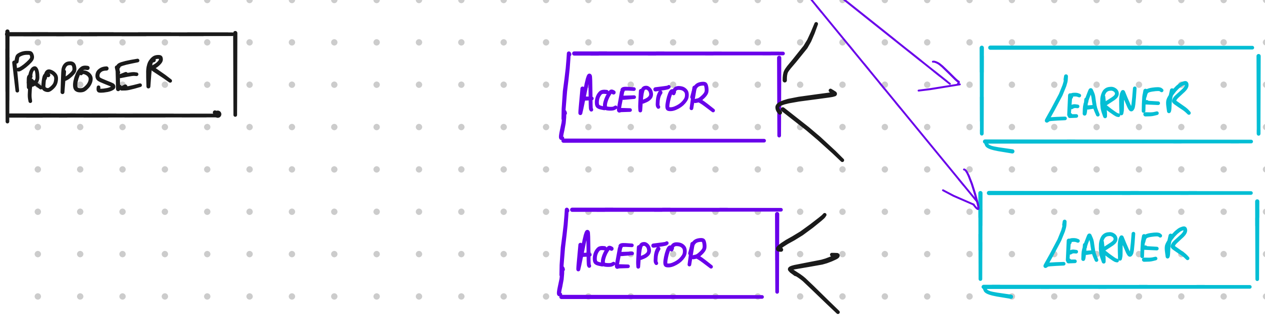
|  | Value to propose | Possibly committed |
|---|---|---|
| ⊥, ⊥ | Any | ~~x~~ |
| (x,7) ⊥ | X |  |

$(x, 2)$ $(y, 3)$ $y$ $\{y, \perp\}$



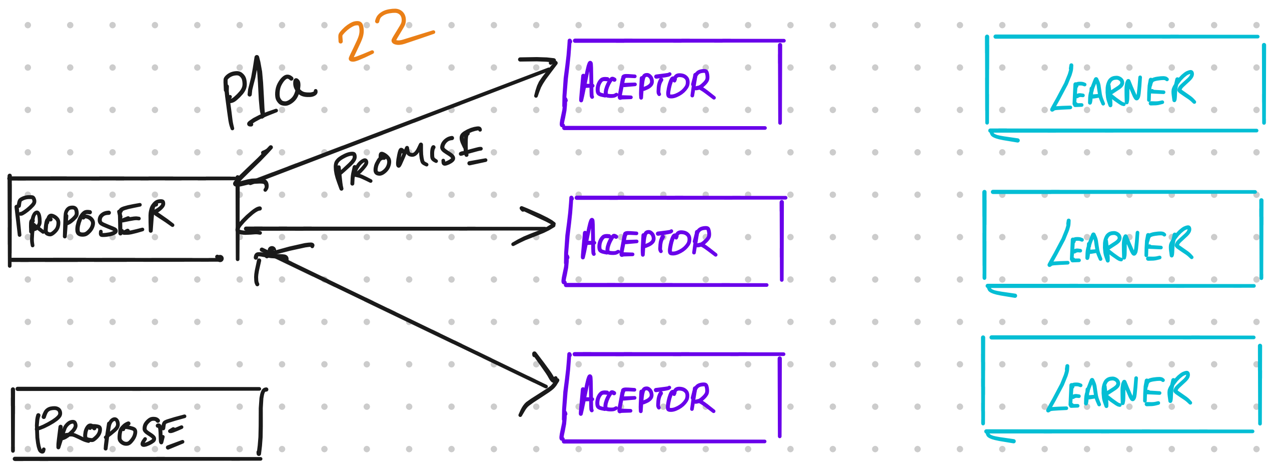
When should the ACCEPTOR not accept?

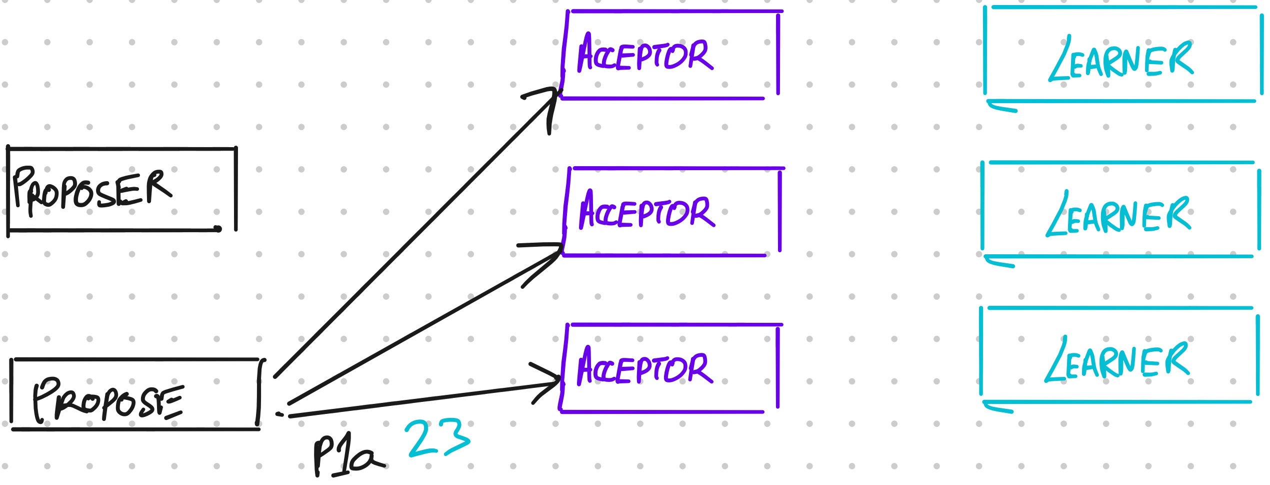




- When should learners execute/act on value?

# Why leader?

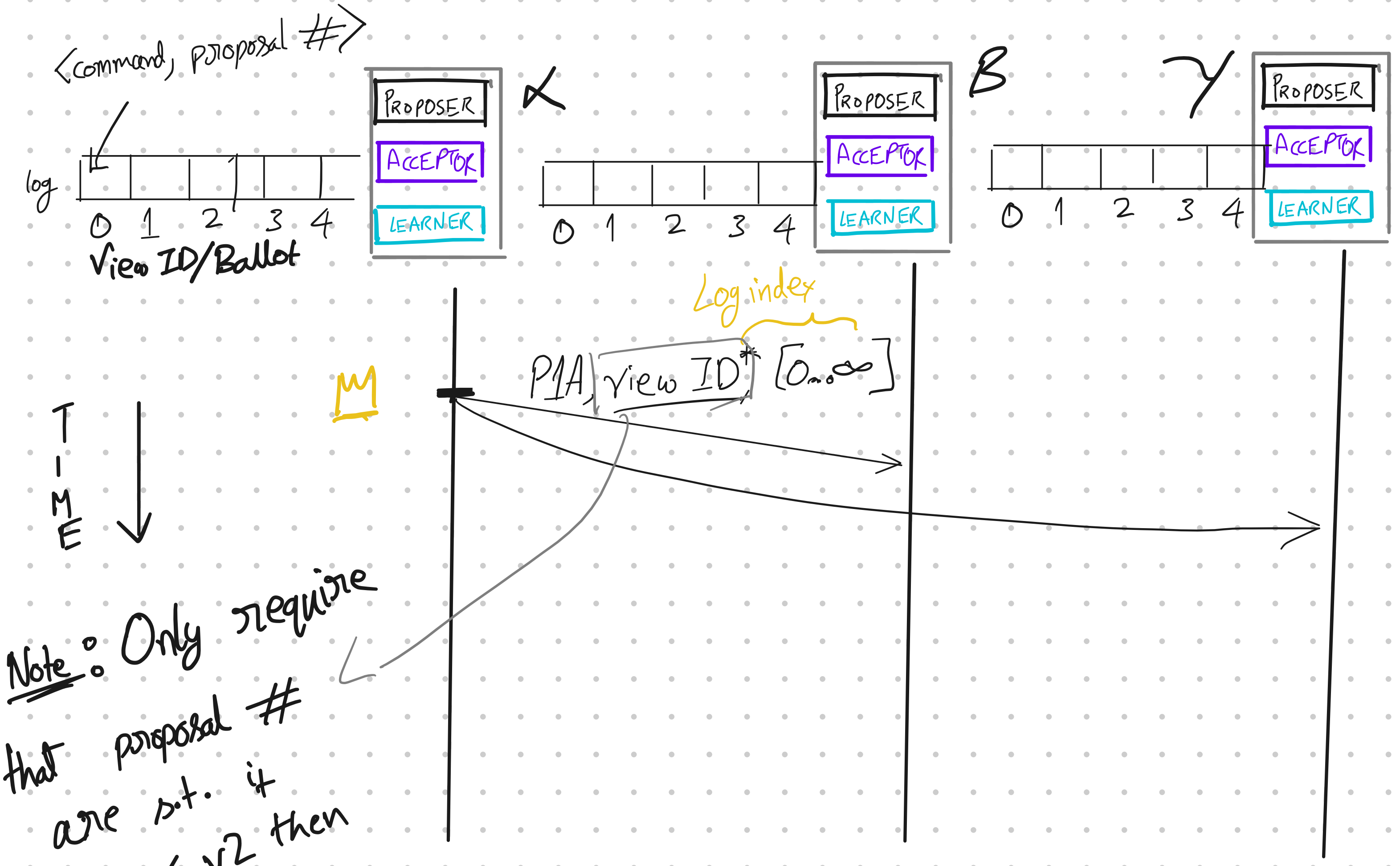




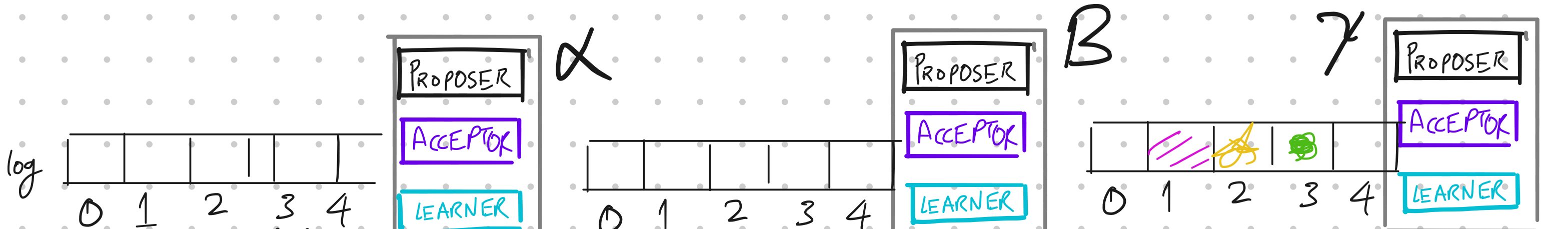
# Duelling proposers are a problem!

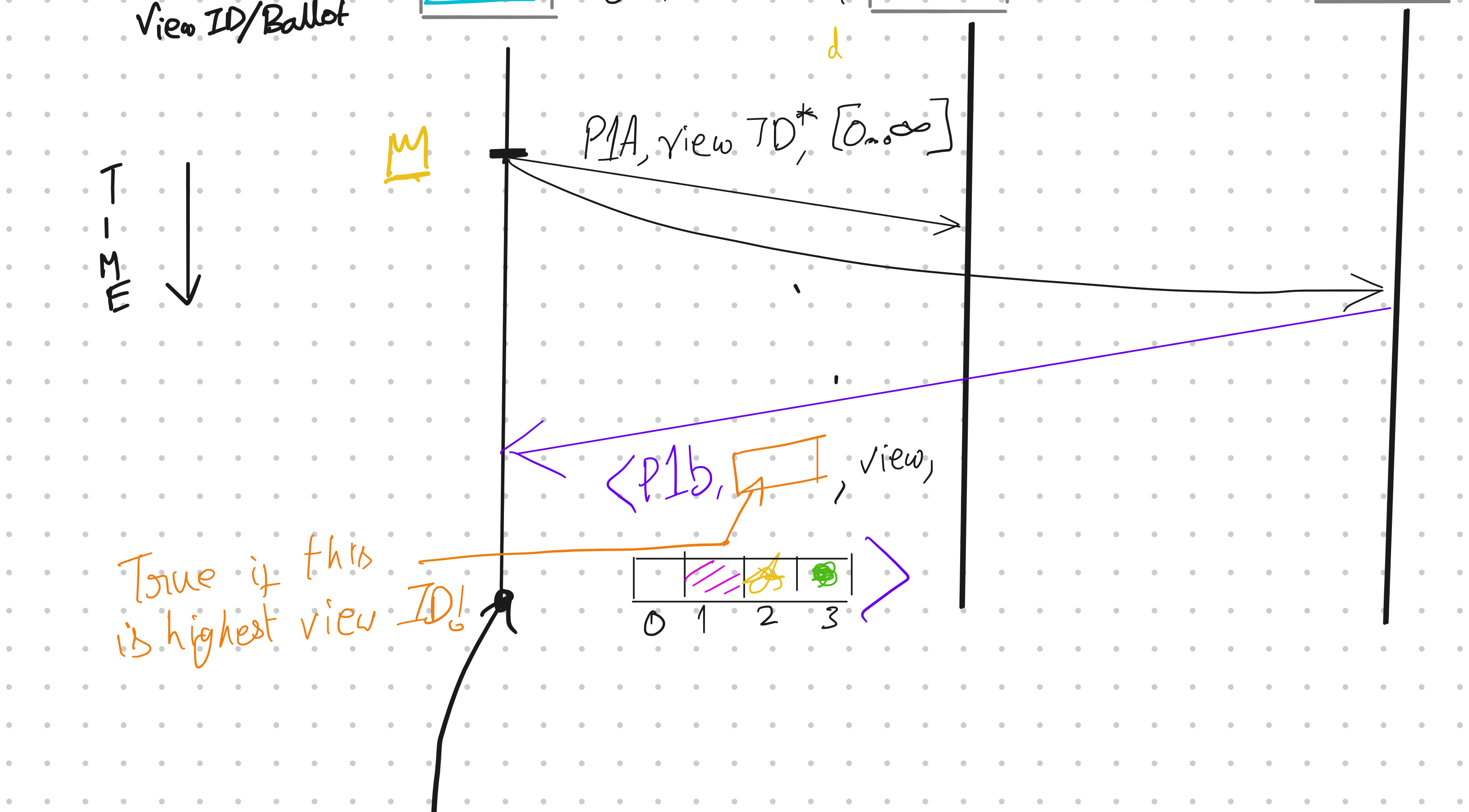
## Putting it all together

So far: Focus on a single log idx. But what about RSMs? "Fast Paxos" Lamport '05



Note: Only require that proposal # are s.t. if v1 < v2 then prop# in v1 < prop# in v2

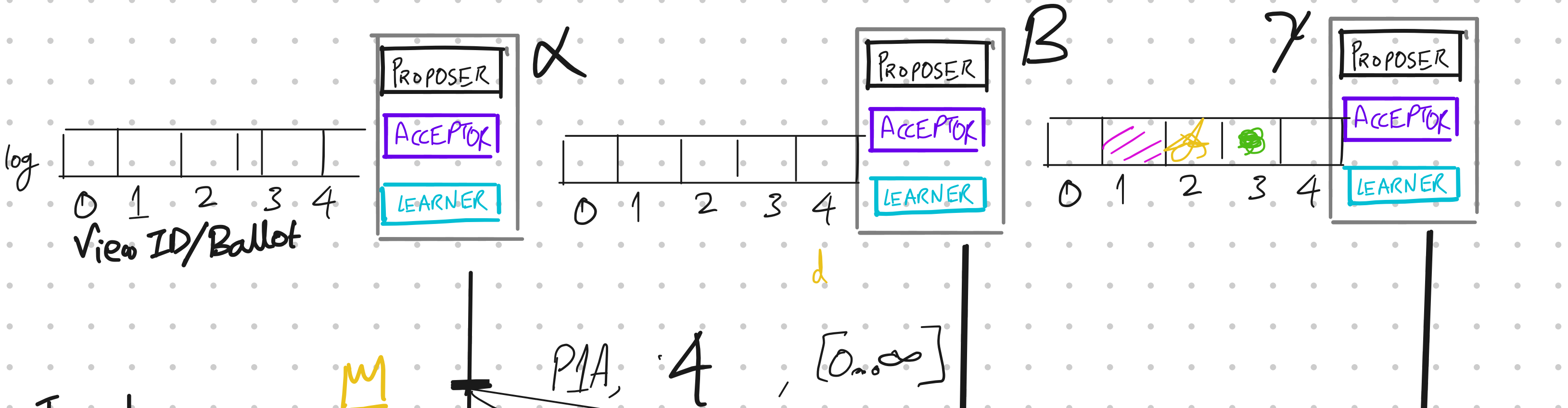


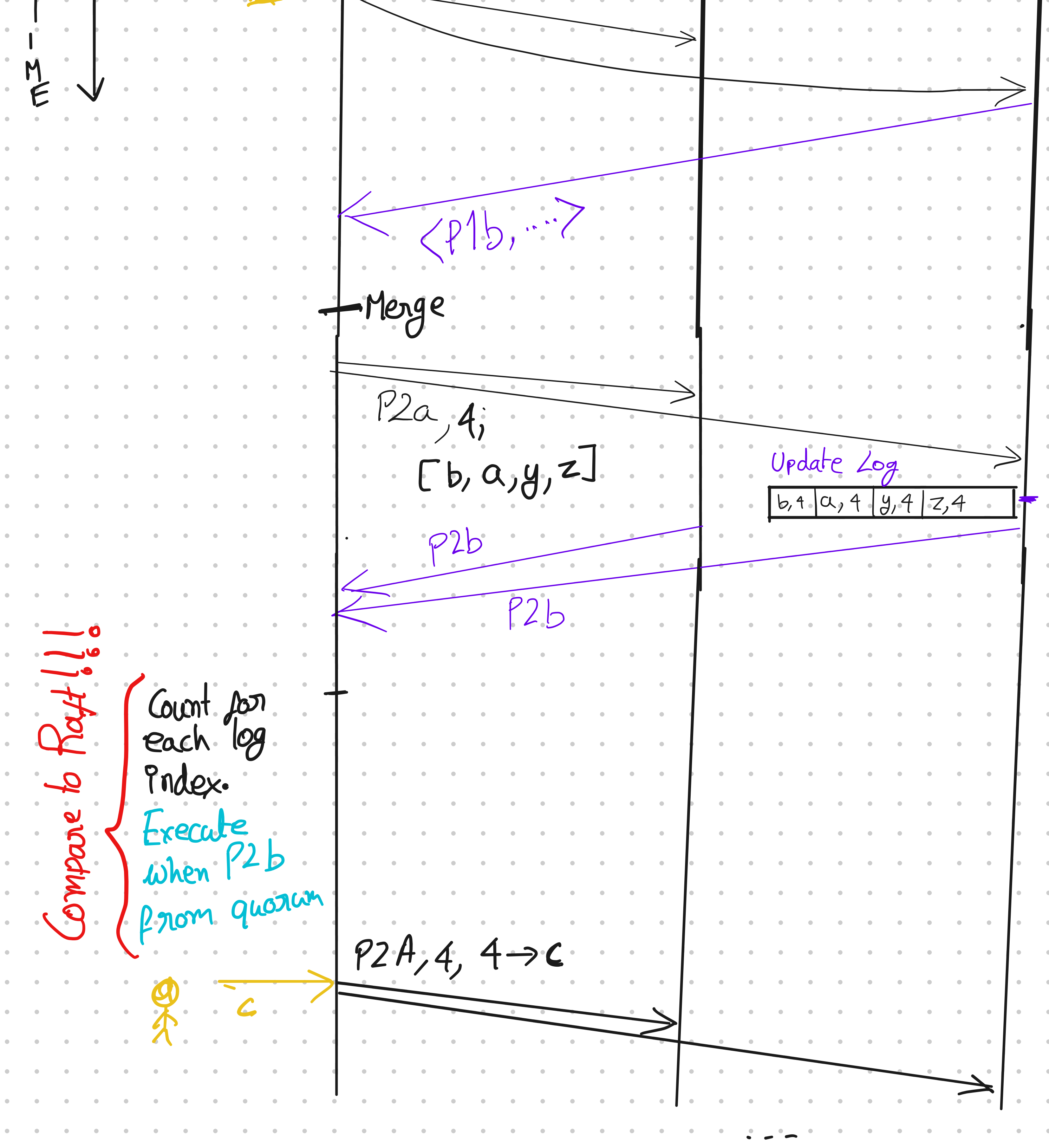


Wait for P1b from a quorum.

Merge logs

| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| α | b,1 | x,1 | y,1 | z,1 |
| γ | ⊥ | a, 2 | y,1 | z,1 |
| ML : | b1 | a,2 | y,1 | z,1 |





Count for each log index.
Execute when P2b from quorum

Compare to Raft!!!

Electing a leader?

- Only requirement is Election Safety; that is at most one leader

- Any protocol that meets this requirement suffices

Changing Configuration

- Hard problem, for the same reason as last week : need to avoid quorums of nodes that are not up-to-date
  Vertical Paxos.

What does this generality help with

- Disk Paxos
  - ↳ Acceptors and learners are networked disks
    - ↳ Read, Write

- Mencius
  - ↳ Multiple leaders for multiple datacenters
  - → Later improved by EPaxos

...

Taking a step back

- RSMs

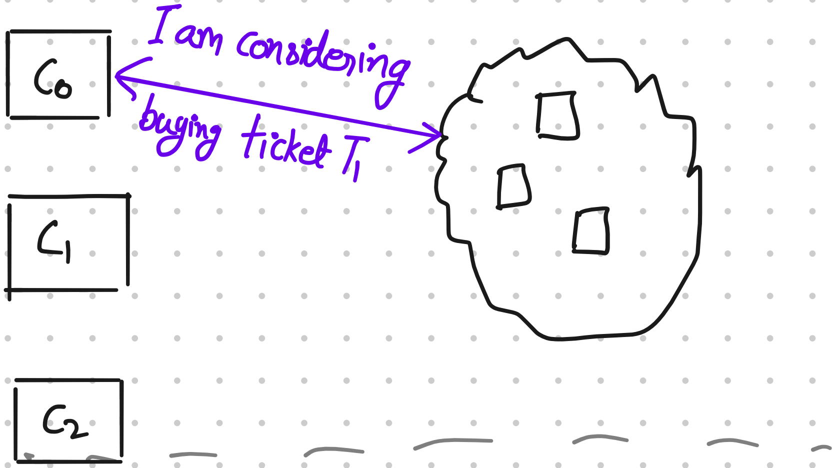
- Quorum intersection

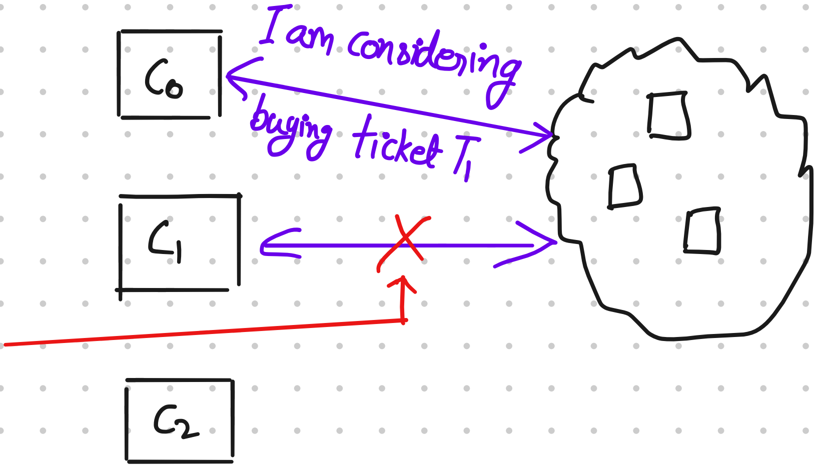
→ Leader safety

→ State machine safety

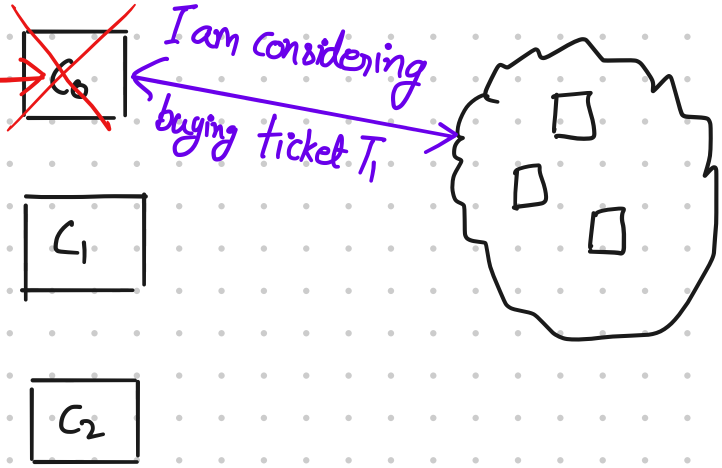
# Leader Leases / Leases

Lease





Don't offer $C_1$ the same ticket

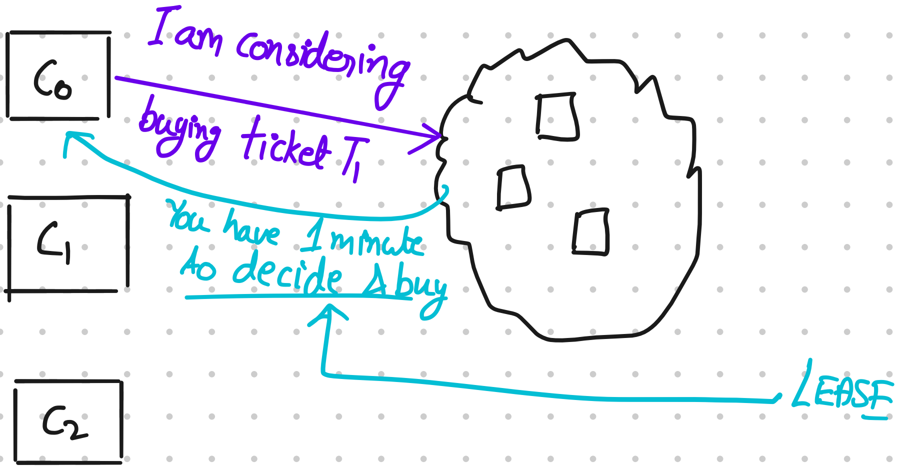


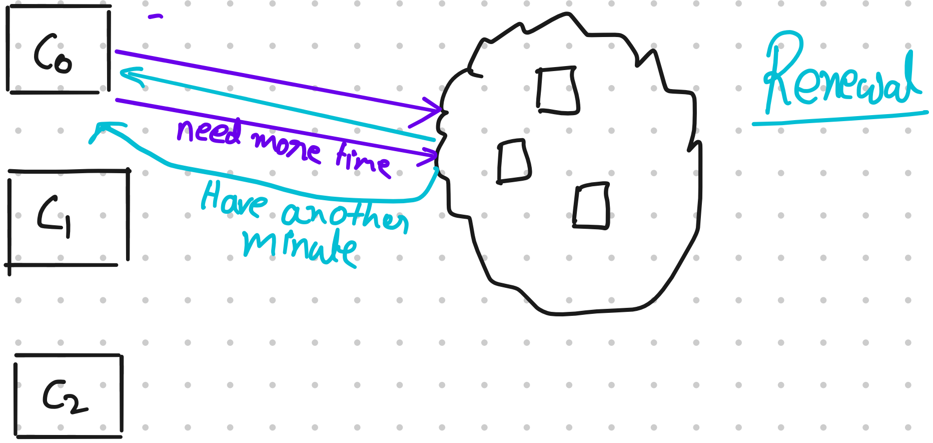
Make sure failure at $C_0$ cannot cause $T_1$ to be resold forever [Resource Leak]

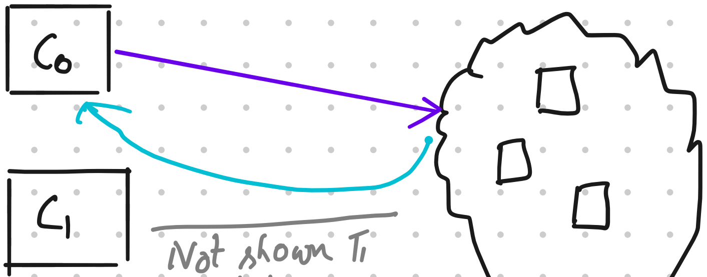
A common problem

- Locks (mutex)
- Resources (memory, etc.)
- ...

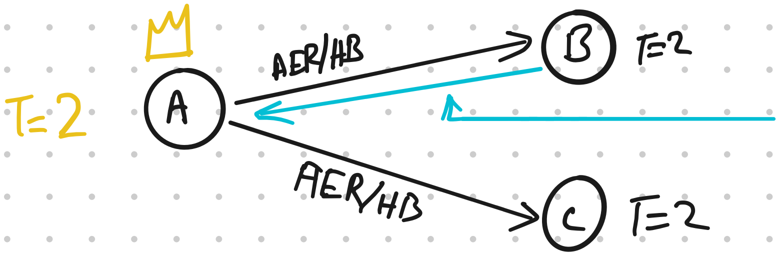
Leases [Gray & Cheriton '89]


C₀
C₁
C₂
I am considering buying ticket T₁
You have 1 minute to decide & buy
LEASE

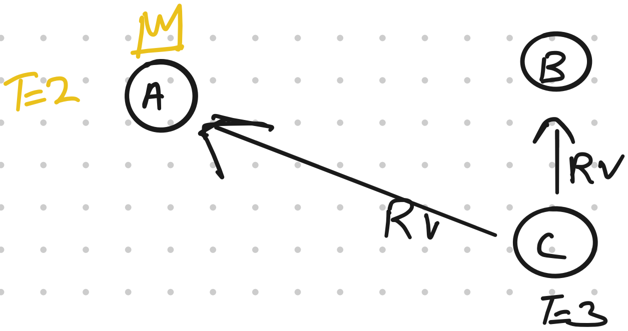




Observe: Raft HB/Leader Election is a Lease Mechanism



Grant: Promise to not start a new election for ____

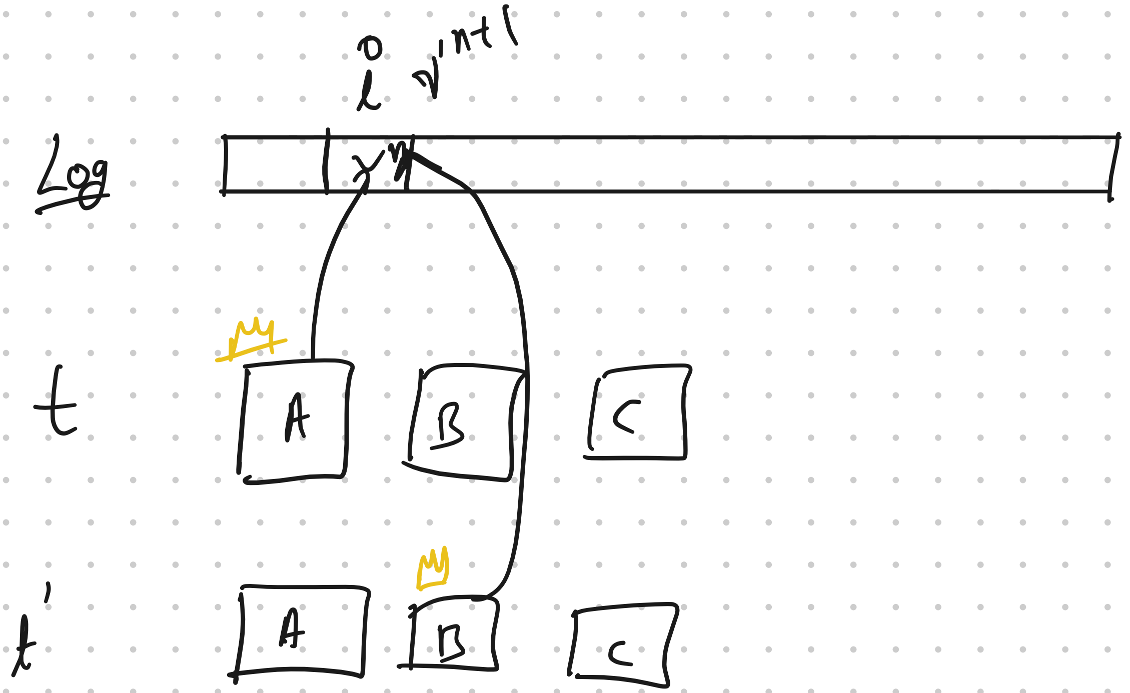


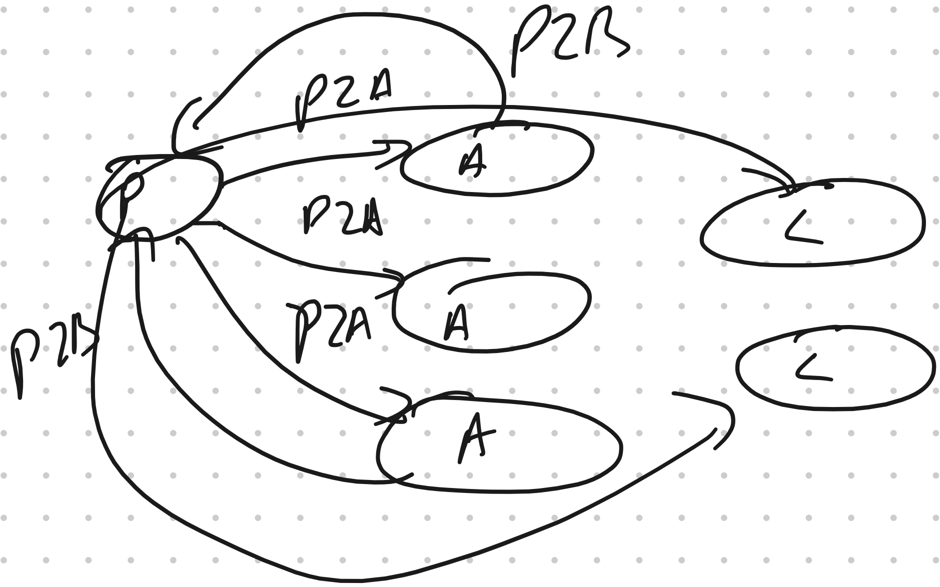
Lease expired

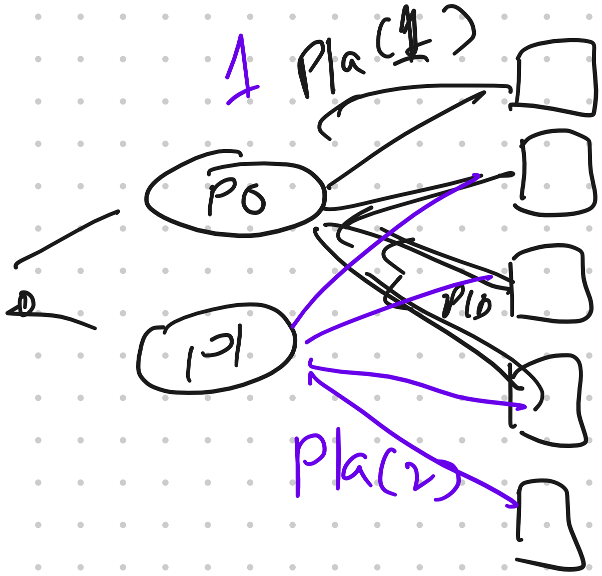
## Leader Leases

Goal:

- Stable leadership
- Availability.


l
v'n+1
Log
x
y
t
A
B
C
t'
A
B
C


P2B
P2A
P
A
C
P2A
A
P2A
P2B
C
A


1
P1a(1)
P0
P1b
P1
P1a(2)