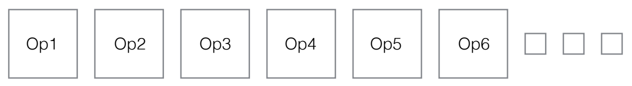
Paxos Made Moderately Complex Made Moderately Simple

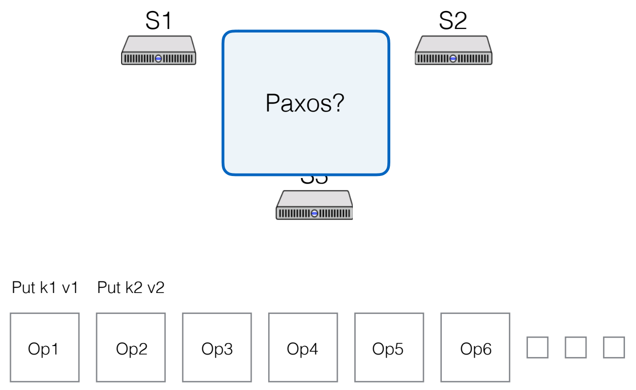
Tom Anderson and Doug Woos

State machine replication

Reminder: want to agree on order of ops

Can think of operations as a log





Lab 3

Paxos = Phase 1
- Send prepare messages
- Pick value to accept

Phase 2
- Send accept messages

Can we do better?

Phase 1: “leader election”

- Deciding whose value we will use

Phase 2: “commit”

- Leader makes sure it’s still leader, commits value

What if we split these phases?

- Lets us do operations with one round-trip

Roles in PMMC

Replicas (like learners)

- Keep log of operations, state machine, configs

Leaders (like proposers)

- Get elected, drive the consensus protocol

Acceptors (*simpler* than in Paxos Made Simple!)

- “Vote” on leaders

A note about ballots in PMMC

(leader, seqnum) pairs

Isomorphic to the system we discussed Mon, Wed

- 0 0, 4, 8, 12, 16, …
- 1 1, 5, 9, 13, 17, …
- 2 2, 6, 10, 14, 18, …
- 3 3, 7, 11, 15, 19, …

A note about ballots in PMMC

(leader, seqnum) pairs

Isomorphic to the system we discussed Mon, Wed

- 0 (0, 0), (0, 1), (0, 2), (0, 3), (0, 4), …
- 1 (1, 0), (1, 1), (1, 2), (1, 3), (1, 4), …
- 2 (2, 0), (2, 1), (2, 2), (2, 3), (2, 4), …
- 3 (3, 0), (3, 1), (3, 2), (3, 3), (3, 4), …

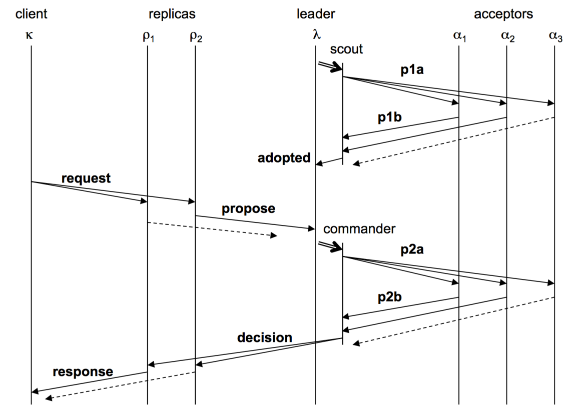
A note about ballots in PMMC

(leader, seqnum) pairs

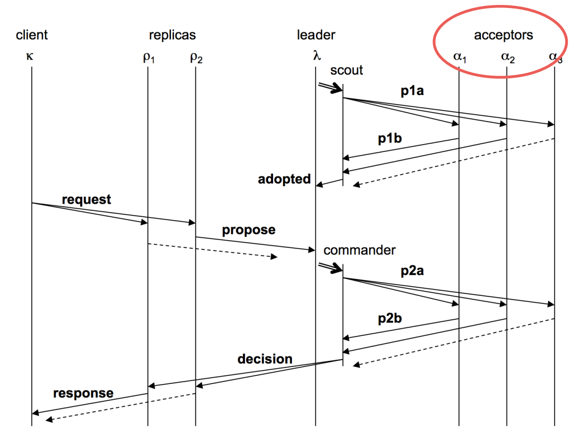
Isomorphic to the system we discussed Mon, Wed

- 0 0.0, 1.0, 2.0, 3.0, 4.0, …
- 1 0.1, 1.1, 2.1, 3.1, 4.1, …
- 2 0.2, 1.2, 2.2, 3.2, 4.2, …
- 3 0.3, 1.3, 2.3, 3.3, 4.3, …

Paxos Made Moderately Complex Made Simple



Paxos Made Moderately Complex Made Simple

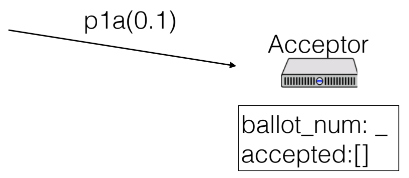


Acceptors

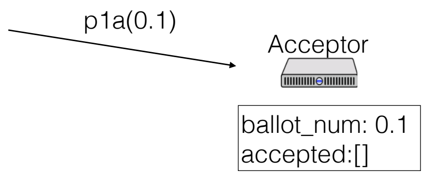
Acceptor

ballot_num: 0
accepted:[]

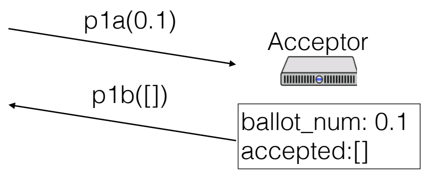
Acceptors



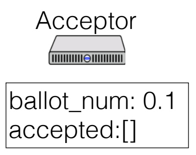
Acceptors



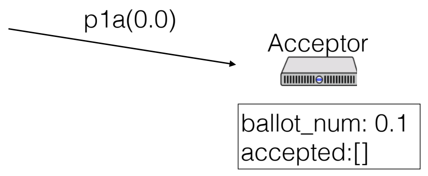
Acceptors



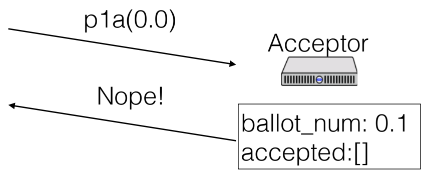
Acceptors



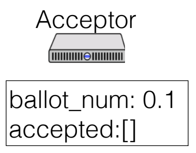
Acceptors



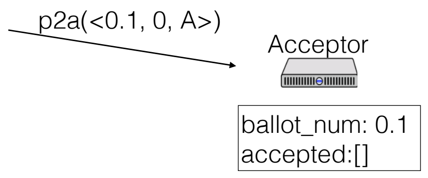
Acceptors



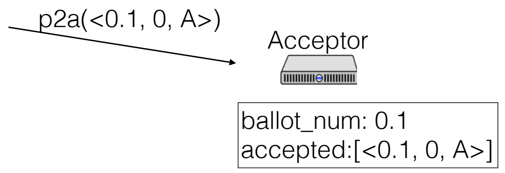
Acceptors



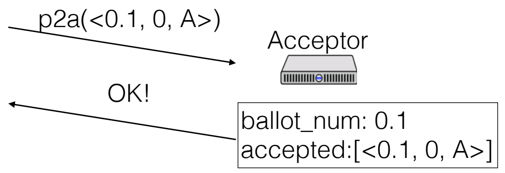
Acceptors



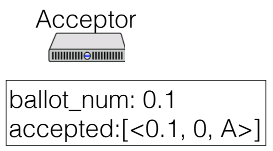
Acceptors



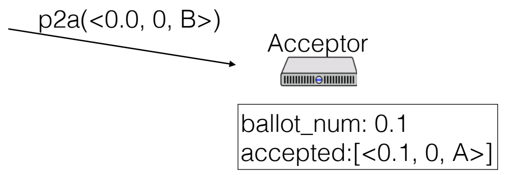
Acceptors



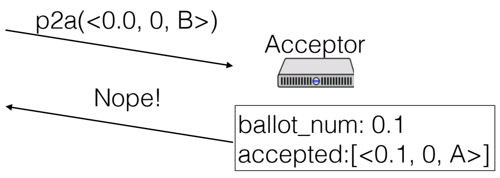
Acceptors



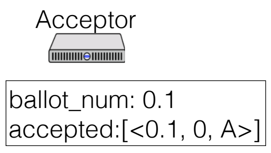
Acceptors



Acceptors



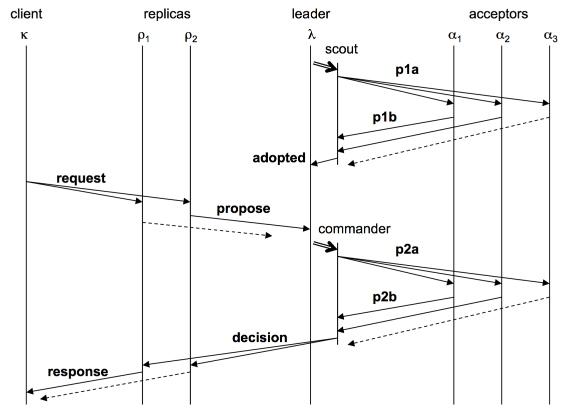
Acceptors



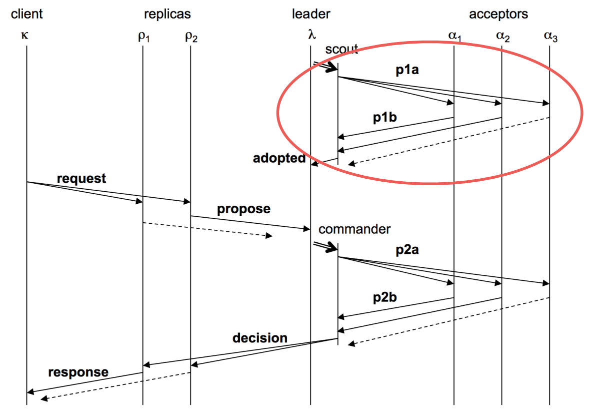
Acceptors

- Ballot numbers increase
- Only accept values from current ballot
- Never remove ballots
- If a value *v* is chosen by a majority on ballot *b*, then any value accepted by any acceptor in the same slot on ballot *b'* > *b* has the same value

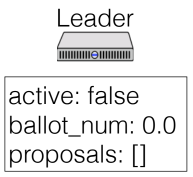
Paxos Made Moderately Complex Made Simple



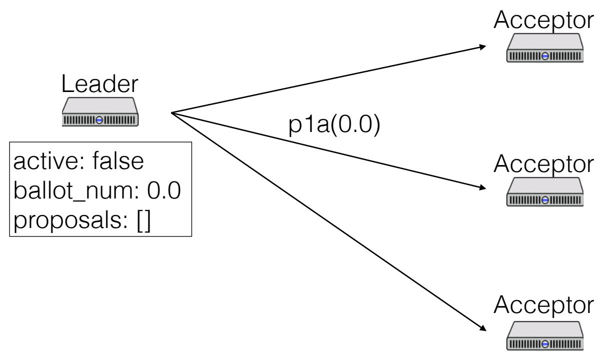
Paxos Made Moderately Complex Made Simple



Leader: Getting Elected

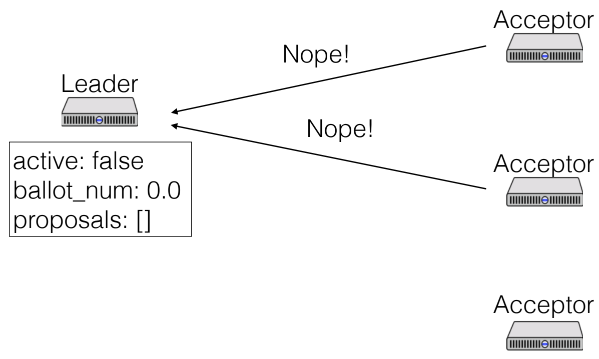


Leader: Getting Elected



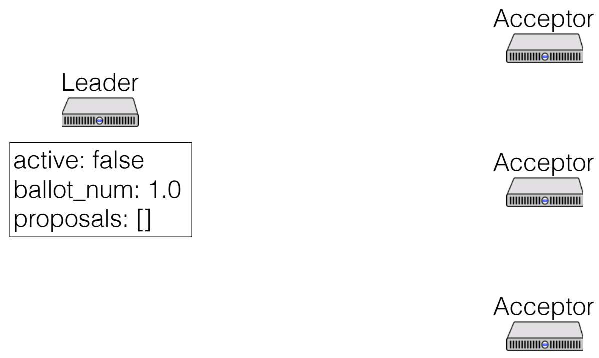
active: false
ballot_num: 0.0
proposals: []

Leader: Getting Elected



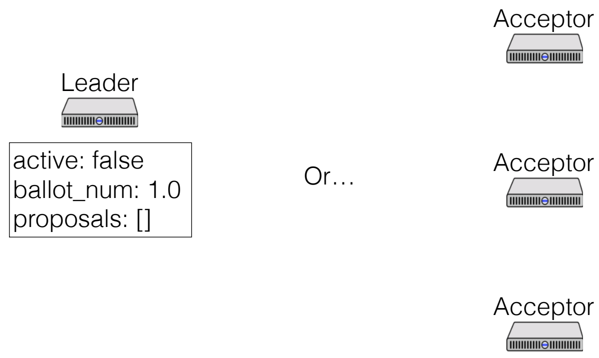
active: false
ballot_num: 0.0
proposals: []

Leader: Getting Elected



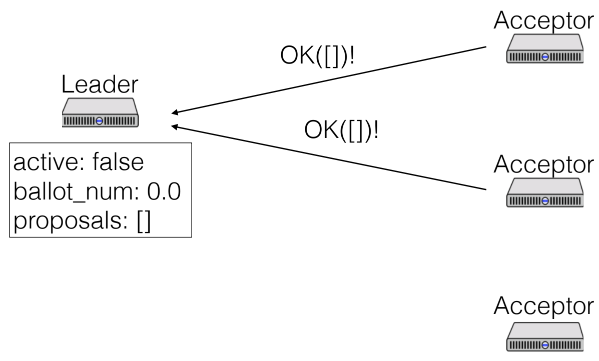
active: false
ballot_num: 1.0
proposals: []

Leader: Getting Elected



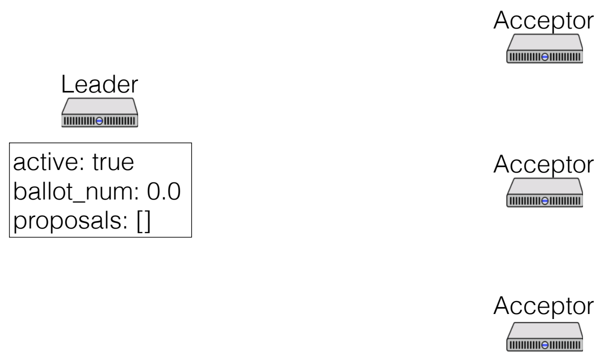
active: false
ballot_num: 1.0
proposals: []

Leader: Getting Elected

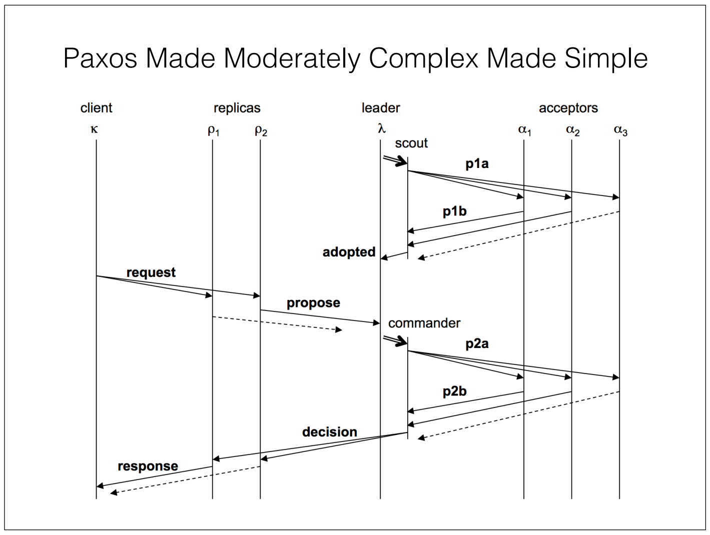


active: false
ballot_num: 0.0
proposals: []

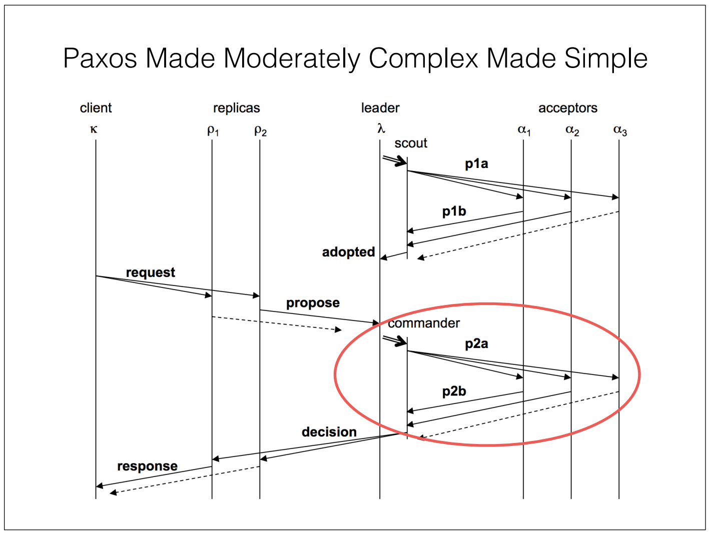
Leader: Getting Elected



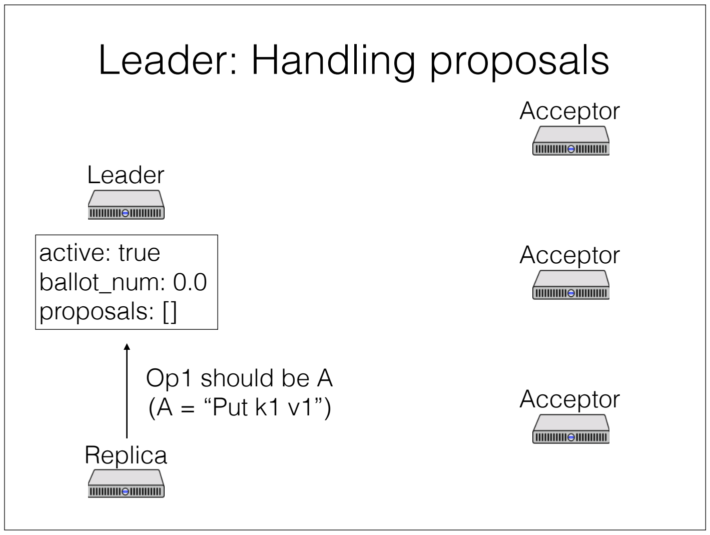
active: true
ballot_num: 0.0
proposals: []



Paxos Made Moderately Complex Made Simple



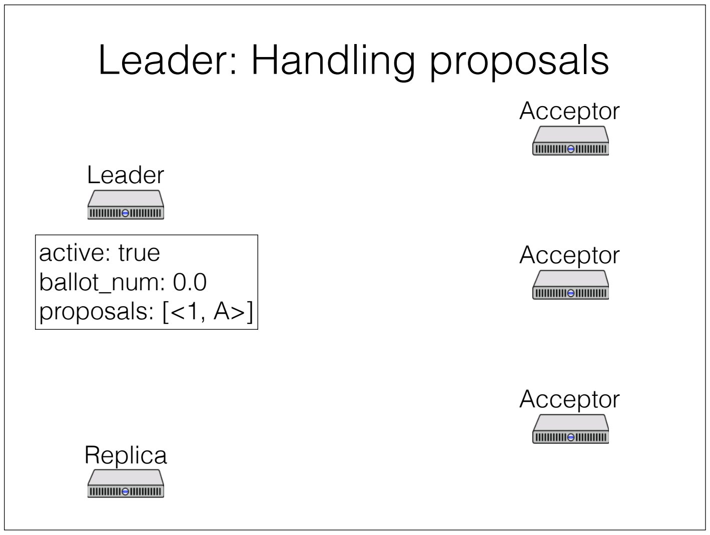
Paxos Made Moderately Complex Made Simple



Leader: Handling proposals

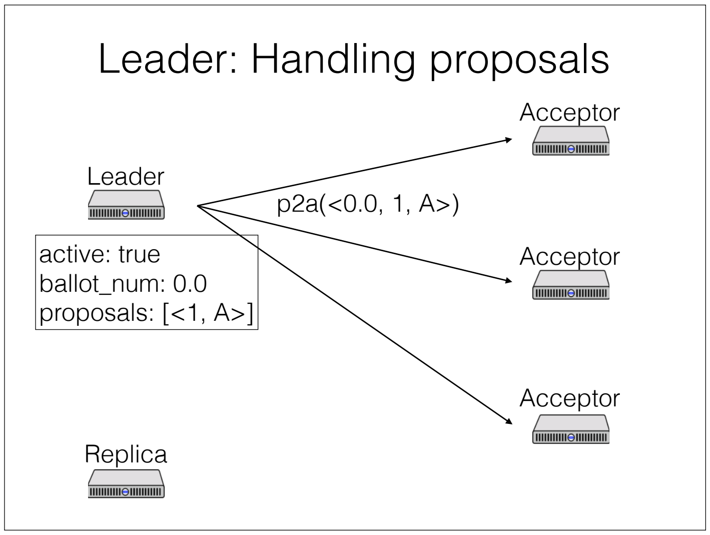
active: true
ballot_num: 0.0
proposals: []

Op1 should be A
(A = “Put k1 v1”)



Leader: Handling proposals

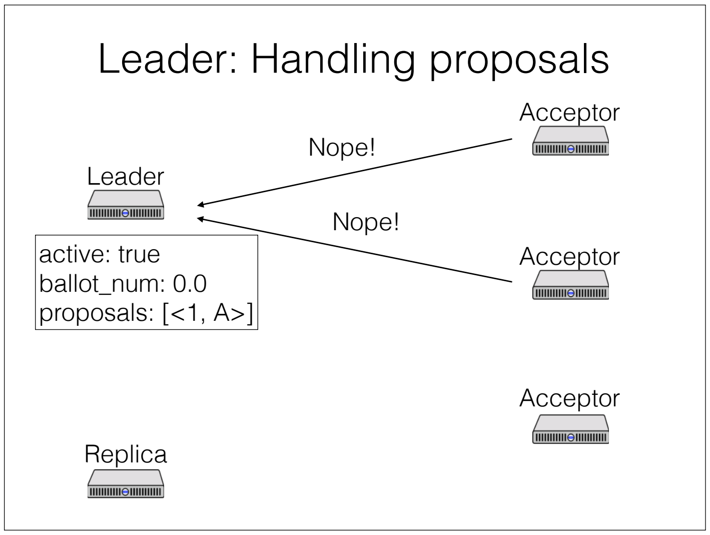
active: true
ballot_num: 0.0
proposals: [<1, A>]



Leader: Handling proposals

active: true
ballot_num: 0.0
proposals: [<1, A>]

p2a(<0.0, 1, A>)



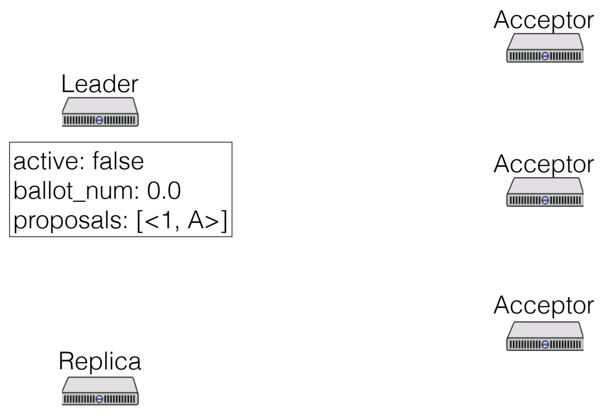
Leader: Handling proposals

Nope!

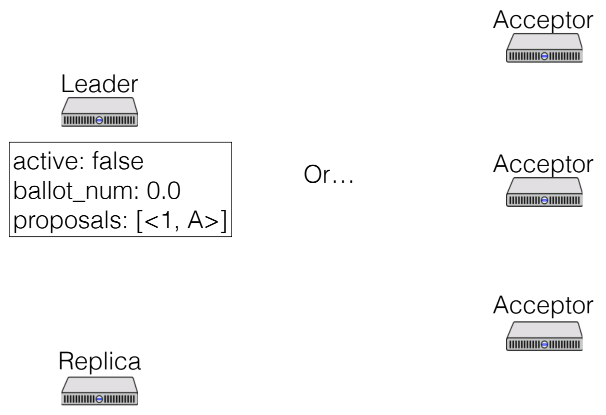
Nope!

active: true
ballot_num: 0.0
proposals: [<1, A>]

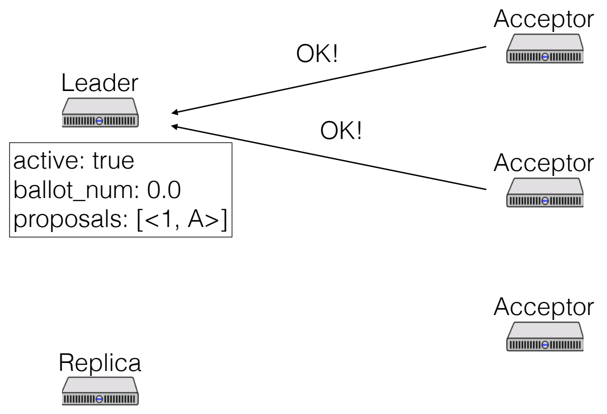
Leader: Handling proposals



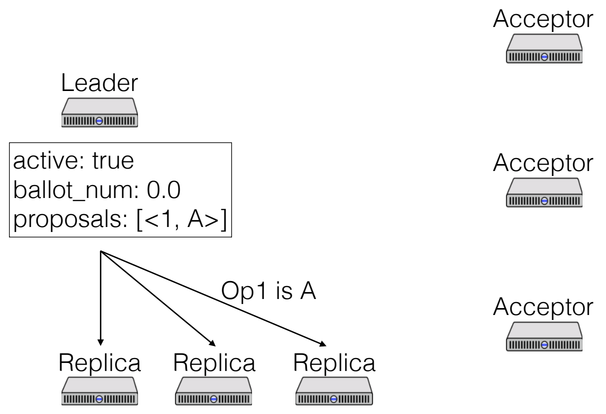
Leader: Handling proposals



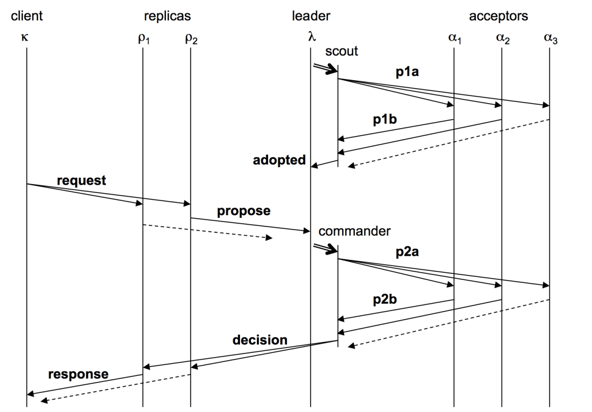
Leader: Handling proposals



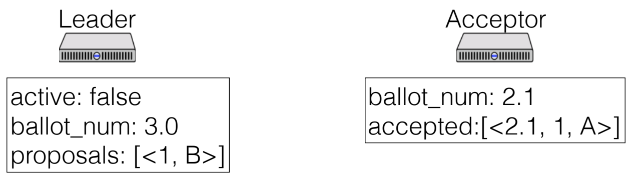
Leader: Handling proposals



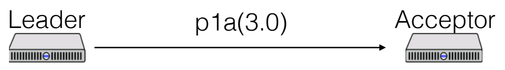
Paxos Made Moderately Complex Made Simple



Election revisited



Election revisited



active: false
ballot_num: 3.0
proposals: [<1, B>]

ballot_num: 2.1
accepted:[<2.1, 1, A>]

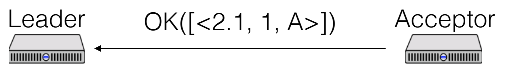
Election revisited

Leader
Acceptor

active: false
ballot_num: 3.0
proposals: [<1, B>]

ballot_num: 3.0
accepted:[<2.1, 1, A>]

Election revisited



active: false
ballot_num: 3.0
proposals: [<1, B>]

ballot_num: 3.0
accepted:[<2.1, 1, A>]

Election revisited

Leader
Acceptor

active: true
ballot_num: 3.0
proposals: [<1, A>]

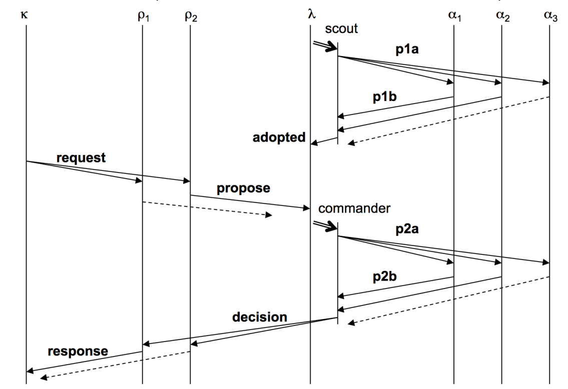
ballot_num: 3.0
accepted:[<2.1, 1, A>]

Leaders

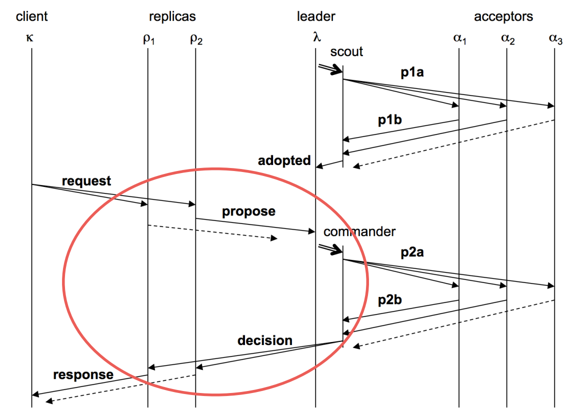
- Only propose one value per ballot and slot

- If a value *v* is chosen by a majority on ballot *b*, then any value proposed by any leader in the same slot on ballot *b'* > *b* has the same value

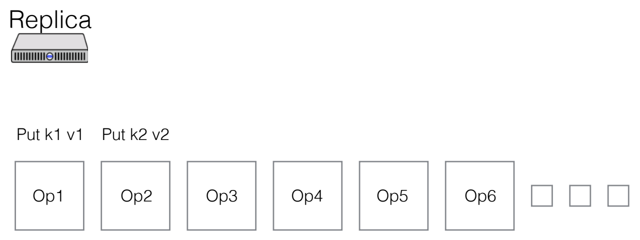
Paxos Made Moderately Complex Made Simple



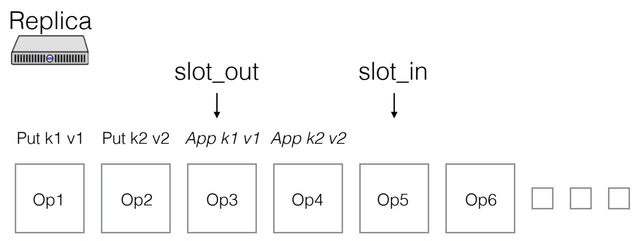
Paxos Made Moderately Complex Made Simple



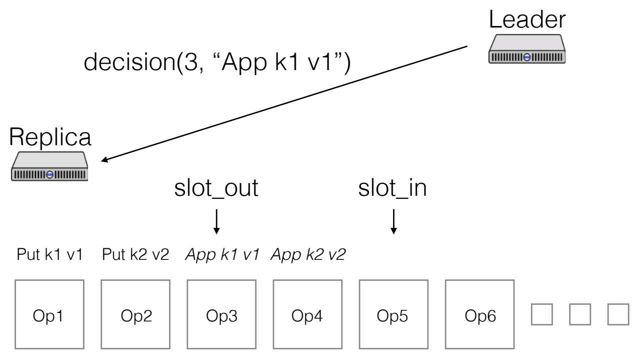
Replicas



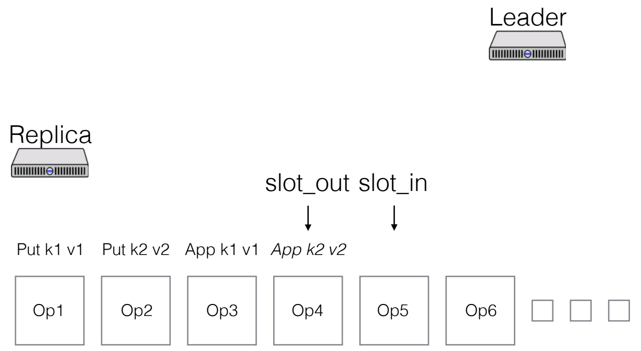
Replicas



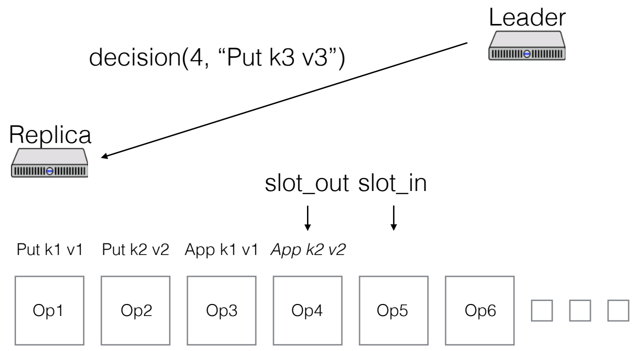
Replicas



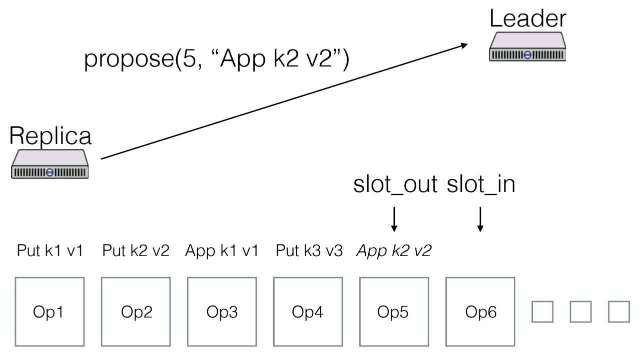
Replicas



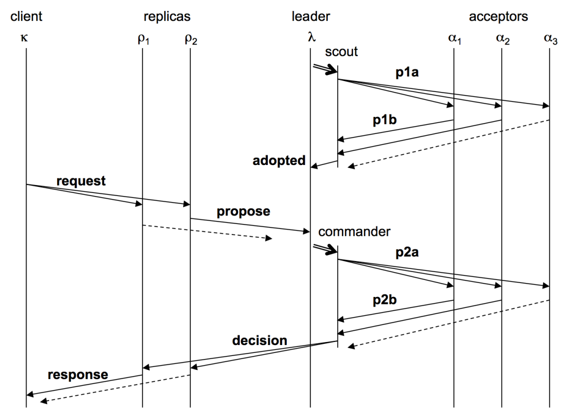
Replicas



Replicas



Paxos Made Moderately Complex Made Simple



When to run for office

When should a leader try to get elected?

- At the beginning of time
- When the current leader seems to have failed

Paper describes an algorithm, based on pinging the leader and timing out

If you get preempted, don’t immediately try for election again!

Reconfiguration

All replicas *must* agree on who the leaders and acceptors are

How do we do this?

Reconfiguration

All replicas *must* agree on who the leaders and acceptors are

How do we do this?

- Use the log!
- Commit a special reconfiguration command
- New config applies after WINDOW slots

Reconfiguration

What if we need to reconfigure *now* and client requests aren’t coming in?

Reconfiguration

What if we need to reconfigure *now* and client requests aren't coming in?

- Commit no-ops until WINDOW is cleared

Other complications

State simplifications

- Can track much less information, esp. on replicas

Garbage collection

- Unbounded memory growth is bad
- Lab 3: track finished slots across all instances, garbage collect when everyone has learned result

Read-only commands

- Can't just read from replica (why?)
- But, don't need their own slot