Byzantine Paxos
(From Paxos to PBFT)

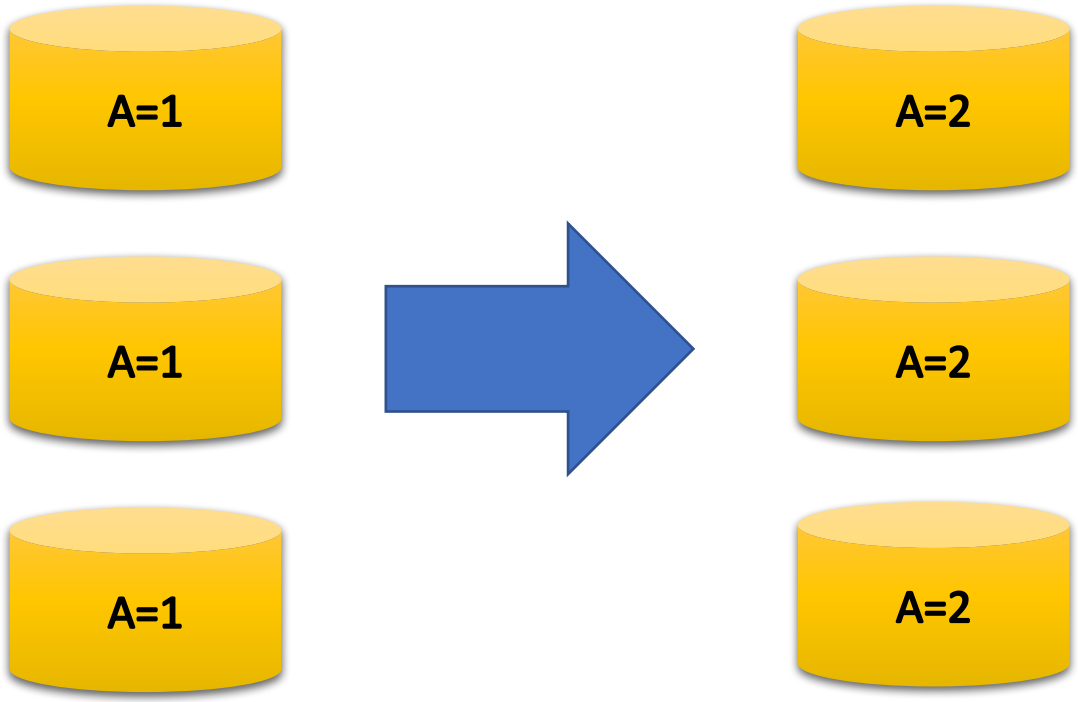
Presenter: Shuai Mu

Structure

- **Background**
  - Replication and consensus
  - Challenges
  - Byzantine failures
- Paxos Review
- ByzPaxos (Non-leader lies)
- ByzPaxos (Leader lies)

Replication and consensus

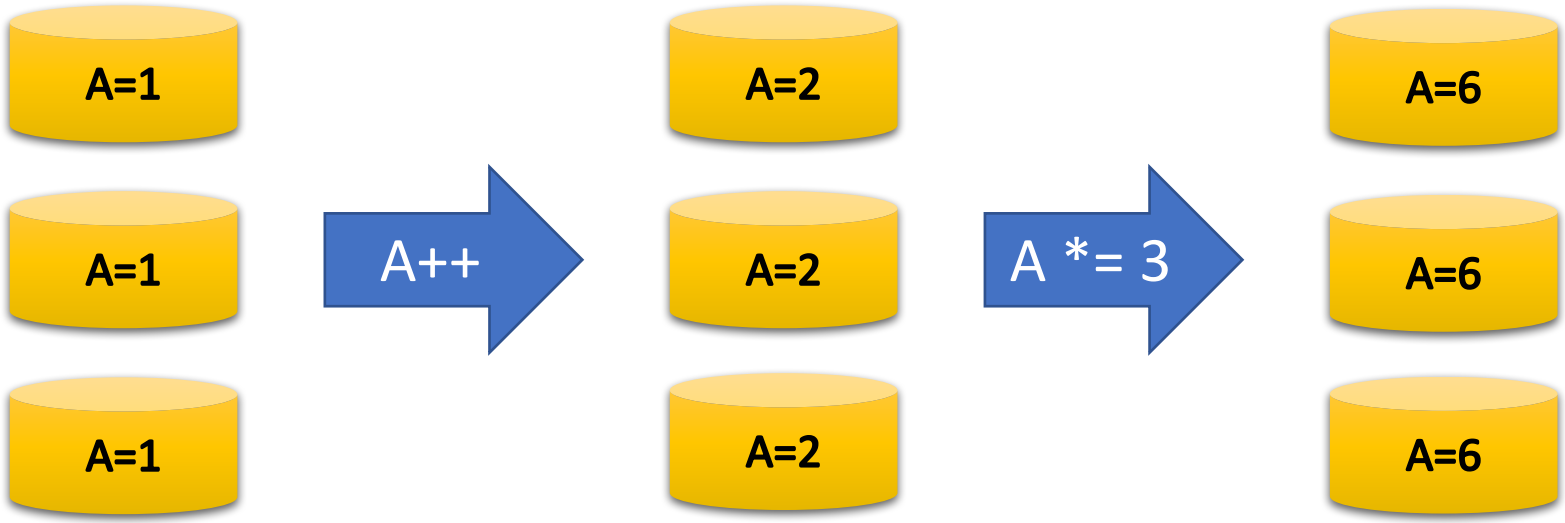
- Replication is usual way to achieve reliability
- Each replica must be consistent
- Safety and liveness with a number of failures



Replicated state machine

- Each step must be identical across replicas
- you can make one decision, iff. you can make a series of decisions

Alice: A++
Bob: A*=3



Why making a decision is difficult?

- Asynchronous
  - Any network partitions
- Fault tolerance (Crashes)
  - Hopefully tolerate *f* crashes with $N = 2f + 1$ replicas
- Nobody has a God view (there is no almighty leader)
  - Cannot detect failures
- A Classic Solution: Paxos

Let us makes it even harder!

- Byzantine Failures
  - Beyond crashes, any message is possible
- They actually happen:
  - Malicious attackers
  - Unexpected failures in network, disks, etc.
- What we cannot protect
  - A malicious leader following the protocol issuing a harmful command, e.g. delete a table

Restrict BFT failures

- The problem is too difficult with arbitrary forging.
  - If anyone can pretend to be anyone else, then it looks like anyone is faulty

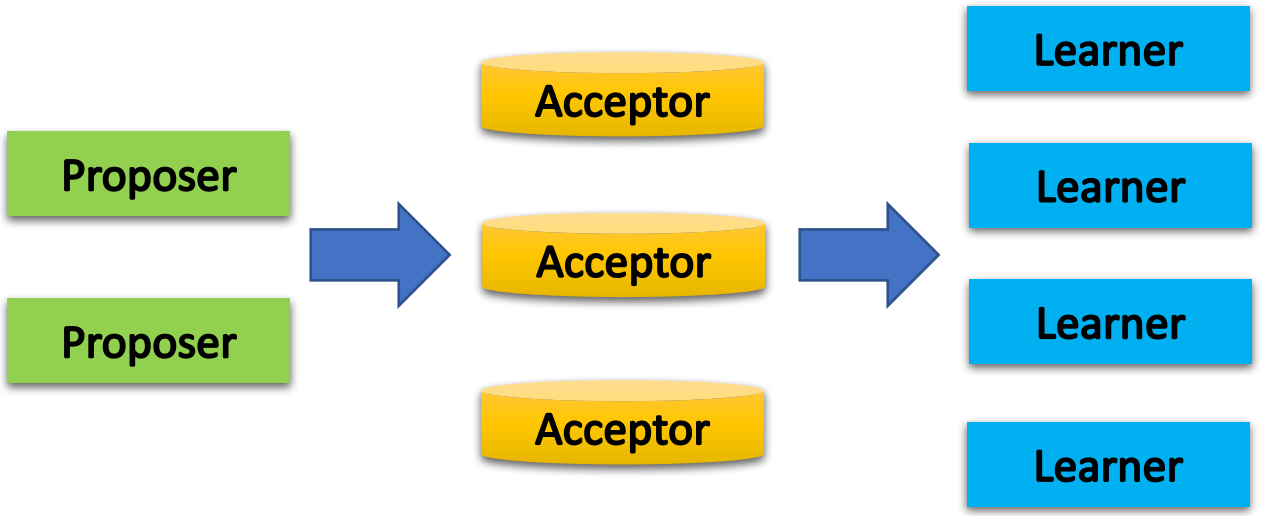
- Disable forging by signing.
  - Everyone knows public key of everyone else
  - Everyone signs its message with private key

Structure

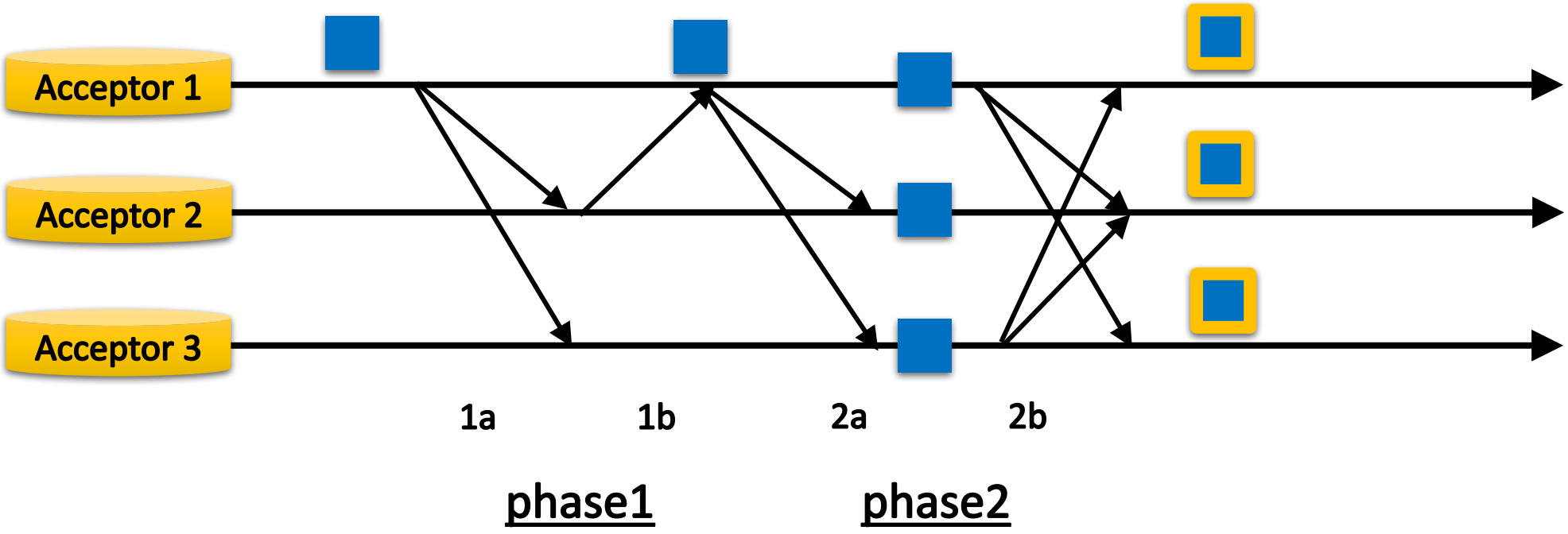
- Background
- **Paxos Review**
  - Overview
  - Protocols
  - Example
- ByzPaxos (Non-leader lies)
- ByzPaxos (Leader lies)

Paxos Review

- Problem
  - Make a decision
- Roles
  - Proposer
  - Acceptor
  - Learner



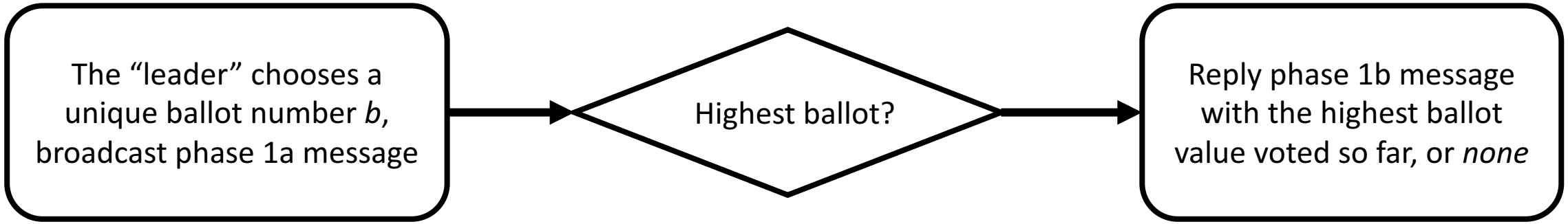
Protocol Overview



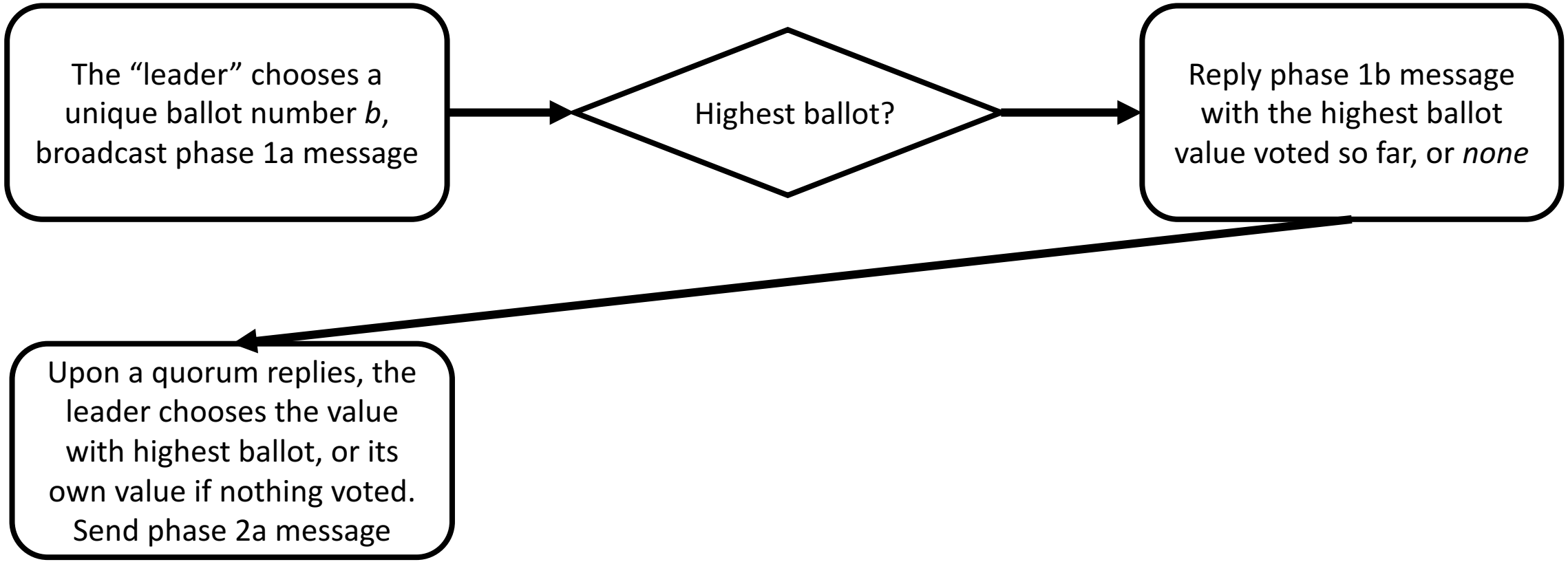
Phase 1a-1b-2a-2b

The “leader” chooses a unique ballot number *b*, broadcast phase 1a message

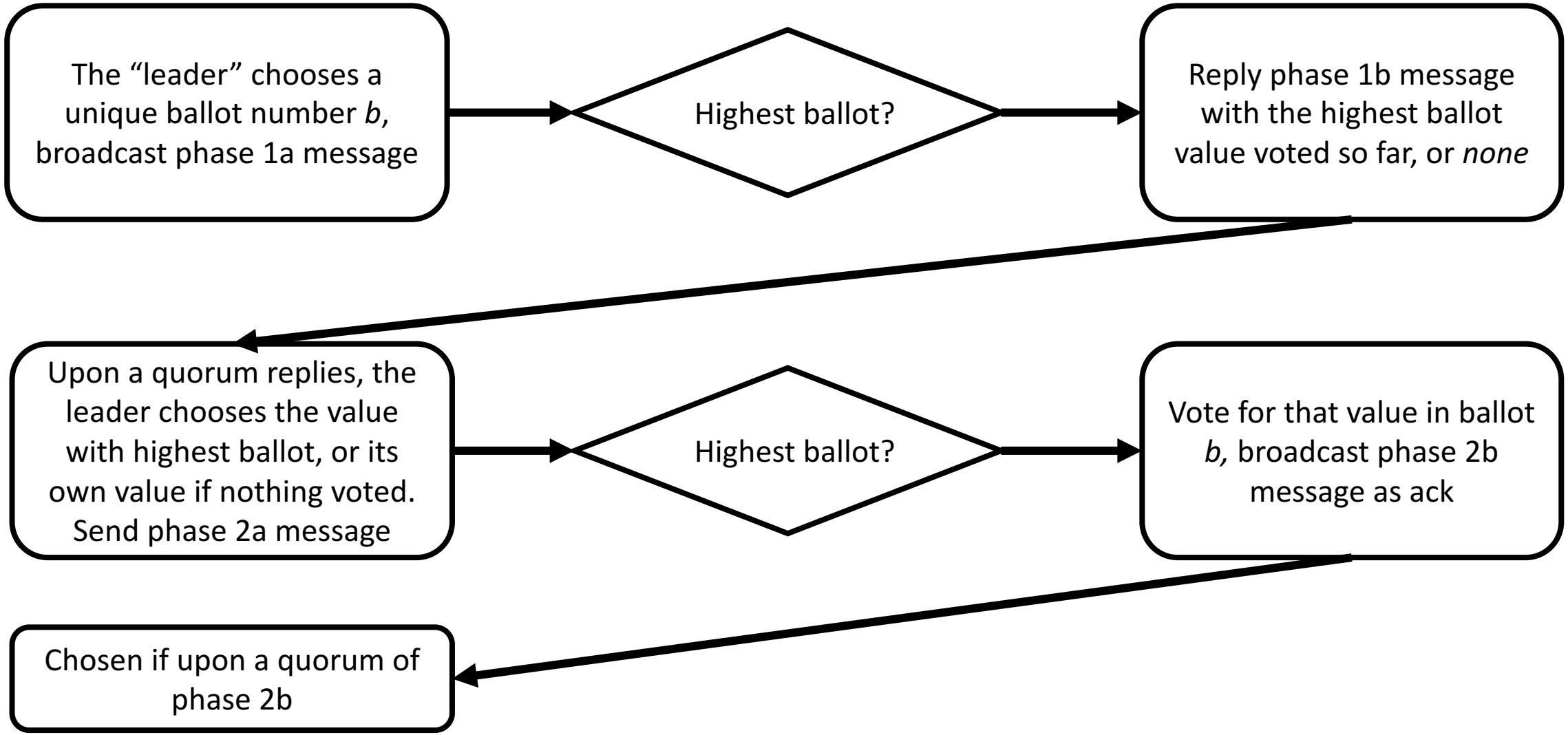
Phase 1a-1b-2a-2b



Phase 1a-1b-2a-2b

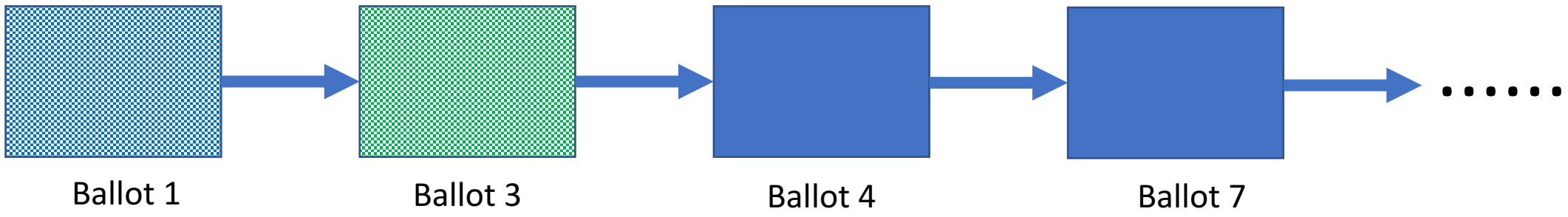


Phase 1a-1b-2a-2b



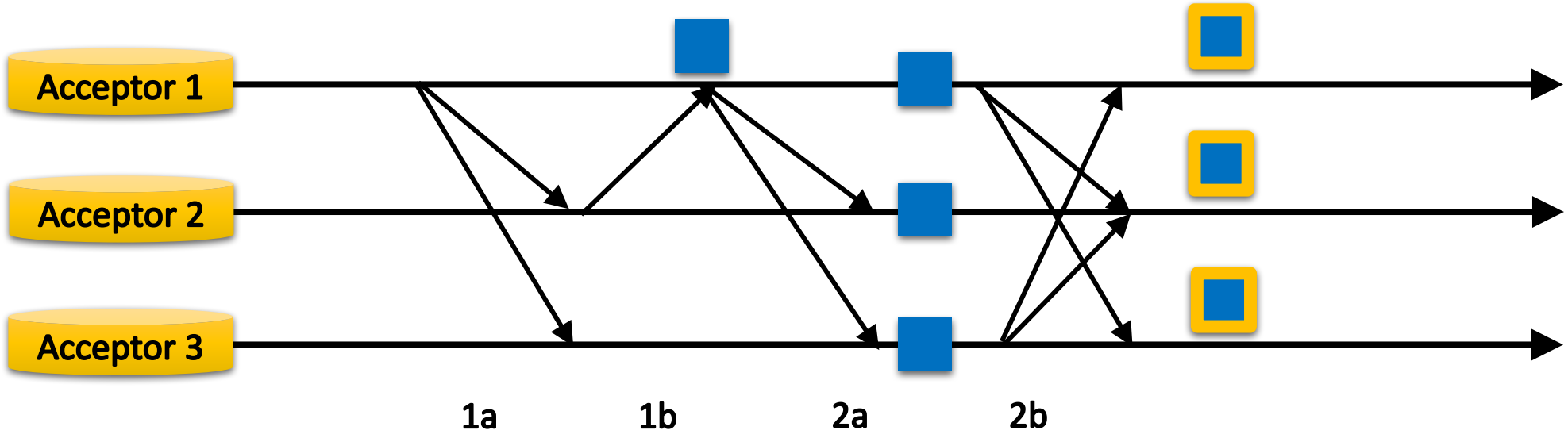
How it works intuitively

- Different acceptors cannot vote for different values in the same ballot.
- An acceptor can vote for different values in different ballots.
- An acceptor can vote for a value iff no other value was(will be) chosen previously

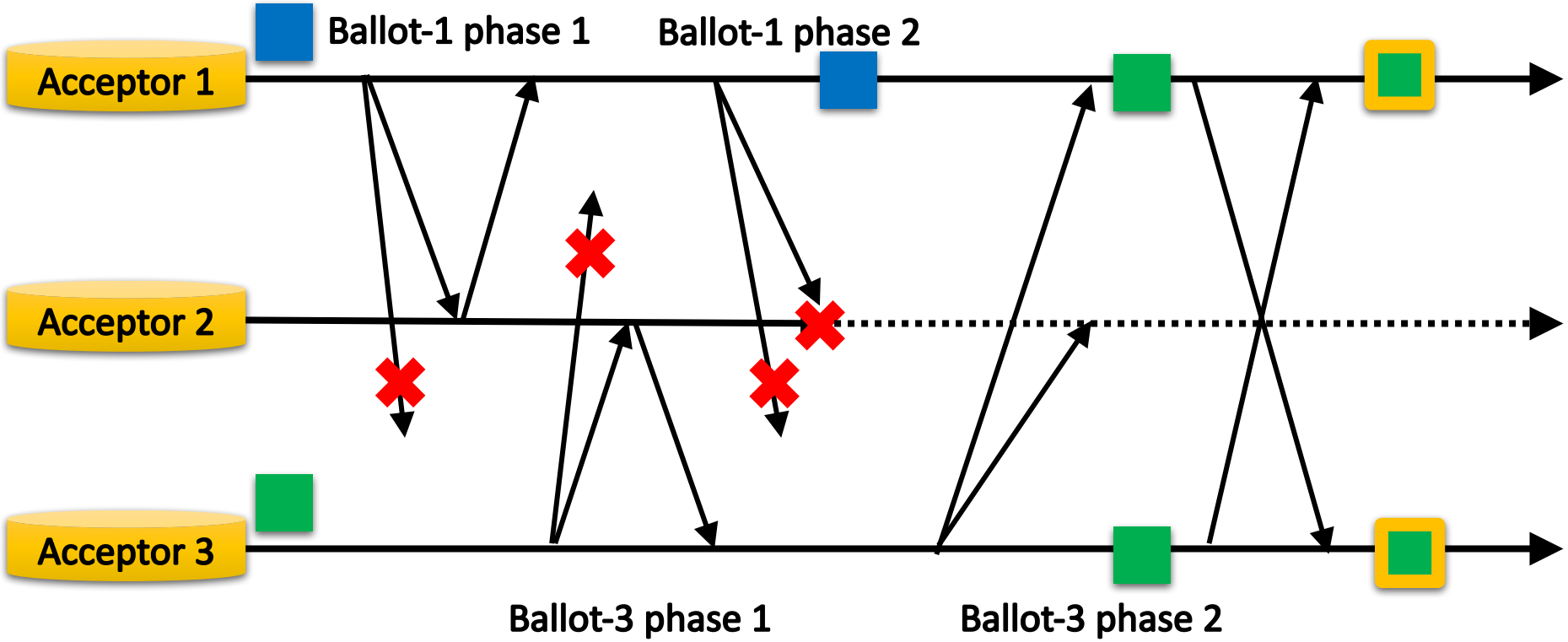


Magic happens between phase 1b and 2a

- If no acceptor in the quorum has voted in a ballot numbered less than *b,* then all values are safe at *b*.
- If some acceptor in the quorum has voted, let *c* be the highest-numbered ballot less than *b* in which such a vote was cast. The value voted for in ballot *c* is safe at *b*.



Example


Ballot-1 phase 1
Ballot-1 phase 2
Acceptor 1
Acceptor 2
Acceptor 3
Ballot-3 phase 1
Ballot-3 phase 2

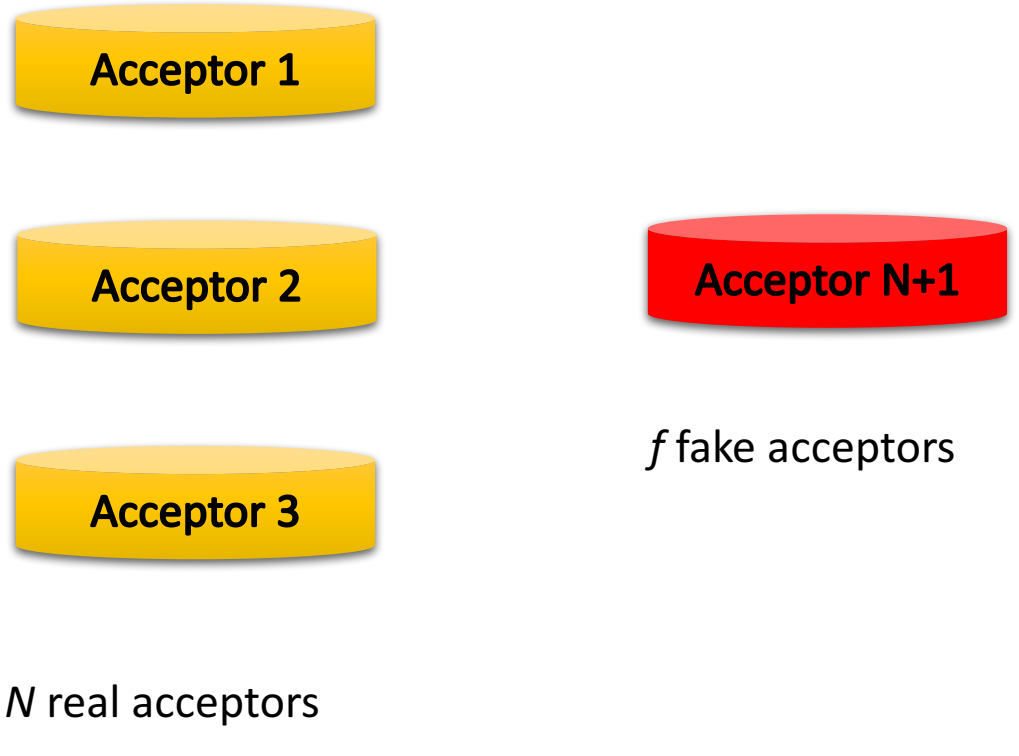
Structure

- Background
- Paxos Review
- **ByzPaxos (Non-leader lies)**
  - Goal
  - Solution w/ larger quorum
  - Reduce quorum size
- ByzPaxos (Leader lies)

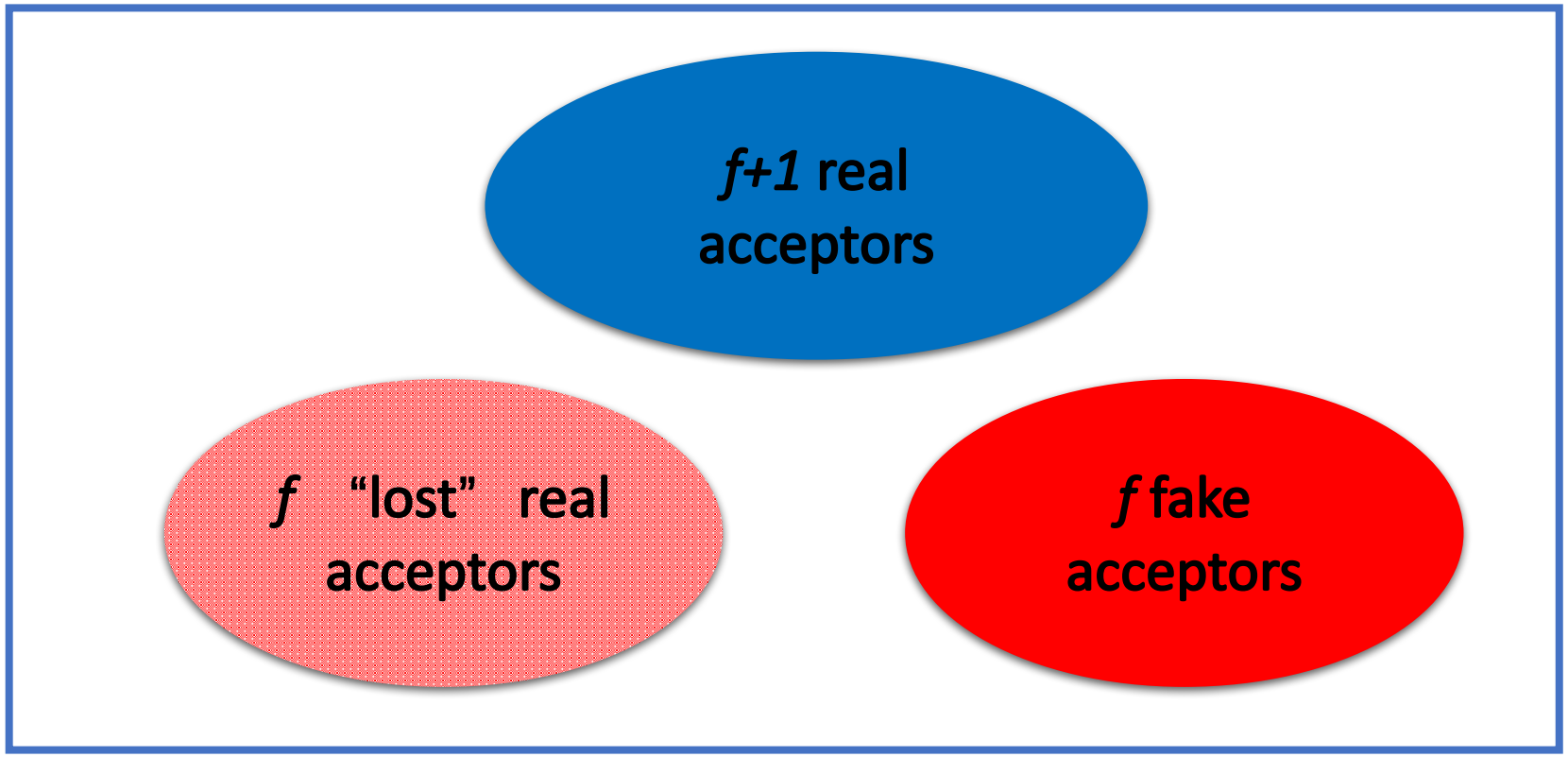
Let us bring in fake acceptors

- *N+f* byz-acceptors
  - *N* real acceptors
  - *f* fake acceptors,
  - Non-distinguishable

- Consensus?
  - If so, what size is *N*?
  - *N,* in the best case, is *2f+1,* which means *N'=N+f=3f+1*



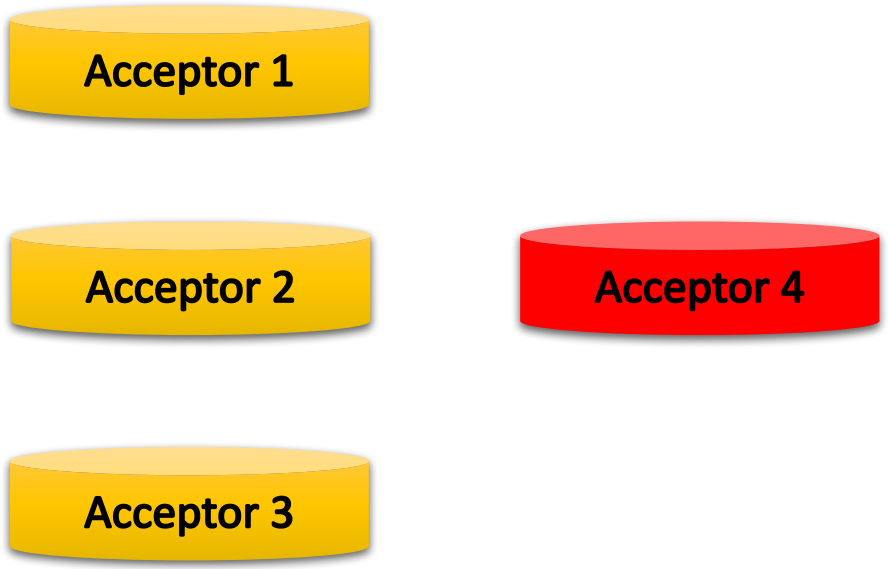
Why at least *3f+1* acceptors?



Naïve solution: bring extra *f* into Paxos

- New byz-quorum *q' = q + f*
  - Every two byz-quorums intersect with *f+1* acceptors
  - *f+1* acceptors must contain a real acceptor.

- Simplify the problem: assume ballot-leaders cannot lie

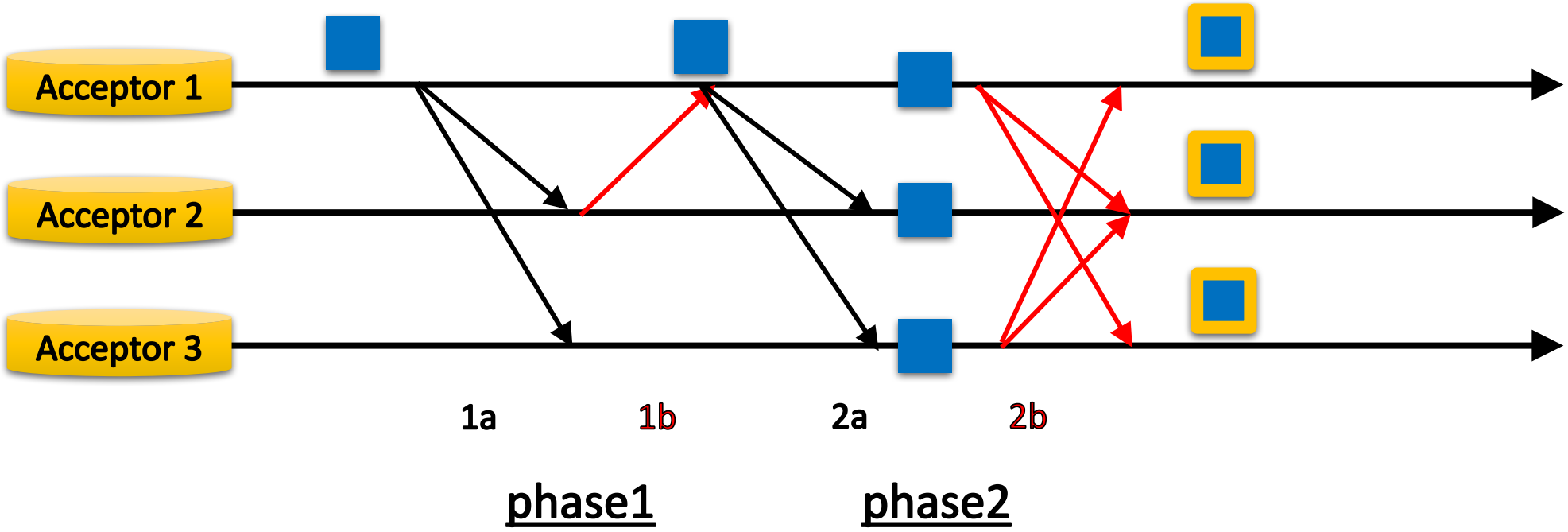


N=3, q=2, intersection=1

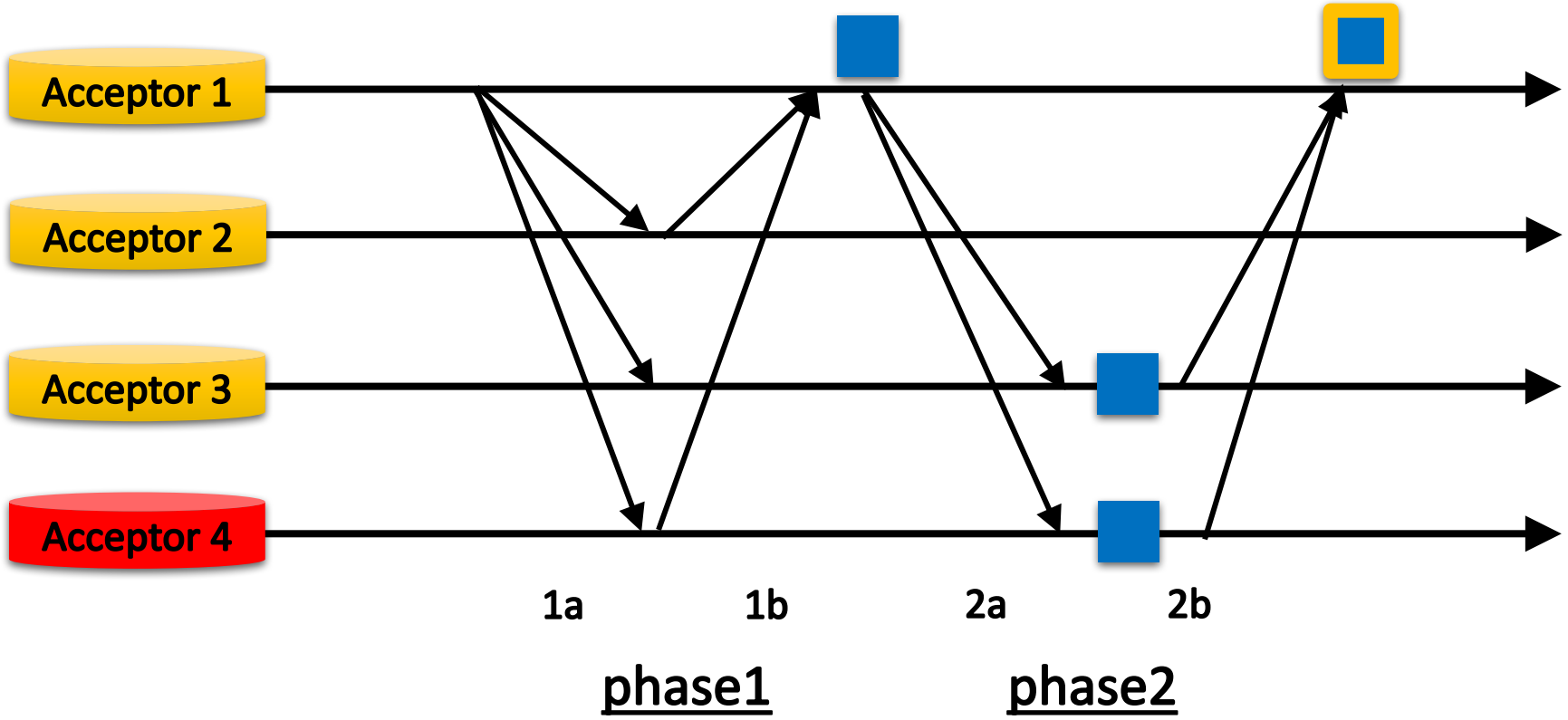
f=1, N'=4, q'=3, intersection=2

Liars in the system

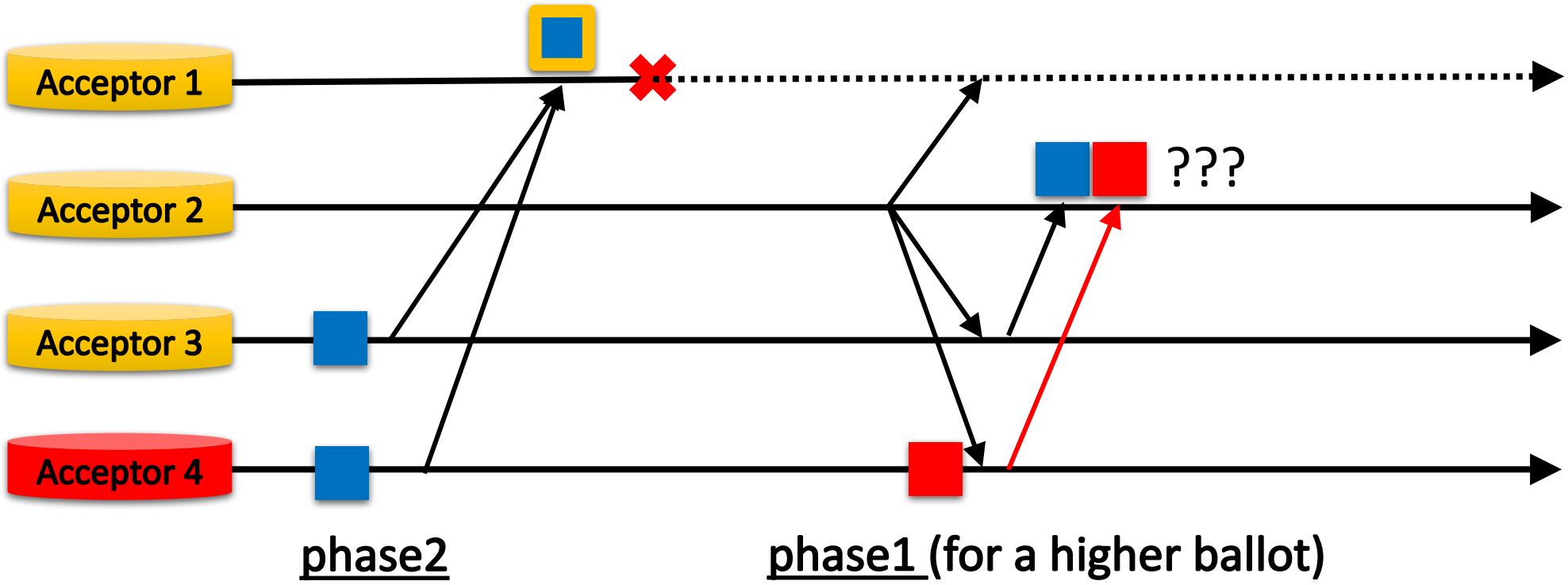
- ~~Lies in Phase 1a, 2a~~
- Lies in Phase 1b, 2b



Modified Protocols with byz-quorum

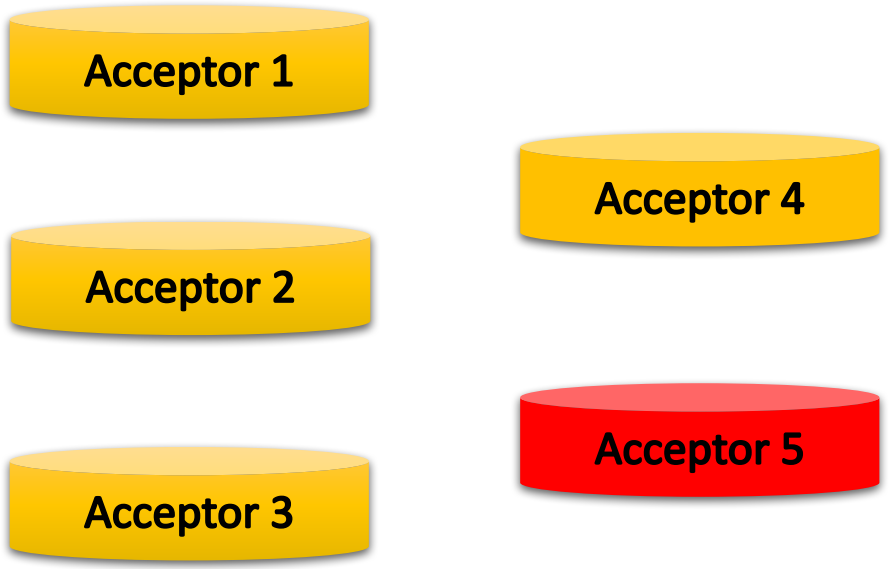


Acceptor can lie in phase 1b



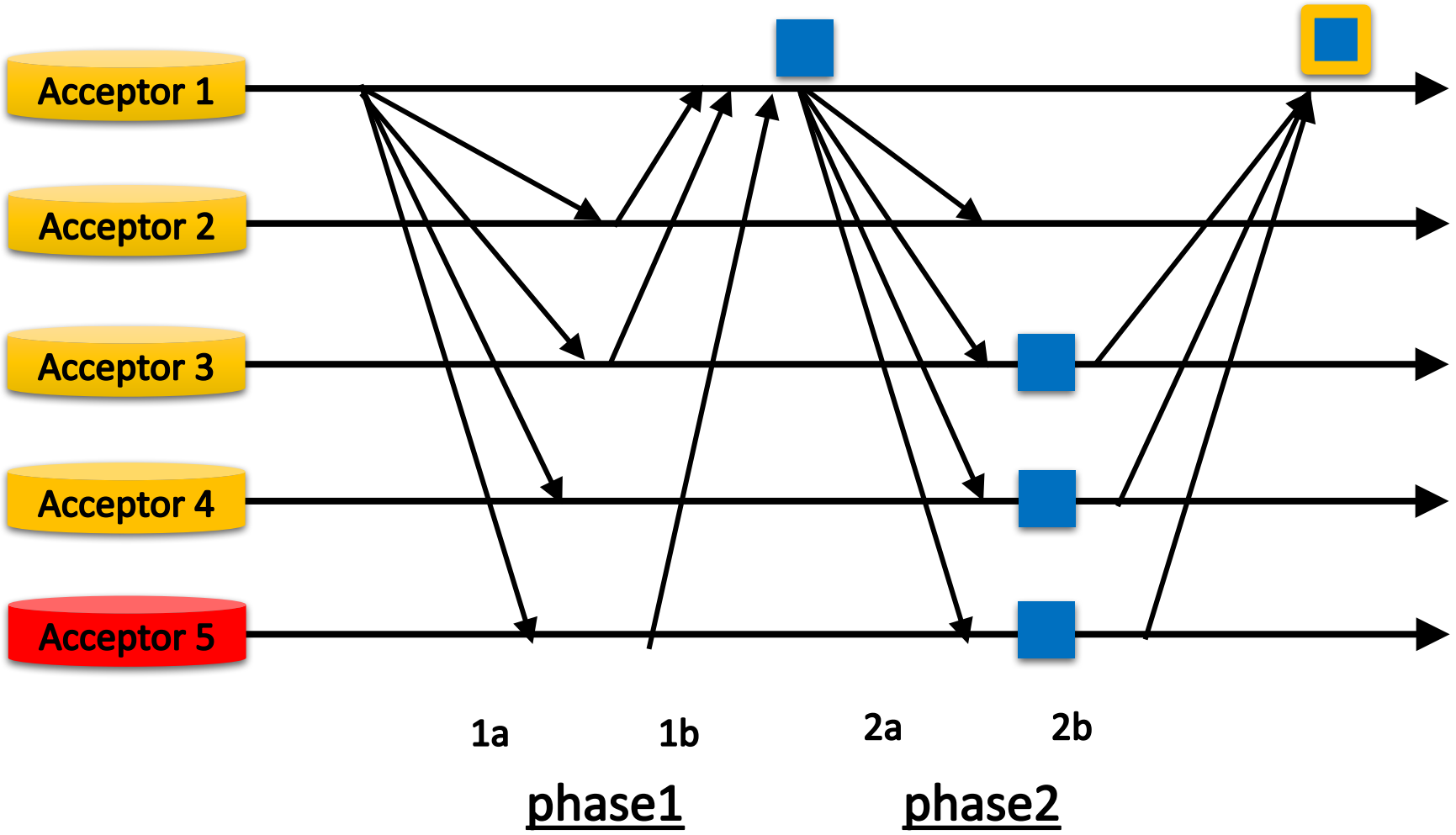
How to outvote liar

- A possible solution: increase the quorum size
  - Original quorum size *q = 2f+1,*
  - Original quorum Intersection = *f+1*
  - Original *N = 3f+1*
  - Byz-quorum *q'=3f+1*
  - Byz-quorum intersection = *2f+1*
  - Byz acceptors *N' = 4f+1*

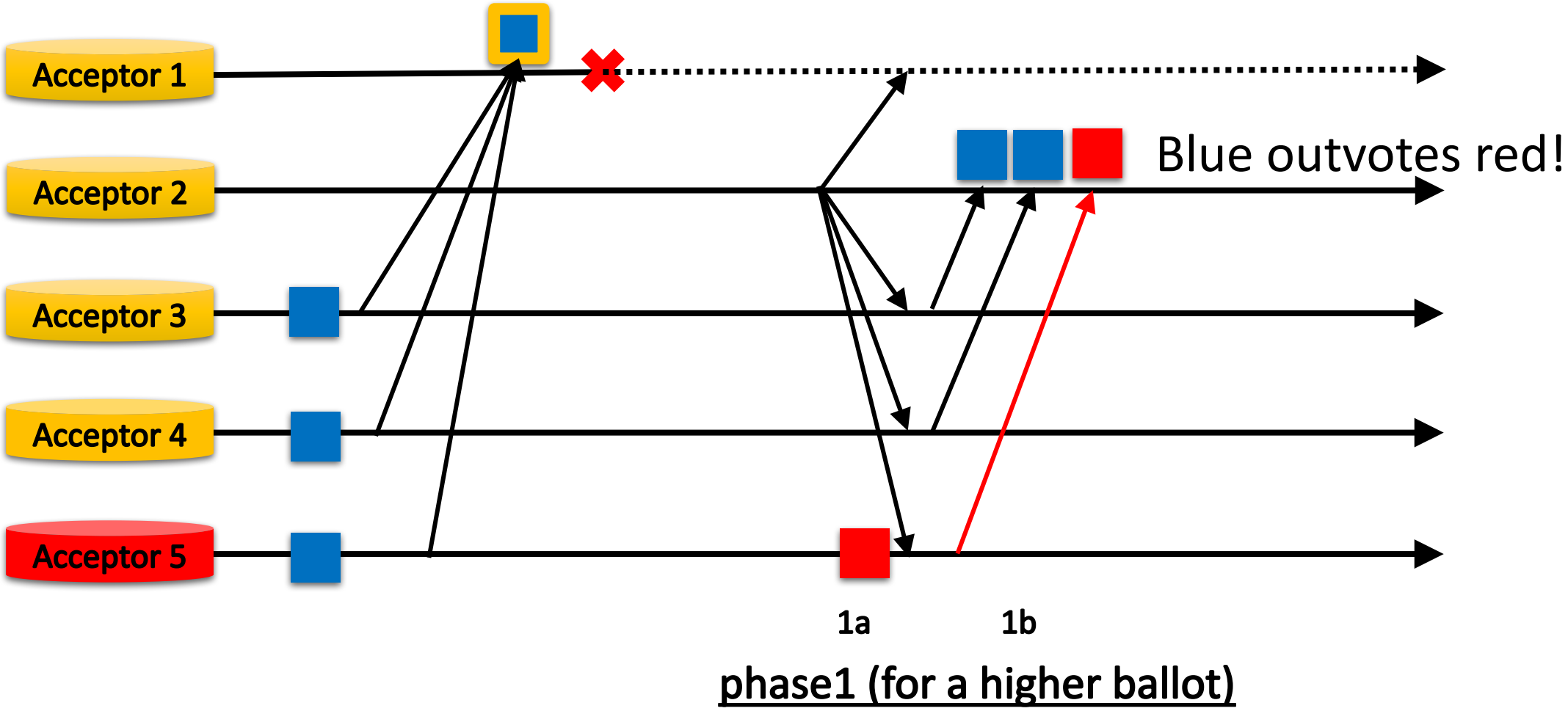


N=4, q=3, intersection=2
f=1, N'=5, q'=4, intersection=3

Modified Protocols with byz-quorum



Example


Acceptor 1
Acceptor 2
Acceptor 3
Acceptor 4
Acceptor 5
Blue outvotes red!
1a
1b
phase1 (for a higher ballot)

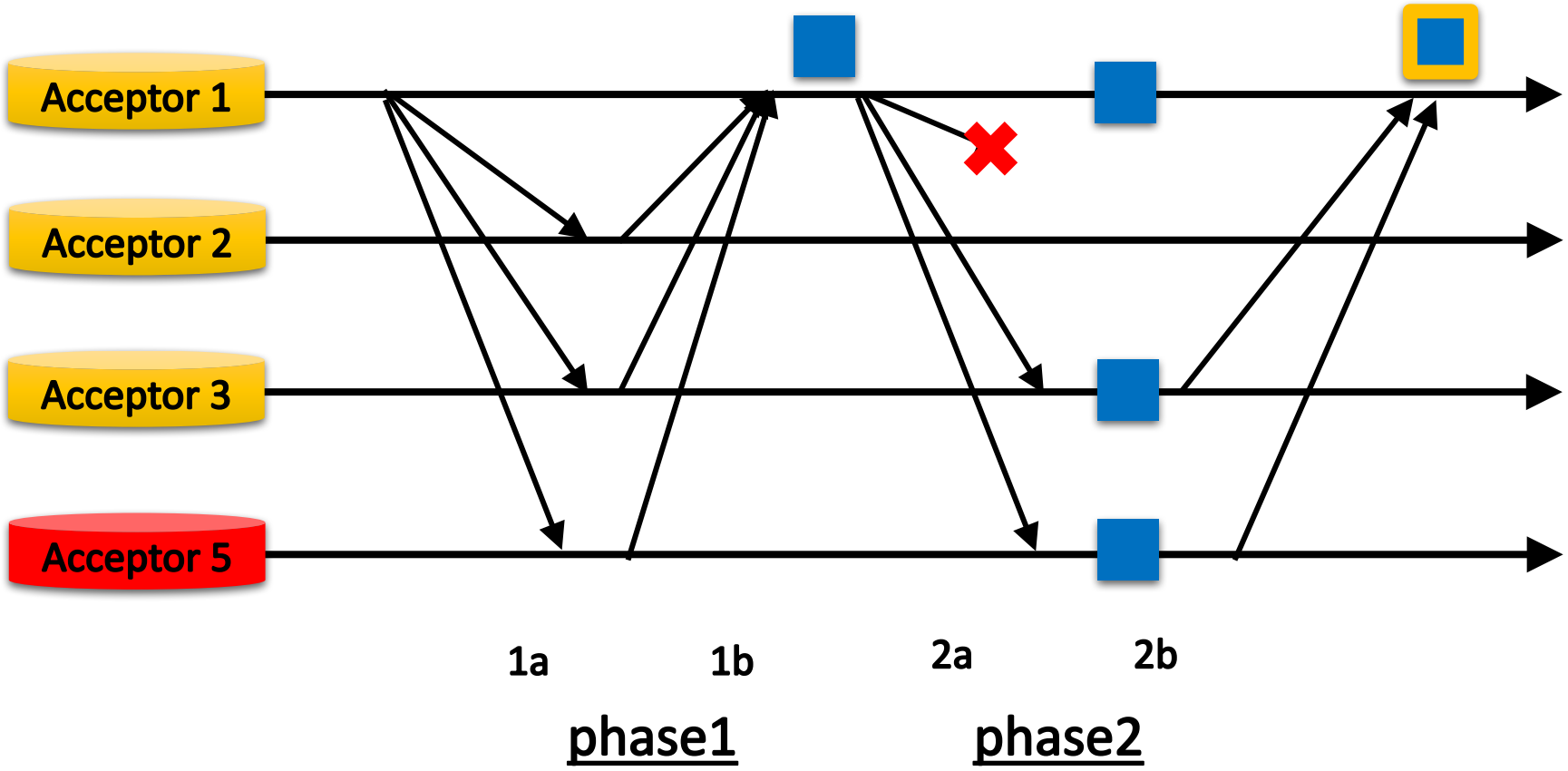
Problem now: too many acceptors

- $N' = N + f = 4f + 1$

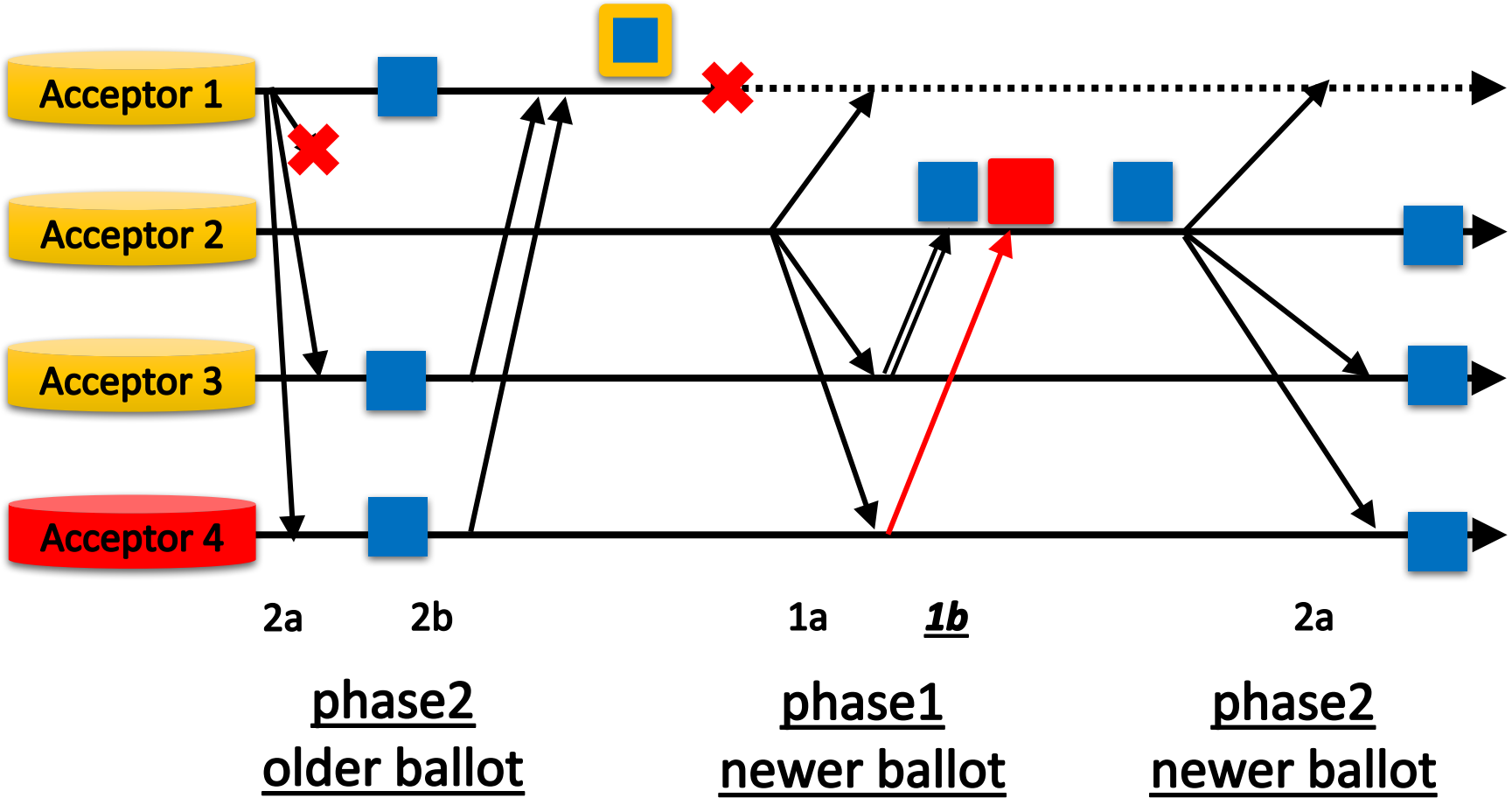
- Can we shrink the size of acceptors?
  - From *4f+1* to *3f+1*?

- Alternative solution: disable the lies in phase 1b
  - Require the value in 1b message be attached with a “proof”
  - Assert that this value was actually a safe value, not a fake value
  - The proof is implied by phase 2a message (assumes no lies of 2a)

Example


Acceptor 1
Acceptor 2
Acceptor 3
Acceptor 5
1a
1b
2a
2b
phase1
phase2

Example (cont.)



Question

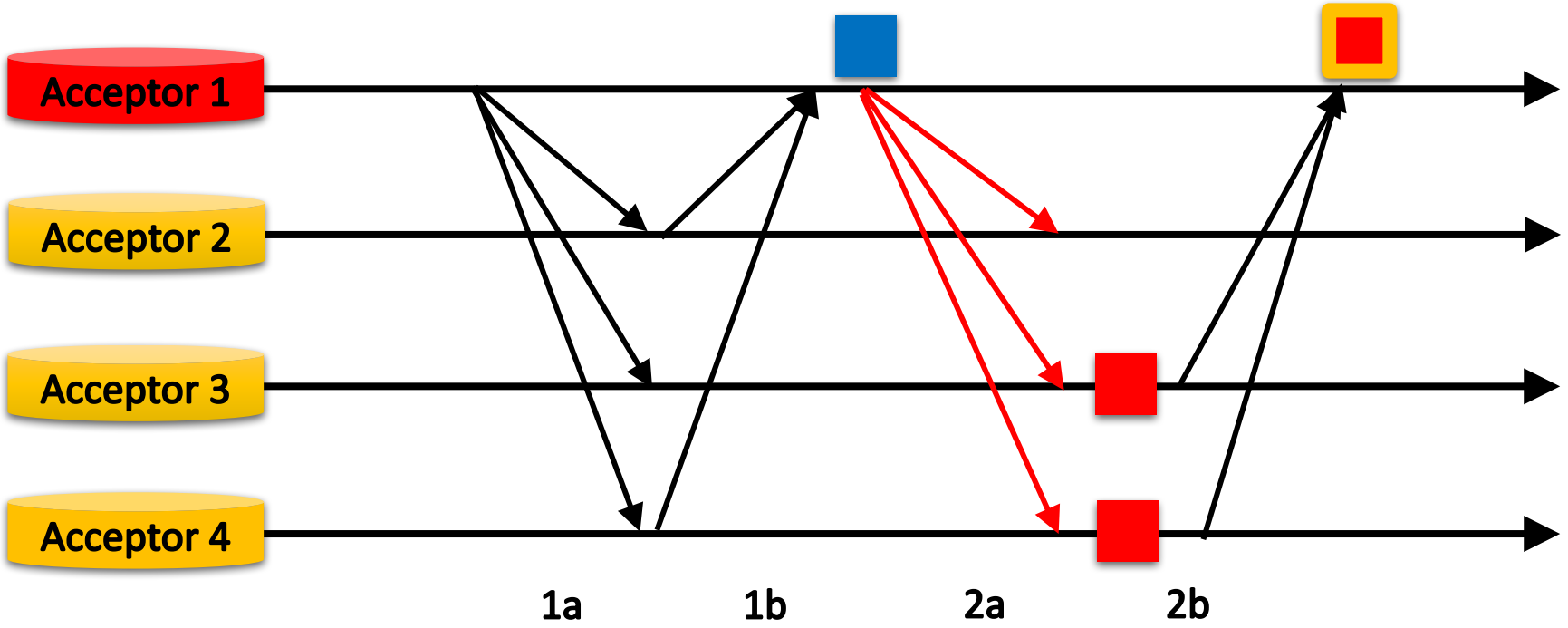
- In phase 1b, can acceptor still lie without being caught?
- If so, does it hurt?
- What about lies in phase 2b?

Structure

- Background
- Paxos Review
- ByzPaxos (Non-leader lies)
- **ByzPaxos (Leader lies)**
  - Break down phase 2a
  - Cooperative phase 2a
  - Multi-instance

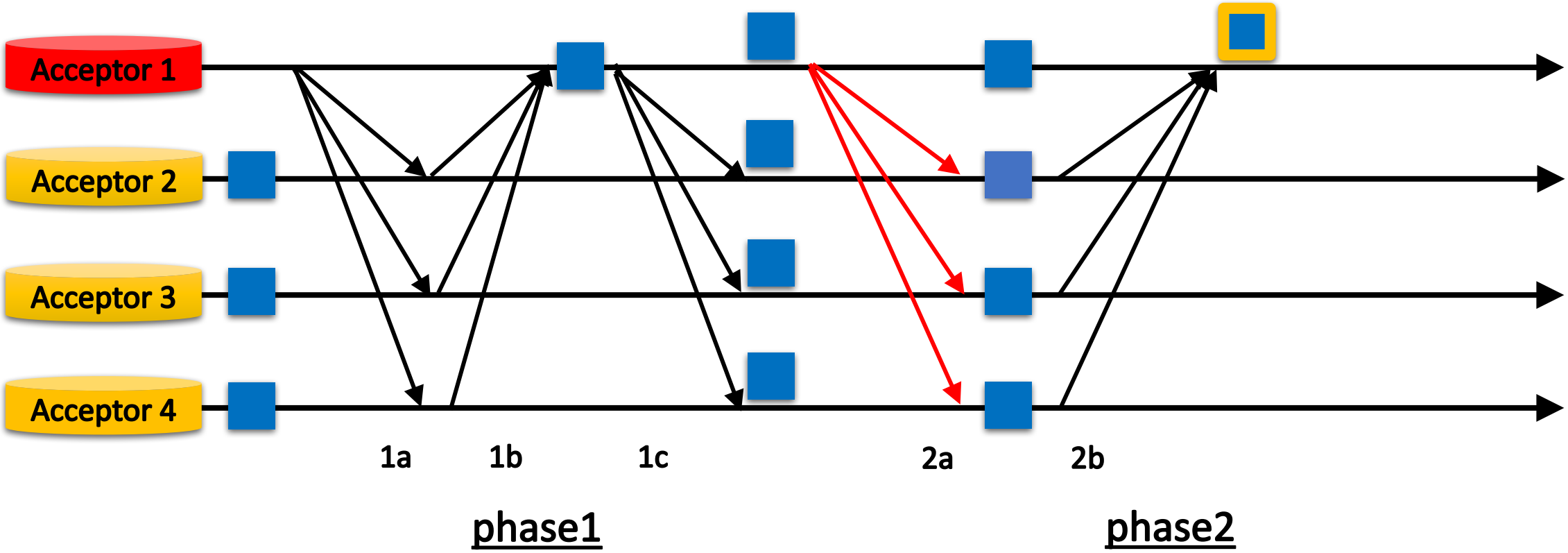
A fake acceptor can lie as leader for a ballot

- Cannot lie in phase 1a
- Lie in phase 2a matters!
  - Lying leader can feed any value
  - Acceptor does not know how to validate if it is “legal” (safe)

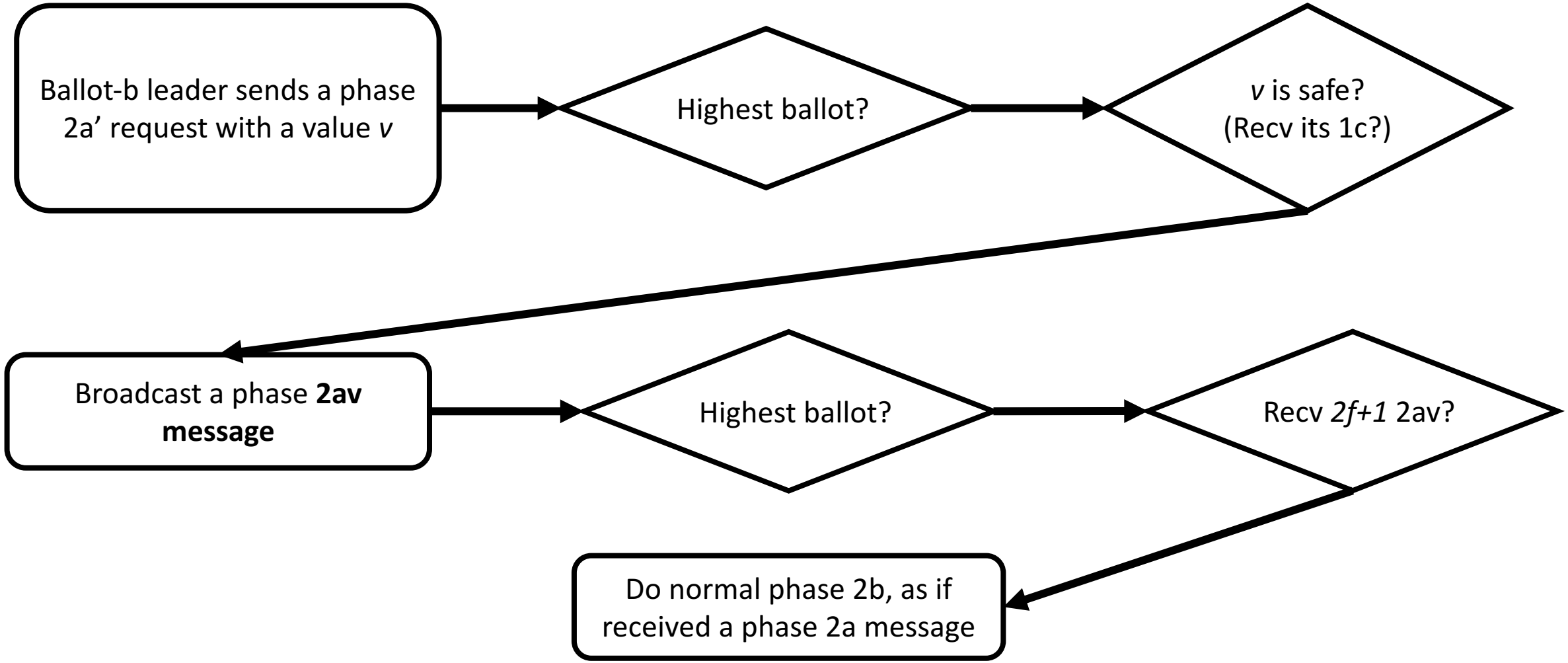


Extract phase 1c from phase 2a

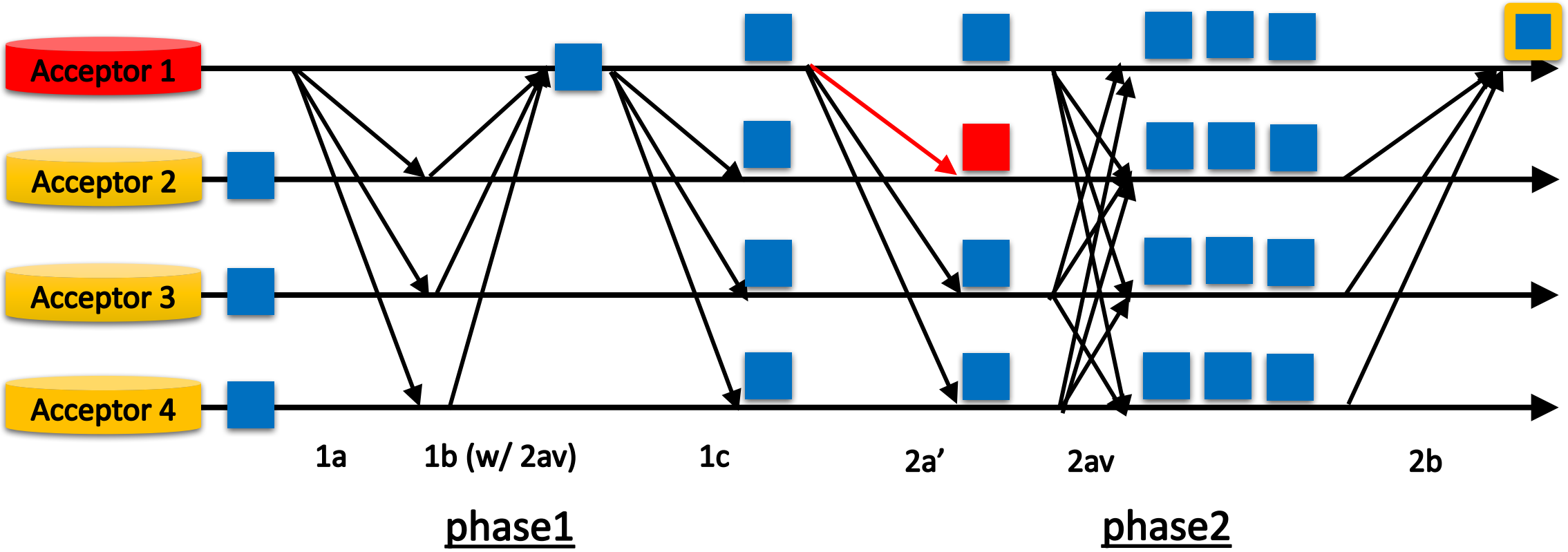
- Give acceptors more information to know what is safe
- Phase 1c message for ballot *b* asserts some (all) values are safe



Dealing with lies w/ cooperative phase 2a

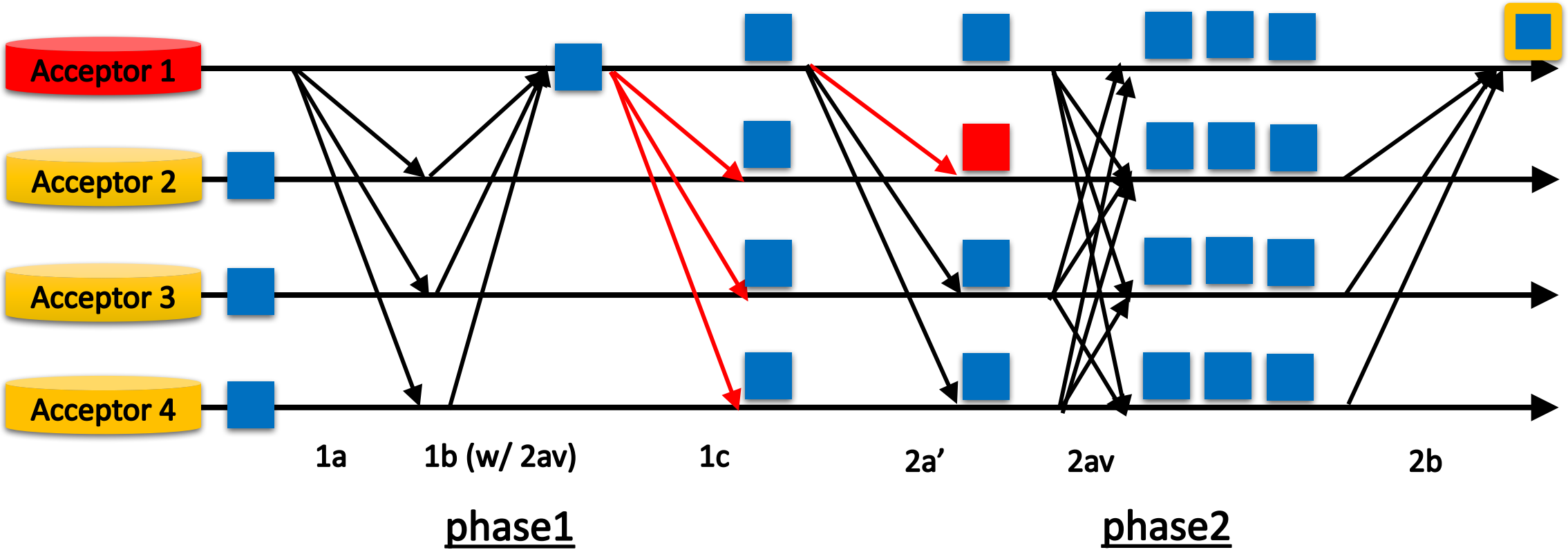


Example


Acceptor 1
Acceptor 2
Acceptor 3
Acceptor 4
1a
1b (w/ 2av)
1c
2a'
2av
2b
phase1
phase2

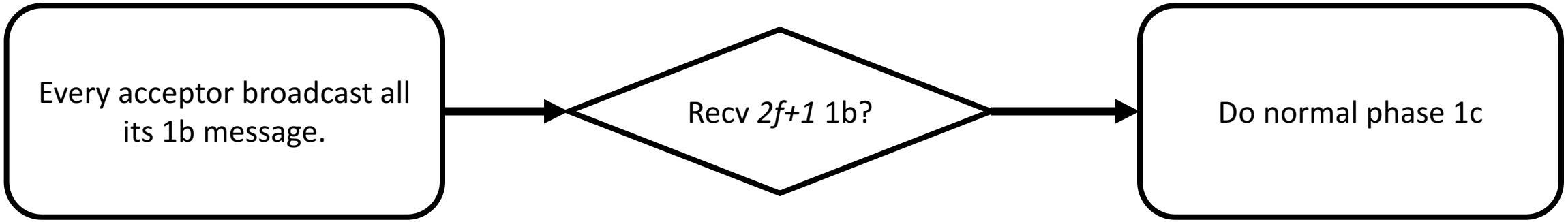
What about lies in phase 1c?

- Can we validate phase 1c message?

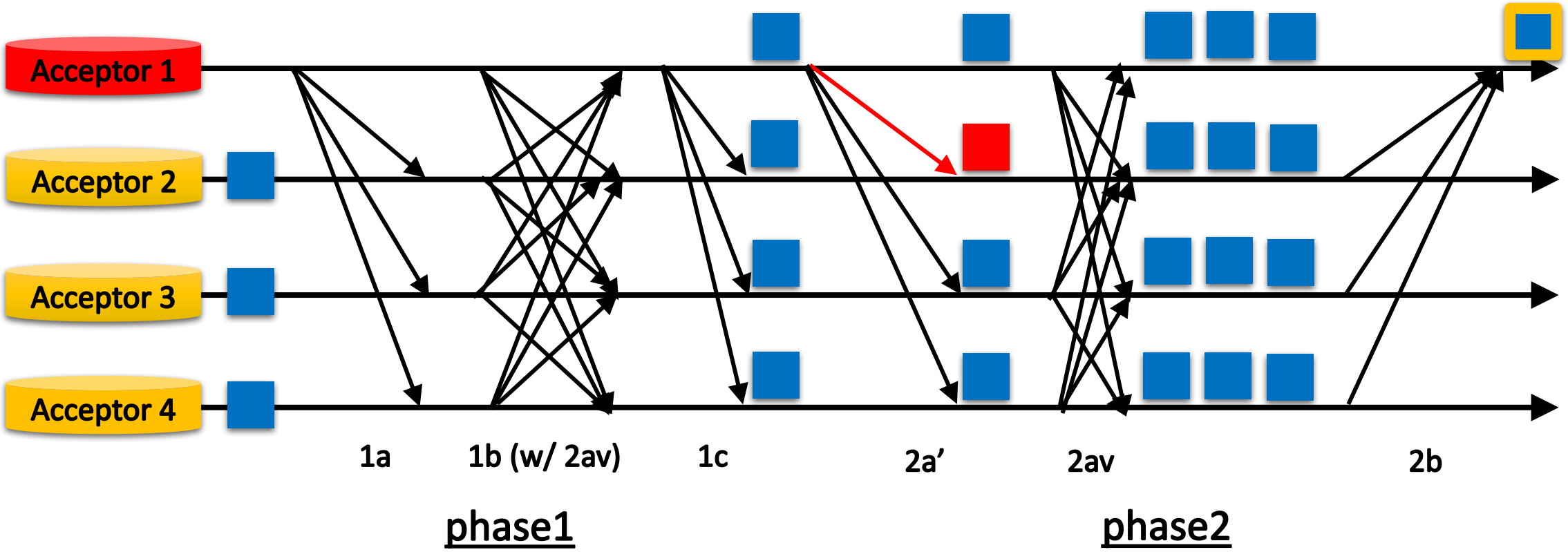


Deduct phase 1c from phase 1b

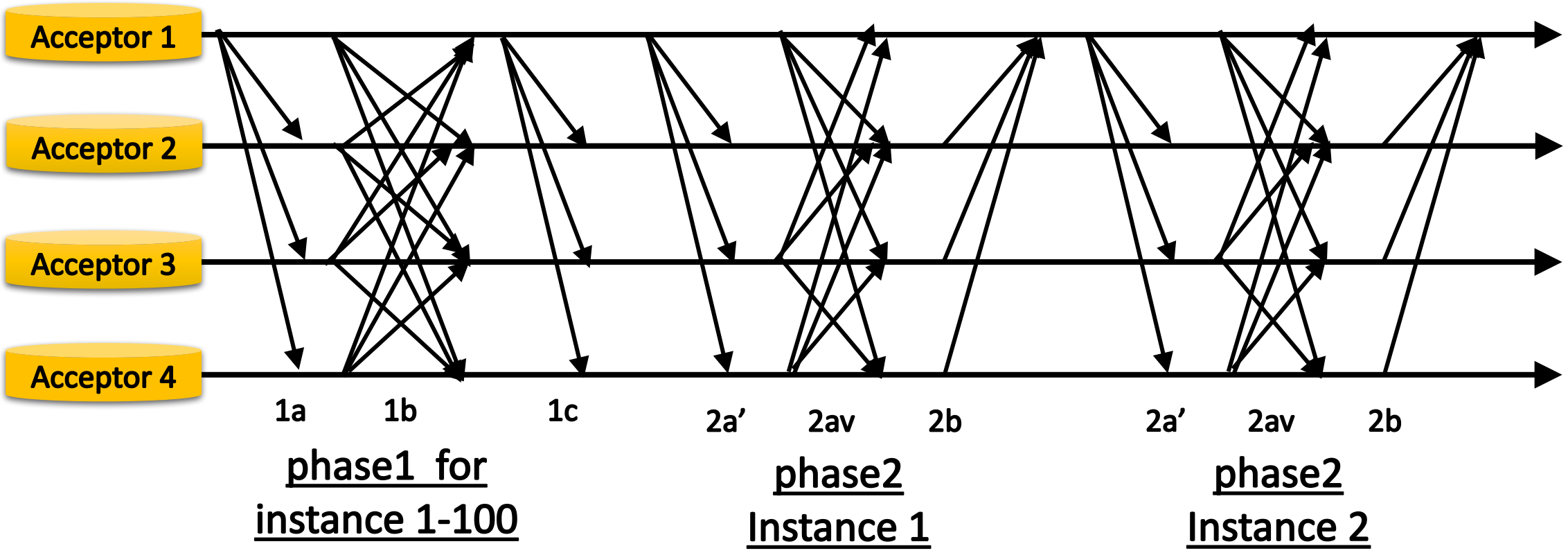
- The ballot-*b* leader sends 1c on receiving a byz-quorum of 1b
- Prevent the leader lying by making every acceptor repeat the process



Example

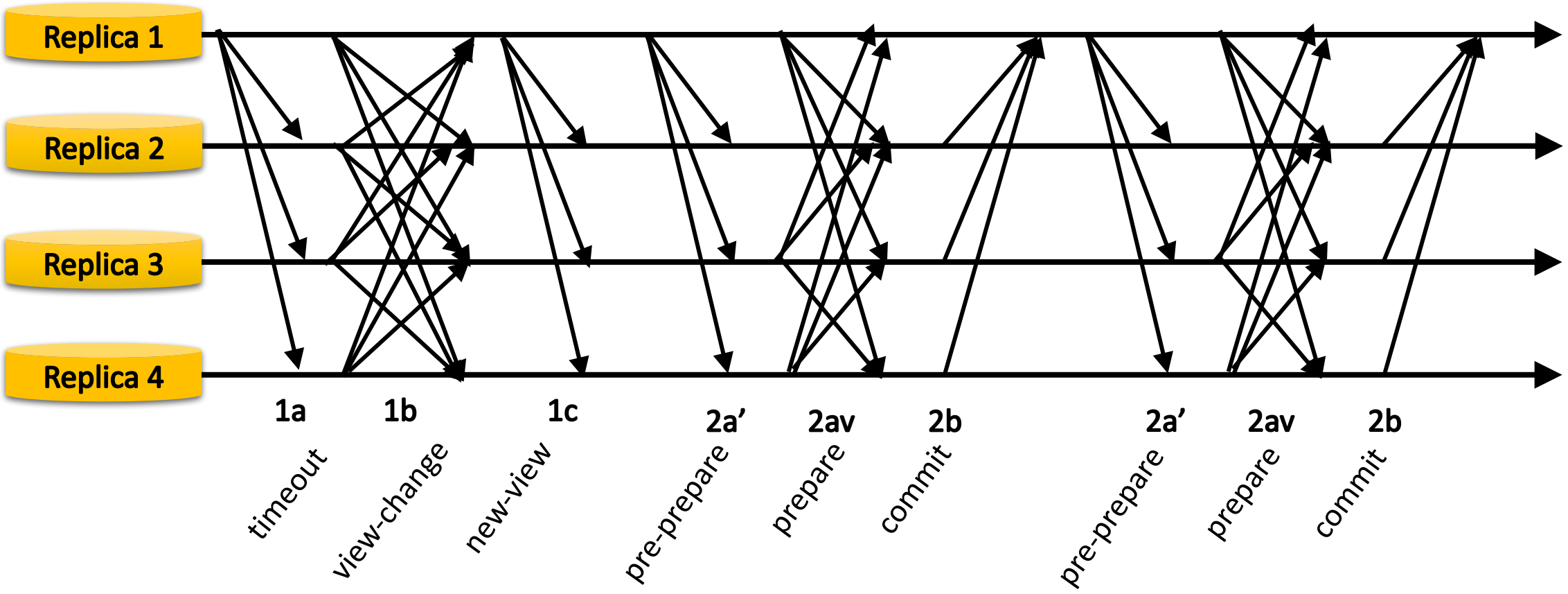

Acceptor 1
Acceptor 2
Acceptor 3
Acceptor 4
1a
1b (w/ 2av)
1c
2a'
2av
2b
phase1
phase2

Multi-instance Example, like Multi-Paxos



A little bit renaming, and look what we get here!

Miguel Castro and Barbara Liskov, *Practical Byzantine Fault Tolerance*, OSDI'99



Summary: the refinements

- Paxos + fake acceptors
- Enlarge quorum to tolerate fake non-leaders
- Attach 1b message with previous 2a message to shrink quorum
- Extract 1c from 2a
- Cooperative 2a using 1c
- Broadcast phase 1b to deduct 1c
- Multiple instances with batched phase 1