



ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ
UNIVERSITY OF CRETE

HY-559

Infrastructure Technologies for Large-Scale Service-Oriented Systems

Kostas Magoutis

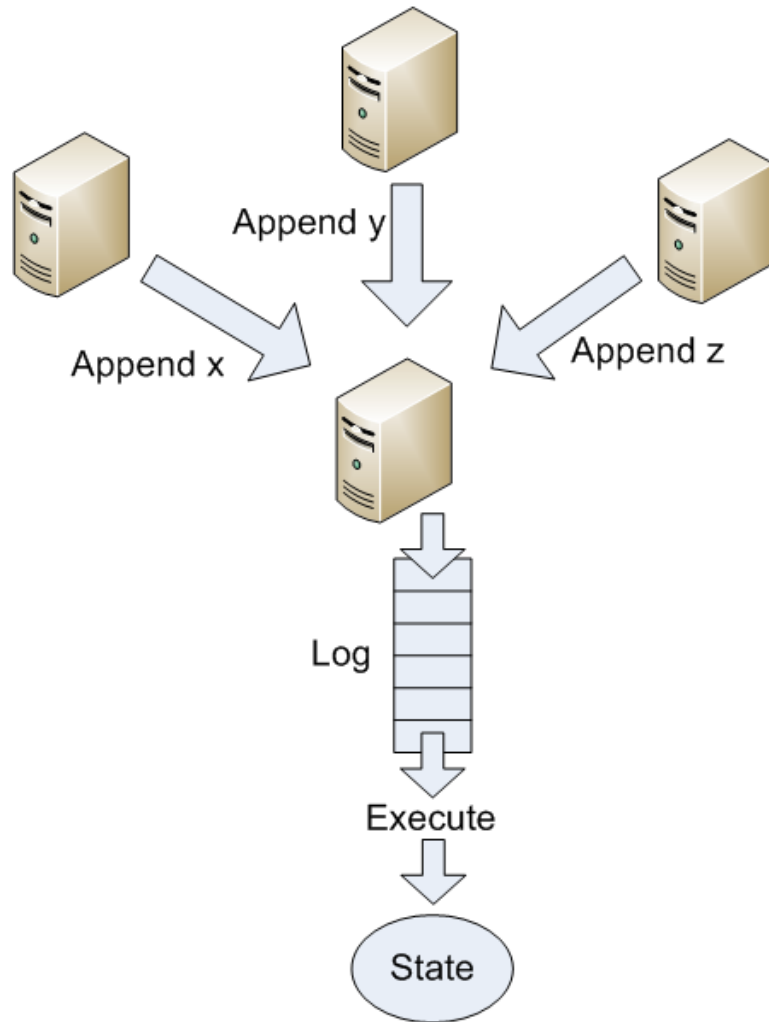
magoutis@csd.uoc.gr

<http://www.csd.uoc.gr/~hy559>

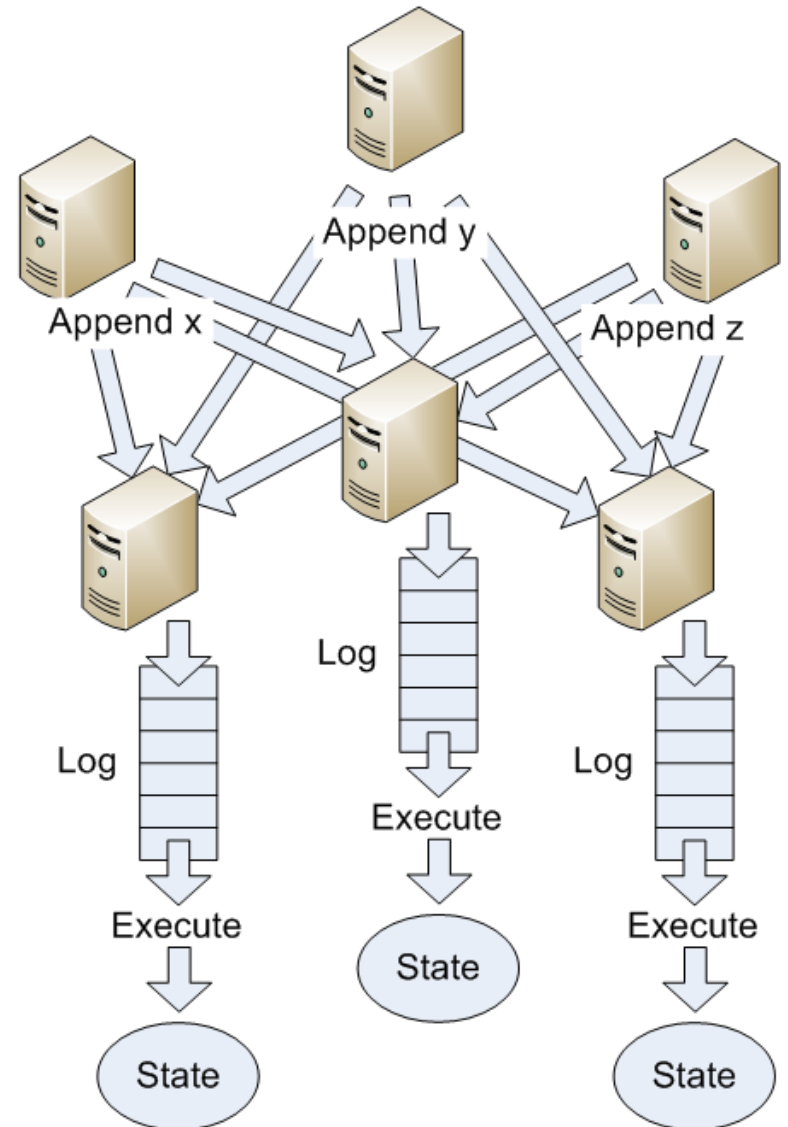
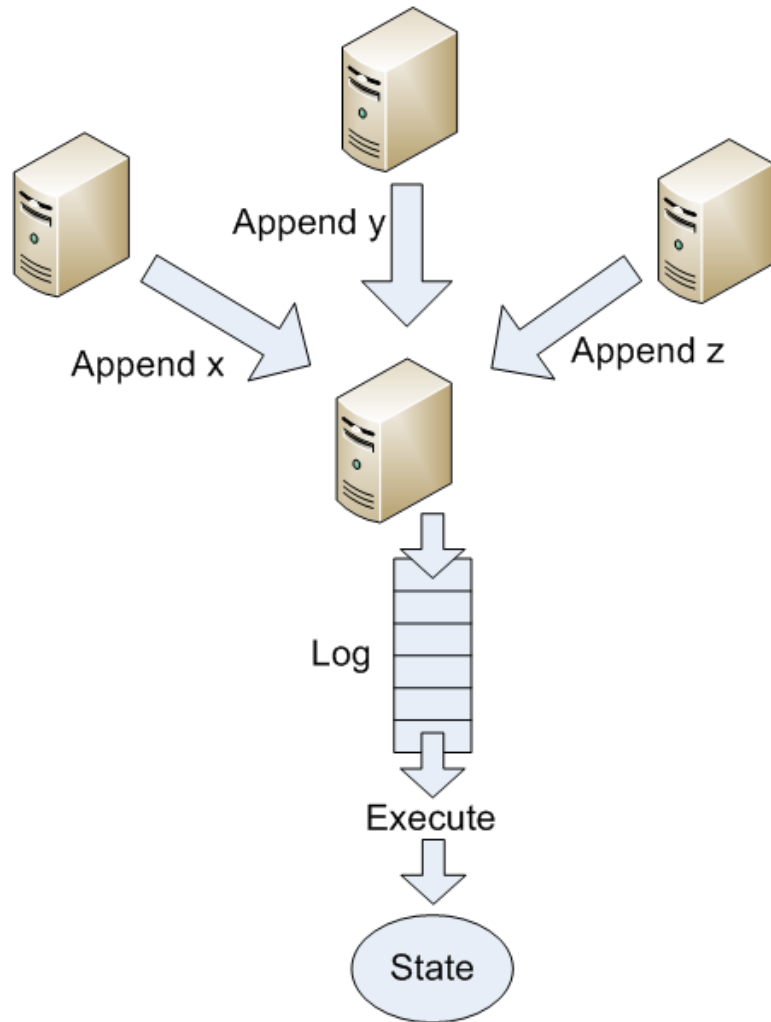
Coordination services

- API for
 - Storing and querying cluster state
 - Live machines, association to services, roles
 - Express interest in conditions, notifications
- High availability and data consistency
 - Replication
 - Order on state updates
- Google Chubby (Paxos), Apache ZooKeeper (ZAB)

Order on state updates



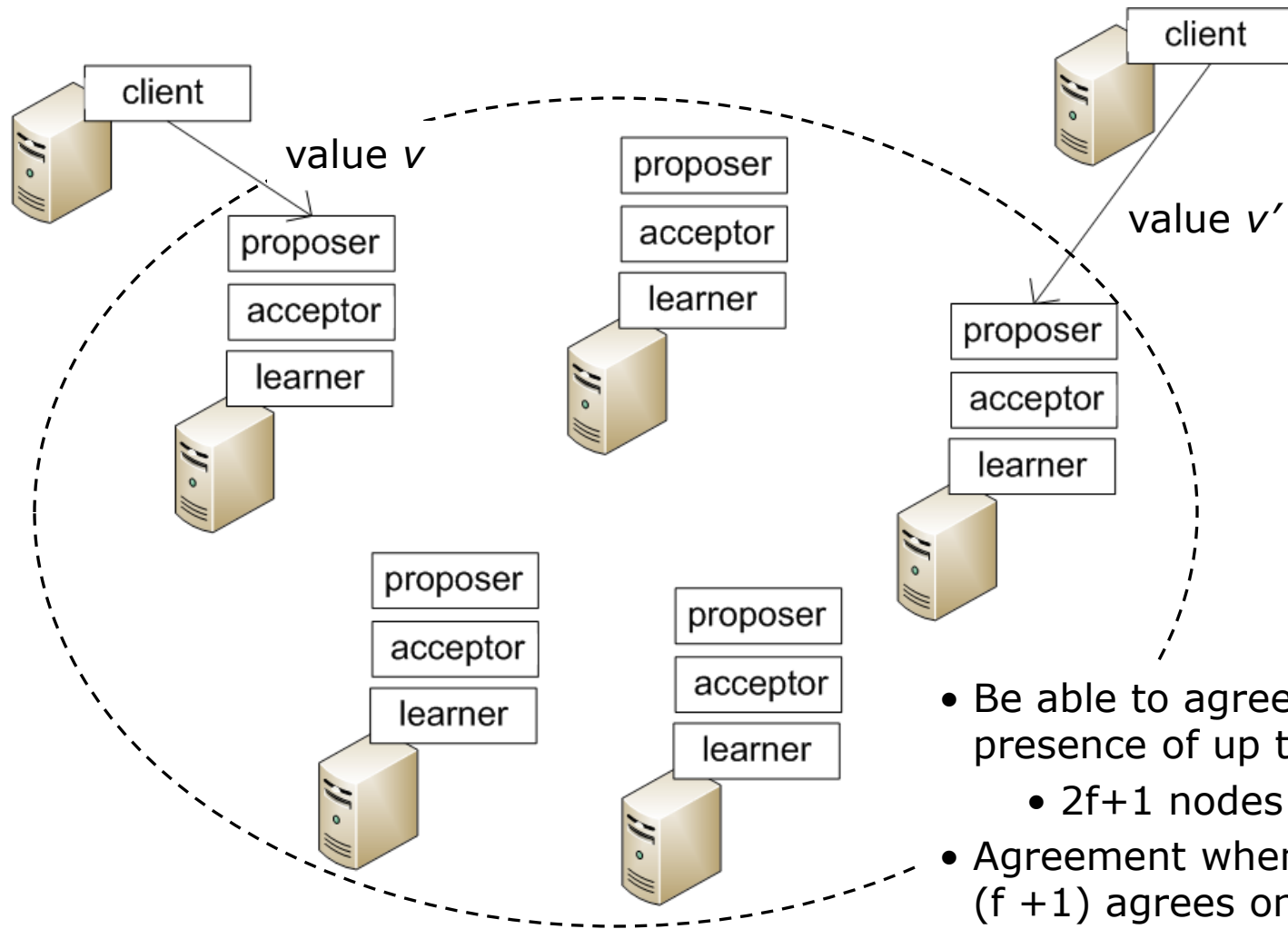
Order on state updates



Paxos algorithm

- Way to build fault-tolerant distributed systems
 - Replicated state machines (RSM)
- Consensus via message exchange
 - Asynchronous: no timing guarantees
 - Network can delay, reorder, lose (but not corrupt) packets
- Can guarantee safety
 - Replicas will agree on a single value
- Need additional assumptions to ensure progress

Paxos setup



Informally

- Simplest, but fault-intolerant solution: single acceptor
- With >1 acceptors, agreement by a majority required
- If single value proposed, that value should be chosen
 - Thus, an acceptor must accept the first value proposed to it
- However, this may lead to fragmented electorate
 - Multiple **ballots** by each proposer should be possible
 - Identify each ballot by a unique integer N

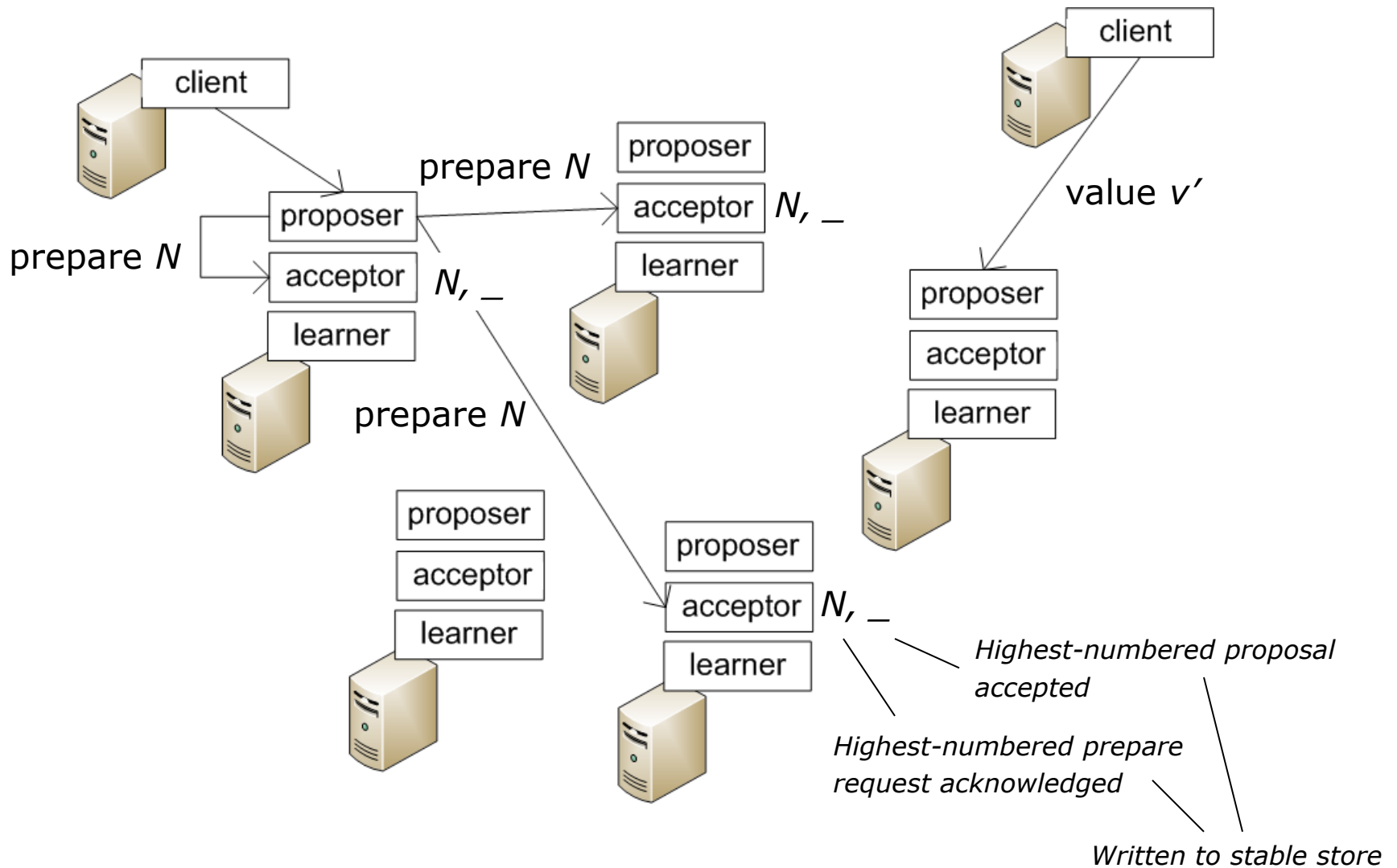
Informally

- After consensus, an acceptor cannot change its mind
 - A value is chosen when a single proposal with that value from ballot N is accepted by a majority of the acceptors
- Allow multiple proposals to be chosen, but guarantee that all chosen proposals have the same value
- If value v is chosen at ballot N , every higher numbered ballot started by any proposer should propose value v
 - However: need to learn what has been accepted so far
 - Two phases: Prepare, then propose

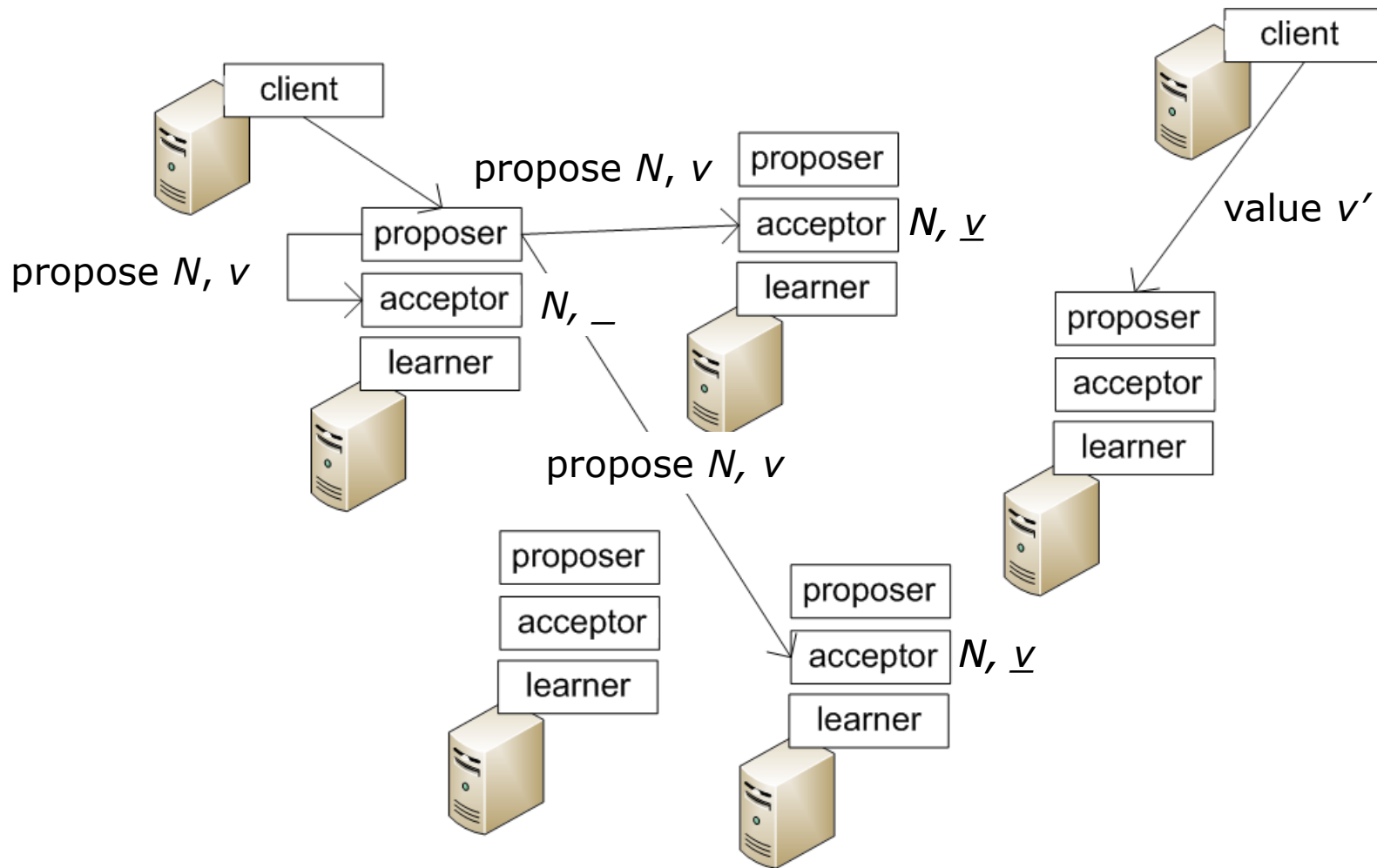
Informally

- A proposer wanting to issue a proposal at ballot N must learn the highest-numbered proposal $< N$ (if any) that has been or will be accepted by a majority
 - Easy to learn about values already accepted
 - Hard to predict the future
- Control the future: extract a promise that there will not be any acceptances of proposals at ballots $< N$

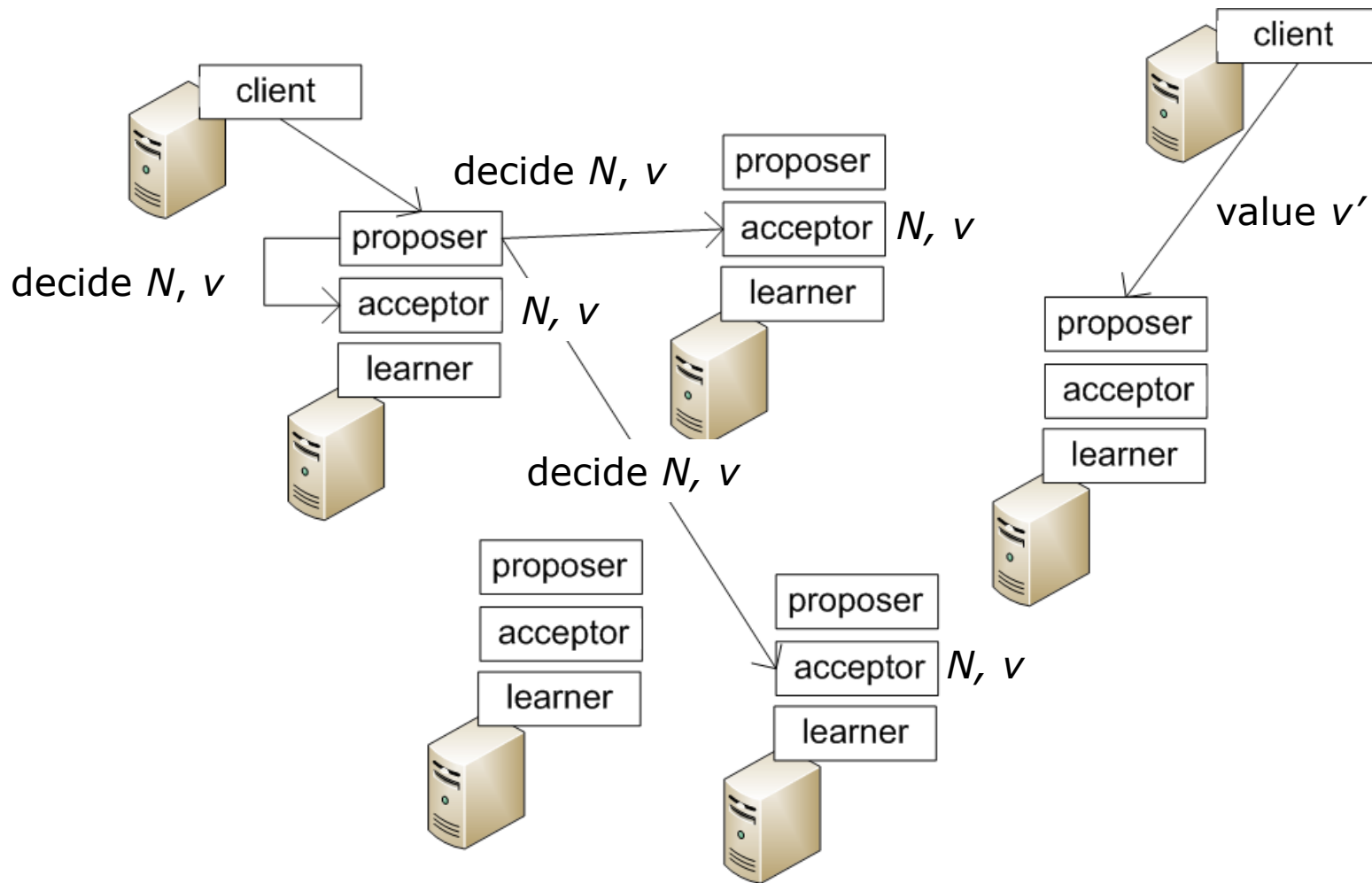
Paxos – phase 1



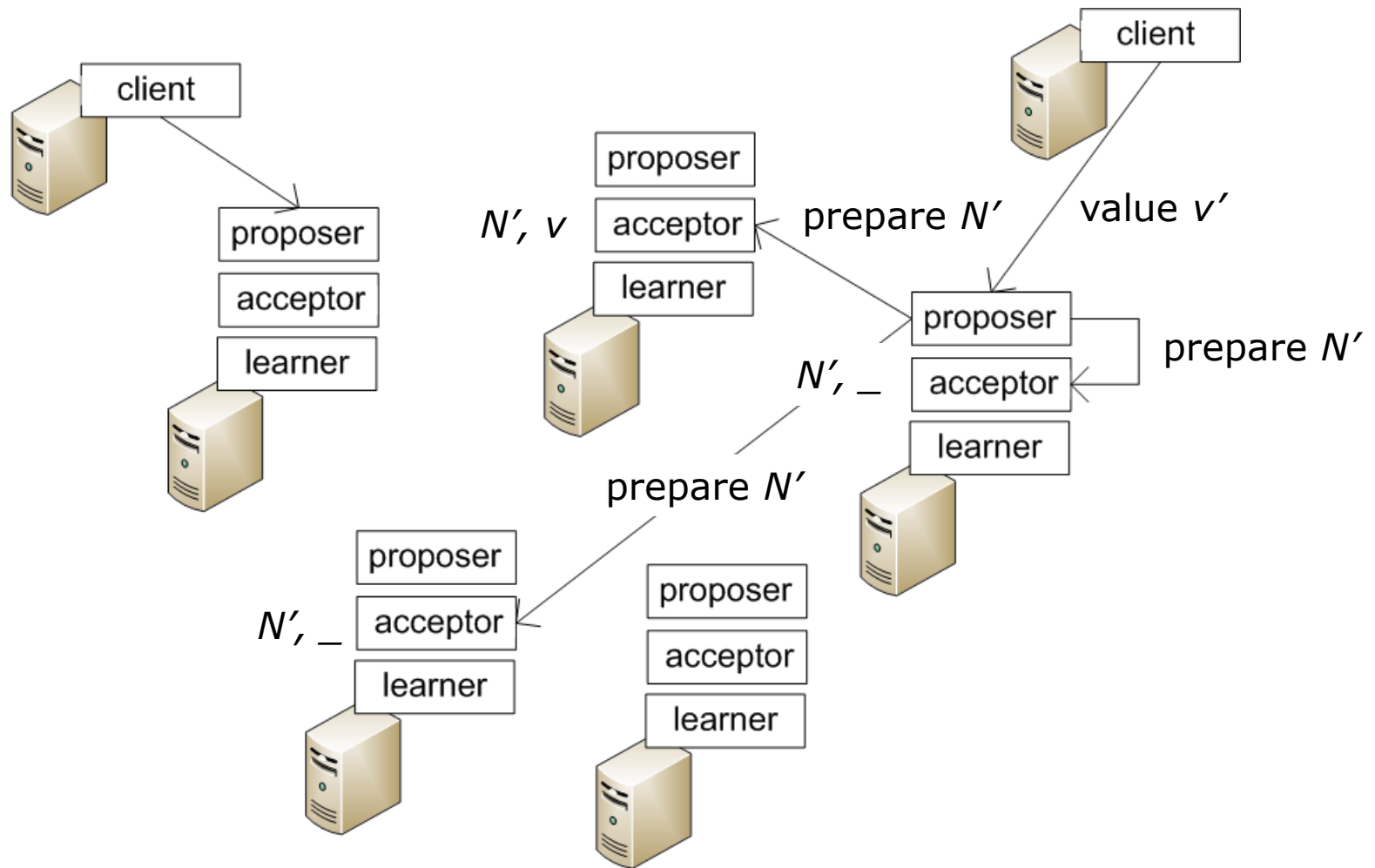
Paxos – phase 2



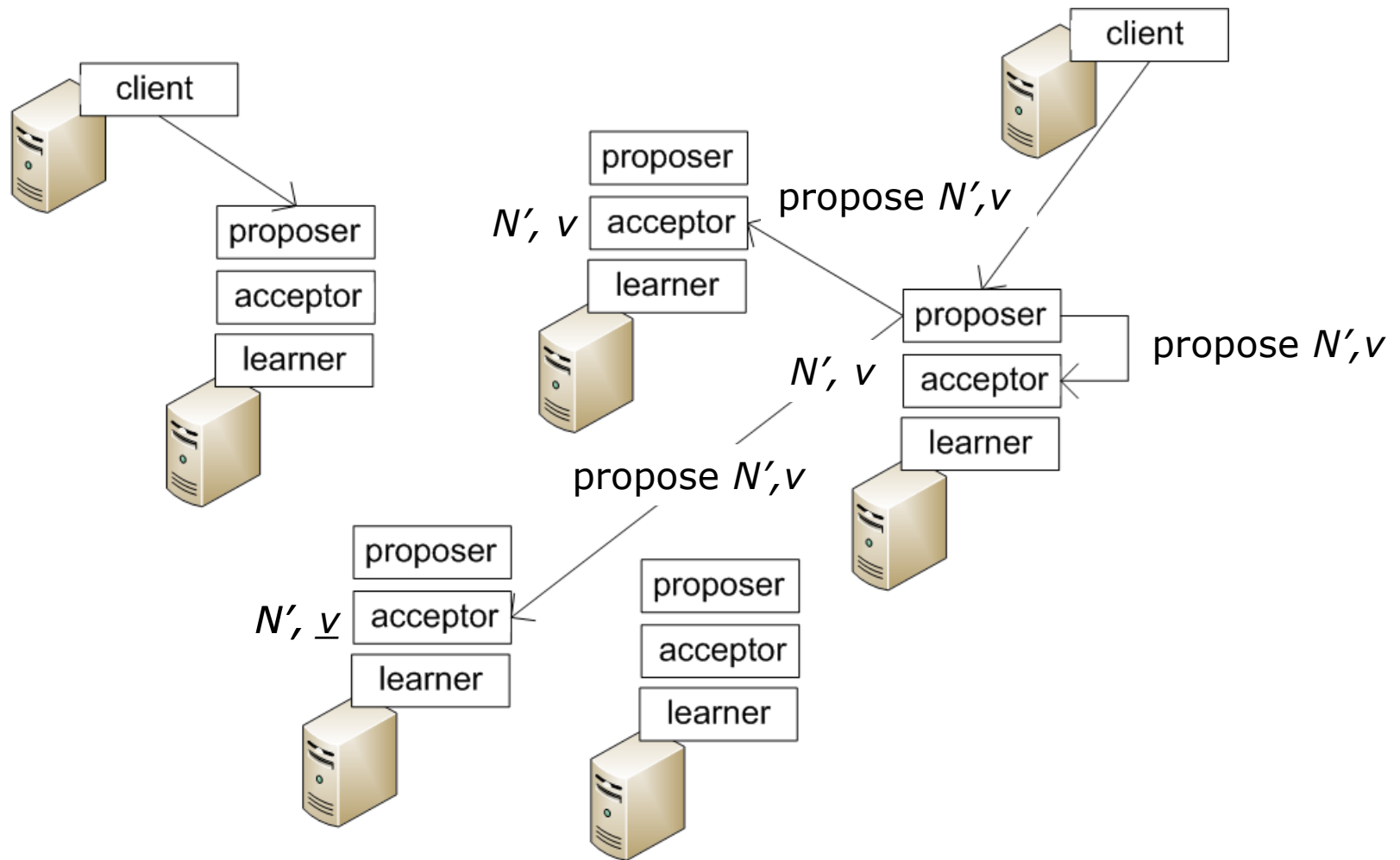
Paxos – communicate agreement



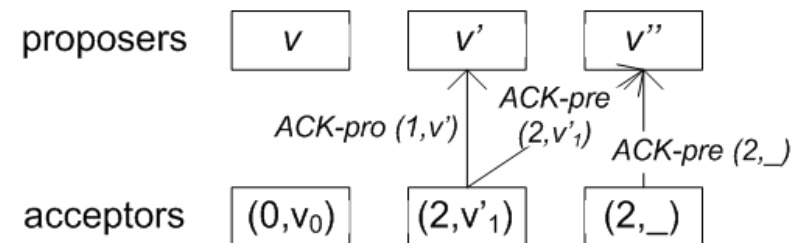
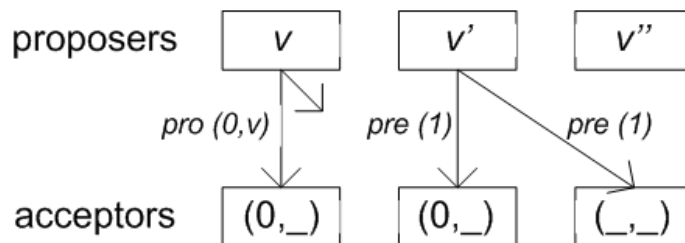
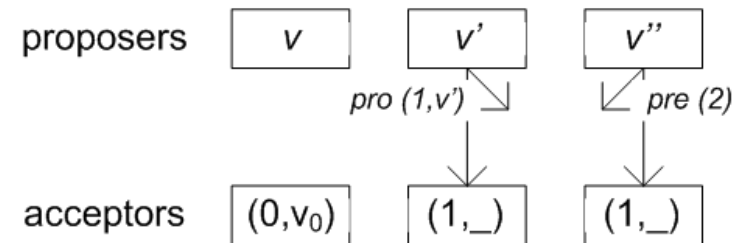
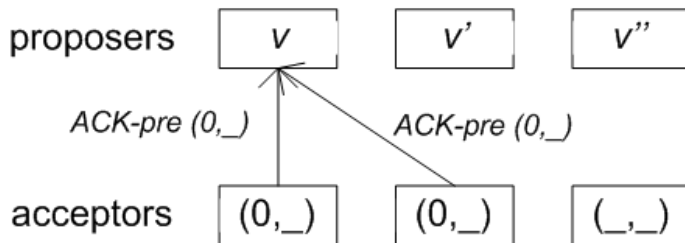
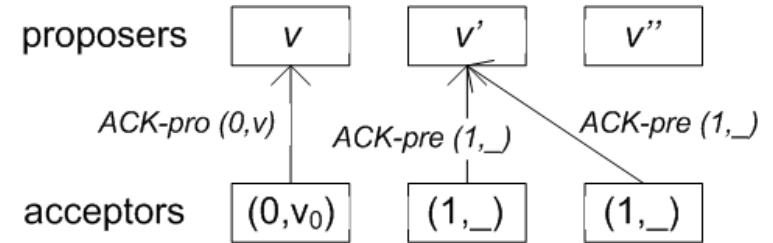
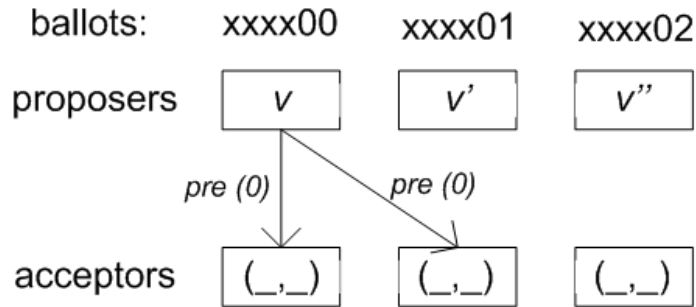
Paxos – learning chosen value



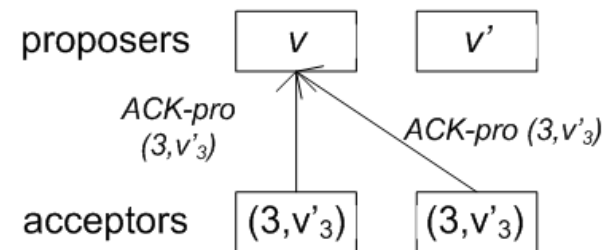
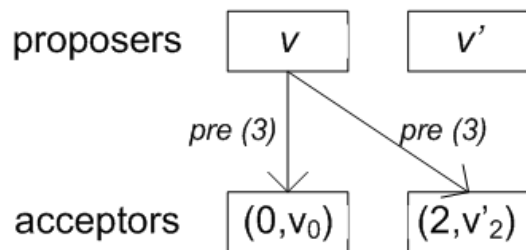
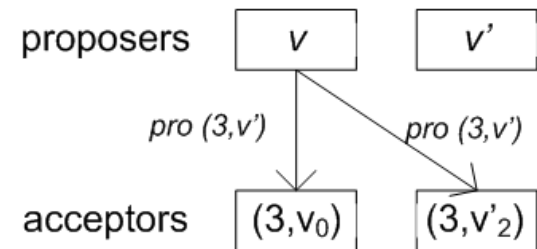
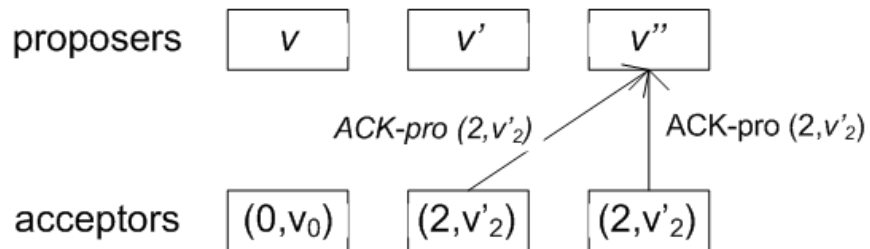
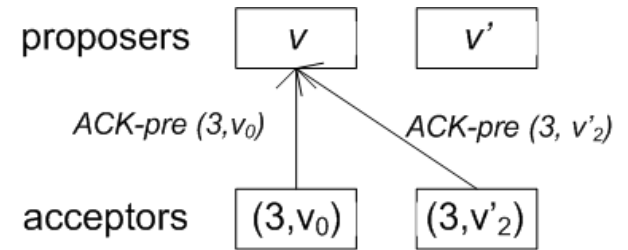
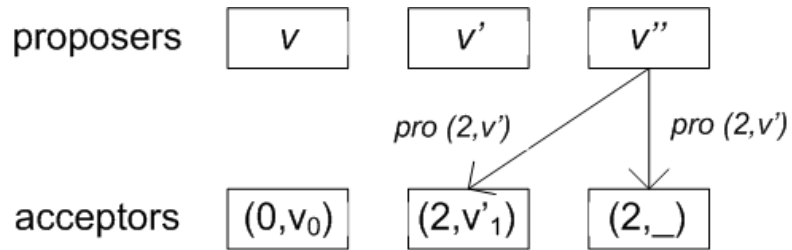
Paxos – propagate chosen value



Example



Example (*contd.*)

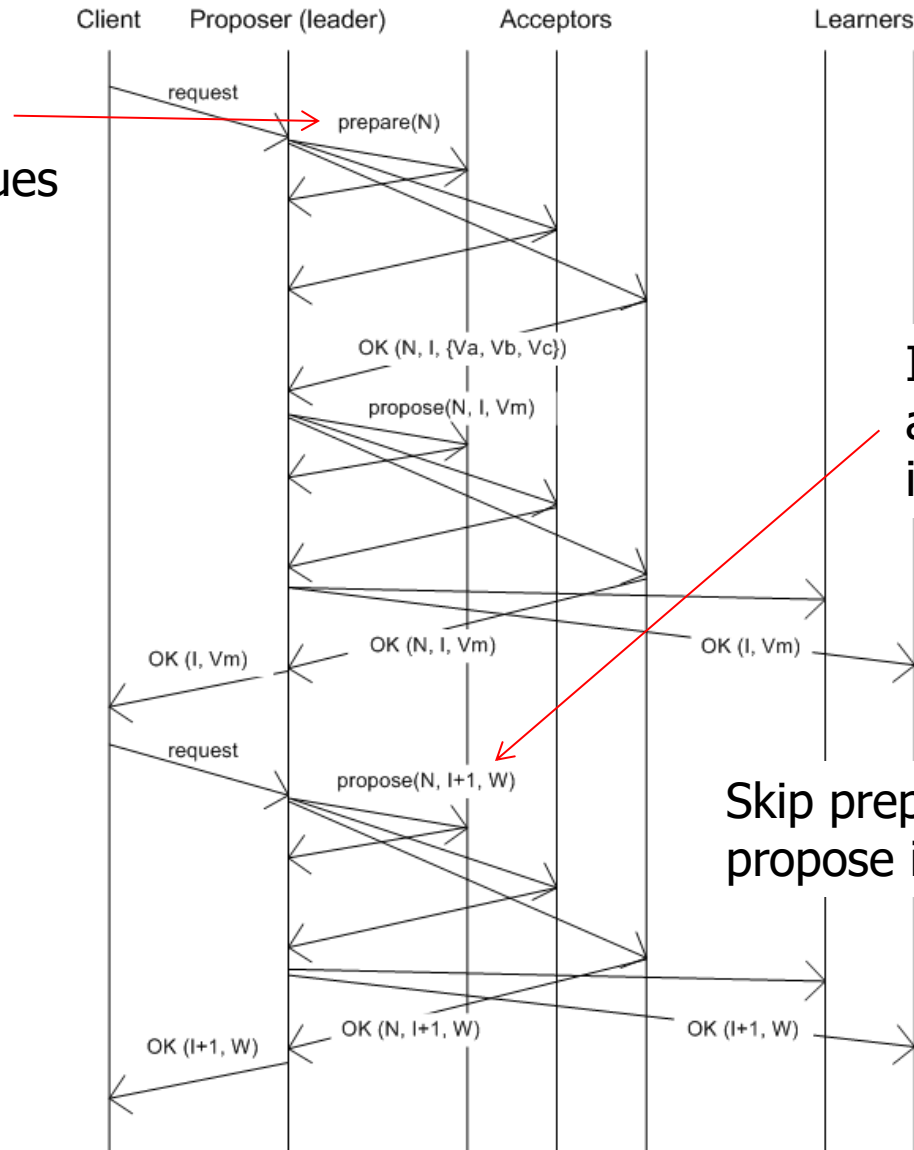


Lamport: implementing a state machine

- How to run multiple instances of Paxos
 - Assume the existence of a distinguished proposer (leader)
 - A leader will run Paxos for a number of instances
 - The leader may crash, at which point there may be gaps in the chosen instances (1-134, 138, ..)
 - A new leader will try to fill in those slots or propose *no-op*
 - As soon as gap fills, commands can be executed
- Multi-Paxos
 - New leader: execute phase 1 for infinitely many instances
 - Acceptors can respond with reasonably short messages
 - Cost of Paxos effectively the cost of executing phase 2

Multi-Paxos

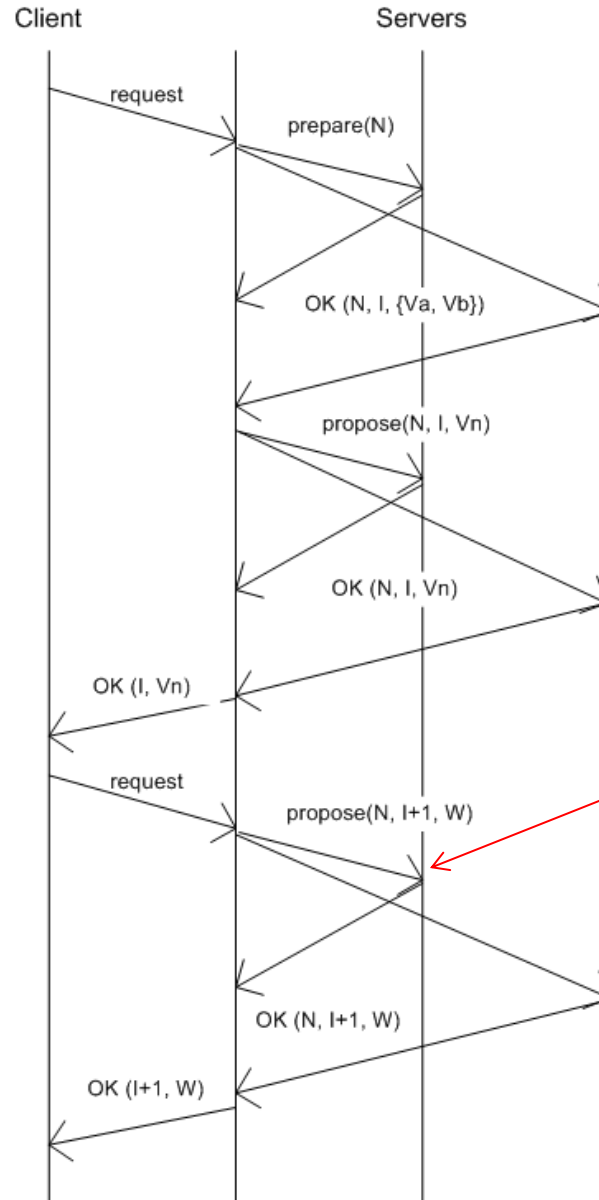
- New leader @N
- Learn accepted values for past instances



If a majority has not accepted anything for instances $> I$

Skip prepare phase until a propose is rejected!

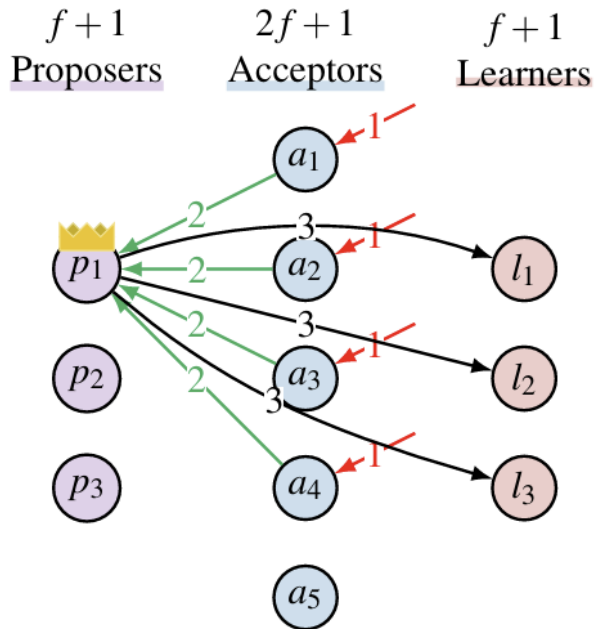
Multi-Paxos



Servers play all roles

Replicas write to disk prior to sending ACK

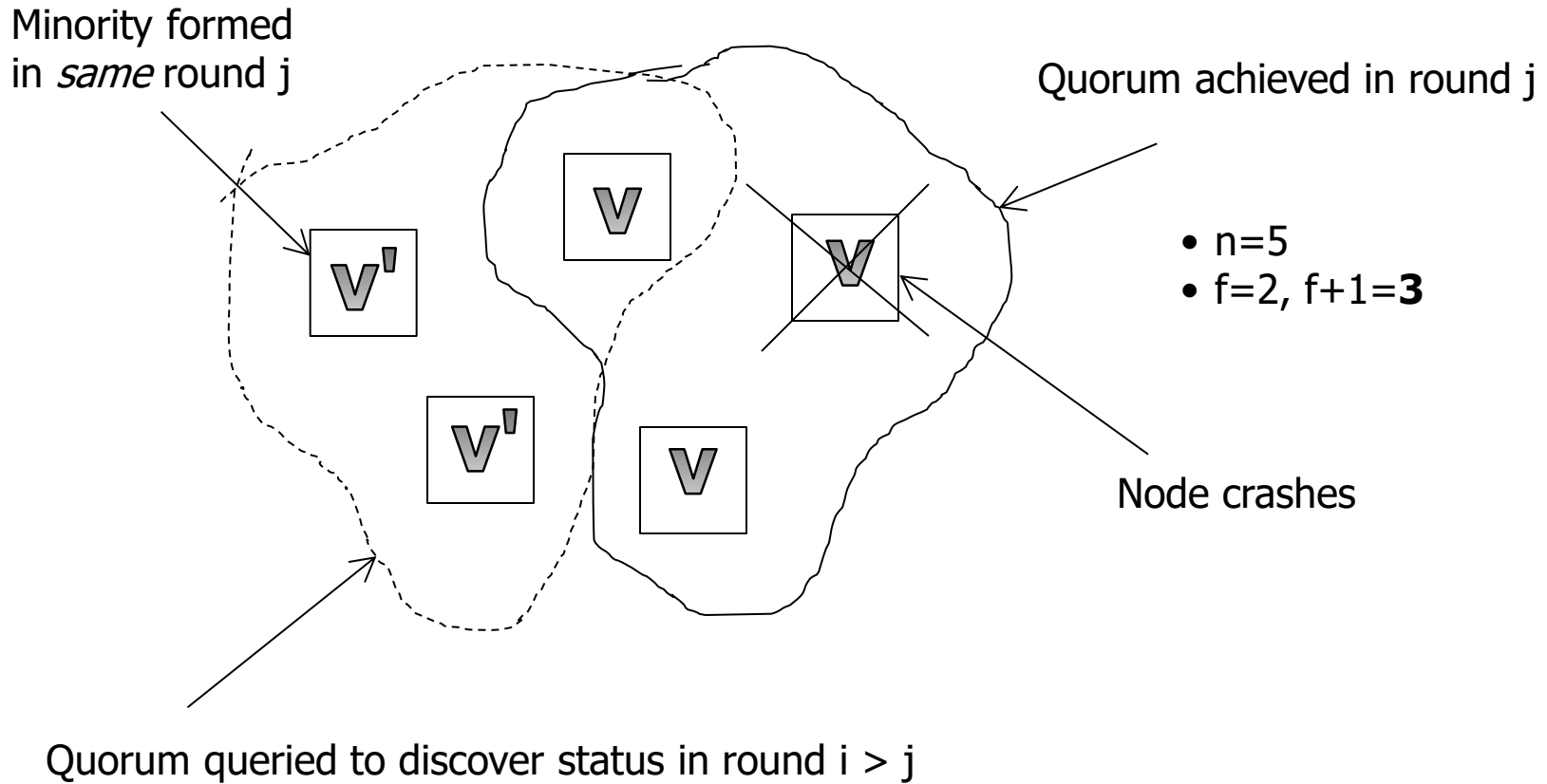
Fast Paxos



(a) Normal case execution.

- Clients directly propose to acceptors in a round
- Acceptors lock on first vote during that round
- Lead proposer counts votes
- Leader of round 0 (fast round) shown with a crown
- **Is it still sufficient to achieve a majority? ($f+1$)**

Fast Paxos – is an $f+1$ quorum enough?



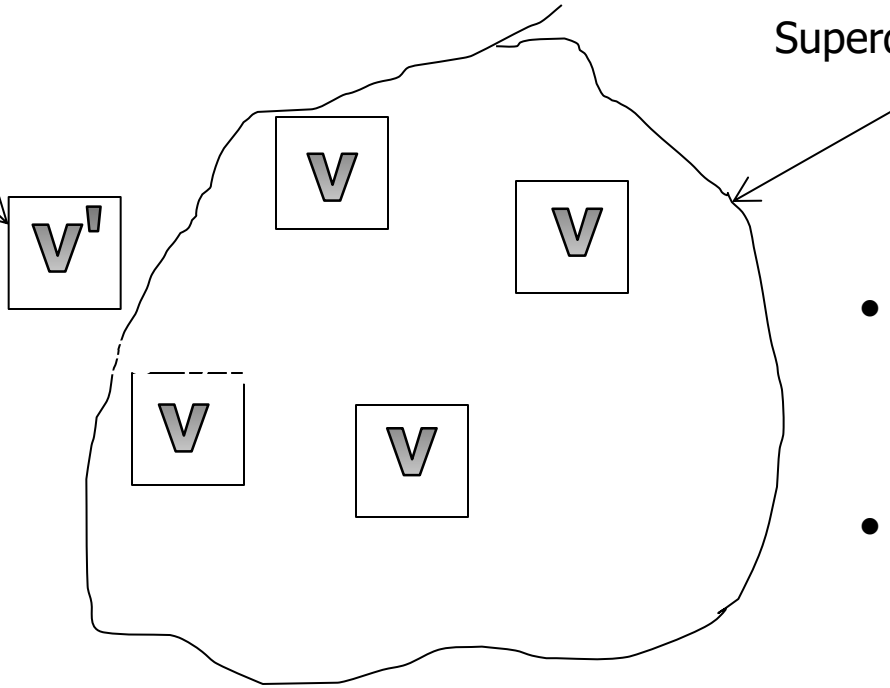
Fast Paxos – Need to use *supermajority*

Minority formed
in round j

Superquorum **Q1** achieved in round j

$$f + \lceil \frac{f+1}{2} \rceil$$

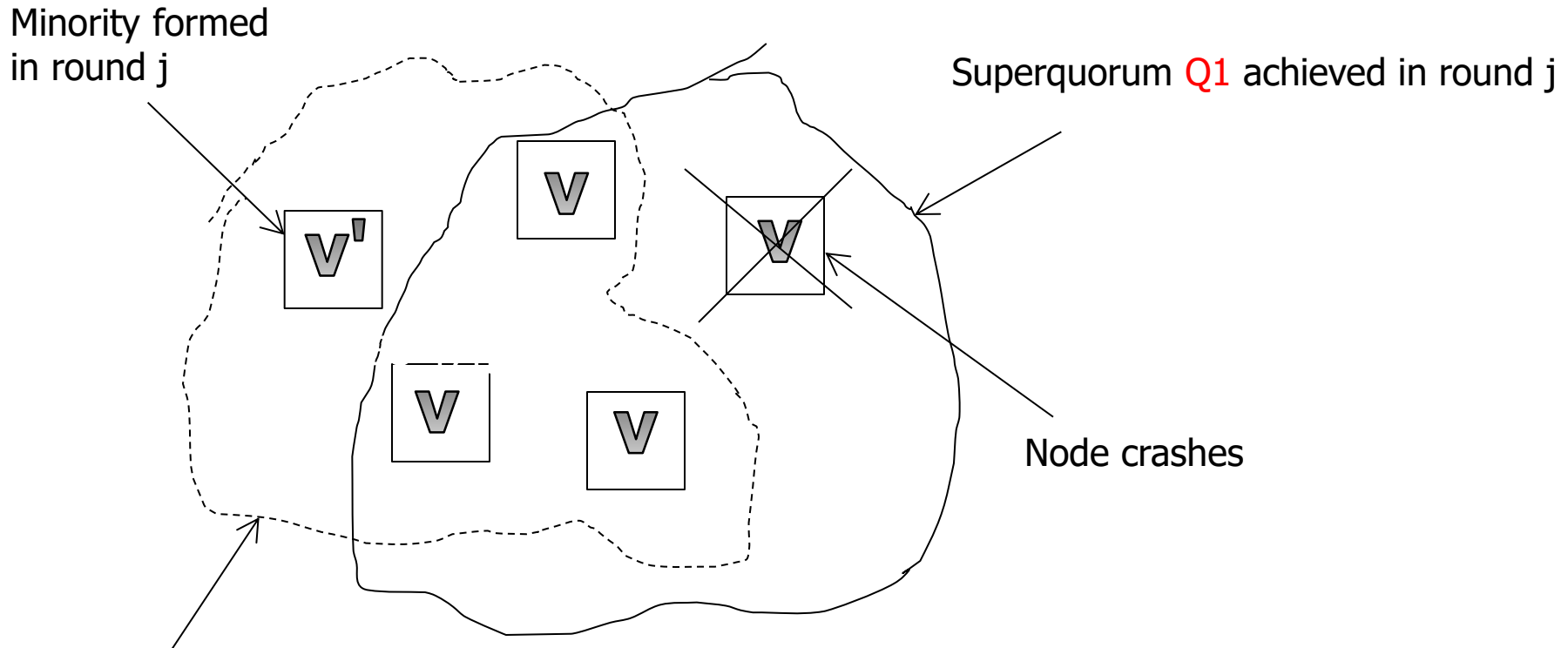
- $f + \text{maj}(f+1)$
- For $n=2f+1=5$
- $\text{SQ} = 4$



- Any single failure leaves at least a majority ($f+1$) of nodes behind
- Two supermajorities intersect on $f+1$ nodes

f	Majority: $f+1$	$\text{Ceil}(f+1)/2$	SQ: $\text{Ceil}(3f/2)+1$	$2f+1$
1	2	1	3	3
2	3	2	4	5
3	4	2	6	7
4	5	3	7	9

Fast Paxos – how to recover a decided round

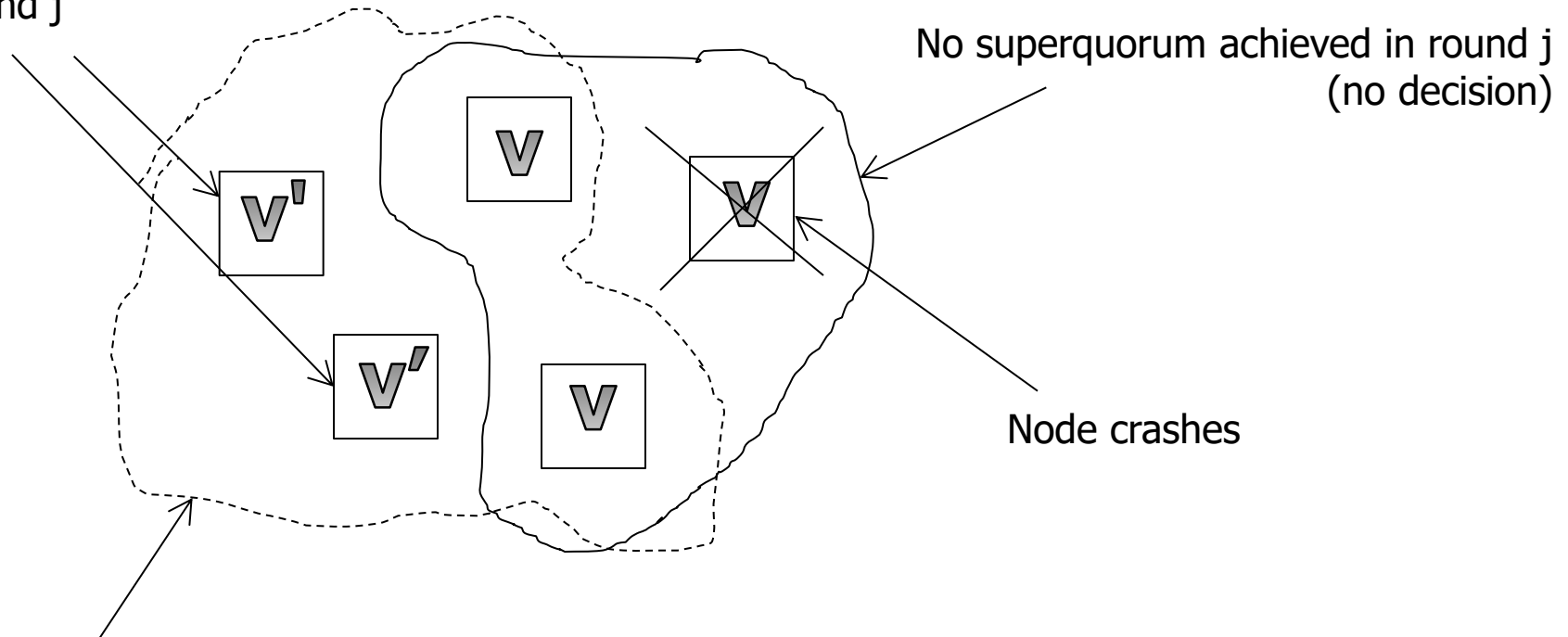


Superquorum **Q2** queried to discover status in round i

- **Q2** guaranteed to overlap with **Q1** on ≥ 3 nodes
- Just one such subset (if *anything* decided, it is v!)

Fast Paxos – how to recover an undecided round

Another minority formed
in round j



Superquorum **Q2** queried to discover status in round i

- Tie (2 votes for v' , 2 votes for v) tells proposer there was no superquorum in previous round, allows it to go with either v or v'