NETWORK → exist of REPLICA

REPLICA → (true) if pending commands

↓

NODE → parent link + proposed command

→ lockedQC → ⊥, highestQC for which a replica voted COMMIT

→ prepareQC → ⊥, highest QC for which a replica voted PRE-COMMIT

for 1, (2) → curView (new phase) —

NEWLEADER → update at every phase

- PREPARE
- PRECOMMIT
- COMMIT
- EXECUTE
- INTERRUPT

→ automatically stamped with curView

MESSAGE → type

→ m.node → proposed node

→ m.justify → carry the QC for different phases

+ partial sig

Quorum certificate (over <type, viewNumber, node>)

↓

(combine collection of signatures for the same tuple signed by (n-f) replicas)

PREPARE

LEADER

→ wait for (n-f) NEW-VIEW messages (curView-1) => set M

→ highQC → highest viewNumber out of m

→ curProposal → CREATELEAF (highQC.node, cmd)

↓

LEADER's proposal => broadcast it with PREPARE type message

REPLICAS

→ wait for PREPARE message

→ if okay => send back ⌐ PREPARE to leader
VOTEMSG

PRE-COMMIT

LEADER

→ wait for (n-f) PREPARE → V set

→ prepareQC → QC(V)

→ broadcast PRE-COMMIT

REPLICA

→ wait for PREPARE message from leader

→ prepareQC = m.justify

→ send back ⌐ PRE-COMMIT
VOTEMSG

COMMIT

LEADER

→ wait for (n-f) PRE-COMMIT VOTES

→ precommitQC → QC(V)

→ broadcast COMMIT

REPLICA

→ wait for PRE-COMMIT message

→ lockedQC = m.justify

→ send back (COMMIT VOTEMSG

DECIDE

LEADER

→ wait for (n-f) COMMIT VOTES

→ commitQC → QC(V)

→ broadcast DECIDE

REPLICA

→ wait for COMMIT message

→ execute new commands (m.justify.node)

→ respond to clients

NEXTVIEW

↳ send if wait for > time threshold

send NEW-VIEW to leader with (curView+1)