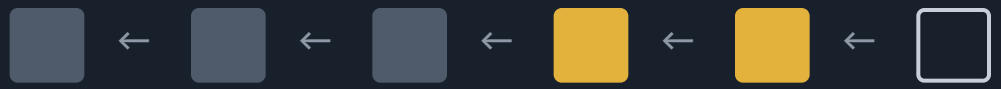
FAST FINALITY · DESIGN DISCUSSION

Goldilocks

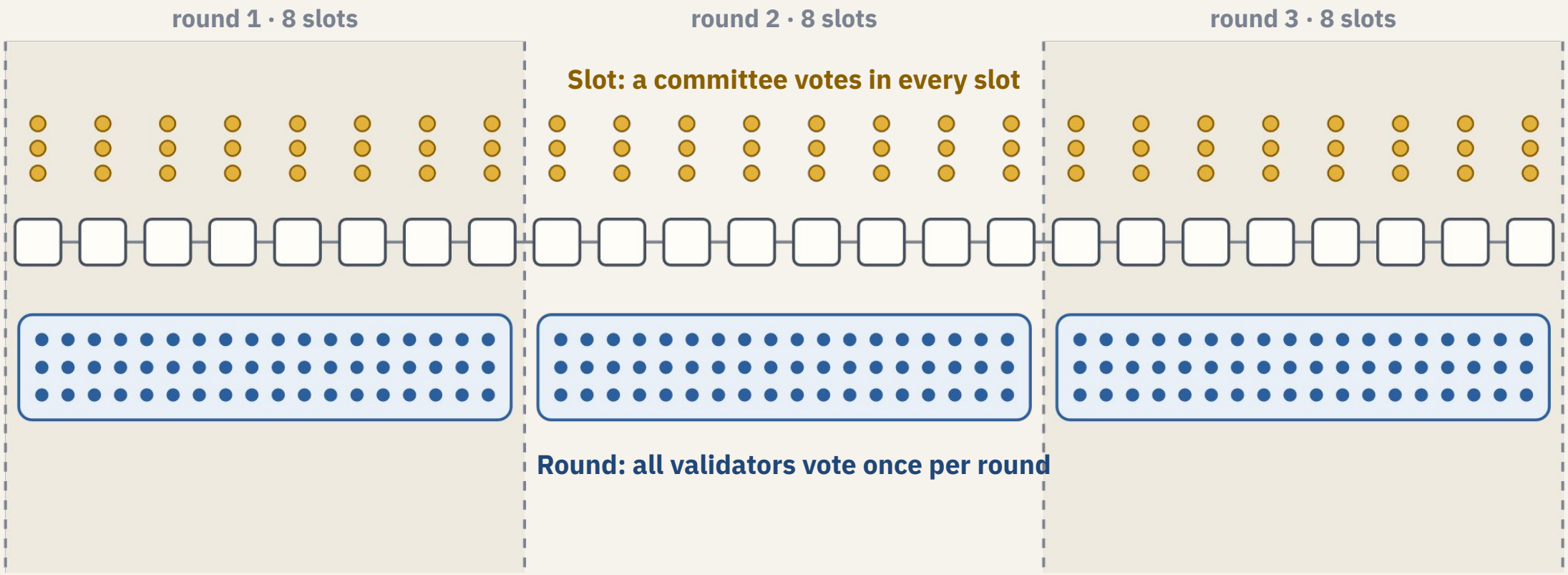
Goldfish with locks, composed with a finality gadget.



finalized (FG) · confirmed and locked (AC) · head

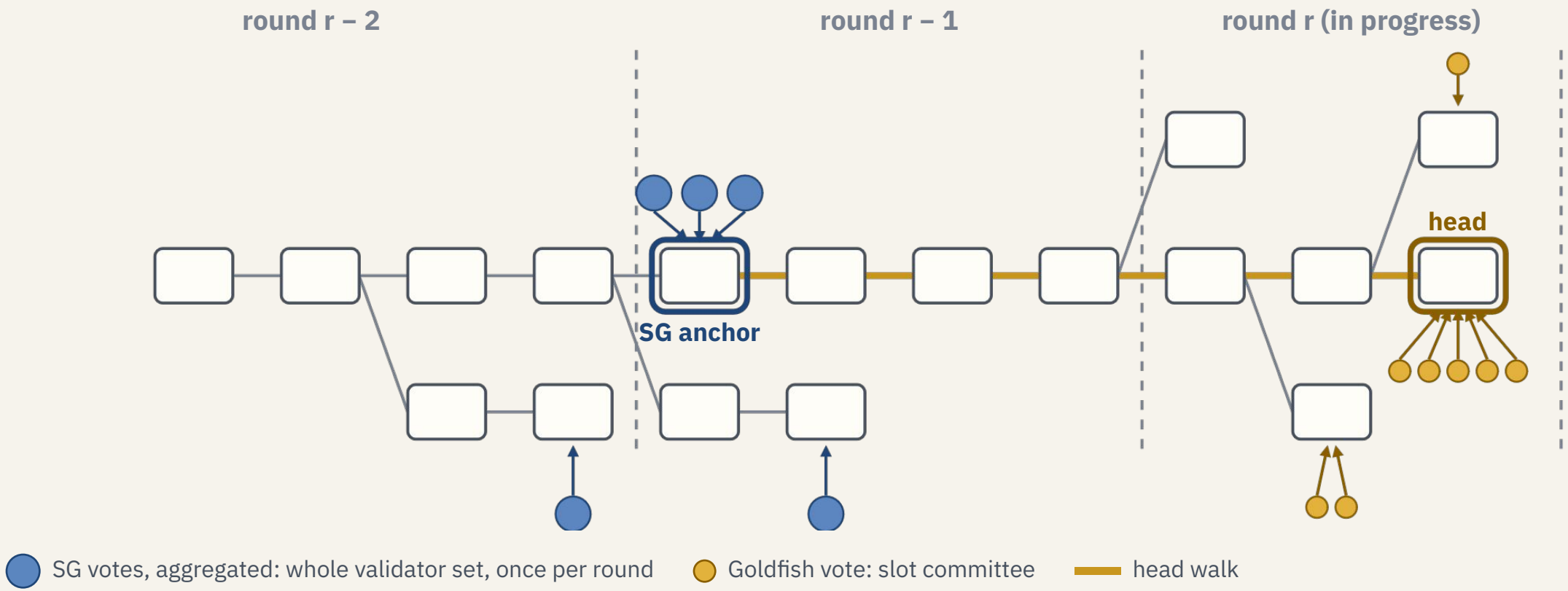
THE COMMON STRUCTURE

Decoupled consensus: two independent voting processes

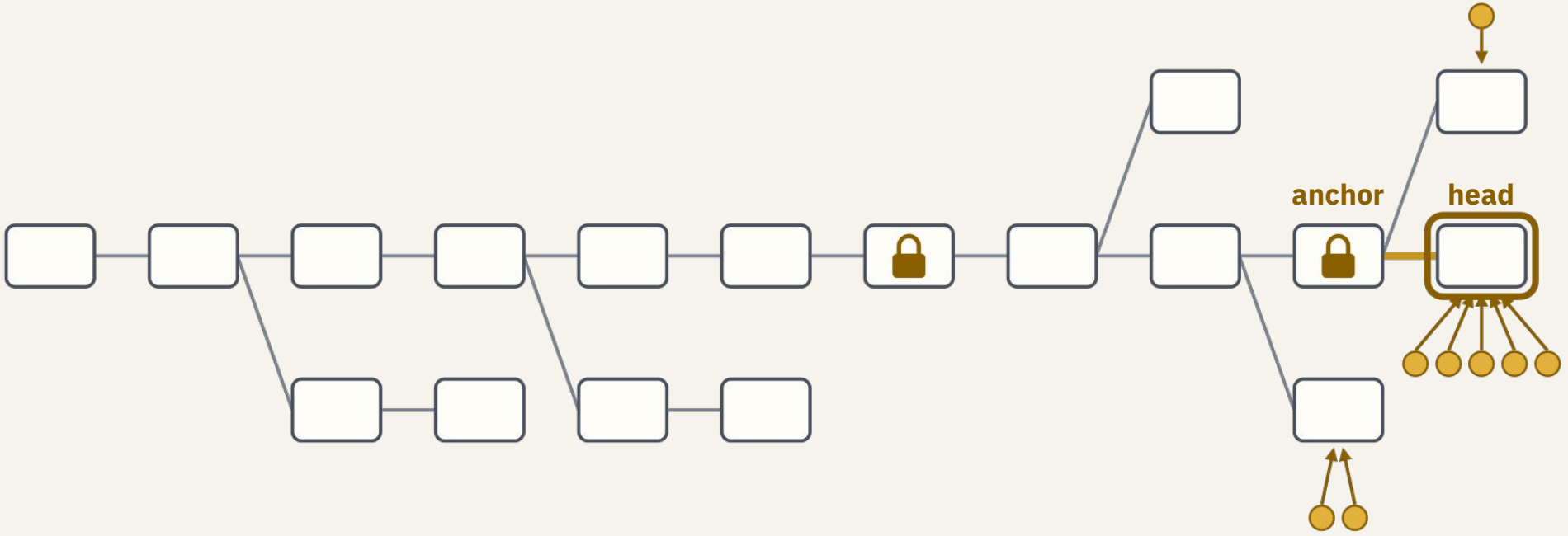


AVAILABLE CHAIN · DC WITH SG

SG + Goldfish



Goldfish + locks

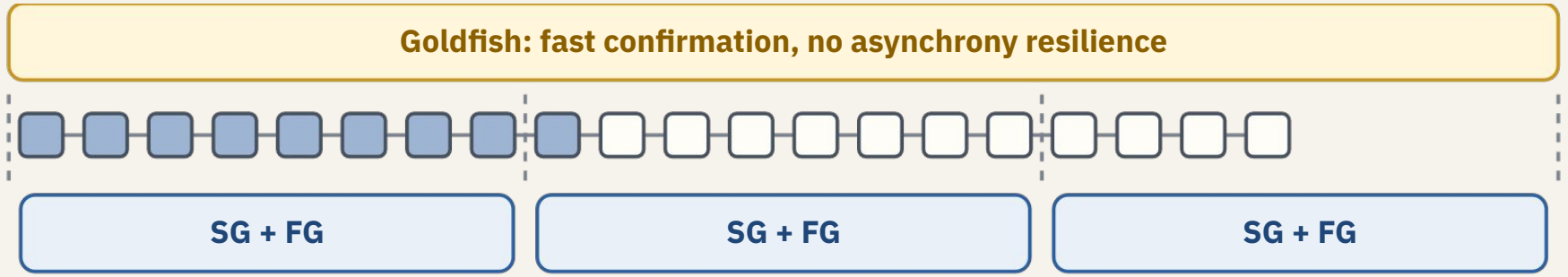


live lock Goldfish vote: slot committee head walk

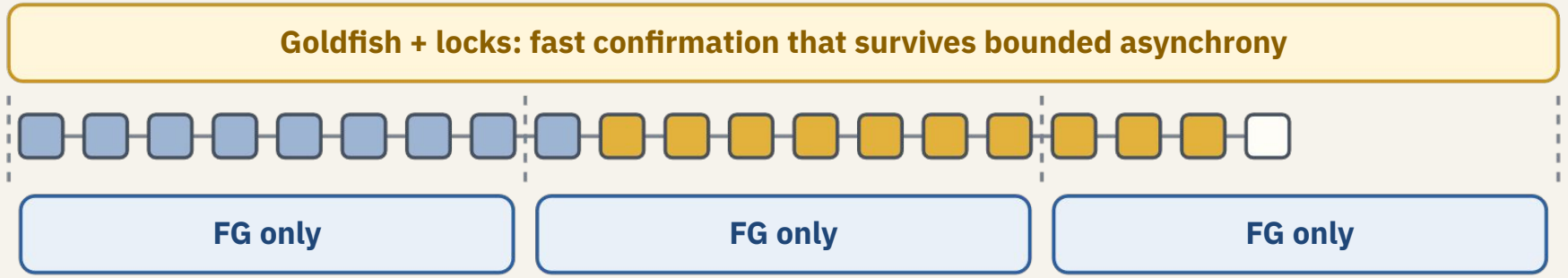
ASYNCHRONY RESILIENCE WITH SG AND WITH LOCKS

How/when do we get stabilization?

DC with SG: 1 round



Goldilocks: 1 slot



Stabilization moves from the round layer to the committee layer. The round layer keeps only one job: finality.

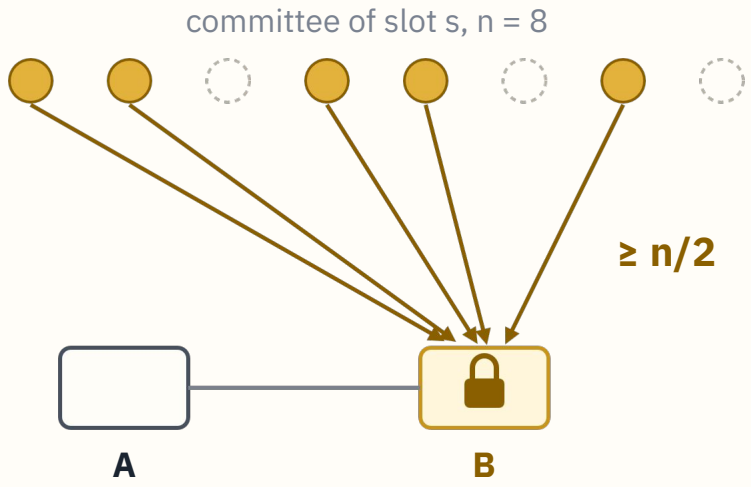
SG VS GOLDILOCKS

Why Goldilocks

	DC with stabilization gadget	Goldilocks
Latency to asynchrony resilience	1 round	1 slot (full participation), k slots (low participation)
Attestations	FG + SG, one per round	FG only, one per height: no progress => no new attestations
Fork choice inputs	blocks, committee votes and attestations: must process old attestations	blocks and current committee votes only: network attestations are safe to drop
Healing cost	High complexity and extra latency in each round (grades mechanism for SG)	Low: two simple unlocking triggers (expiry or majority against the lock)
PQ available chain	needs PQ attestations (full validator set)	needs PQ committees only
Low participation	guarantees from the whole validator set	guarantees from committees only

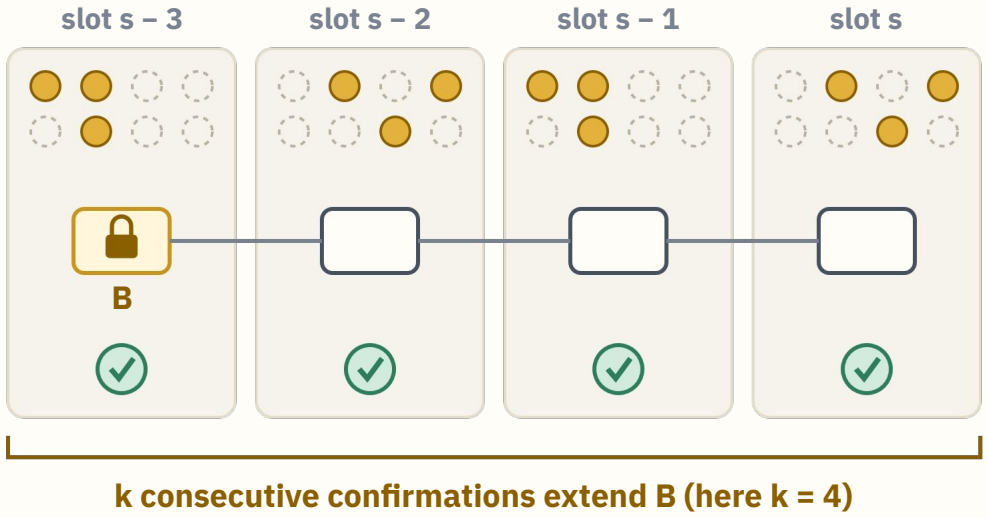
Two ways to lock

Fast: half of one committee



The confirmation update of slot s sees at least n/2 of the whole committee on B.

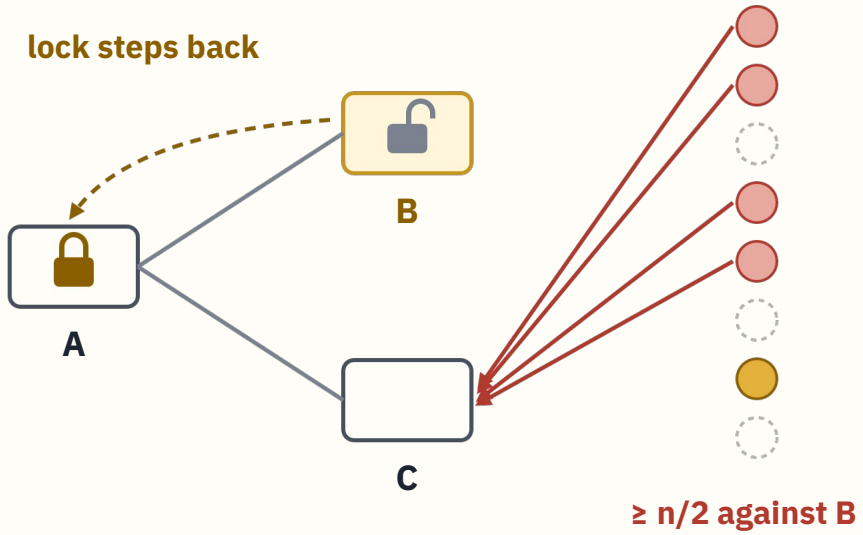
Slow: k confirmations in a row



Few voters are awake, so no slot has n/2 on B. But every slot confirms a chain through B. Lock B.

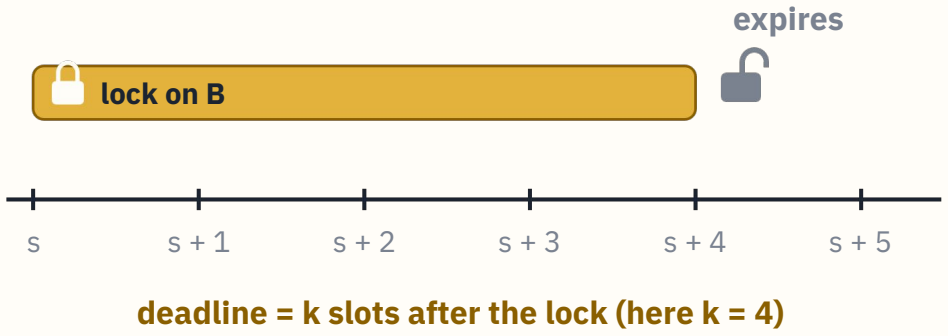
Two ways to unlock

Fast: half of one committee against



Before each vote, a lock moves back to the deepest ancestor with fewer than n/2 opponents.

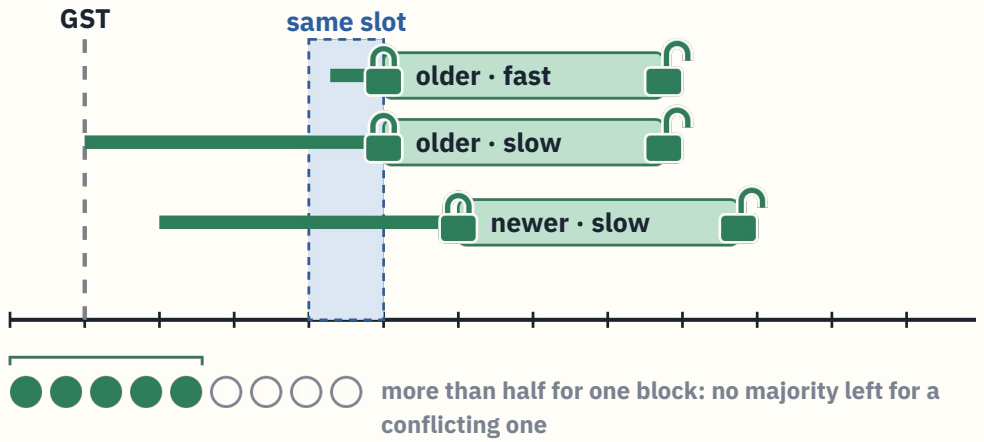
Slow: expiry in k slots



All locks expire after k slots if not renewed

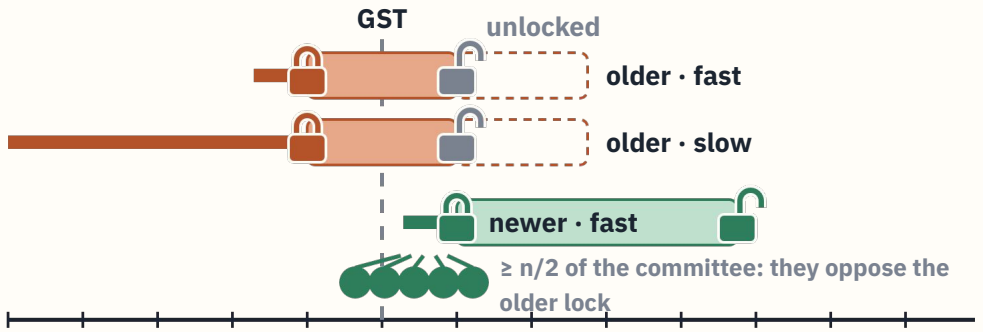
No conflicting live locks after GST

The newer lock is slow



The formation windows of the two locks must overlap. With the time-shifted quorum (TSQ), honest validators cannot confirm two conflicting blocks in the same slot after GST.

The newer lock is fast

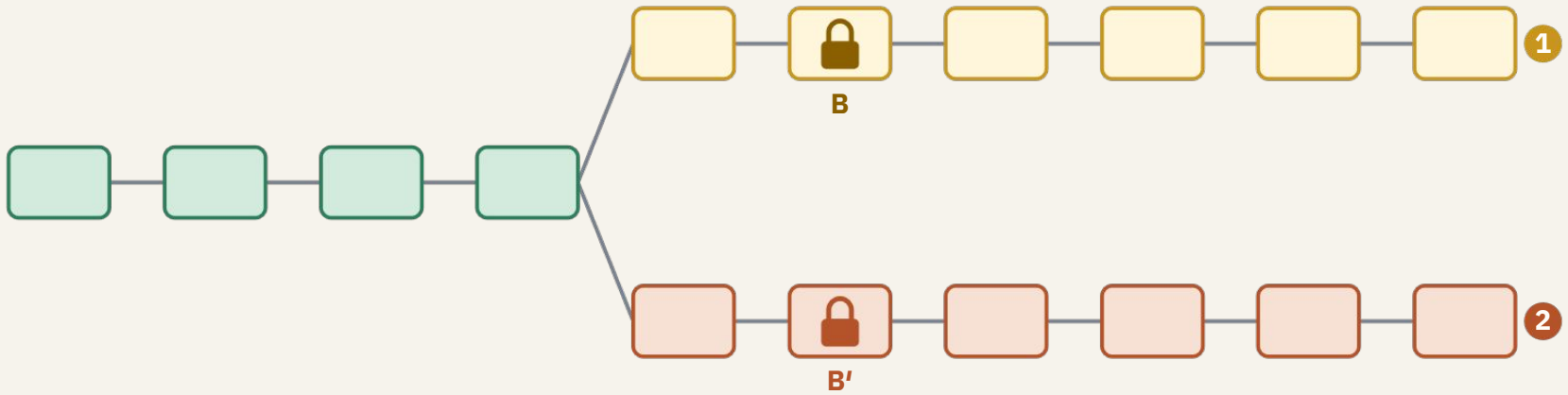


At least n/2 of the committee voted for it. They oppose any conflicting lock, which is unlocked, even one from before GST.



Conflicting locks can form

before GST | GST → GST + L | GST + L → GST + 2L | after GST + 2L

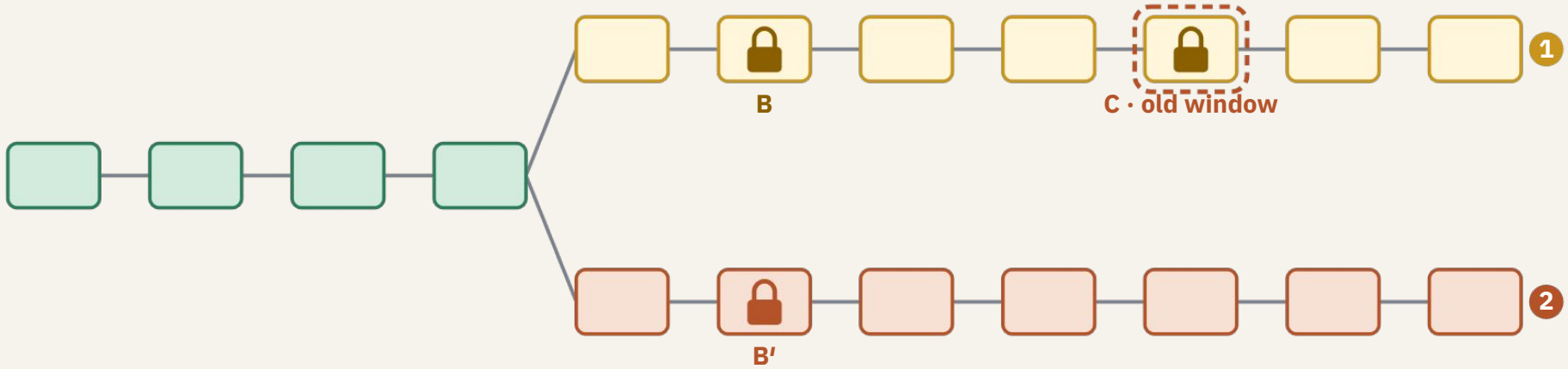


Before GST, nodes saw different votes. Group 1 locked B, group 2 locked B′, and each group walks from its own lock. Green: both groups follow it.

GOLDILOCKS · HEALING

No new conflicting locks, but old ones still alive

before GST | **GST → GST + L** | GST + L → GST + 2L | after GST + 2L

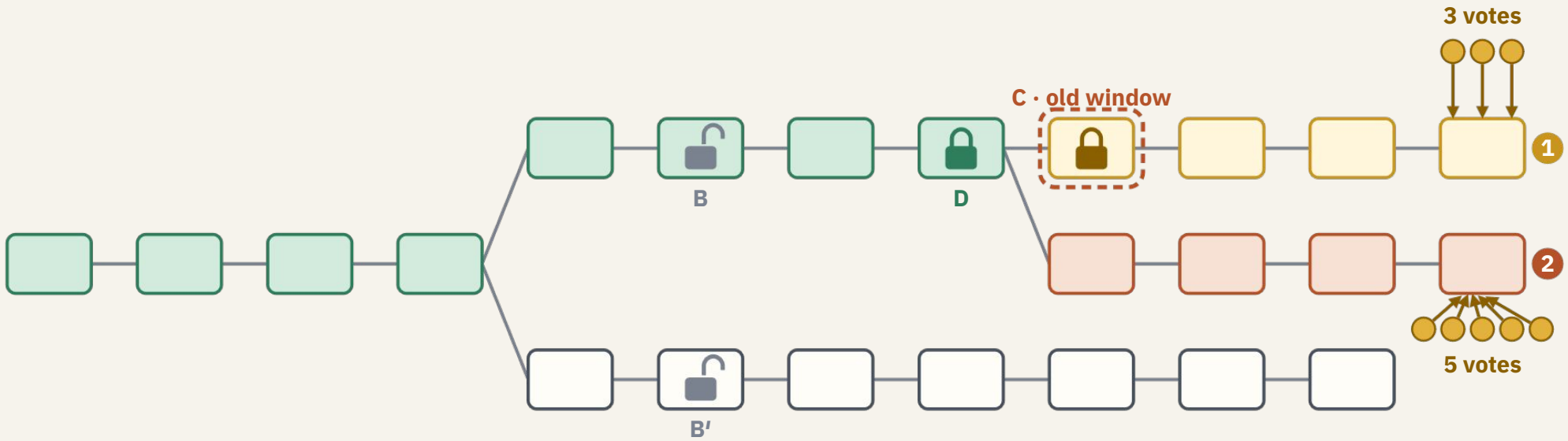


After GST, new locks never stay in conflict (slide 9). But B and B′ are still live. Group 1 also locks C, partly from confirmations before GST.

GOLDILOCKS · HEALING

Compatible locks, but not every lock is followed by votes

before GST | GST → GST + L | **GST + L → GST + 2L** | after GST + 2L

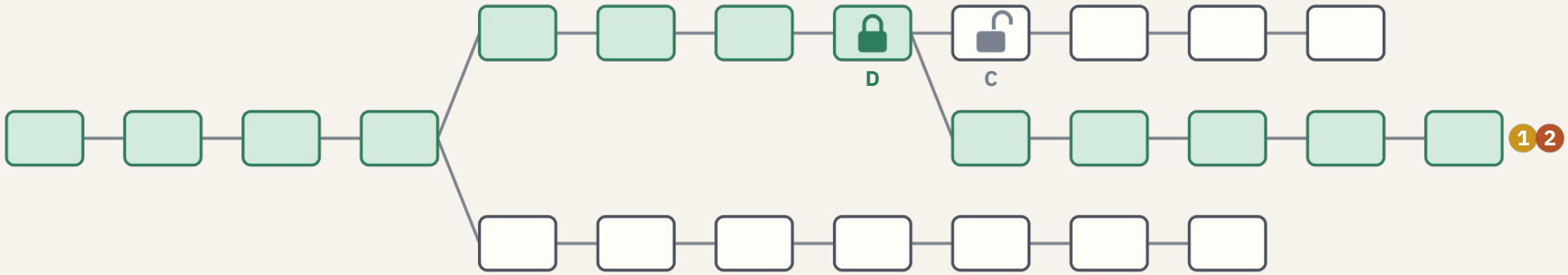


B and B′ have expired: no live locks conflict. D, from a synchronous window, is followed by everyone. More Goldfish votes are on the lower branch, so group 2 follows them past D; group 1 stays on C, held by its lock until it expires.

GOLDILOCKS · HEALING

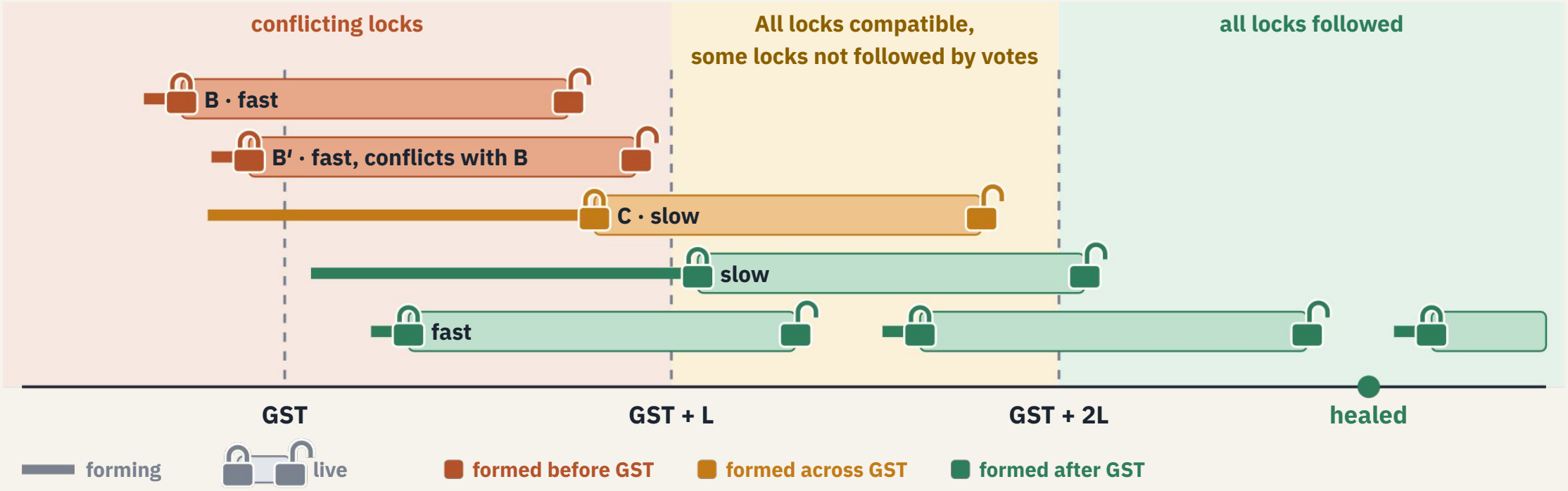
Every live lock is followed by everyone

before GST | GST → GST + L | GST + L → GST + 2L | **after GST + 2L**



C has expired, so group 1 is free to follow the votes and joins group 2. Every live lock is fully from after GST, so every honest head extends it. The next honest proposal is confirmed by all

Bad locks drain away within 2L



1 No new conflicting locks after GST

2 Locks formed fully after GST are followed by everyone

3 Conflicting locks expire by GST + L, unfollowed ones by GST + 2L ⇒ healed

FULL ARCHITECTURE WITH FINALITY GADGET

Finality Gadget <> Available Chain

INTERFACE

Finality Gadget <> Available Chain

Finality gadget (FG)

block tree and the chain state σ of each block

F, J, h_max, attestation pool

accepts blocks · attests

FG -> AC: fork-choice constraints

`(block tree, fc root, viable heads)`

AC -> FG: attestation targets

`(fg_confirmed, head)`

Available chain (AC)

Goldfish votes, locks, saved confirmations

no block store of its own

runs fork-choice for proposals, voting, confirming

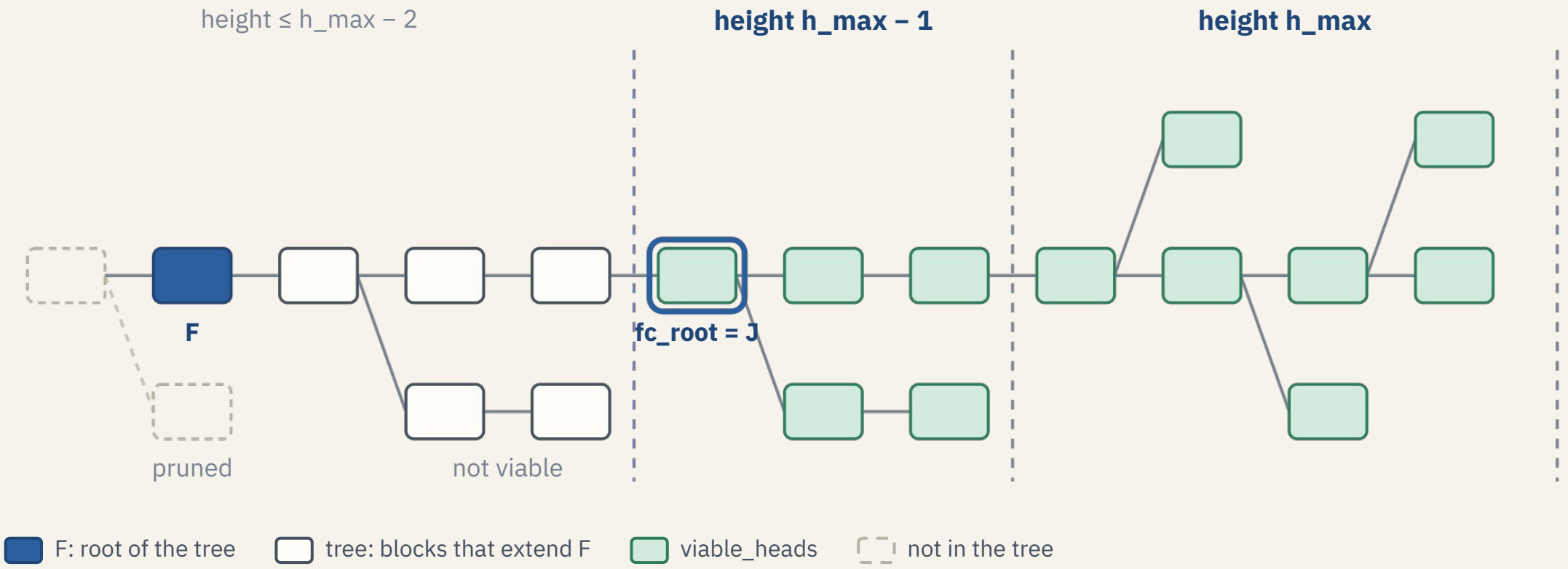


finalized · from the FG

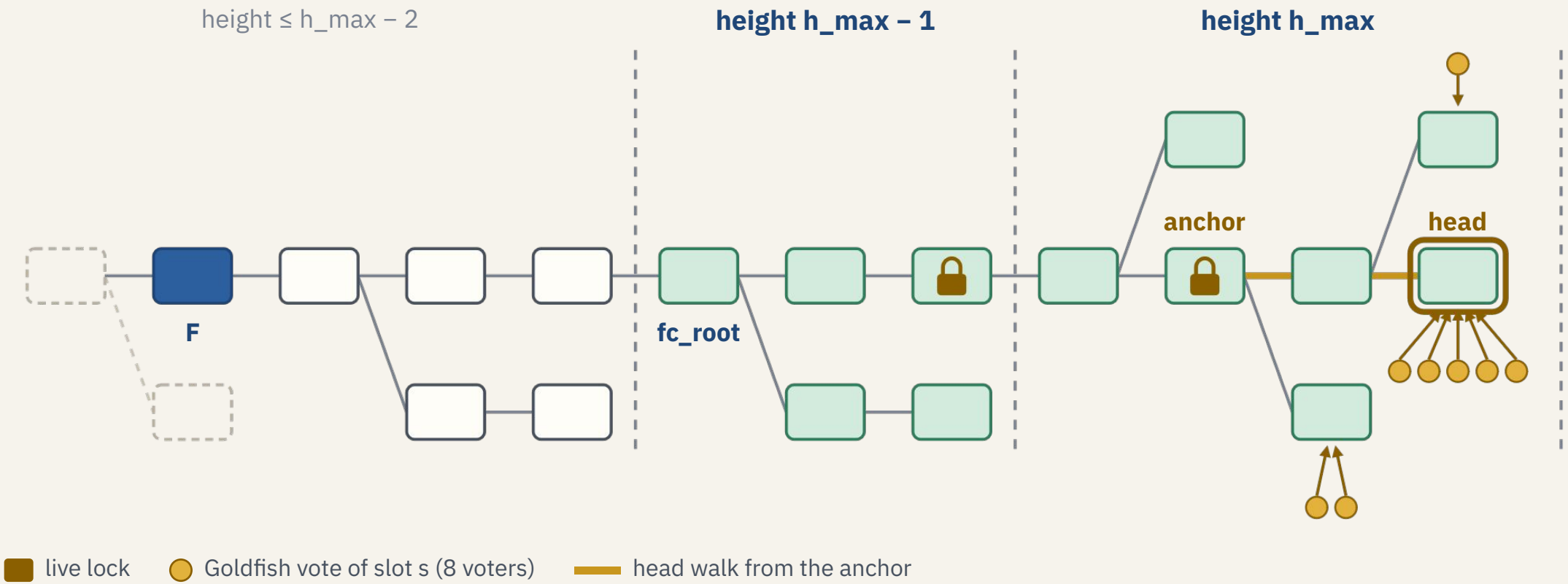
confirmed · locked

head · from the AC

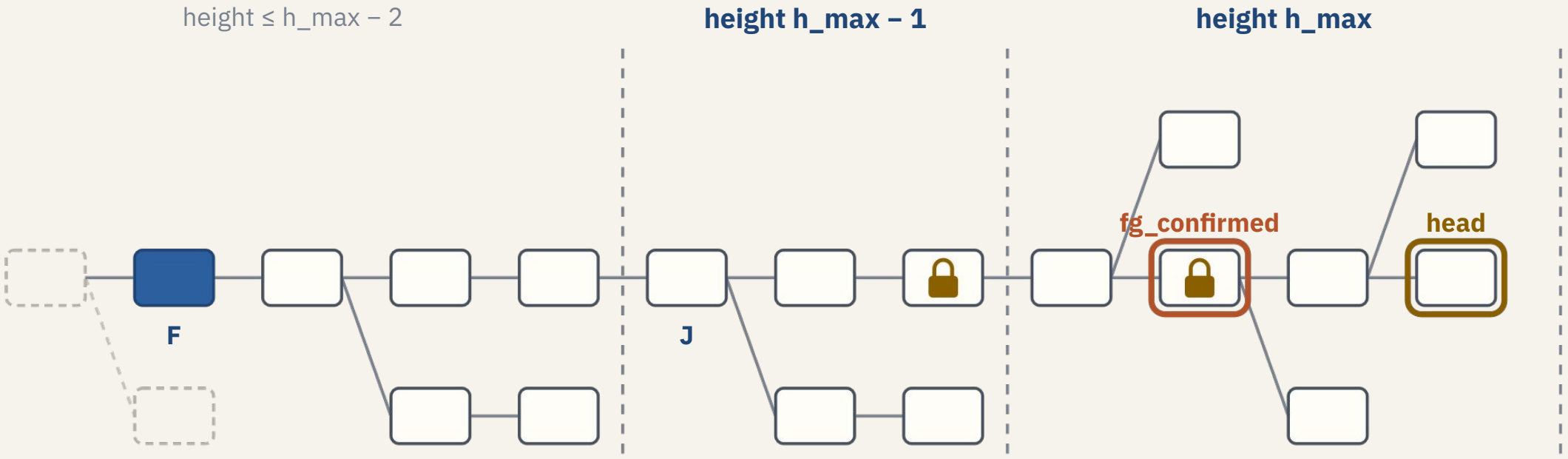
Tree rooted at F, fork-choice root, viable heads



Locks give the anchor, votes give the walk



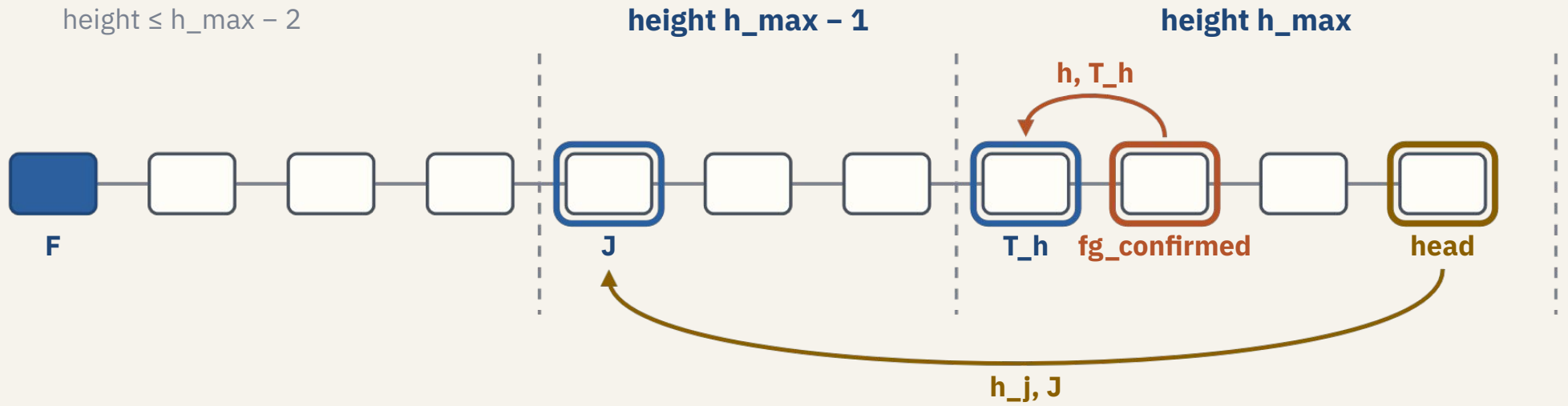
Two blocks: fg_confirmed and head



fg_confirmed: the deepest block that the latest confirmation update fast locked. Half of the committee or more voted for it.

head: the output of the fork choice that we just ran.

What the FG reads from fg_confirmed and head



Height pair, from the state of fg_confirmed

`(h, T_h, timeout = true/false)`

T_h is the first block of the height. Whether we vote for timeout or not depends on if we previously voted for a conflicting target

Finality pair, from the state of head

`(h_j, J)`

The latest justification on the head chain. Vote to finalize it.

THE FG SIDE

Attestations are pure finality votes

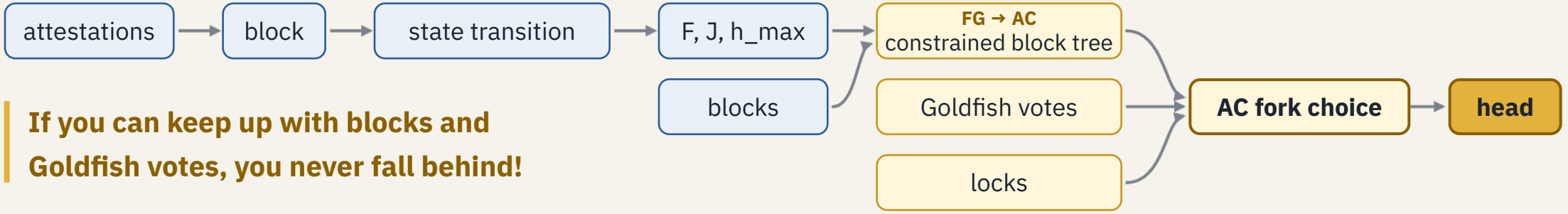
```
(validator, h, T, timeout, h_f, T_f)
```

On chain, in the state transition	Effect
2/3 target votes for (h, T)	justify T, advance the height
2/3 targets + timeouts, after a delay	advance the height, no justification
2/3 finality votes for a pending (j, T)	finalize T

No rounds in attestations. Only need to vote once "per height per chain": only vote again if we reorg to a different T_h, in which case you vote with (h, T_h, timeout)

TWO CONSEQUENCES

1 · Network attestations do not influence the fork choice



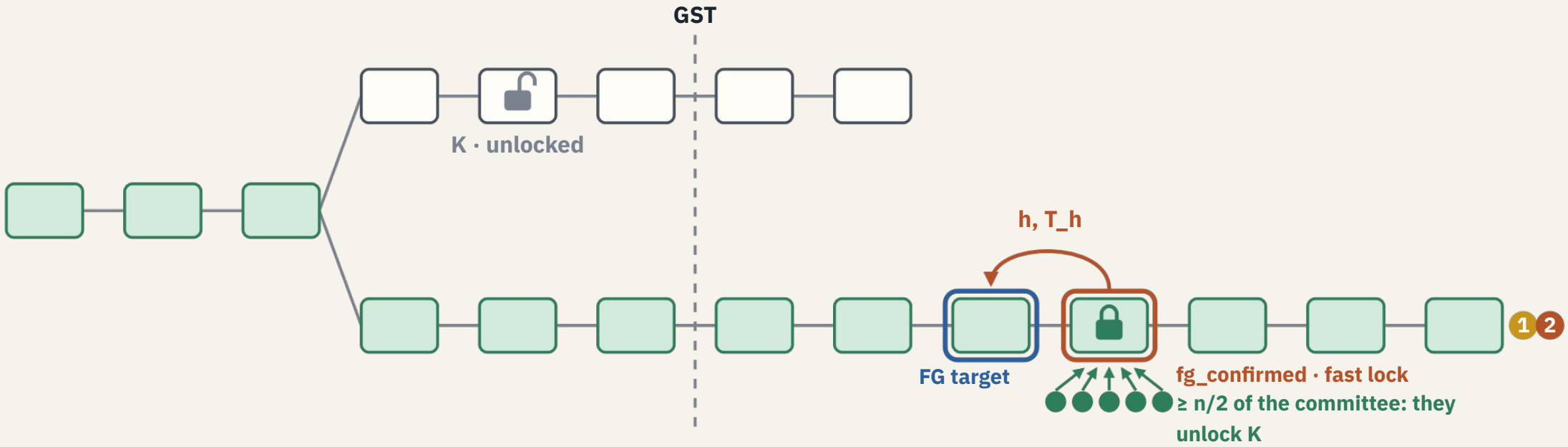
If you can keep up with blocks and Goldfish votes, you never fall behind!

2 · When finality stalls, attestation traffic dies down by itself

No height progress means no new target, so a validator whose vote landed is done. A backoff with no backoff rule: bandwidth goes to propagating and aggregating the current votes, not to sending new ones.

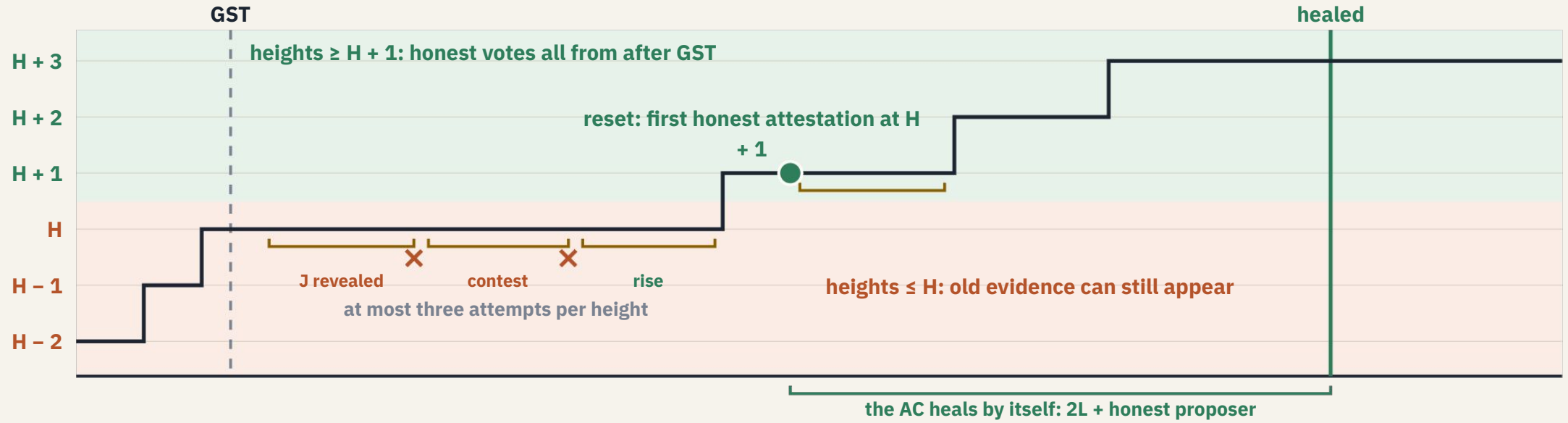
HEALING WITH THE FG

Fast locks make FG targets safe



After GST a fast lock unlocks every conflicting lock, and every honest head keeps it. A target read from its state is never dropped later, so it cannot set up an FG reorg.

The FG falls in line above the old heights



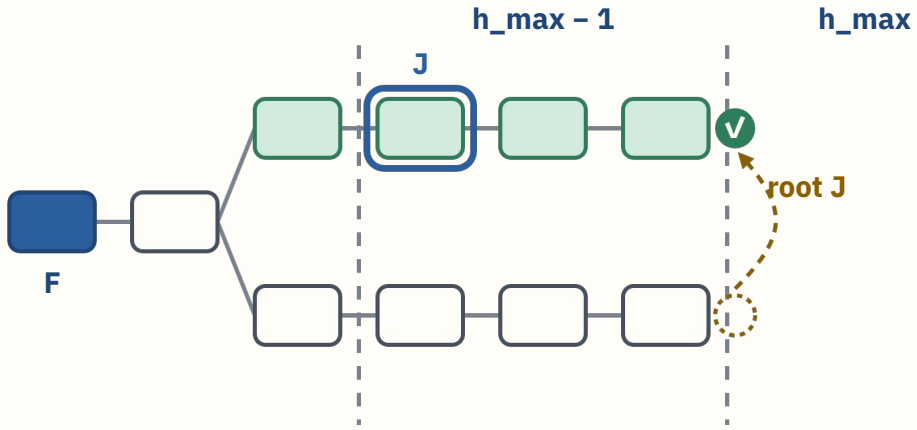
1 At most two frame changes per height: each height rises within three attempts

2 Targets come from fast locks: new justifications agree with the AC

3 From the first honest attestation at H + 1, the FG follows the AC; the AC heals 2L + one honest proposal later

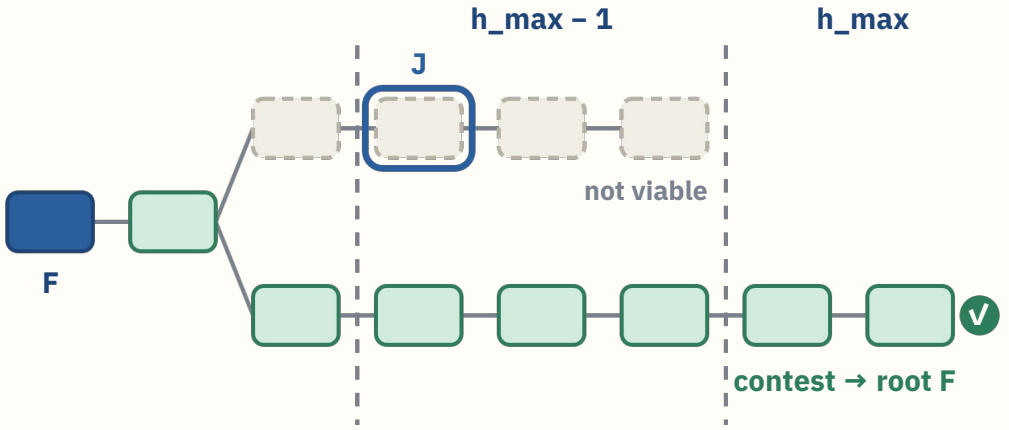
J = fc root only if no conflicting branch at hmax

Our branch still at h_max − 1



A J revealed late becomes the root and pulls us there. Once revealed, that J is spent.

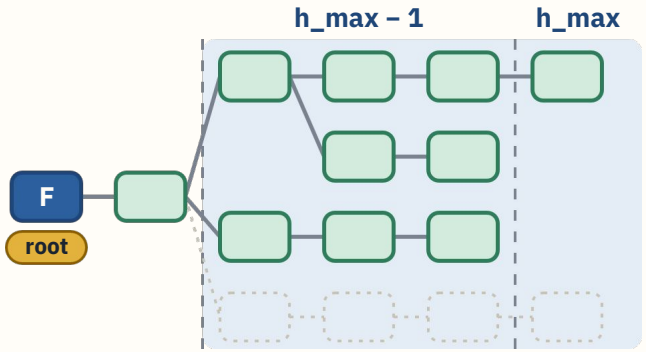
Our branch already at h_max



Our branch contests J, so the root falls back to F. No reorg from J being revealed.

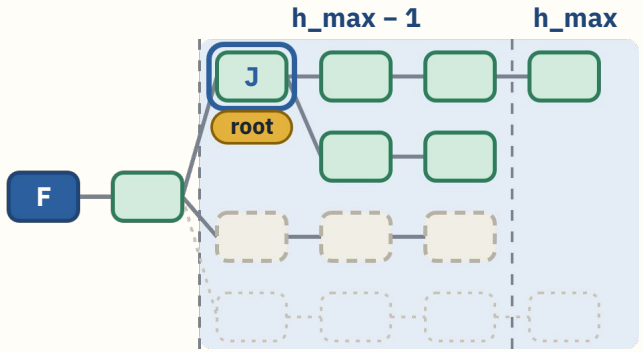
Two frame changes at most, then the height rises

1 · Before J



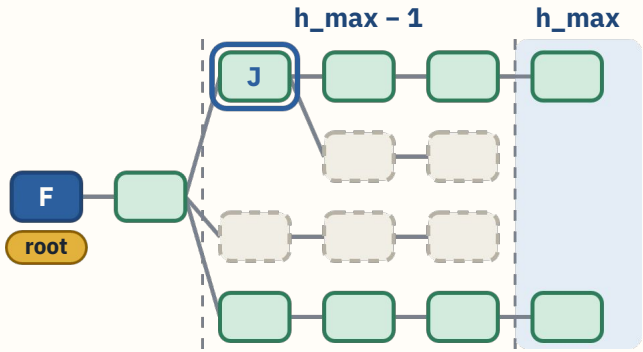
No J at h_max – 1 yet. Root F; height floor h_max – 1. If nothing changes, the height rises.

2 · J revealed



Root J. Branches against J drop out. If nothing else changes, the height rises.

3 · Contest



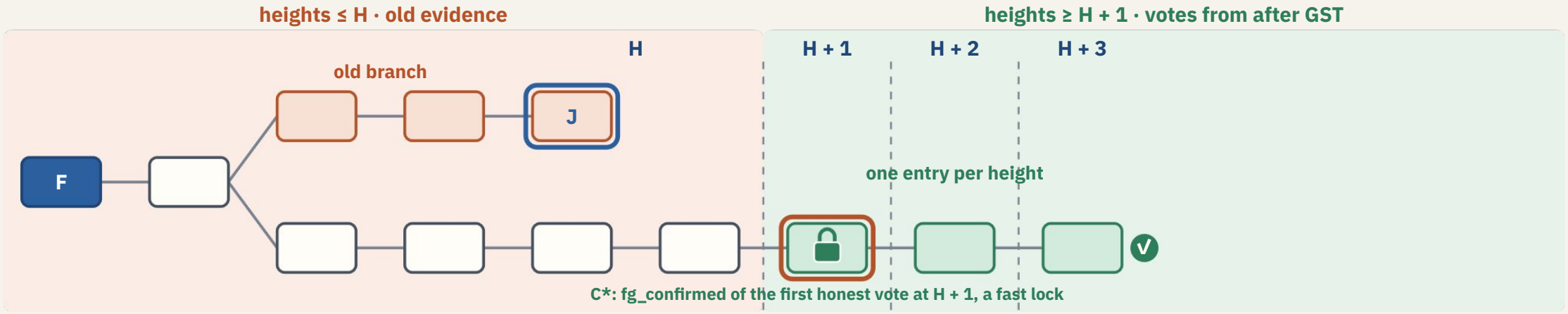
A block at h_max not on J. Root F; height floor h_max. Nothing can change any more: the height rises.

viable · dropped by the frame · not seen yet · ≥ height floor: must reach it · root fork-choice root

One J per height, and a contest never goes away: every height rises within three attempts.

HEALING WITH THE FG

From height H + 1 on, the FG follows the AC



H: the highest height that honest nodes reach shortly after GST. Evidence from before GST stays at H or below.

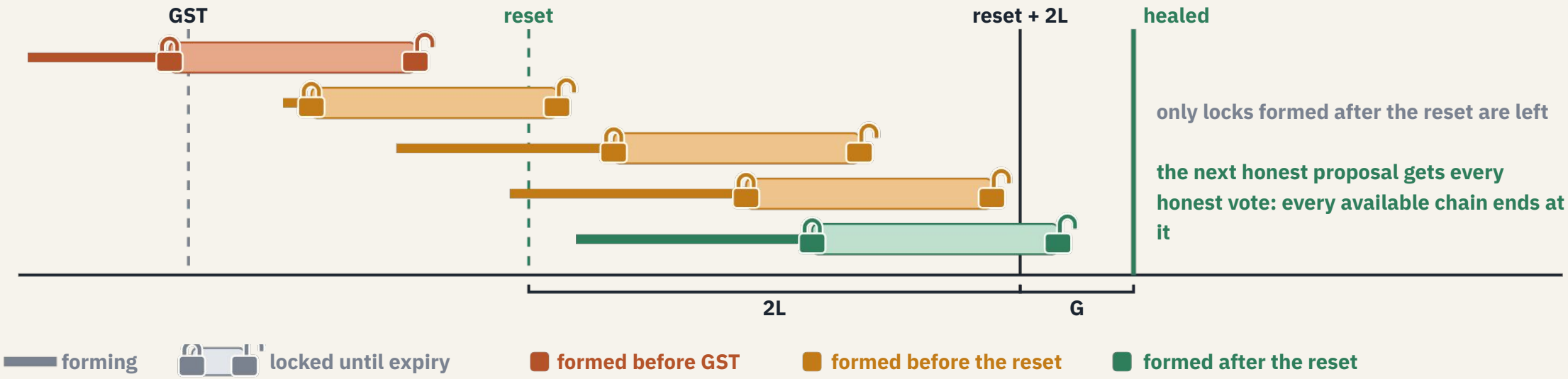
1 h_max = H + 1: an old J is either below C*, or C* contests it. C* stays

2 h_max ≥ H + 2: an old J is too low to be the fork-choice root

3 New justifications name one entry per height, on C*'s chain

HEALING WITH THE FG

After the reset, the AC heals by itself



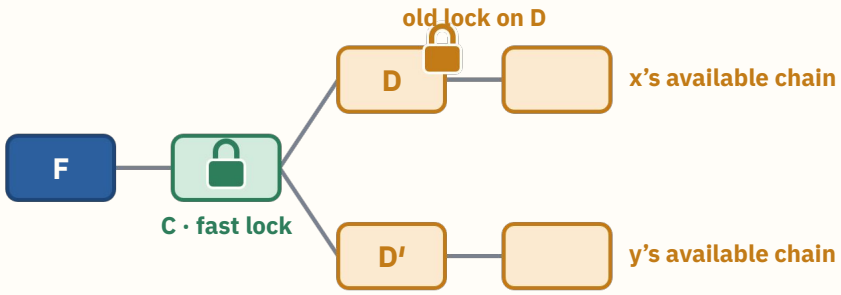
1 From the reset on, the FG follows the AC

2 Locks formed after the reset are followed by everyone

3 Older locks expire by reset + 2L; the next honest proposal heals ⇒ reset + 2L + G

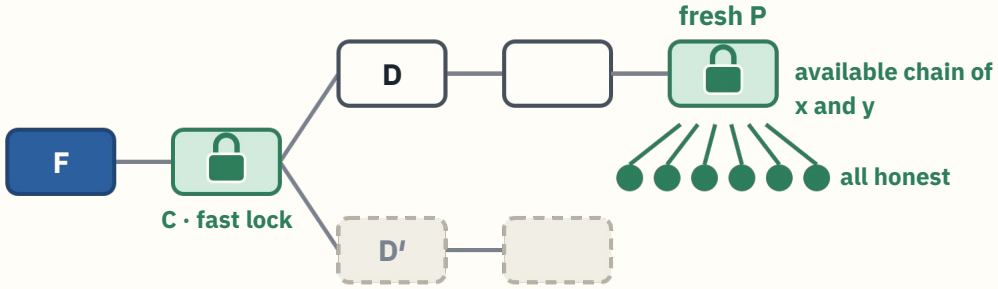
Why the AC still needs 2L after the reset

A fast lock on C



C is below both tips, and an available chain never moves back to an ancestor: x and y still disagree. The old lock on D also holds x's head there.

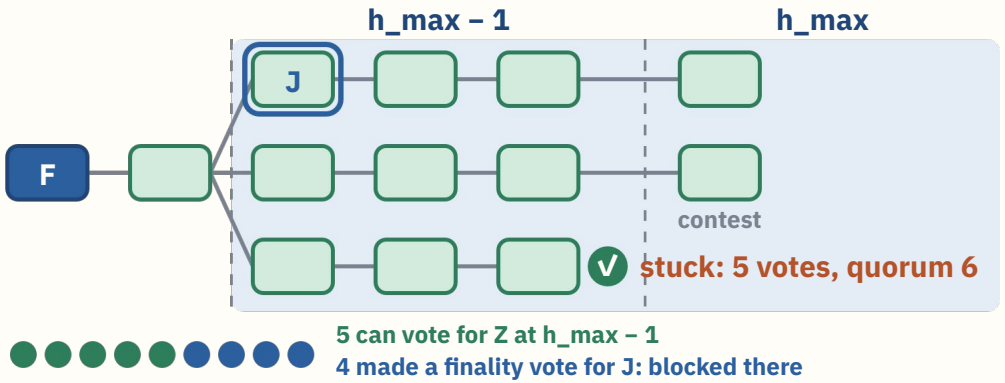
First honest proposal after old locks expire



No old tip extends a fresh block, and every honest validator votes for P: every available chain now ends at P.

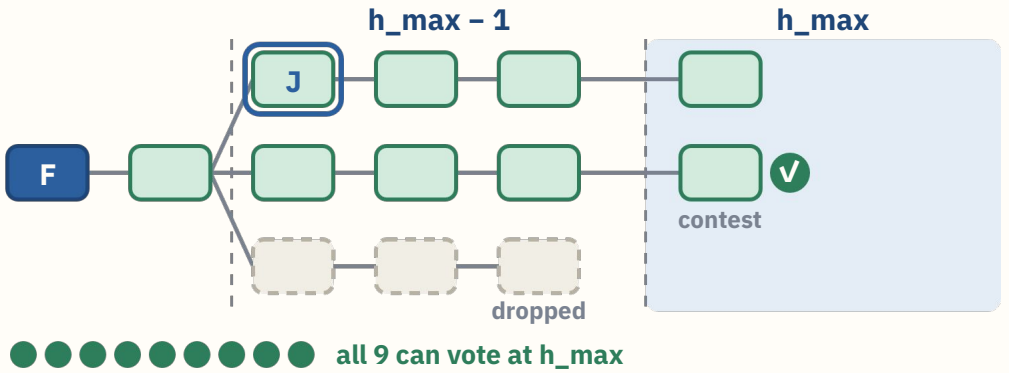
Why a contest raises the height floor

Height floor stays at h_max − 1



The head can sit on Z, which still needs a quorum at h_max − 1. Validators that made a finality vote for J vote for nothing else there.

Height floor rises to h_max



Branches at h_max need no more votes at h_max − 1. The finality votes for J block nothing.