Partial sync: periods of sync + async.

(DLS) ← Dijkstra

1. Unknown Global stabilization time (GST): + known $\Delta$:

$\Delta$ assumption does not hold $\Rightarrow$ $\Delta$ assumption holds.

GST

equivalent $\Leftrightarrow$

2. Unknown $\Delta$:

Both defines are equivalent to each other.

State machine replication (SMR)

state machine

cmd, cmd executed

ledges
Safety: Any two honest replicas should output different values at the same past command.

Liveness: Every honest client should eventually output a value sent by a client.

Agreement and termination & validity:

→ One value vs a ledger of values (ordering).

→ Notion of client: t+1 replicas.

BB: Boulware-strong $t < n-1$ faults.

Partial synchrony (PS):

Under PS, it is impossible to tolerate $t > \frac{n}{3}$ Byzantine faults.

$t > \frac{n}{2}$ crash.

BA: "Split-brain attack"
World 1: A & B are honest, C has crashed. A & B start with input 1. 
\[ t \geq \frac{n}{3} \] faults. 

By validity constraint, A & B output 1.

World 2: 
B & C are honest 
By validity constraint, B & C output 0.

World 3:
B is Byzantine 
\[ \text{brain} \] works as in Inst.
A: This world is exactly the same as world 1.

Observations:

- Setup assumptions.
- Existence of partial synchrony & the fault being Byzantine.
  \[ n > 2t \]
- \[ t \geq \frac{n}{3} \] is not possible.
- \[ t < \frac{n}{3}, \quad n = 3t + 1 \]
- If one honest party decides on a value, then all honest parties should decide on the same value.
- Not all honest parties can communicate with one another; we need a majority of the honest parties to "agree" on the same value.
\[ n = 3t + 1 \]

Byzantine: \( t + 1 \) honest parties, \( t \) with you

"accept-commit" paradigm.

PBFT: Practical Byzantine Fault Tolerance


Turing

Primary - backup paradigm:

"primary/leader": drive progress in the system; order in which values are

replicas: ensuring safety holds.

& ensuring primary is doing its job correctly,

pros: In a good world: progress - streamlined.

cons: subtle attacks: ordering.

Steady State (v): Replica i's perspective.
Propose : Leader proposes \(<\text{propose, B, v, ...}>\)

Vote 1! \((\text{prepare})\): \(<\text{vote1, B, v, ...}>\)

"If it is safe" to vote for it.

Vote 2! \((\text{commit})\): Replica i collects \(2t+1\) vote1's

Replica i sends \(<\text{vote2, B, v, 2t+1 vote1's}>\)

Commit \((\text{committed})\): Replica i collects \(2t+1\) vote2's

\(B\) then it commits \(B\).

\(\Box\)
If an honest party commits \( B \), i.e., it receives \( 2t+1 \) vote2 messages, then at least \( t+1 \) honest parties have receive a quorum of vote1 messages (have locked on \( B \)).

Uniqueness within the view: first round of votes.

\[
\text{Voted for } B, v \quad \text{and} \quad \text{Voted1 for } B', v
\]

\[
\begin{align*}
\mathbb{Q} + t & = 4t + 1
\end{align*}
\]
up to $t$ parties will vote for both.

1. Only a unique value can be locked on in a given view.

2. If an honest party commits, the $2t+1$ honest parties lock on the committed value.

"Blame the leader": View-change.

Blame: send "blame, $y$, lock; $>$ to the next leader.

New leader:

On collecting $2t+1$ of blame messages & start a new-view.

Propose: pick a value with the
Byzantine skagglers. Locked.

"if it is safe to do so": Replica \( i \) votes if
- the value \( B \) is the same as the lock replica holds,
- if the proposed is value is \( B' \neq B \) but there is a higher lock in the status.

View \( v \): Honest party \( h \) is locked \((B, \nu)\)

View \( v+1 \): \( h \)'s status does not arrive
$h_2$ locked on $(b', v+1)$

$V + t + 1$

$(h_{t+1}$ on $b', v+t+1)$

makes me commit a value $(b', v')$

makes my commit unsafe "status"
\[2t+1\]

One slot to many slots:

$$S \quad S+1 \quad \sqrt{1} \quad \square$$

- propose
- propose
- vote
- vote
- commit

Byzantine parties:
- Increase bynos,
- Arbitrarily
- Lo-watermark
- Hi-watermark

Check pointing:

Moved very fast

1 million
State Transfer: (Dolev-Reischuk) \cdot n^2

Broadcast Extension protocols.

HotStuff / Casper / Tendermint : 2019
$O(n^2)$ communication within a view & during view-change.

propose vote1 vote2 blame propose.

$O(n)$ $O(n)$ $O(n)$

$O(n^2)$ $O(n^2)$ $O(n^2)$
**Linear-view-change**: 

Safety: If \( t+1 \) honest parties had locked, status cut included at least one of them. 

\[ q_{t+1} \leftarrow \left( 0, \ldots \right) \]

locks. 

\[ \downarrow \]

“It is safe to vote”.

Liveness: A party can unlock

Idea: Do not include a status cut, just include the highest lock.

Fix safety: No party will ever unlock.
quorum who guards my commits safely.

\[ n \geq t+1 \text{ locks} \]

\[ n^2 + 1 \text{ vote 2's.} \]

propose  vote 0 key  vote 1  lock  vote 2  commit  blame  purpose.

\[ \square \]