BFT Consensus: State Machine Replication Definition

- $f$ out of all $n$ nodes could exhibit arbitrarily faulty behavior
- The rest of $(n-f)$ nodes replicate the same sequence of commands
- Total ordering of commands in asynchronous network: \textit{safety}’
- It’d better make useful progress: \textit{liveness}’

Replica... 1 2 3 4
Replica... 1 2 3 4
Byz...
Replica... 1 2 3 4
BFT Consensus: Why Another Protocol?
BFT Consensus: Why Another Protocol?

The Saddest Moment, Mickens 2013
BFT Consensus: “Nakamoto Consensus”

“Proof-of-Work” (PoW)

“Genesis Block”

“Longest Chain” Rule

Bitcoin: A Peer-to-Peer Electronic Cash System
Satoshi Nakamoto
Engineers want a simple, well-specified algorithm, so its implementation is less prone to critical bugs.

Algorithmic complexity is the most exciting thing for the researchers!
Network Model

Messages can be delayed for arbitrarily long time.

Asynchronous

Informally worded as “always safe even in asynchrony, live only when network is synchronous”

Randomized protocols...

Partially Synchronous

DLS/PBFT/.../HotStuff

“Global Stabilization Time” (GST)

Synchronous

All messages have a known bound of delivery time.
Network Model

Synchronous

All messages have a known bound of delivery time.

Partially Synchronous

Informally worded as "always safe even in asynchrony, live only when network is synchronous"

DLS/PBFT/.../HotStuff

"Global Stabilization Time" (GST)

Synchonous

Linearity

The total number of exchanged authenticators (digital signatures/MACs) is $O(n)$.

“The communication cost is linear”
Linearity

The total number of exchanged authenticators (digital signatures/MACs) is $O(n)$.

“The communication cost is linear”

(Optimistic) Responsiveness

After GST, any correct leader, once designated, needs to wait just for the first $(n-f)$ responses to guarantee that it can create a proposal that will make progress.

“As fast as the network propagates, on a good day”
<table>
<thead>
<tr>
<th>Protocol</th>
<th>Correct Leader</th>
<th>Leader Failure</th>
<th>f Leader Failures</th>
<th>Responsive</th>
</tr>
</thead>
<tbody>
<tr>
<td>DLS</td>
<td>$O(n^4)$</td>
<td>$O(n^4)$</td>
<td>$O(n^4)$</td>
<td></td>
</tr>
<tr>
<td>PBFT</td>
<td>$O(n^2)$</td>
<td>$O(n^3)$</td>
<td>$O(fn^3)$</td>
<td>✓</td>
</tr>
<tr>
<td>SBFT</td>
<td>$O(n)$</td>
<td>$O(n^2)$</td>
<td>$O(fn^2)$</td>
<td>✓</td>
</tr>
<tr>
<td>Tendermint/Casper</td>
<td>$O(n^2)$</td>
<td>$O(n^2)$</td>
<td>$O(fn^2)$</td>
<td></td>
</tr>
<tr>
<td>Tendermint/Casper*</td>
<td>$O(n)$</td>
<td>$O(n)$</td>
<td>$O(fn)$</td>
<td></td>
</tr>
<tr>
<td>HotStuff</td>
<td>$O(n)$</td>
<td>$O(n)$</td>
<td>$O(fn)$</td>
<td>✓</td>
</tr>
</tbody>
</table>
# Protocol Complexity

<table>
<thead>
<tr>
<th>Quorum-based Consensus</th>
<th>Nakamoto Consensus</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Quorum votes</td>
<td>● PoW based</td>
</tr>
<tr>
<td>● Views/Slots</td>
<td>● Longest chain</td>
</tr>
<tr>
<td>● Fast, leader-based</td>
<td>● Slow, leaderless</td>
</tr>
<tr>
<td>● Complicated view change</td>
<td>● No explicit view change</td>
</tr>
<tr>
<td>● Spec is complicated</td>
<td>● Clearly specified</td>
</tr>
</tbody>
</table>
HotStuff: BFT Consensus with Linearity and Responsiveness

<table>
<thead>
<tr>
<th>Theoretical</th>
<th>Practical</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Linearity: $O(n)$</td>
<td>● View change made simple</td>
</tr>
<tr>
<td>● Responsiveness like PBFT</td>
<td>● Safety is by voting and commit rules over a graph of nodes (blocks)</td>
</tr>
<tr>
<td>● No additional cost for VC</td>
<td>● “Pacemaker”: decouples safety and liveness at the algorithmic level</td>
</tr>
<tr>
<td>● Unified interpretation of Quorums</td>
<td></td>
</tr>
</tbody>
</table>
Challenge: BFT Consensus in 10 Min

Ingredients to Make a HotStuff

● Messages
● Protocol state variables
● Voting rule
● Commit rule
Challenge: BFT Consensus in 10 Min

PBFT communication pattern
Concept: Quorum Certificate (QC)

Proof of the existence of $2f+1$ votes

Can only wait for $2f+1$ responses

Responses form correct replicas

$2f + 1$: QC

$\geq f + 1$

$\geq 1$

$2f + 1$: QC

$\geq f + 1$

$\geq 1$
Challenge: BFT Consensus in 10 Min

Messages

- `<propose, b_{new}>`
- `<vote, <u, b_{new} signed by u>`

- Proposer broadcasts the propose message carrying block $b_{new}$
- Voters give back their opinions to the next proposer via votes
- Only one type of messages for voting/view change, etc.
Challenge: BFT Consensus in 10 Min

Client

<propose, $b_1$> <vote> <propose, $b_2$> <vote> <propose, $b_3$> <vote> <propose, ...>
“Blockchain”: a Tree of Blocks (“Nodes”)

Every block contains a QC for some previous block (the genesis block, for example), some blocks’ QCs are omitted in the diagram for clarity. A QC has to refer to an ancestor.
Challenge: BFT Consensus in 10 Min

Protocol State Variables

- $b_{lock} = \text{block leading the preferred branch}$
- $b_{exec} = \text{last committed block}$
- $vheight = \text{height of the block last voted for}$
“Longest” Chain Rule: Branch Preference

$b_{\text{lock}}$: the “locked block” that leads the preferred branch

Locking mechanism: a replica only votes for the block on the preferred branch, unless...
Challenge: BFT Consensus in 10 Min

How to Vote (Safety Rule)?

- Only vote for $b_{\text{new}}$ if the following constraints hold:
  - $b_{\text{new}}.\text{height} > v_{\text{height}}$
  - $(b_{\text{new}} \text{ is on the same branch as } b_{\text{lock}})$ or $(b_{\text{new}}.\text{justify}.\text{node.height} > b_{\text{lock}}.\text{height})$
Challenge: BFT Consensus in 10 Min

How to Vote (Liveness Rule)?

- Only vote for $b_{\text{new}}$ if the following constraints hold:
  - $b_{\text{new}}.\text{height} > v_{\text{height}}$
  - ($b_{\text{new}}$ is on the same branch as $b_{\text{lock}}$) or ($b_{\text{new}}.\text{justify}.\text{node.height} > b_{\text{lock}}.\text{height}$)
Challenge: BFT Consensus in 10 Min

When to Commit?
Challenge: BFT Consensus in 10 Min

When to Commit?
HotStuff Framework: Commit Rule

3-phase HS

b_1 \rightarrow b_2 \rightarrow b_3 \rightarrow b_4

b_7 \rightarrow b_5 \rightarrow b_6 \rightarrow b_8 \rightarrow \ldots

QC

2-phase HS

b_1 \rightarrow b_2 \rightarrow b_3 \rightarrow b_4

b_7 \rightarrow b_5 \rightarrow b_6 \rightarrow b_8 \rightarrow \ldots

QC
HotStuff Framework: Branch Preference

3-phase HS

b₁ → b₂ → b₃ → b₄ → QC

b₇ → b₈

2-phase HS

b₁ → b₂ → b₃ → b₄ → b₅ → QC

b₇ → b₈

f + 1 correct replicas know qc_{high} is on b₄
Listed projects use/claim to use/is based on HotStuff, but we choose not to comment on any one of them. Nor did we receive any compensation by listing them.

ThunderCore PaLa is based on HotStuff.
Evolution (incomprehensive)

- **DLS**
  - VC: $O(N^3)$
  - Normal: $O(N^4)$
  - Responsive

- **PBFT**
  - VC: $O(N^3)$ or $O(N^2)$
  - Normal: $O(N^2)$
  - Responsive

- **PBFT***
  - VC: $O(N^2)$
  - Normal: $O(N)$
  - Responsive

- **Casper**
  - VC: $O(N^2)$
  - Normal: $O(N^2)$
  - Not responsive

- **Casper***
  - VC: $O(N)$
  - Normal: $O(N)$
  - Responsive

- **Tendermint**
  - VC: $O(N^2)$
  - Normal: $O(N^2)$
  - Not responsive

- **Tendermint***
  - VC: $O(N)$
  - Normal: $O(N)$

- **HotStuff**
  - VC: $O(N)$
  - Normal: $O(N)$
  - Not responsive

DLS (88): Dwork et al.
PBFT (99): Castro et al.
Tendermint (16): Buchman
SBFT (18): Gueta et al.
That’d be all.

Special Thanks

VMware Research Group
Calibra

All people contributed to this paper

← Longer version of the paper

Code is now open-sourced →