Randomized vs. Deterministic? Practical Randomized Synchronous BFT in Expected Constant Time

Xufeng Zhang[1], Baohan Huang[1], Sisi Duan[2], Haibin Zhang[3]

[1] Beijing Institute of Technology, China
[2] Tsinghua University, China
[3] Yangtze Delta Region Institute of Tsinghua University, Zhejiang, China

September 30, 2025

BFT Under Different Network Assumptions

Type	Synchronous	Partially Synchronous	Asynchronous
Assumption	Known upper bound for message transmission and processing	Unknown upper bound	No upper bound
Resilience Bound	$f < n/2$	$f < n/3$	$f < n/3$
Protocols	Sync HotStuff	PBFT, HotStuff	HoneyBadger, BEAT, Dumbo

BFT Under Different Network Assumptions

Type	Synchronous	Partially Synchronous	Asynchronous
Assumption	Known upper bound for message transmission and processing	Unknown upper bound	No upper bound
Resilience Bound	$f < n/2$	$f < n/3$	$f < n/3$
Protocols	Sync HotStuff	PBFT, HotStuff	HoneyBadger, BEAT, Dumbo

Synchronous BFT achieves better resilience !

Challenges in Practical Synchronous BFT

However, synchronous BFT has not yet been practical enough ...

Challenges in Practical Synchronous BFT

- Two types : **Deterministic** / Randomized.

Protocol	Time	Msg	Latency (failure-free)	Adversary
Sync HotStuff	$O(n)$	$O(n^2)$	$2\Delta + \delta$	static / adaptive
1Δ-SMR	$O(n)$	$O(n^2)$	$\Delta + 2\delta$	static / adaptive
Sync-RR	$O(n)$	$O(n^2)$	$2\Delta + O(\delta)$	static / adaptive
OptSync	$O(n)$	$O(n^2)$	2δ	static / adaptive

Challenges in Practical Synchronous BFT

- Two types : **Deterministic** / Randomized.
- Deterministic protocols have decent performance in failure-free scenarios.

Protocol	Time	Msg	Latency (failure-free)	Adversary
Sync HotStuff	$O(n)$	$O(n^2)$	$2\Delta + \delta$	static / adaptive
1Δ-SMR	$O(n)$	$O(n^2)$	$\Delta + 2\delta$	static / adaptive
Sync-RR	$O(n)$	$O(n^2)$	$2\Delta + O(\delta)$	static / adaptive
OptSync	$O(n)$	$O(n^2)$	2δ	static / adaptive

Challenges in Practical Synchronous BFT

- Two types : **Deterministic** / Randomized.
- Deterministic protocols have decent performance in failure-free scenarios.
- But they inevitably have $O(n)$ time as the lower bound !

Protocol	Time	Msg	Latency (failure-free)	Adversary
Sync HotStuff	$O(n)$	$O(n^2)$	$2\Delta + \delta$	static / adaptive
1Δ-SMR	$O(n)$	$O(n^2)$	$\Delta + 2\delta$	static / adaptive
Sync-RR	$O(n)$	$O(n^2)$	$2\Delta + O(\delta)$	static / adaptive
OptSync	$O(n)$	$O(n^2)$	2δ	static / adaptive

Challenges in Practical Synchronous BFT

- Two types : Deterministic / **Randomized**.

Protocol	Time	Msg	Latency (failure-free)	Adversary
PiLi	$O(1)$	$O(n^2)$	40Δ~65Δ	static
Streamlet	$O(1)$	$O(n^3)$	12Δ	static
Dfinity	$O(1)$	$O(n^2)$	8Δ	mildly adaptive
	$O(n)$	$O(n^3)$	12Δ	(standard) adaptive
ADDNR	$O(1)$	$O(n^2)$	5Δ	static
			8Δ	strongly rushing adaptive

Challenges in Practical Synchronous BFT

- Two types : Deterministic / **Randomized**.
- Randomized protocols can achieve $O(1)$ time.

Protocol	Time	Msg	Latency (failure-free)	Adversary
PiLi	$O(1)$	$O(n^2)$	40Δ~65Δ	static
Streamlet	$O(1)$	$O(n^3)$	12Δ	static
Dfinity	$O(1)$	$O(n^2)$	8Δ	mildly adaptive
	$O(n)$	$O(n^3)$	12Δ	(standard) adaptive
ADDNR	$O(1)$	$O(n^2)$	5Δ	static
			8Δ	strongly rushing adaptive

Challenges in Practical Synchronous BFT

- Two types : Deterministic / **Randomized**.
- Randomized protocols can achieve $O(1)$ time.
- But they have high latency even in failure-free scenarios !

Protocol	Time	Msg	Latency (failure-free)	Adversary
PiLi	$O(1)$	$O(n^2)$	40Δ~65Δ	static
Streamlet	$O(1)$	$O(n^3)$	12Δ	static
Dfinity	$O(1)$	$O(n^2)$	8Δ	mildly adaptive
	$O(n)$	$O(n^3)$	12Δ	(standard) adaptive
ADDNR	$O(1)$	$O(n^2)$	5Δ	static
			8Δ	strongly rushing adaptive

Challenge 1 : High Complexity

- Two types : Deterministic / **Randomized**.
- Randomized protocols can achieve $O(1)$ time.
- But they have high latency even in failure-free scenarios !

Protocol	Time	Msg	Latency (failure-free)	Adversary
PiLi	$O(1)$	$O(n^2)$	40Δ~65Δ	static
Streamlet	$O(1)$	$O(n^3)$	12Δ	static
Dfinity	$O(1)$	$O(n^2)$	8Δ	mildly adaptive
	$O(n)$	$O(n^3)$	12Δ	(standard) adaptive
ADDNR	$O(1)$	$O(n^2)$	5Δ	static
			8Δ	strongly rushing adaptive

Challenges in Practical Synchronous BFT

In contrast, **partially sync** and **async** BFT have been widely studied in practical settings :

- Improving throughput by **multiple concurrent proposers**.
- Delivering $O(n)$ block proposals in each epoch.

Challenges in Practical Synchronous BFT

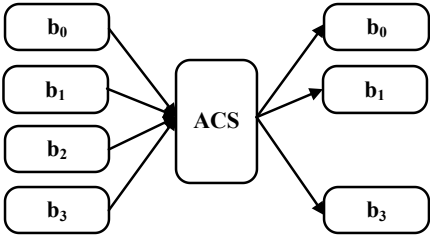
In contrast, **partially sync** and **async** BFT have been widely studied in practical settings :

- Improving throughput by **multiple concurrent proposers**.
- Delivering $O(n)$ block proposals in each epoch.
- $O(n)$ amortized message per block.

Challenges in Practical Synchronous BFT

In contrast, **partially sync** and **async** BFT have been widely studied in practical settings :

- Improving throughput by **multiple concurrent proposers**.
- Delivering $O(n)$ block proposals in each epoch.
- $O(n)$ amortized message per block.
- Such as async BFT in async common subset (ACS) family.

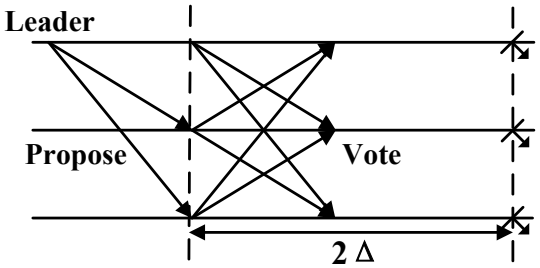


Challenges in Practical Synchronous BFT

- However, such a paradigm has never been explored for sync BFT.

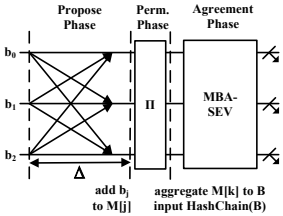
Challenge 2 : Single-Proposer Design

- However, such a paradigm has never been explored for sync BFT.
- Most sync BFT just allows leader to propose one block in each epoch.



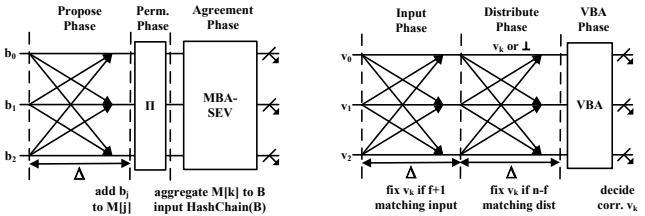
Step-by-Step Reduction

BFT for full blocks



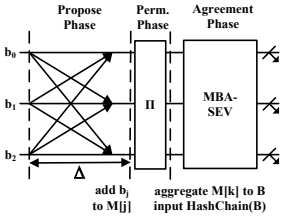
Step-by-Step Reduction

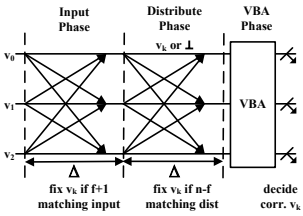
BFT for full blocks $\Longrightarrow$ MBA for compact hashes

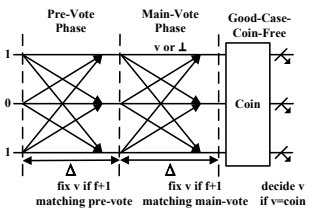


Step-by-Step Reduction

BFT for full blocks $\Longrightarrow$ MBA for compact hashes $\Longrightarrow$ BA for one-bit values

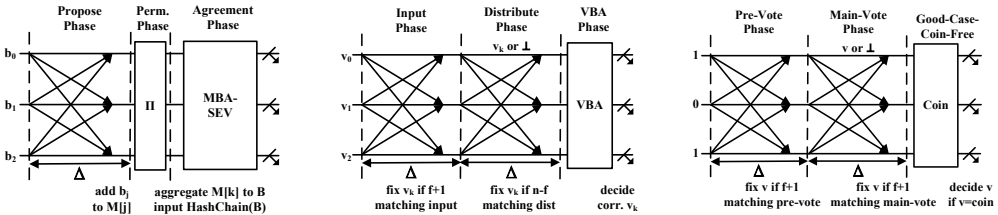






Step-by-Step Reduction

BFT for full blocks $\Longrightarrow$ MBA for compact hashes $\Longrightarrow$ BA for one-bit values



$O(1)$ **time,** $O(n^2)$ **message, and fewer steps.**

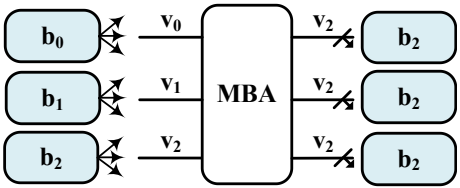
Goal : Reducing BFT for full blocks to MBA for compact hashes.

When we just use standard MBA protocols ...

Goal : Reducing BFT for full blocks to MBA for compact hashes.

When we just use standard MBA protocols ...

- Deciding a hash value from correct replica.



Goal : Reducing BFT for full blocks to MBA for compact hashes.

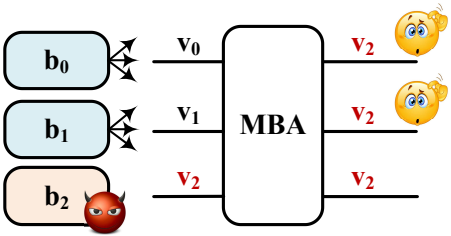
When we just use standard MBA protocols ...

- What if deciding a hash value from Byzantine replica ?

Goal : Reducing BFT for full blocks to MBA for compact hashes.

When we just use standard MBA protocols ...

- What if deciding a hash value from Byzantine replica ?



- Correct replicas cannot fetch the corresponding block !

Goal : Reducing BFT for full blocks to MBA for compact hashes.

- **Non-Intrusion** : Ensuring decided hash value from at least one correct replica, who has the corresponding block.

Goal : Reducing BFT for full blocks to MBA for compact hashes.

MBA-SEV : Multi-Valued Byzantine Agreement with Strong External Validity

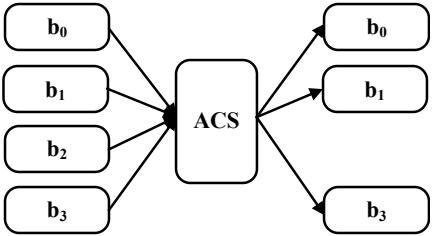
- Add 3 properties on top of MBA.
- **External Validity** : Validation, similar to multi-valued validated BA (MVBA).
- **Unanimous Validity** : Making progress if input the same hash.
- **Non-Intrusion** : Ensuring decided hash value from at least one correct replica, who has the corresponding block.

Goal : Reducing BFT for full blocks to MBA for compact hashes.

MBA-SEV : Multi-Valued Byzantine Agreement with Strong External Validity

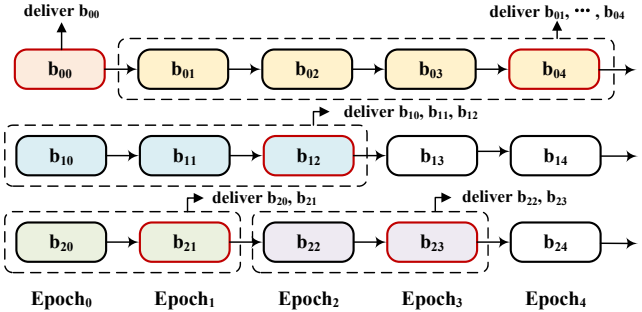
- Add 3 properties on top of MBA.
- **External Validity** : Validation, similar to multi-valued validated BA (MVBA).
- **Unanimous Validity** : Making progress if input the same hash.
- **Non-Intrusion** : Ensuring decided hash value from at least one correct replica, who has the corresponding block.

Batch Delivery : Prior Approach



- Each replicas proposes a block in each epoch.
- Each time $O(n)$ blocks are included in ACS and delivered.

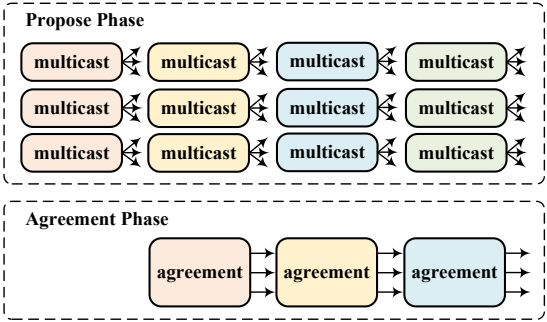
Batch Delivery : A Different Approach



- Deliver historical blocks proposed by the selected p_k together.
- Each time $O(n)$ historical blocks are delivered.
- Input of MBA-SEV is still only one hash value !

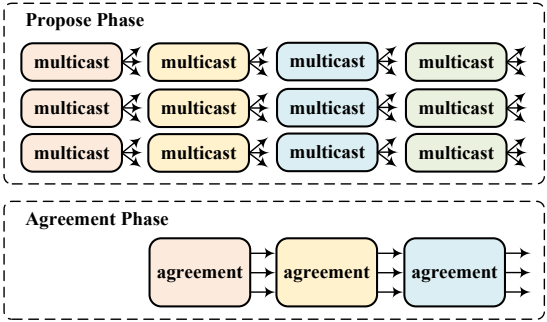
Pipelined Version

- Decoupling propose phase from agreement phase.
- Improving performance !



Pipelined Version

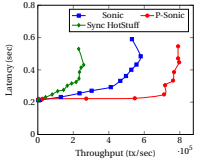
- Decoupling propose phase from agreement phase.
- Improving performance !
- But much trickier to tune timer ...



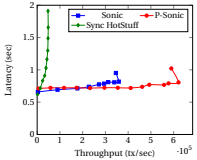
- Evaluated on AWS m5.xlarge, up to 91 VMs, in the same region.
- Transaction size : 50 bytes.
- Timer Δ : Large enough. A block with 10,000 TXs can be timely received.

Network Size	$f = 1$	$f = 10$	$f = 20$	$f = 30$	$f = 45$
Timer Δ	100 ms	300 ms	800 ms	1,200 ms	2,000 ms

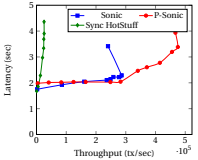
Evaluation Results



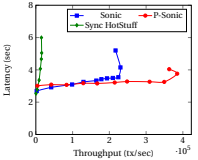
(a) Throughput vs. Latency when $f = 1$.



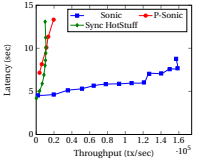
(b) Throughput vs. Latency when $f = 10$.



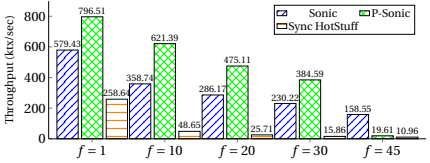
(c) Throughput vs. Latency when $f = 20$.



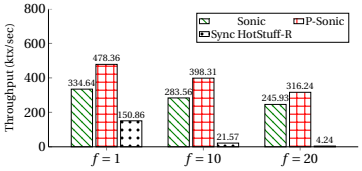
(d) Throughput vs. Latency when $f = 30$.



(e) Throughput vs. Latency when $f = 45$.



(f) Peak Throughput.



(g) Peak Throughput under f crash failures.

Conclusion

- Randomized sync BFT paradigm. **Expected $O(1)$ time**. **$O(n)$ amortized message per block proposal** based on new primitive MBA-SEV.

Conclusion

- Randomized sync BFT paradigm. **Expected $O(1)$ time**. **$O(n)$ amortized message per block proposal** based on new primitive MBA-SEV.
- Sync BFT : Sonic. **Near-optimal latency of 2Δ** under failure-free scenarios. Beating prior solutions.

Conclusion

- Randomized sync BFT paradigm. **Expected $O(1)$ time**. **$O(n)$ amortized message per block proposal** based on new primitive MBA-SEV.
- Sync BFT : Sonic. **Near-optimal latency of 2Δ** under failure-free scenarios. Beating prior solutions.
- Sync BA & MBA : Expected $O(1)$ time, $O(n^2)$ message, and **fewer steps** than SOTA.

Conclusion

- Randomized sync BFT paradigm. **Expected $O(1)$ time**. **$O(n)$ amortized message per block proposal** based on new primitive MBA-SEV.
- Sync BFT : Sonic. **Near-optimal latency of 2Δ** under failure-free scenarios. Beating prior solutions.
- Sync BA & MBA : Expected $O(1)$ time, $O(n^2)$ message, and **fewer steps** than SOTA.
- Peak throughput of Sonic and P-Sonic is **2.24x-14.52x** and **3.08x-24.25x** that of Sync HotStuff, respectively.

Conclusion

- Randomized sync BFT paradigm. **Expected $O(1)$ time**. **$O(n)$ amortized message per block proposal** based on new primitive MBA-SEV.
- Sync BFT : Sonic. **Near-optimal latency of 2Δ** under failure-free scenarios. Beating prior solutions.
- Sync BA & MBA : Expected $O(1)$ time, $O(n^2)$ message, and **fewer steps** than SOTA.
- Peak throughput of Sonic and P-Sonic is **2.24x-14.52x** and **3.08x-24.25x** that of Sync HotStuff, respectively.
- Randomized synchronous BFT can be both practical and efficient !

Thank you for listening !

Xufeng Zhang

xufengzhang.crypto@gmail.com

Beijing Institute of Technology