ICDE 2023

Scaling Blockchain Consensus via a Robust Shared Mempool

Fangyu Gai[1,†], Jianyu Niu[2,†], Ivan Beschastnikh[3], Chen Feng[1], Sheng Wang[4]
[1]{fangyu.gai, chen.feng}@ubc.ca [2]niujy@sustech.edu.cn [3]bestchai@cs.ubc.ca [4]sh.wang@alibaba-inc.com
University of British Columbia ([1]Okanagan Campus, [3]Vancouver Campus)
[2]Southern University of Science and Technology [4]Alibaba Group

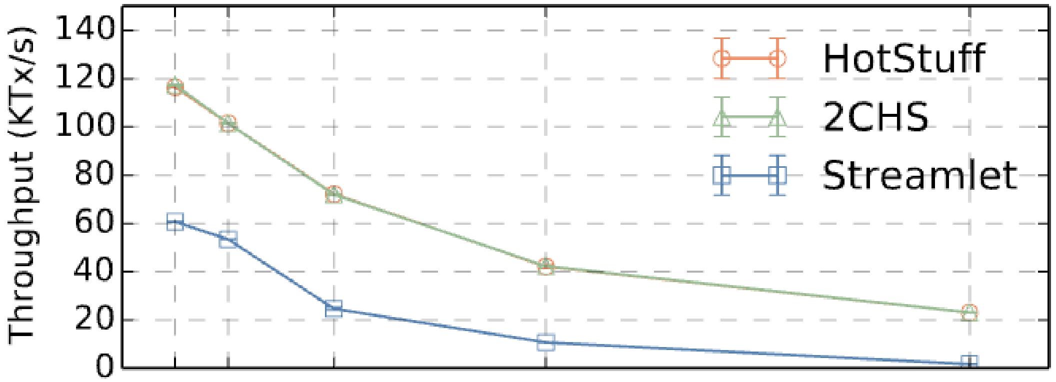
2024. 09. 02

서울대 분산시스템연구실

석사과정 문보설

Background

- Leader based BFT(LBFT) lacks scalability



Throughput of LBFT protocols drops from 120K tps (transaction per second) with 4 replicas to 20K tps with 64 replicas

Background

- Leader bottleneck
  - A key scalability challenge for Leader based BFT (LBFT)
  - Proposing and commit are handled by leader

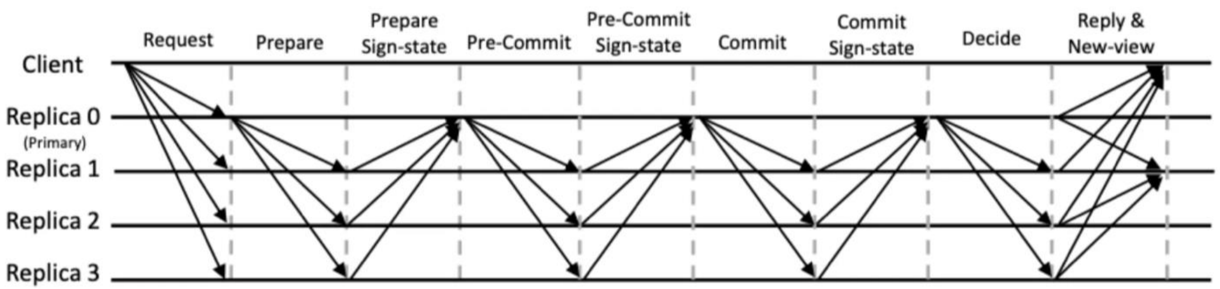


Fig. 4. The workflow of the 4-phase replication under normal operation in HotStuff.

TABLE III: Outbound bandwidth consumption comparison (MB/s) with $N = 64$ replicas. The bandwidth of each replica is throttled to 100 MB/s. The results are collected when the network is saturated.

Role/Messages		N-HS	SMP-HS	**S-HS (this paper)**
Leader	Proposals	75.4	4.7	9.8
	Microblocks	N/A	50.5	50.3
	SUM	**75.4**	**55.2**	**60.1**
Non-leader	Microblocks	N/A	50.4	50.3
	Votes	0.5	2.5	2.4
	Acks	N/A	N/A	4.7
	SUM	**0.5**	**52.9**	**57.4**

Background

1) Proposing phase

- Leader : Forms a proposal and broadcasts it to the other replicas
- Replicas : Verify the proposal

Permissioned Network

2) Commit phase

- Leader : Checks whether all correct replicas have committed to the same proposal

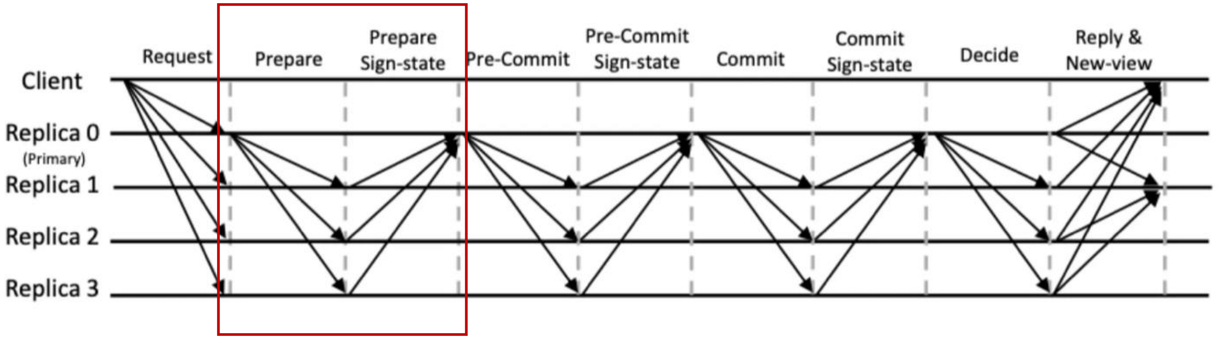


Fig. 4. The workflow of the 4-phase replication under normal operation in HotStuff.

Shared Mempool Abstraction

- **Decouples transaction distribution from consensus**

1) Transaction data is disseminated among replicas
   - Transactions can be batched == **Microblock**
2) Proposals contain only transaction ids
   - Proposal size can be further reduced through batching
3) Non-leader replicas reconstruct the proposal pulling txs from their local mempool
   - If there is **missing transaction**, they fetch it from other replicas (defined by shared mempool protocol)
   - Independent from the consensus algorithm
   - **Ensuring transaction availability is the role of the shared mempool**

Shared Mempool Abstraction

1) Upon receiving a new tx from the network, a replica adds tx into the mempool
2) Replica broadcasts tx if tx is from a client
3) Leader replica obtains a proposal p from local mempool
4) Leader proposes the proposal
5) Non-leader replicas reconstruct p
6) Send committed proposals to the executor

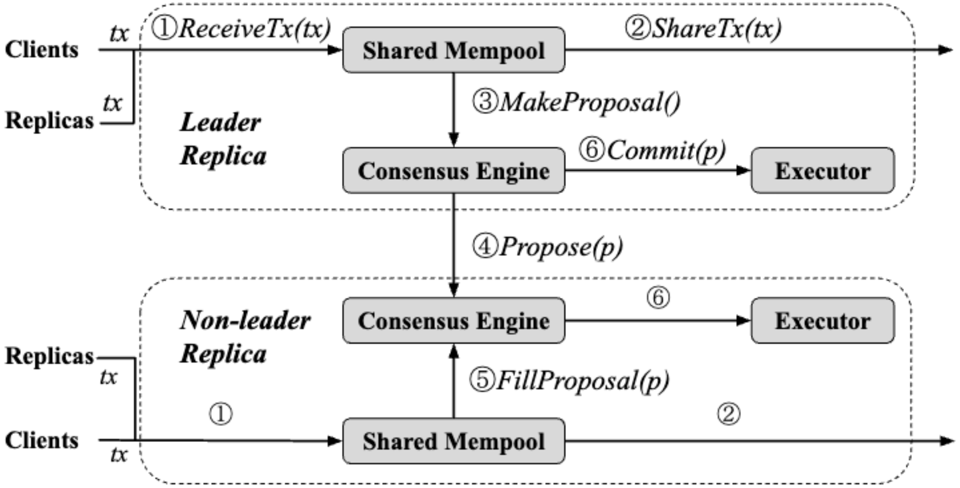
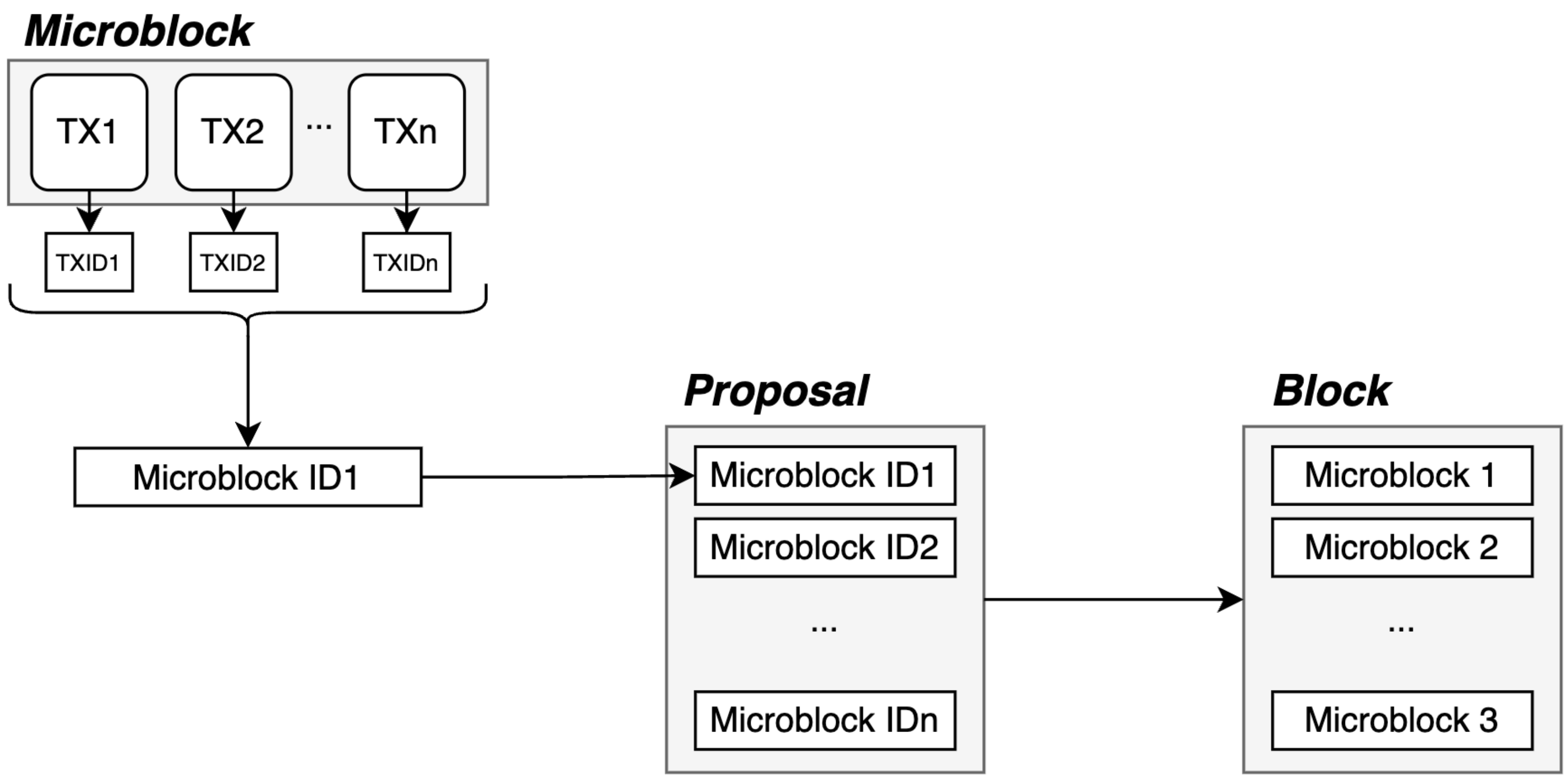


Fig. 1: The processing of transactions in state machine replication using SMP.

Data structure

- MicroBlock
  - Batched transaction
- Proposal
  - List of the microblock ids
- Block
  - Obtained by FillProposal(p)



Challenge 1: Missing Transaction

- **Integrity** of a proposal depends on the **availability** of referenced transactions
- Byzantine replica(R5) can only share a tx with the leader (R1) to
  1) Make frequent view-change (bottleneck)
  2) Make replicas fetch missing tx from the leader (bottleneck)

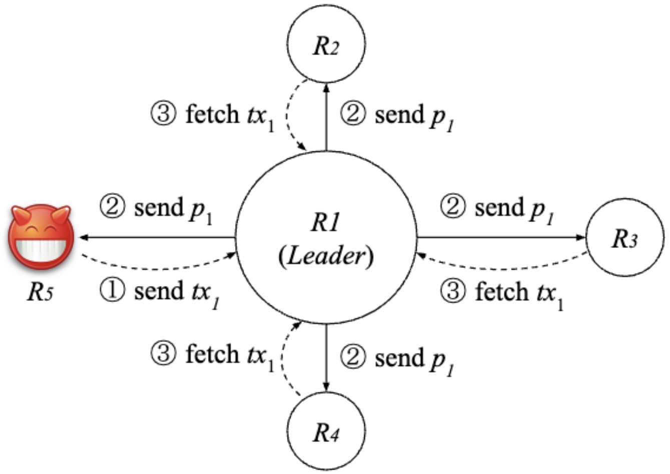


Fig. 2: In a system with SMP, consisting of 5 replicas in which R_5 is Byzantine and R_1 is the current leader.

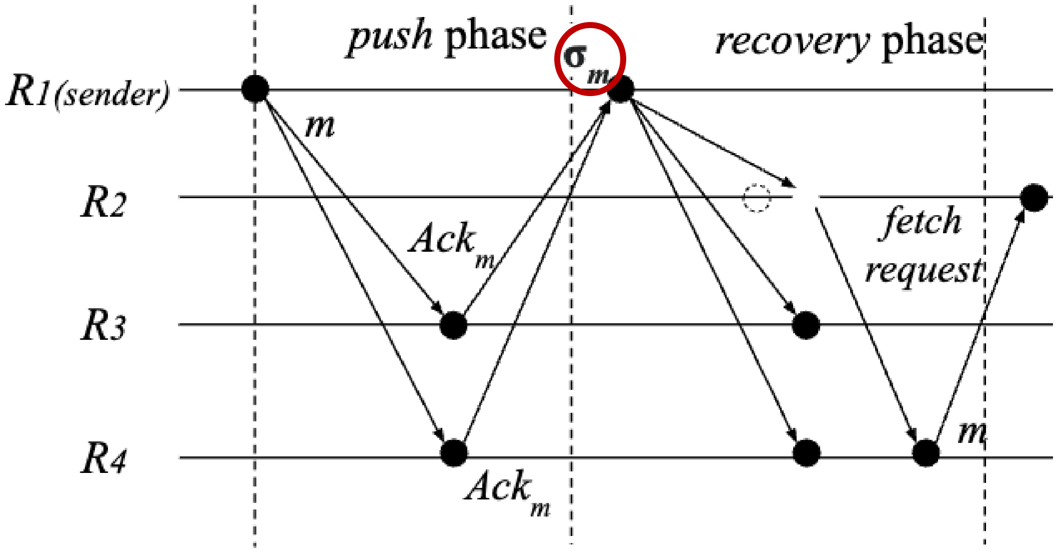
Solution 1: PAB(Provably Available Broadcast)

- **Idea**
  - A valid microblock requires a quorum of q signatures from replicas

- In previous example,
  1) If the missing transactions have valid signatures ➔ No view change is needed
  2) Fetch missing transactions from one of the q replicas ➔ Fetch request is distributed

Solution 1: PAB(Provably Available Broadcast)

- **Push phase**
  - Leader broadcasts microblock
  - Replicas send signature on ⟨PAB-Ack|m.id⟩
  - Leader produce succinct proof σ from a quorum of q signatures
  - e.g. q=f+1



- **Recovery Phase**
  - Leader broadcasts proof σ
  - Replicas missing the microblock fetch it from one of the signer of σ

Decoupling

- When a replica receives a proposal p:

1) Verify all proofs included in p
    1) Fail -> Trigger view change
    2) Success -> Move to the commit phase
2) Pull the content of microblocks associated with p
3) Fetch missing transactions
4) Execute filled proposal (block)

Decoupling

- When a replica receives a proposal p:

1) Verify all proofs included in p
    1) Fail -> Trigger view change
    2) Success -> Move to the commit phase
2) Pull the content of microblocks associated with p
3) Fetch missing transactions
4) Execute filled proposal (block)

Challenge 2: Unbalanced workload

- Nodes have varying resources.
- Clients are unevenly distributed

➔ Replicas with a low workload-to-bandwidth ratio can become bottlenecks

Solution 2: DLB(Distributed Load Balancing)

- Busy replicas forward their load to less busy replicas (proxy)
    1) A busy replica randomly **samples d replicas**
    2) Forwards its load to the least loaded replica (proxy)
    3) Proxy replica sends PAB-Proof σ back to original replica

- Proxy timeout ➔ Restart from step 1 (Re-sample)
- Optimal d=3

How to determine

1) **whether a replica is busy?**
2) **how much the replica is overloaded?**

Solution 2: DLB(Distributed Load Balancing)

- Workload Estimation: **ST(Stable Time)**
  - Duration from microblock **broadcast** to **stabilization** (Stabilization time – broadcast time)
  - Stabilization : Receiving q ⟨PAB-Ack|m.id⟩

- ST for a replica == N-th percentile of ST values for microblock
  - If ST > $\alpha + \varepsilon$ ➔ busy! ➔ Forward excess load
  - Choose a replica with the lowest ST as a proxy.

Implementation

- **Stratus**
- Prototyped with Bamboo
  - Open-source project for prototyping, evaluating, benchmarking BFT protocols

Dissecting the Performance of Chained-BFT

Fangyu Gai[*], Ali Farahbakhsh[*], Jianyu Niu[*], Chen Feng[*], Ivan Beschastnikh[†], Hao Duan[‡]
University of British Columbia ([*]Okanagan Campus, [†]Vancouver Campus)
[‡]Hangzhou Qulian Technology Co., Ltd.

- PAB proof : concatenation of q ECDSA signatures
  - Computation efficiency

Implementation

- **Testbeds**
  - 4vGPU, 8GB memory, Ubuntu 20.04
  - LAN and WAN simulation
  - LAN
    - Up to 3Gbit/s of bandwidth
    - Inter-replica RTT less than 10 ms
  - WAN
    - Up to 100Mbit/s of bandwidth
    - Inter-replica RTT less than 100 ms

- **Metrics**
  - Latency: Commit time - Receive time
  - Throughput: TPS(Transactions per second)

Implementation

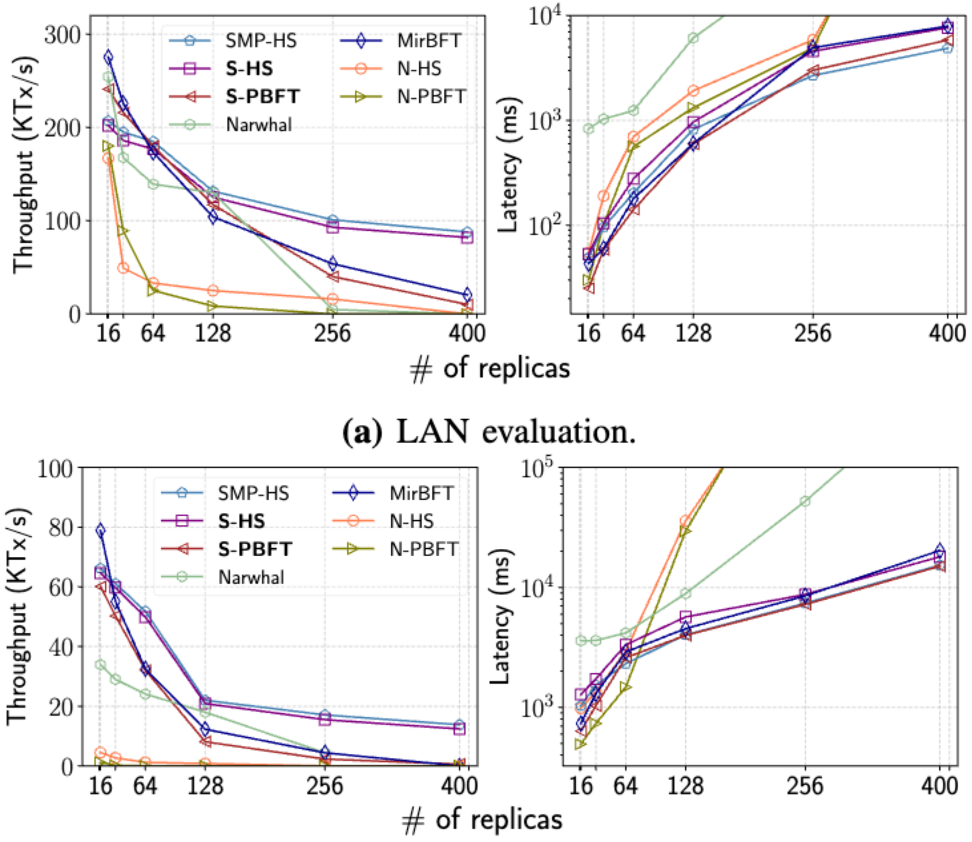
- **Protocols**
  - N- : Native version
  - SMP- : Shared mempool version (w/o PAB, DLB)
  - -G : Gossip version
  - -Even : Even workload
  - S- : Stratus version **(this paper)**

TABLE II: Summary of evaluated protocols.

Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

- SMP-HS (?) vs S-HS ➔ **PAB**
- S-HS-Even(ideal) vs SMP-HS(w/o) vs SMP-HS-G(naïve) vs S-HS(DLB) ➔ **DLB**

Evaluation(1) Scalability



(a) LAN evaluation.

(b) WAN evaluation.

Fig. 5: The throughput (left) and latency (right) of protocols in both LAN and WAN with increasing number of replicas. We use 128-byte payload and 128KB batch size.

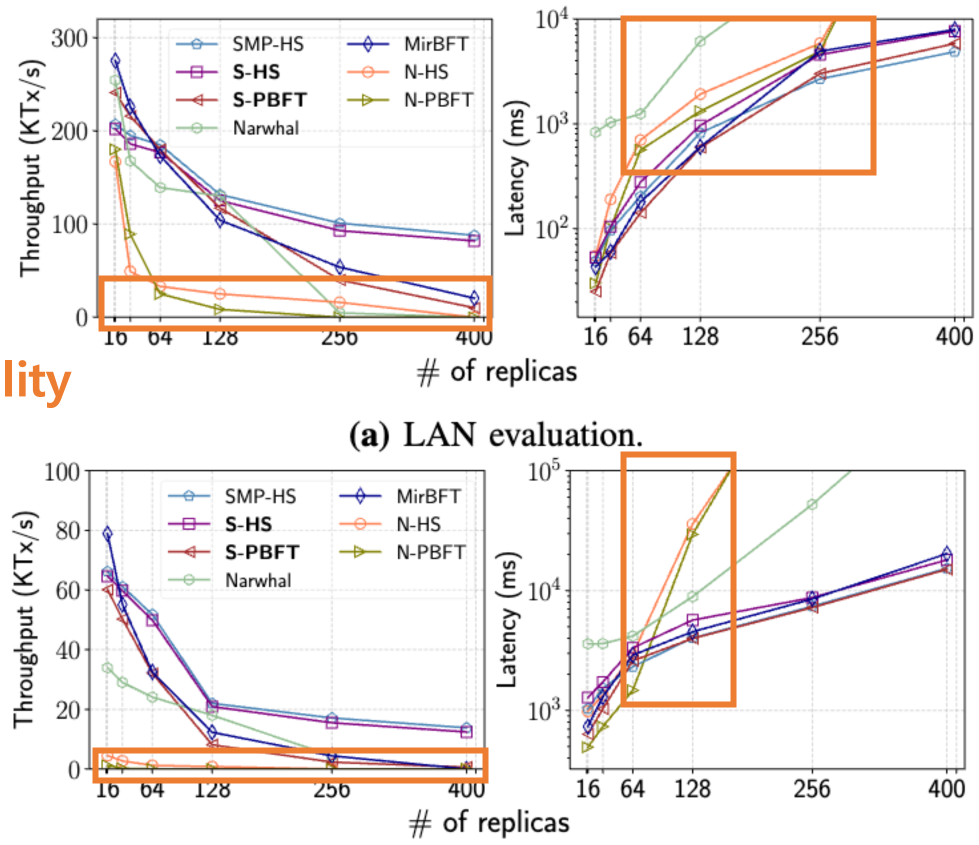
TABLE II: Summary of evaluated protocols.

Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

DCSLAB

Evaluation(1) Scalability

N-HS, N-PBFT lack of scalability



(a) LAN evaluation.

(b) WAN evaluation.

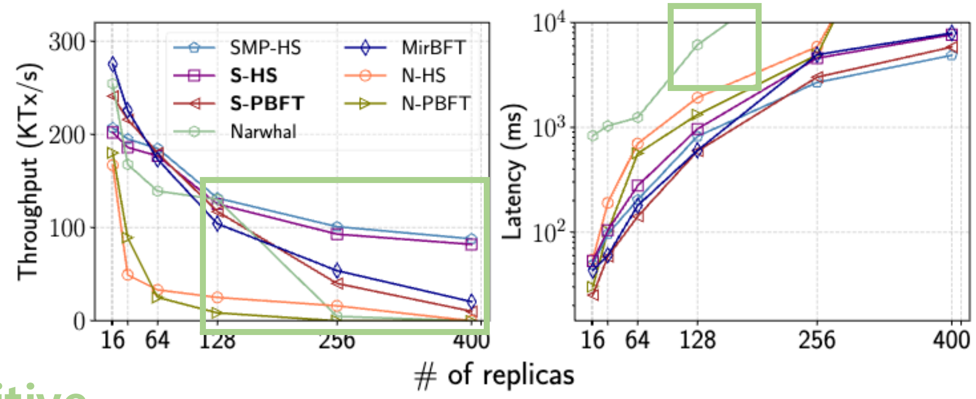
Fig. 5: The throughput (left) and latency (right) of protocols in both LAN and WAN with increasing number of replicas. We use 128-byte payload and 128KB batch size.

TABLE II: Summary of evaluated protocols.

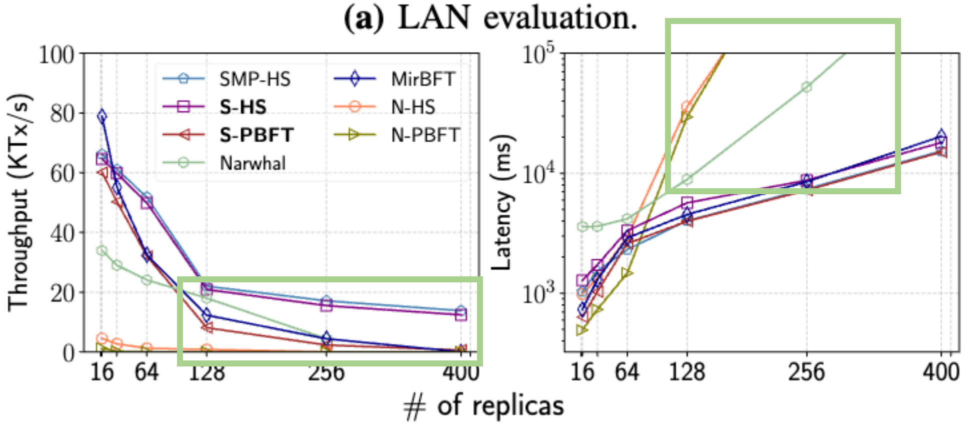
Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

Evaluation(1) Scalability

Narwhal suffers from heavy broadcast primitive



(a) LAN evaluation.



(b) WAN evaluation.

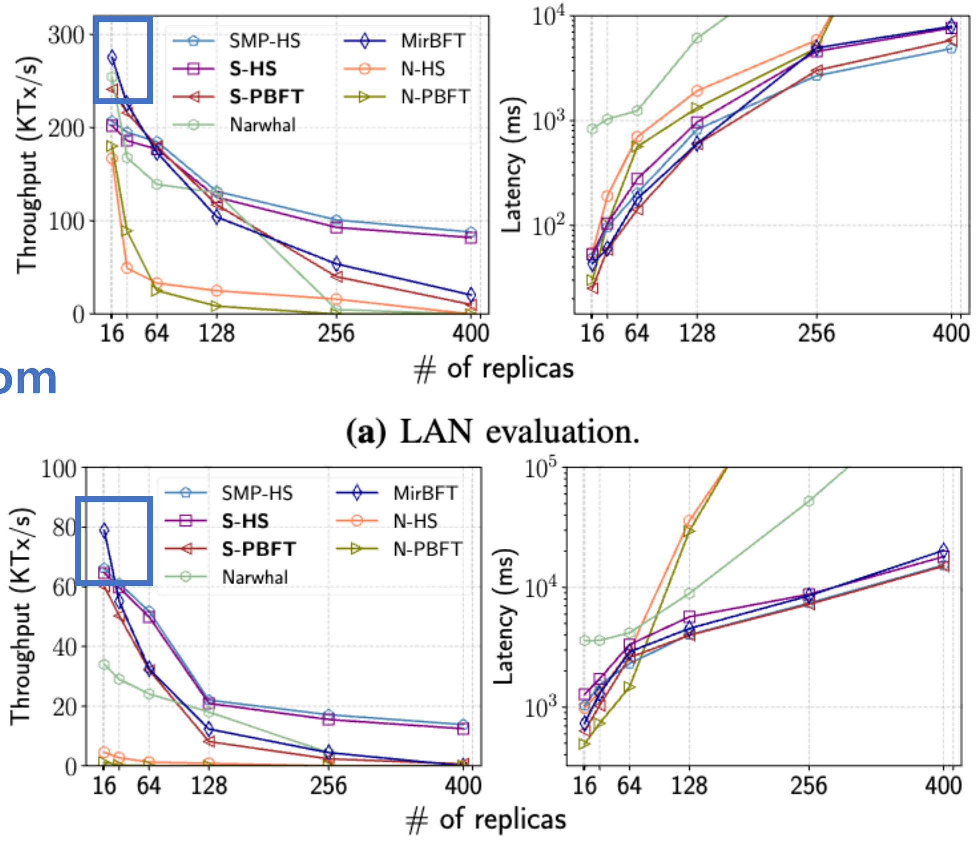
Fig. 5: The throughput (left) and latency (right) of protocols in both LAN and WAN with increasing number of replicas. We use 128-byte payload and 128KB batch size.

TABLE II: Summary of evaluated protocols.

Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

Evaluation(1) Scalability

S-PBFT & MirBFT suffers from higher message complexity



(a) LAN evaluation.

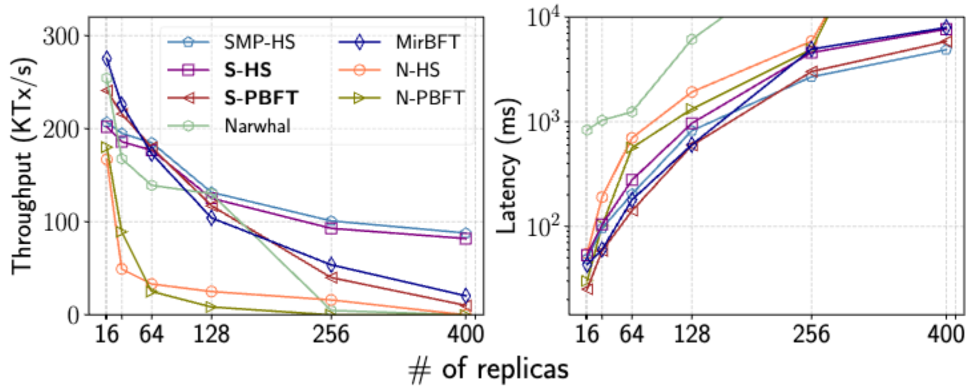
(b) WAN evaluation.

Fig. 5: The throughput (left) and latency (right) of protocols in both LAN and WAN with increasing number of replicas. We use 128-byte payload and 128KB batch size.

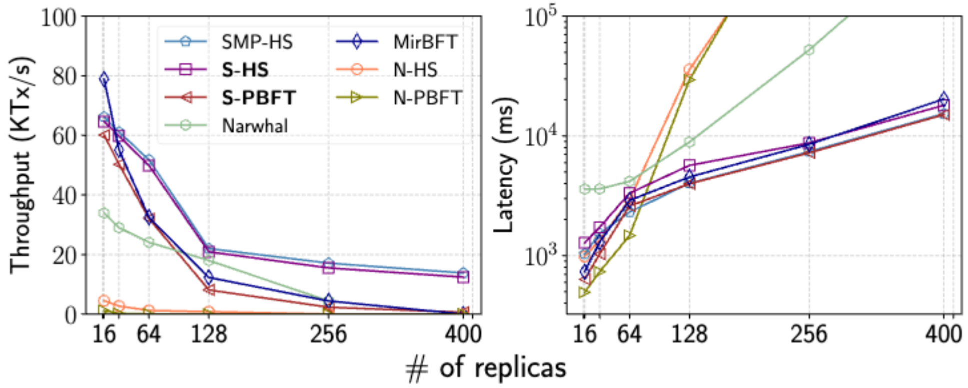
TABLE II: Summary of evaluated protocols.

Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

Evaluation(1) Scalability



(a) LAN evaluation.



(b) WAN evaluation.

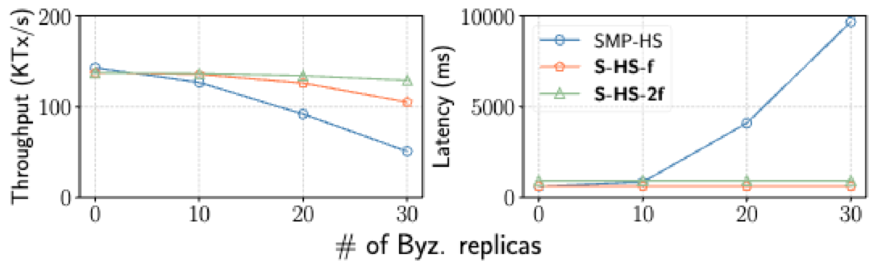
Fig. 5: The throughput (left) and latency (right) of protocols in both LAN and WAN with increasing number of replicas. We use 128-byte payload and 128KB batch size.

TABLE II: Summary of evaluated protocols.

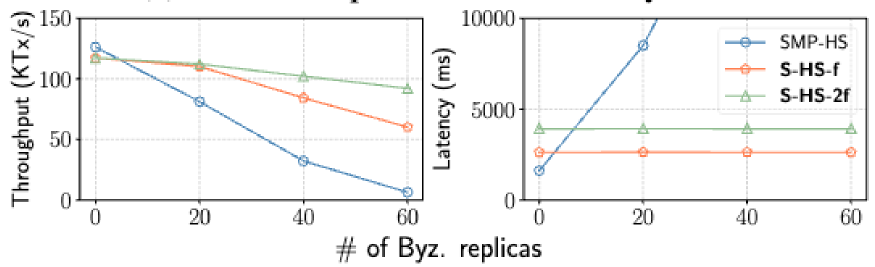
Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

Evaluation(2) Missing transactions (PAB)

- **(1) Byzantine sender scenario**
  - Make missing transactions in leader's proposal
  - SMP-HS
    - Byzantine replicas only send microblocks to the leader
  - S-HS
    - Byzantine replicas send microblocks to the leader and (q-1) replicas



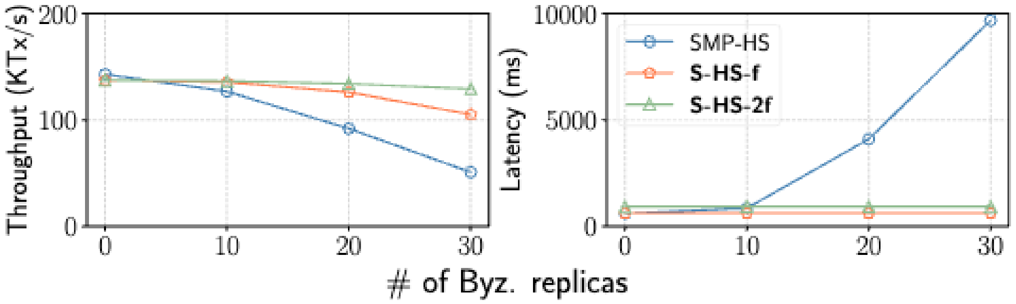
(a) 100 total replicas with 0 to 30 Byz. ones.



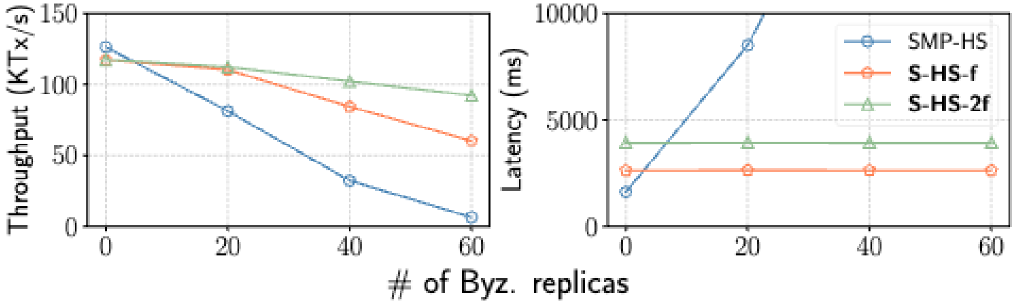
(b) 200 total replicas with 0 to 60 Byz. ones.

Fig. 7: Performance of SMP-HS and S-HS with different quorum parameters (S-HS-d1 and S-HS-d2) and increasing Byzantine replicas.

Evaluation(2) Missing transactions (PAB)



(a) 100 total replicas with 0 to 30 Byz. ones.



(b) 200 total replicas with 0 to 60 Byz. ones.

Fig. 7: Performance of SMP-HS and S-HS with different quorum parameters (S-HS-d1 and S-HS-d2) and increasing Byzantine replicas.

TABLE II: Summary of evaluated protocols.

Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

Evaluation(2) Missing transactions (PAB)

- **(2) Network asynchrony**
  - A proposal is likely to arrive before referenced transactions
  - WAN
  - Network fluctuation via NetEm (for 10s, between 100ms and 300ms)

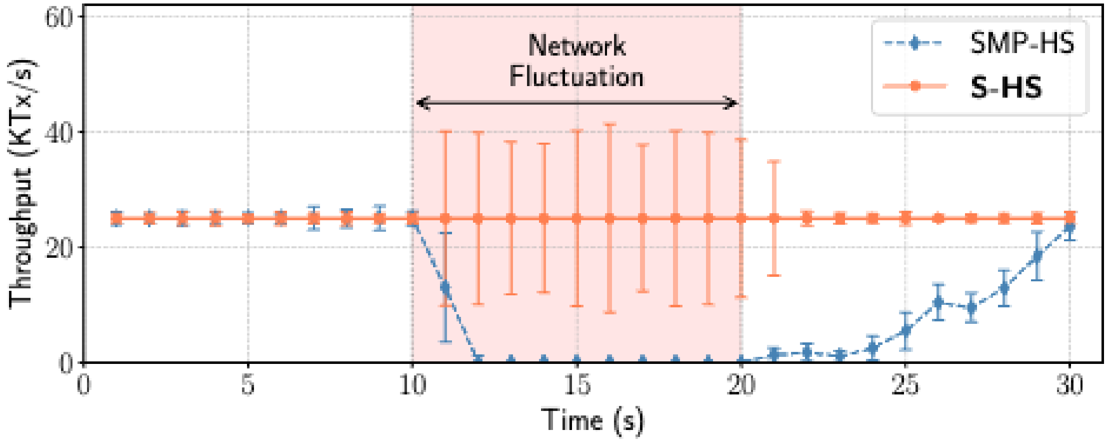


Fig. 6: Delay is injected at time 10 s and lasts for 10 s. The transaction rate is 25KTx/s. Each point is averaged over 10 runs.

TABLE II: Summary of evaluated protocols.

Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

DCSLAB

Evaluation(3) Unbalanced Workload (DLB)

- Zipfian parameter
- d: Sampling parameter
  - d=3 is the optimal

TABLE II: Summary of evaluated protocols.

Acronym	Protocol description
N-HS	Native HotStuff without a shared mempool
N-PBFT	Native PBFT without a shared mempool
SMP-HS	HotStuff integrated with a simple shared mempool
SMP-HS-G	SMP-HS with gossip instead of broadcast
SMP-HS-Even	SMP-HS with an even workload across replicas
S-HS	HotStuff integrated with **Stratus (this paper)**
S-PBFT	PBFT integrated with **Stratus (this paper)**
Narwhal	HotStuff based shared mempool
MirBFT	PBFT based multi-leader protocol

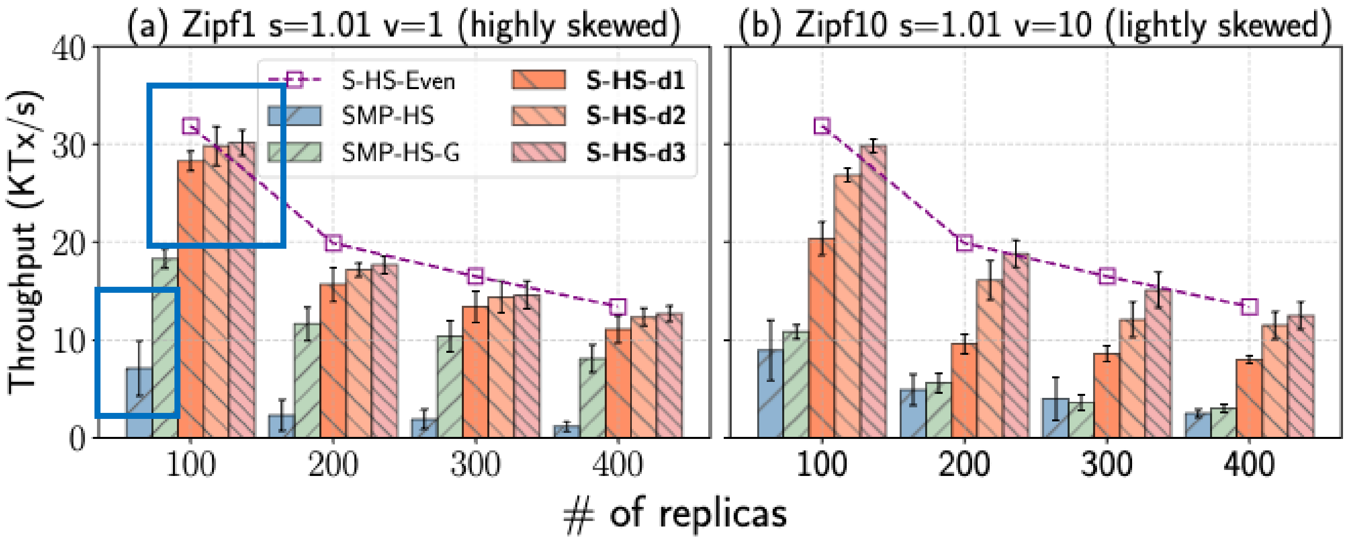


Fig. 9: Throughput with different workload distribution.

Conclusion & Future work

- SMP(Shared Mempool Abstraction) resolves the leader bottleneck.
- Stratus is a novel SMP designed to
  - Address missing tx
  - Handle unbalanced workloads
- S-HS 5x to 20x higher throughput compared to N-HS

- Future work
  - Extend Stratus to support multi-leader BFT protocols