HotStuff-1: Linear Consensus with One-Phase Speculation

Dakai Kang

Suyash Gupta*

Dahlia Malkhi†

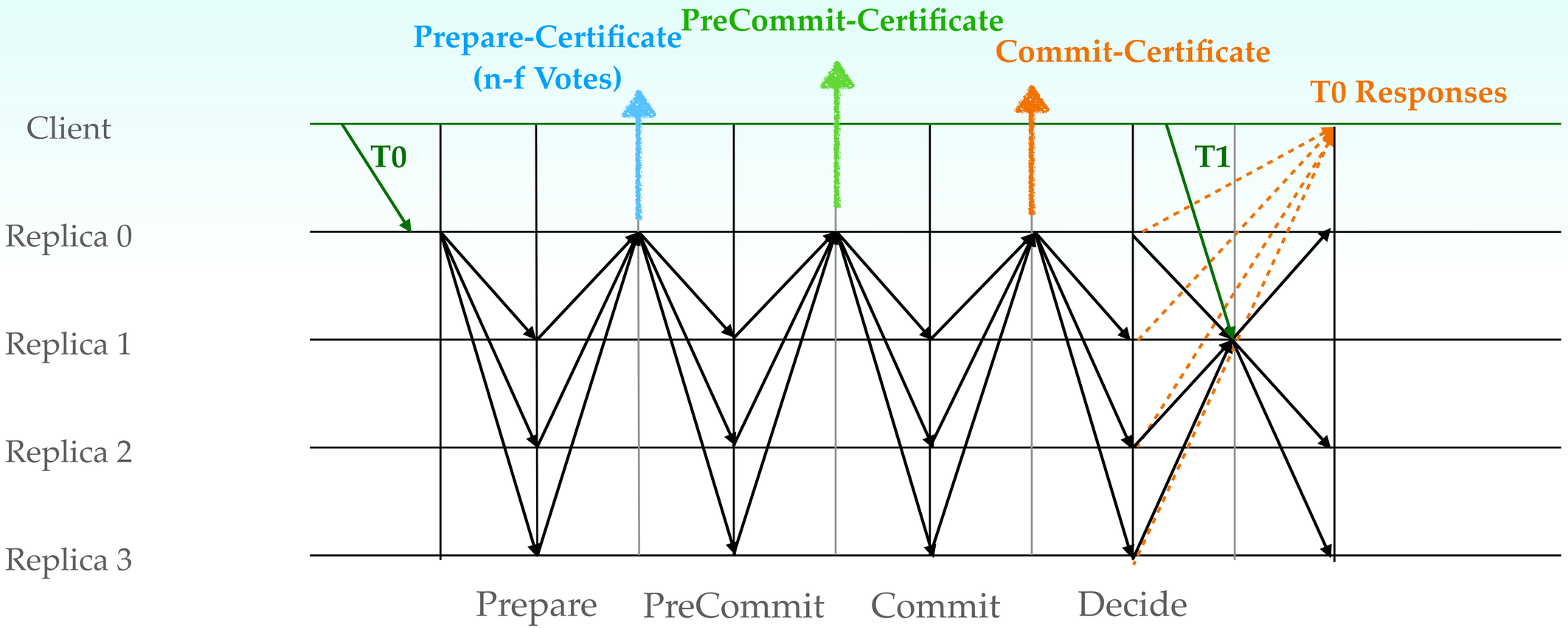
Mohammad Sadoghi

Department of Computer Science, UC Davis

* Distopia Labs, Department of Computer Science, University of Oregon

†Department of Computer Science, UC Santa Barbara

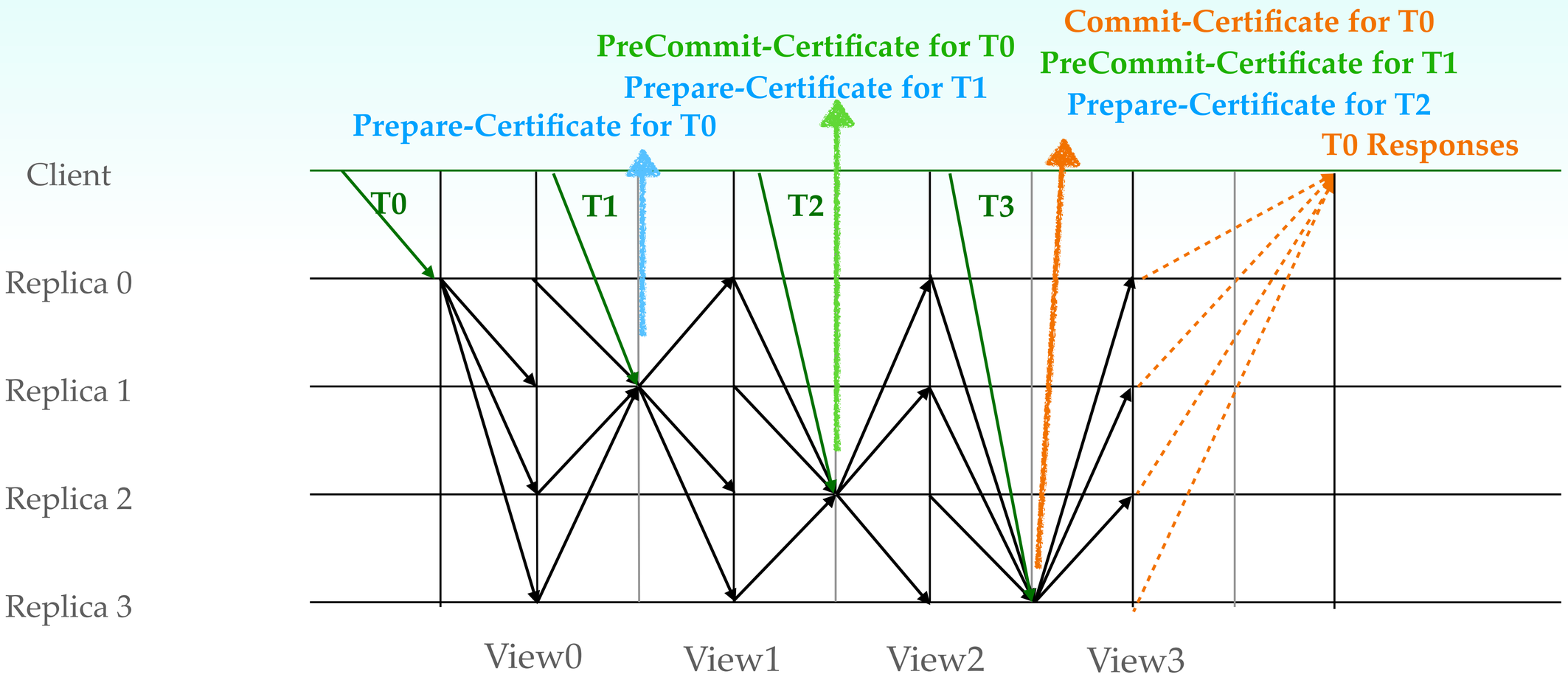
HotStuff: Rotational Consensus



Basic HotStuff: 3.5-Phase BFT Protocol with **Leader Rotation** and **Linear Complexity**

HotStuff with **n** replicas can tolerate at most **f Byzantine** replicas**, if n > 3f**

Streamlining: Boost Throughput



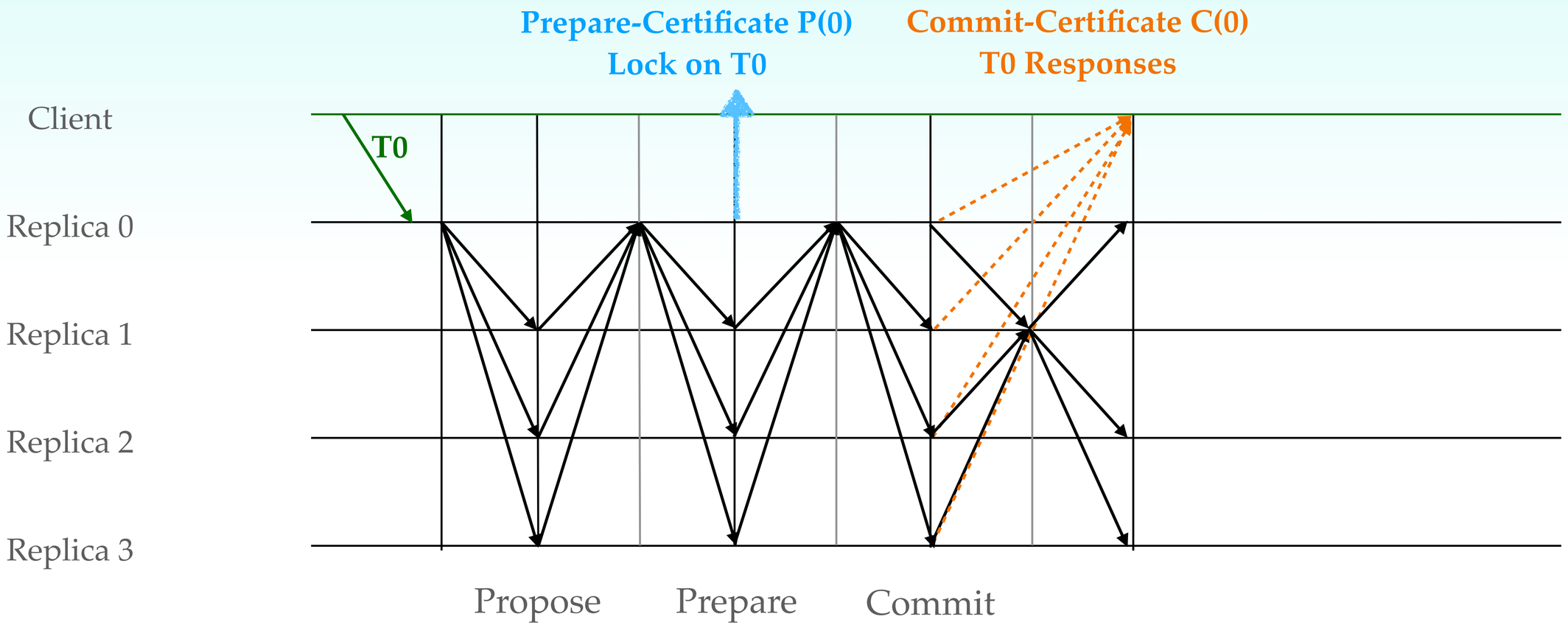
Streamlined HotStuff: Overlapping phases of multiple views

Challenges in HotStuff

High Latency

- **HotStuff**: 4 Broadcast Steps + 3 Vote Steps ==> 7Δ
- **PBFT**: 1 Broadcast Step + 2 All-to-All Vote Steps ==> 3Δ
- **Challenge**: Can we reduce HotStuff's latency?

HotStuff-2: Two-phase HotStuff



HotStuff-2: Removed PreCommit Phase from HotStuff; 3 Broadcast + 2 Vote ==> 5Δ

Challenges in HotStuff

High Latency

- **HotStuff**: 4 Broadcast Steps + 3 Vote Steps ==> 7Δ
- **PBFT**: 1 Broadcast Step + 2 All-to-All Steps ==> 3Δ
- **HotStuff-2**: 3 Broadcast Steps + 2 Vote Steps ==> 5Δ

Challenges in HotStuff

High Latency

- **HotStuff**: 4 Broadcast Steps + 3 Vote Steps ==> 7Δ
- **PBFT**: 1 Broadcast Step + 2 All-to-All Steps ==> 3Δ
- **HotStuff-2**: 3 Broadcast Steps + 2 Steps ==> 5Δ

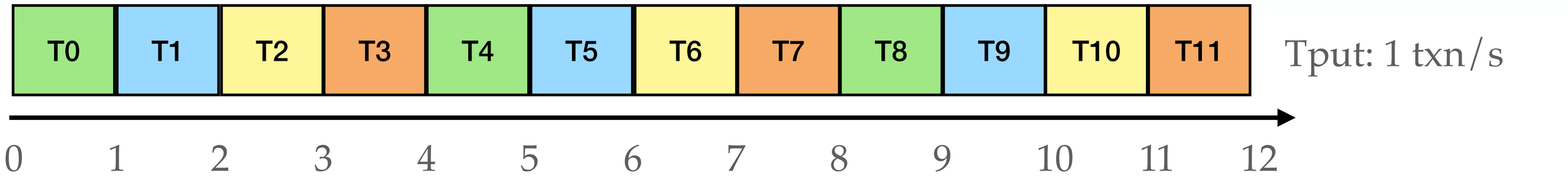
Leader Slowness

- Leader may have **low computational capacity**.
- Leader may target **Maximal Extractable Value (MEV)**.
  - Low Throughput!

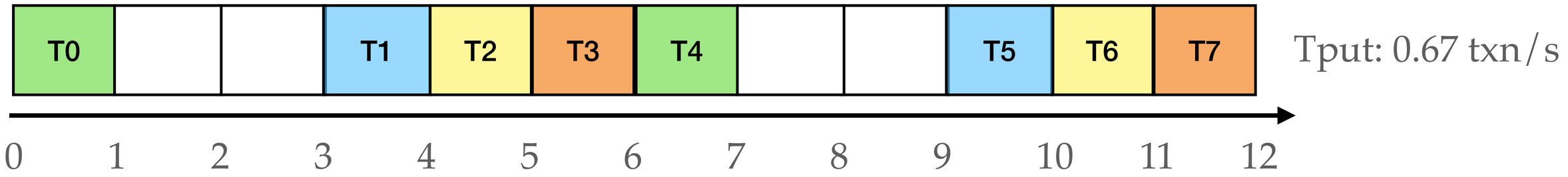
Leader Slowness

- **Timer Length: 3s**
- **Average View Duration for Honest Leaders: 1s**

All Leaders are well-behaving



R1 is a Slow Leader



Challenges in HotStuff

High Latency

- **HotStuff**: 4 Broadcast Steps + 3 Vote Steps ==> 7Δ
- **PBFT**: 1 Broadcast Step + 2 All-to-All Steps ==> 3Δ
- **HotStuff-2**: 3 Broadcast Steps + 2 Vote Steps ==> 5Δ

Leader Slowness

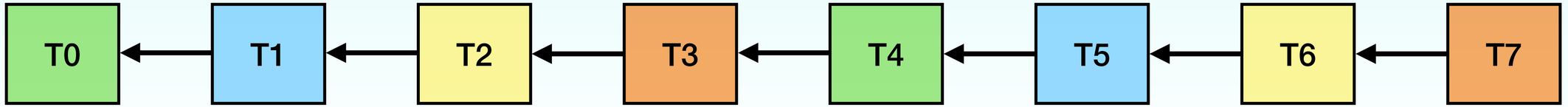
- Leader may have **low computational capacity**.
- Leader may target **Maximal Extractable Value (MEV)**.
  - Low Throughput!

Tail-Forking Attack in Streamlined Version

- Proposals from honest leaders are ignored by the subsequent *malicious* leaders.
  - Low Throughput!
  - Unfair to victim honest leaders.

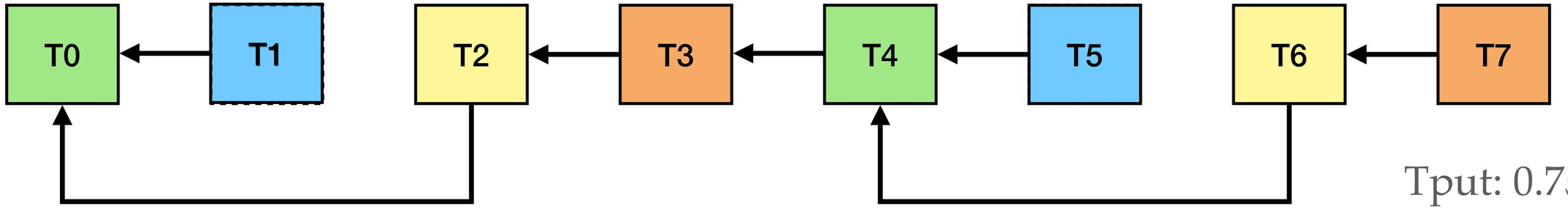
Tail-Forking Attack

All Leaders are well-behaving



Tput: 1 txn/s

R2 is Byzantine: Blocks from R1 are intentionally ignored by R2



Tput: 0.75 txn/s

Magic Sauce for Resolving the Challenges

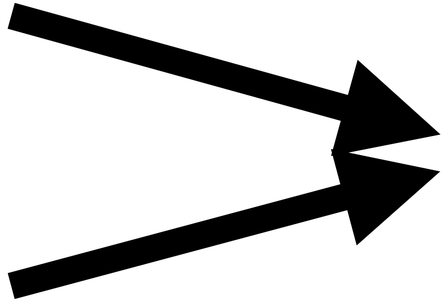
High Latency

Speculation

- Allow replicas to speculatively execute transactions.

Leader Slowness

Tail-Forking Attack



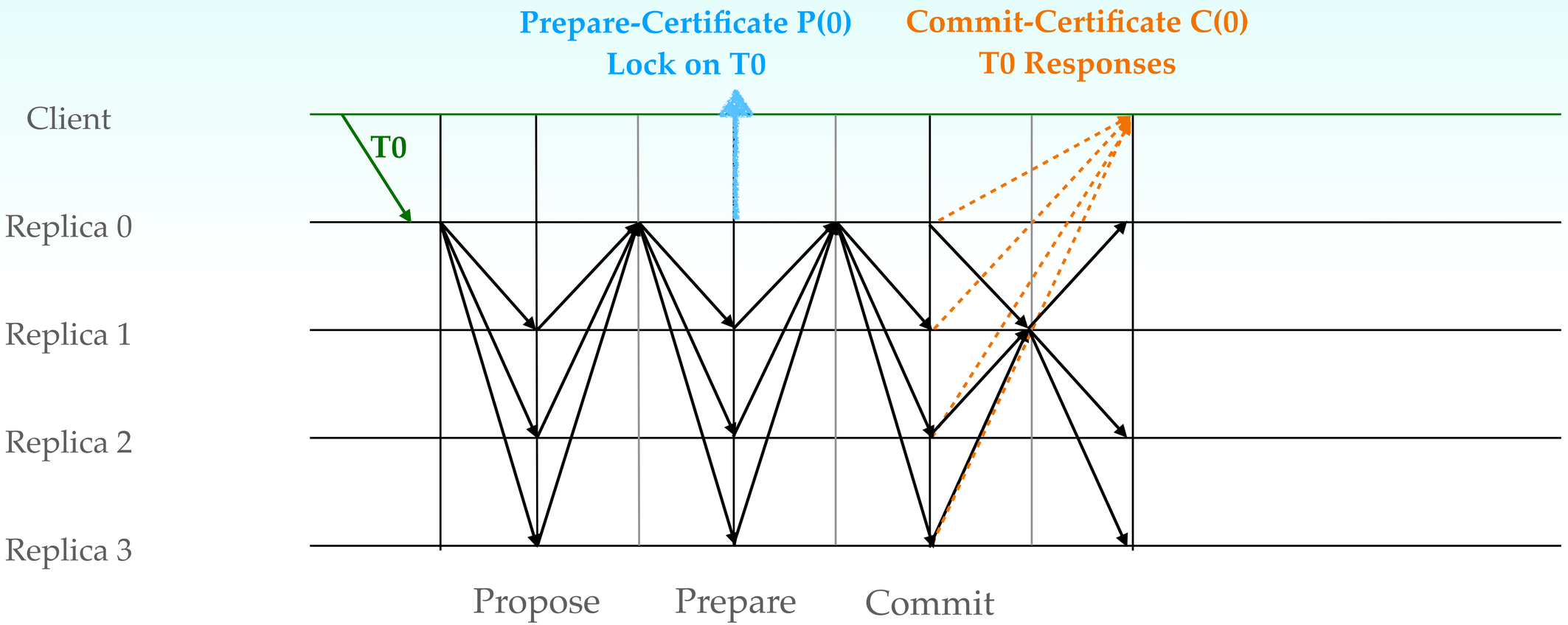
Slotting

- Allow leader to propose multiple slots; one transaction block per slot.

Introducing HotStuff-1 (HS-1)

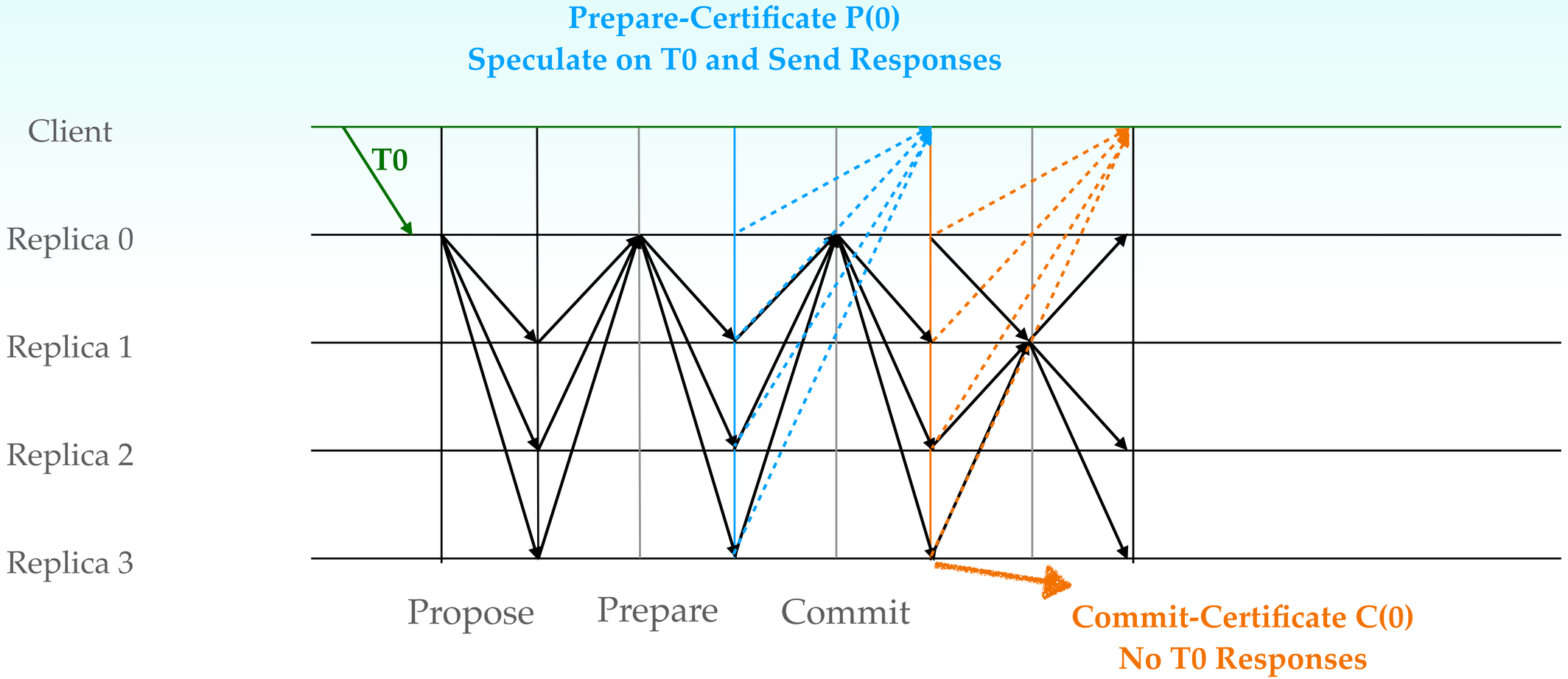
- Ensures **Linear Complexity** as HotStuff/HotStuff-2.
- **First** HotStuff Family Protocol to employ **Speculation.**
- **Resilient** against Leader Slowness via **Slotting.**
- **Resilient** against Tail-Forking via **Slotting.**

HotStuff-2: Two-phase HotStuff



HotStuff-2: Removed PreCommit Phase from HotStuff; 3 Broadcast + 2 Vote ==> 5Δ

HotStuff-1: Linear Consensus with One-Phase Speculation



Basic HotStuff-1: 2 Broadcast + 1 Vote ==> 3Δ

Speculation is not Simple!

(1) Speculative responses **are not** commit-votes but prepare-votes.

- 2f+1 speculative responses are required, rather than f+1 as in prior works.

(2) **Prefix Speculation Dilemma:**

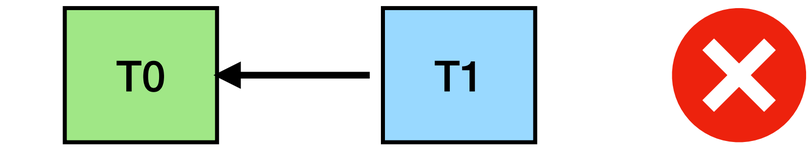
- After sending a speculative response for T, a replica **may switch** to a conflicting branch.
- **Invalidates** previous speculative response!
- Client may **mistakenly collect** 2f+1 invalid speculative responses.

Speculation Rules

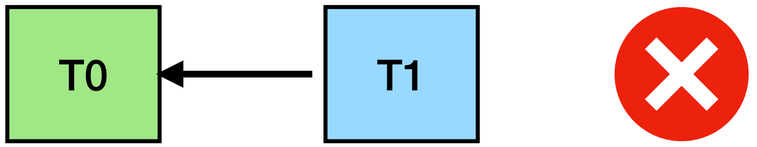
Proof and Detailed Examples in our Extended Report:
https://arxiv.org/pdf/2408.04728

Two rules for a replica R to speculate on a transaction T proposed in view v:

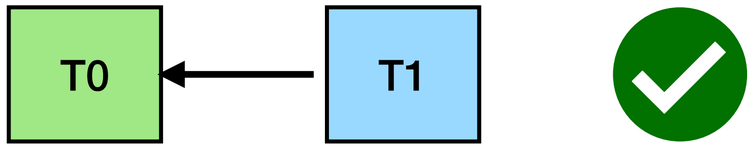
- **Prefix Speculation Rule:** The prefix of T must have been committed;
- **No-Gap Rule:** R receives the prepare-certificate of T in view v



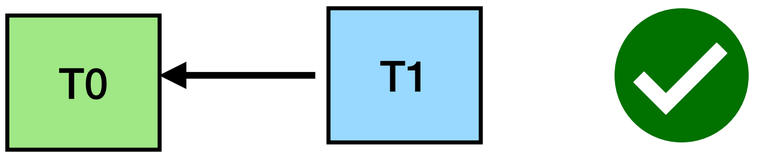
Not committed Receive P(1)



Committed Receive P(1) from **Propose** message in view **2**



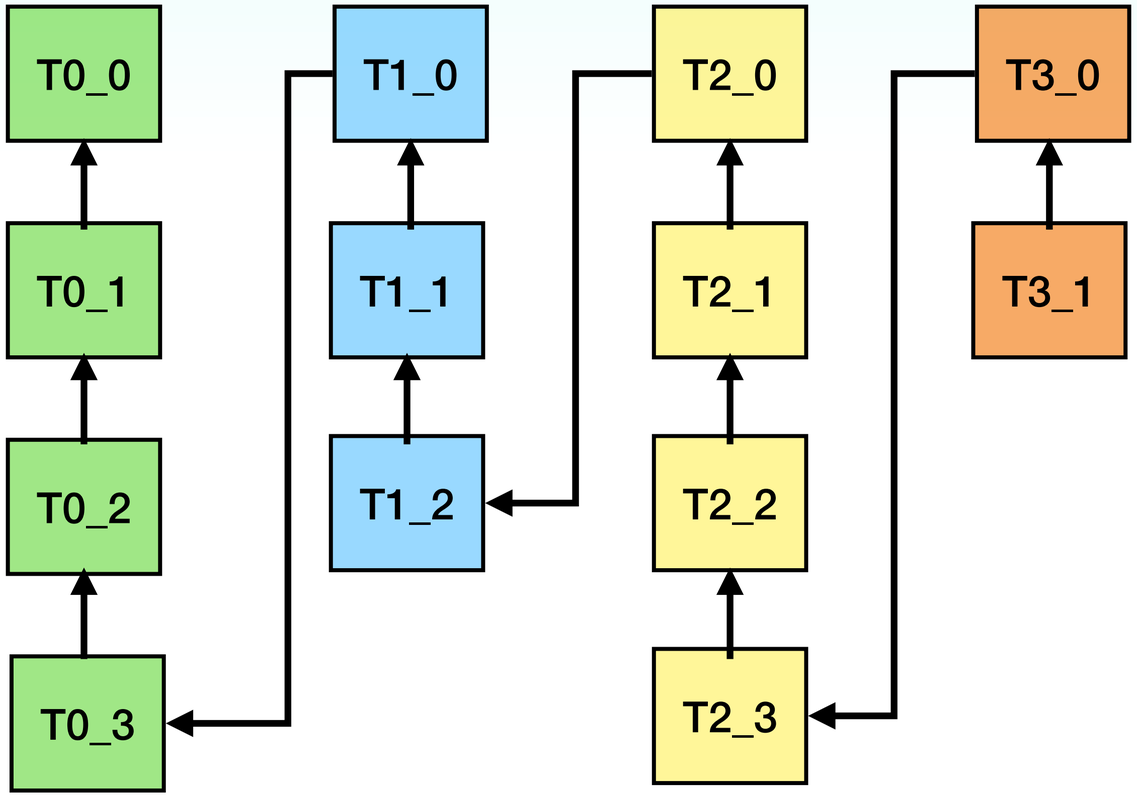
Committed Receive P(1)



Committed Receive P(1) from **Prepare** message in view **1**

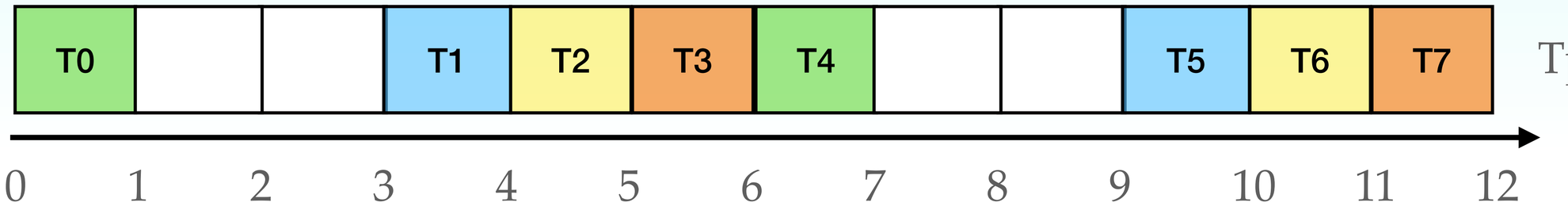
Slotting

- We enable a fast leader to propose as many transactions as possible before timeout
- Tv_s must contain certificate P(v, s-1)



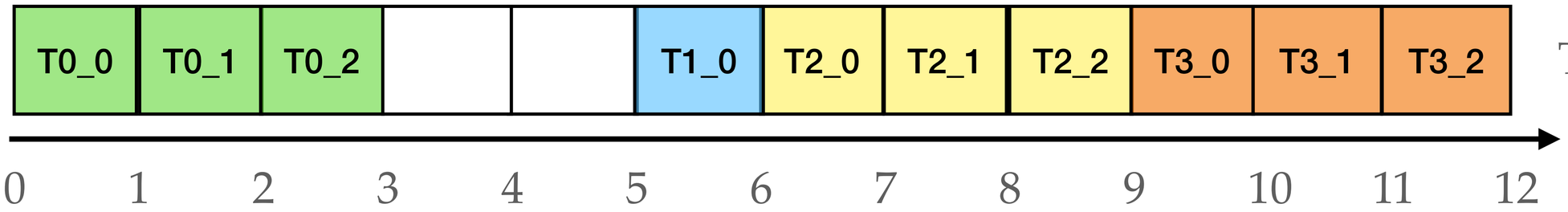
Slotting - Leader Slowness

Without Slotting



Tput: 0.67 txn/s

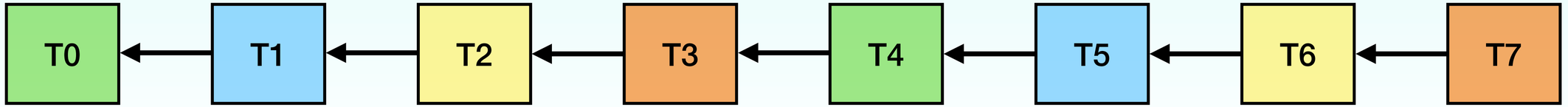
With Slotting



Tput: 0.83 txn/s

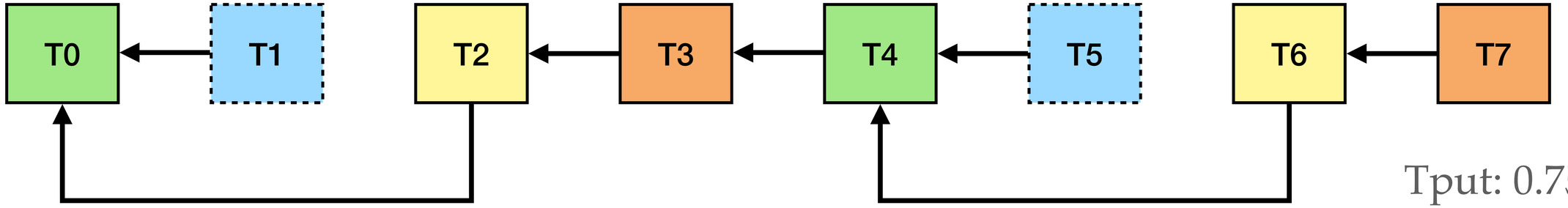
Slotting - Tail-Forking Attack

All Leaders are well-behaving



Tput: 1 txn/s

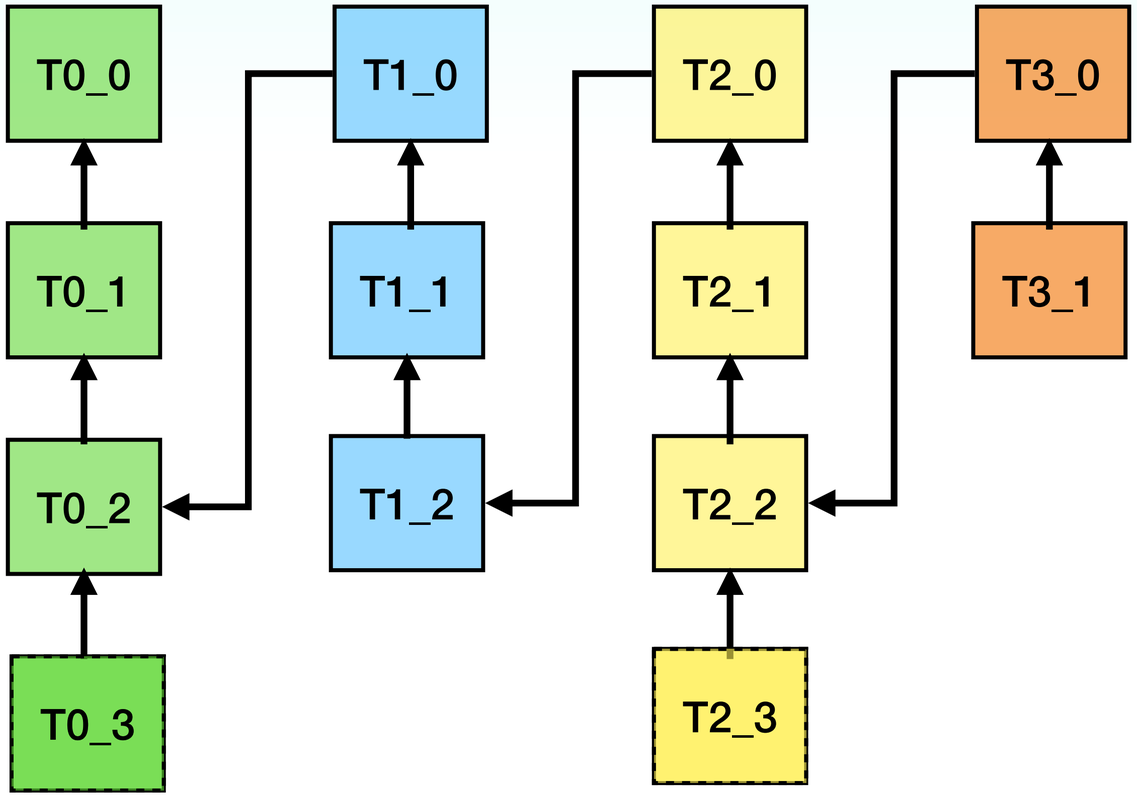
R2 is Byzantine: Transactions from R1 are all skipped



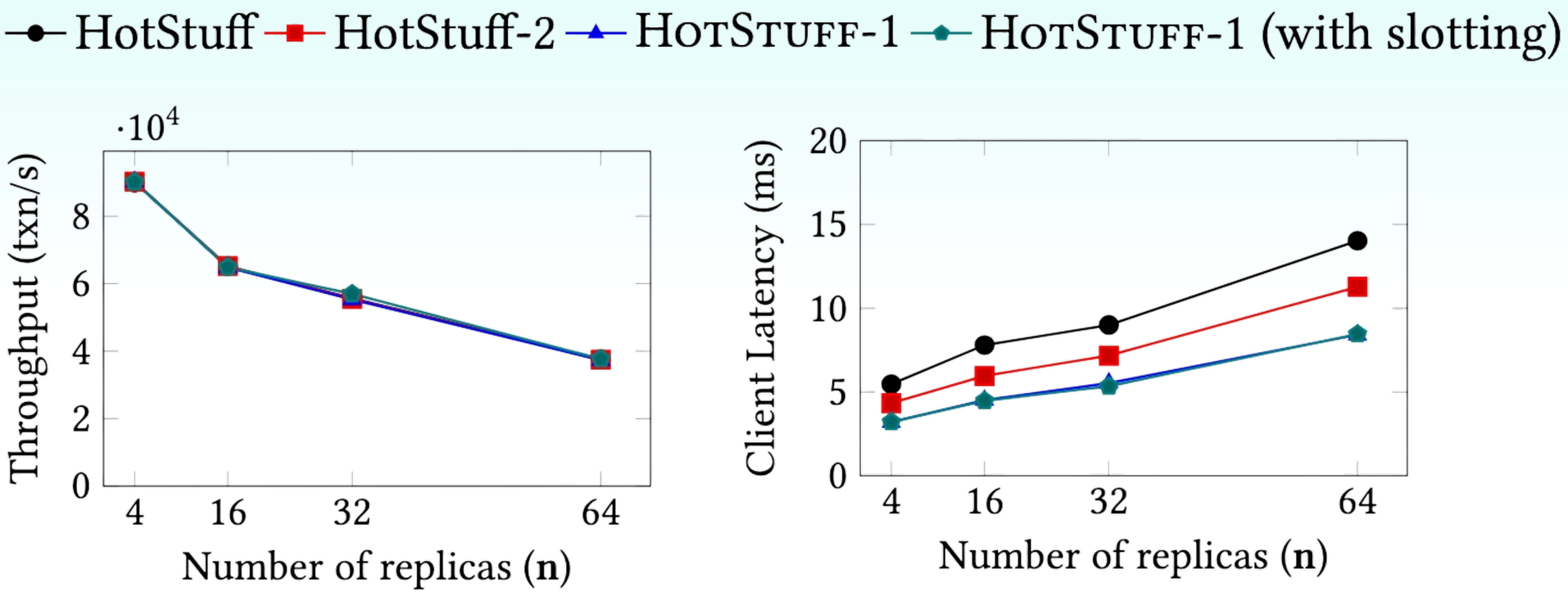
Tput: 0.75 txn/s

Tail-Fork at most 1 slot in HotStuff-1

- A certificate P(v,s) means at least f+1 honest replicas lock on P(v,s-1)
- At most one slot can be skipped per view

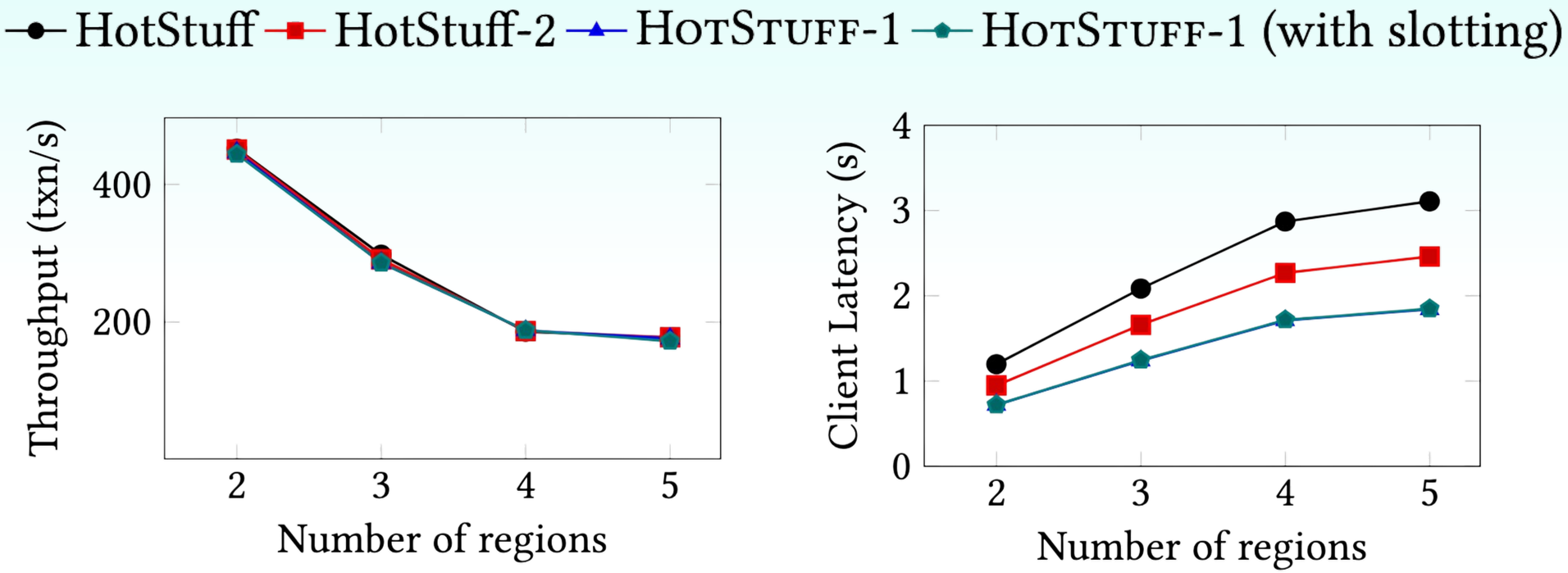


Evaluation - Scalability



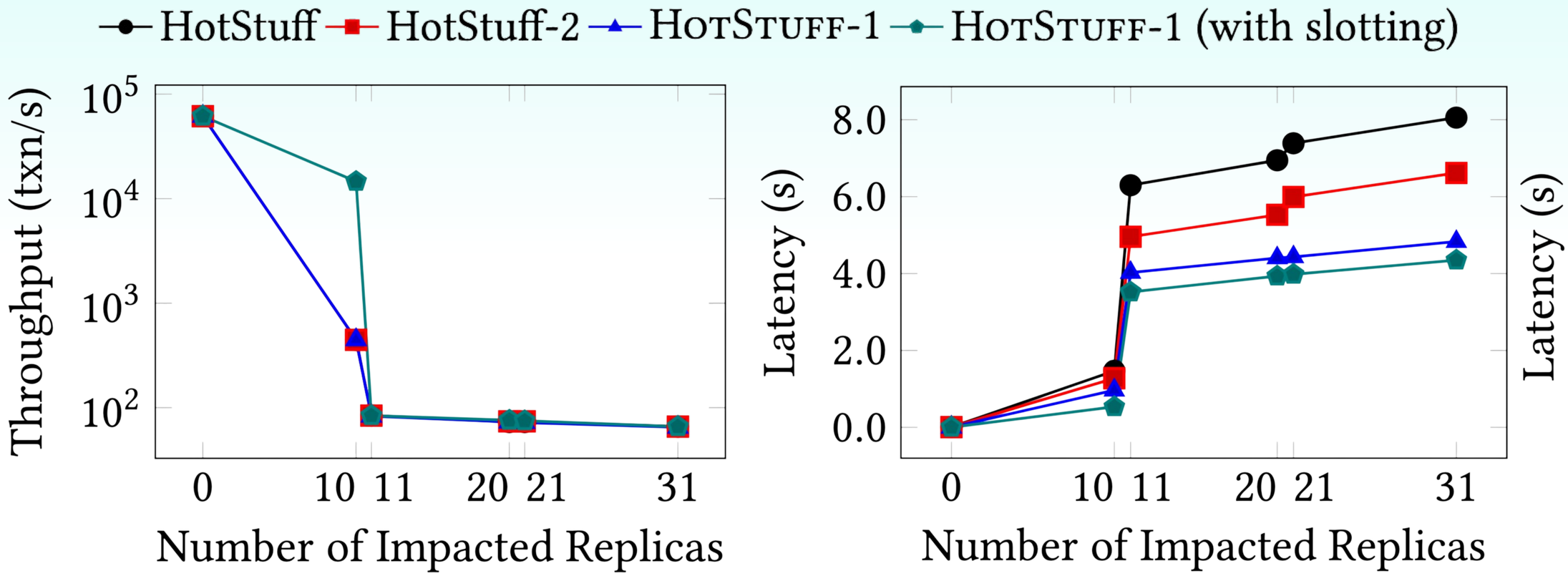
HotStuff-1 exhibits the **same throughput** and **lower latency** than other protocols.

Evaluation - Geographical Scalability



HotStuff-1 exhibits the **same throughput** and **lower latency** than other protocols.

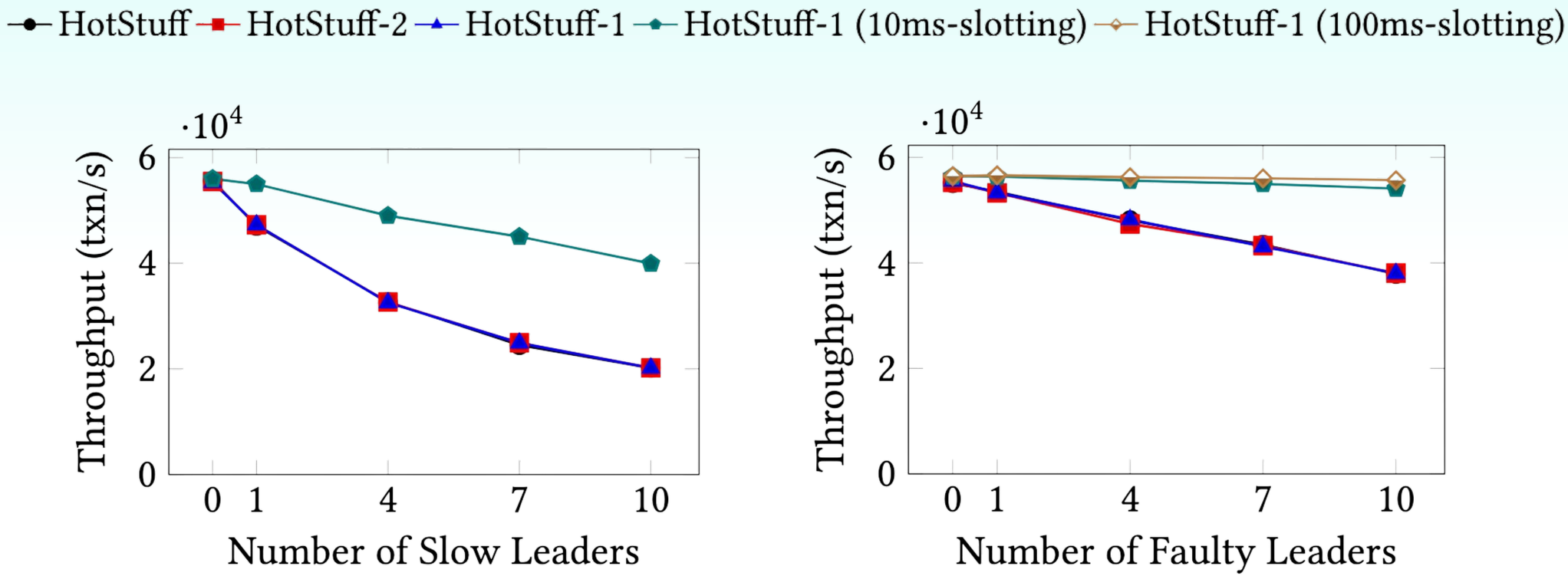
Evaluation - Network Jitters



Performance under unfavorable network condition with **500ms** injected Network delay.

HotStuff-1 with slotting exhibits the **highest resilience** among all protocols.

Evaluation - Attacks



HotStuff-1 with slotting exhibits the **highest resilience** against the attacks.

Conclusion

- **HotStuff-1 employs speculative execution to offer consensus in a single phase.**

- **HotStuff-1 with slotting guards against Leader Slowness and Tail-Forking Attacks.**

UC DAVIS UNIVERSITY OF OREGON UC SANTA BARBARA

Apache ResilientDB Incubating

THANK YOU

Apache ResilientDB Incubating

https://resilientdb.com/

ExpoLab Creativity Unfolded

FifTech Lab ucsb