Brief Announcement: Revisiting Consensus Protocols through Wait-Free Parallelization

Suyash Gupta *Jelle Hellings* Mohammad Sadoghi

Exploratory Systems Lab,
Department of Computer Science,
University of California, Davis, CA, USA

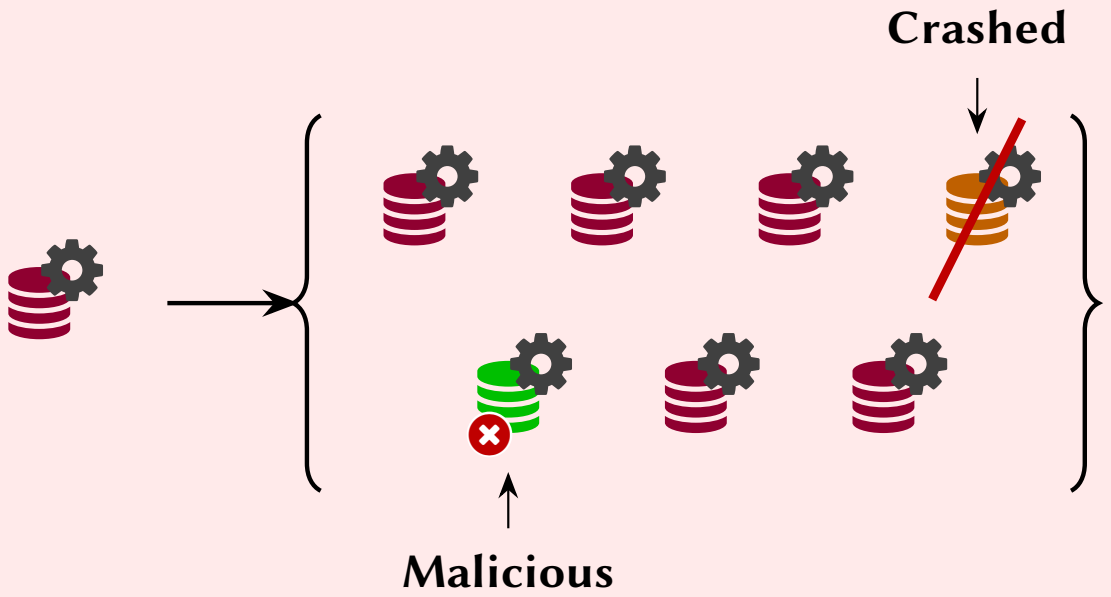
Vision: resilient data processing

Goal

Harden database systems: *crashes* and *malicious* attacks.

Solution

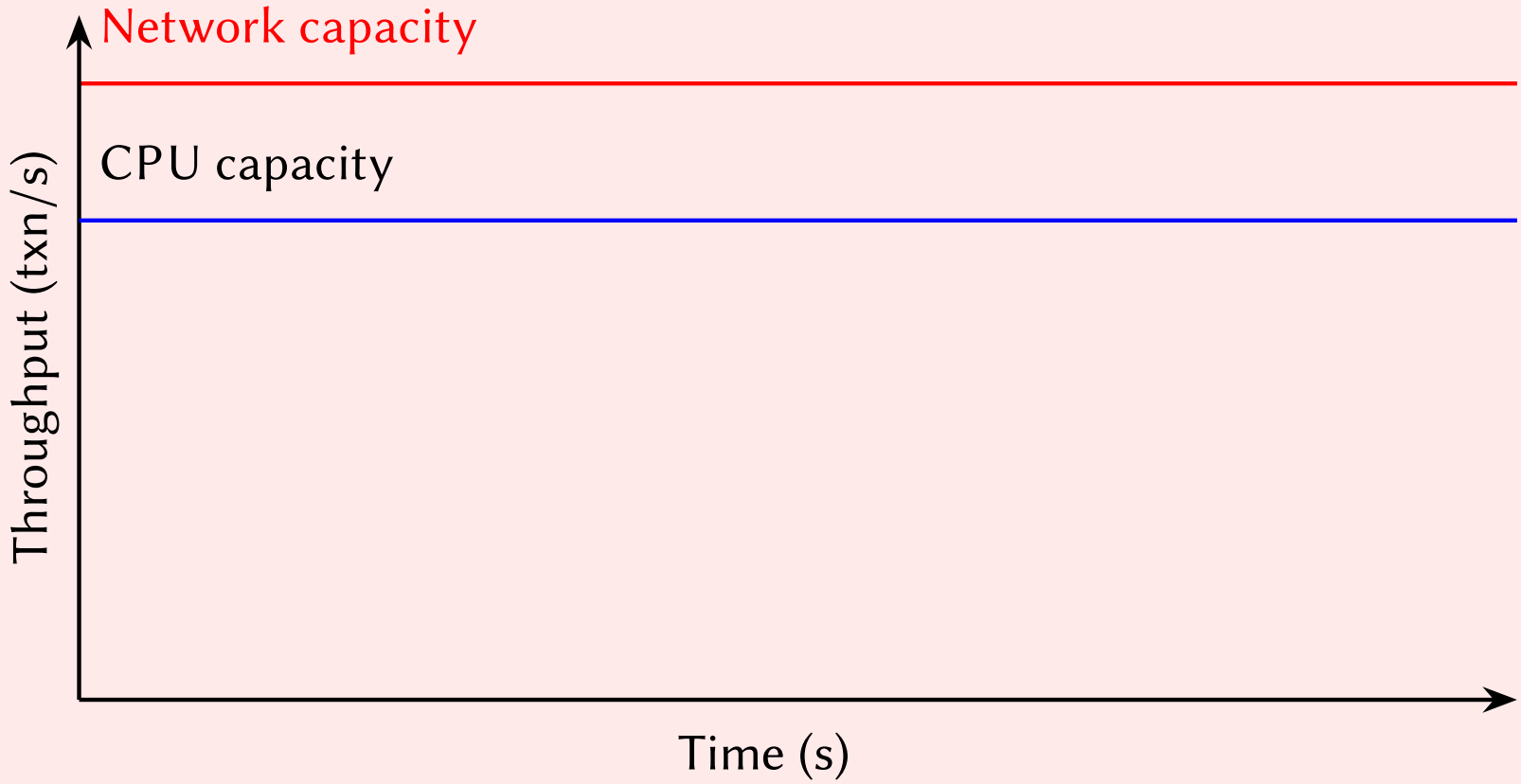
Use Byzantine fault-tolerant replication in database systems.



High-performance BFT: via *primary-backup consensus protocols*.

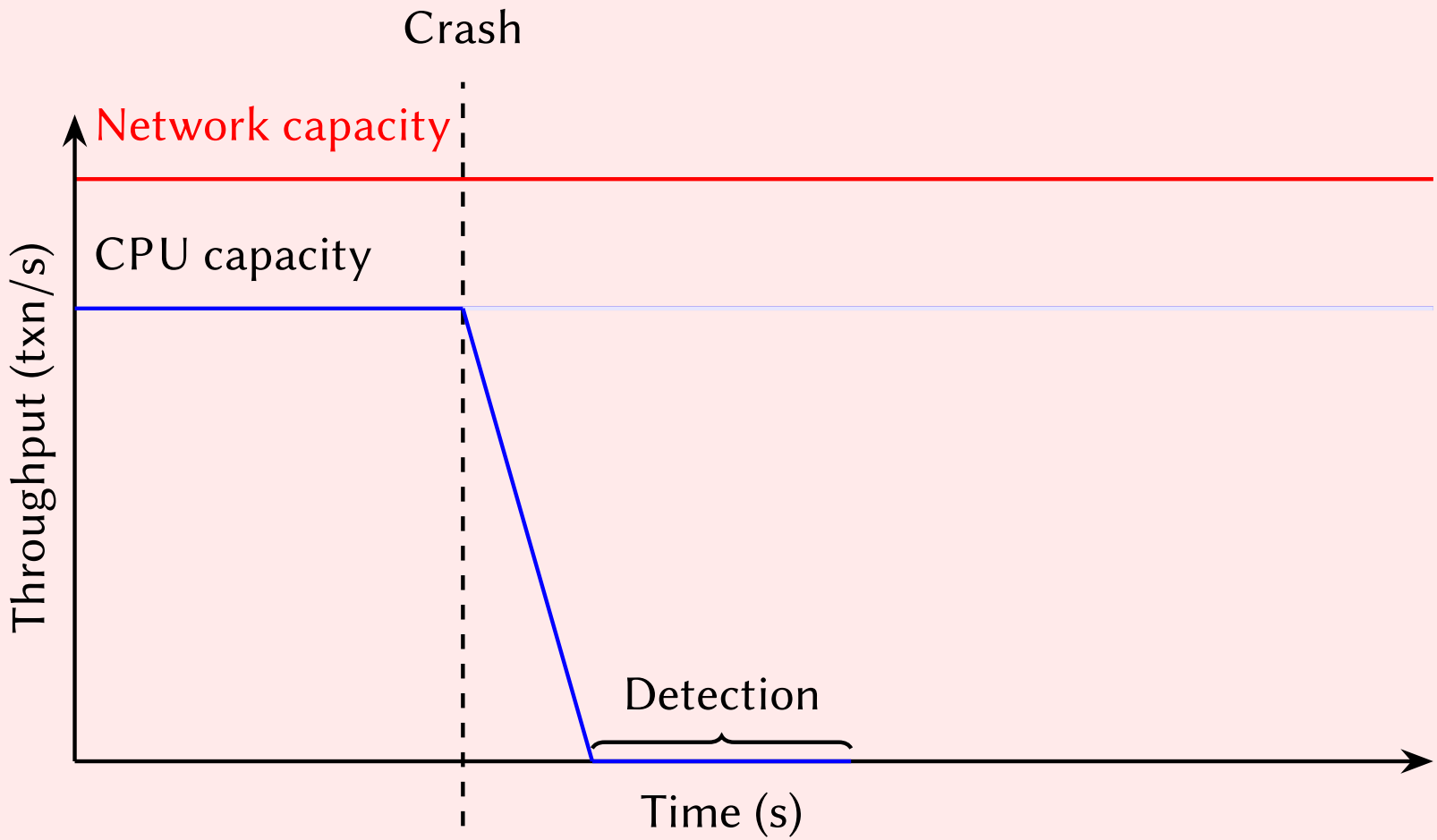
The need for wait-free consensus

Primary-backup consensus protocols have *weaknesses*!



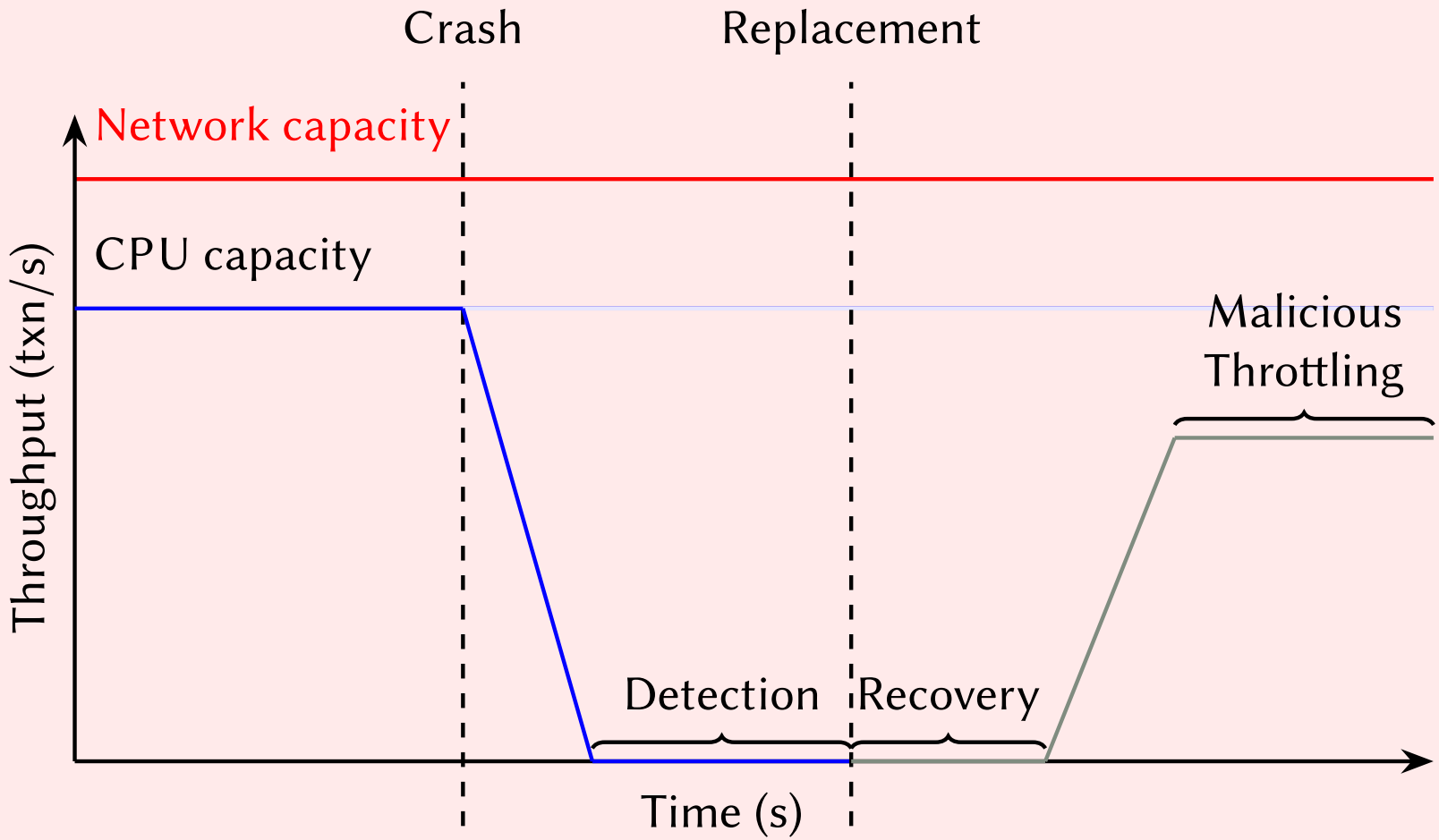
The need for wait-free consensus

Primary-backup consensus protocols have *weaknesses*!



The need for wait-free consensus

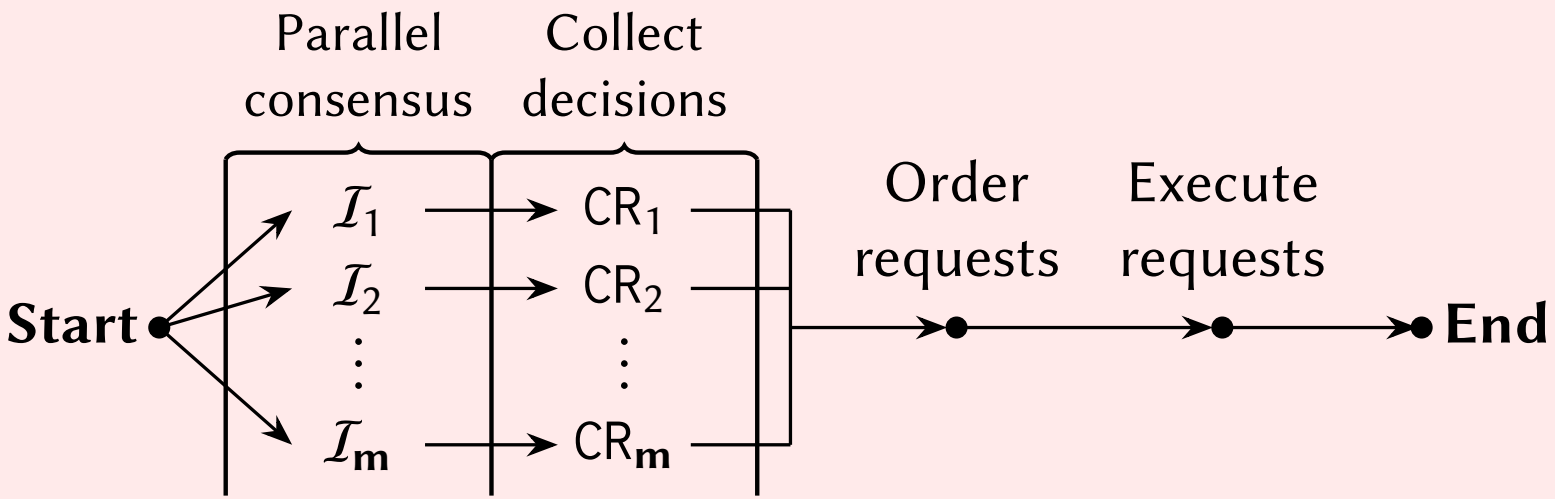
Primary-backup consensus protocols have *weaknesses*!



Solution: parallelize consensus

Basic idea

Run **m** > **f** instances of a BFT protocol P in parallel!



Challenges

- Dealing with *faulty* instances.
- Eliminate influence of *faulty* instances on *other* instances.
- New *attack vectors* due to parallelization?

Core techniques employed

- Consensus does not need coordination!
  Only execution needs coordination between instances.
- Order requests based on the decided requests.
  Not predictable, very hard to influence.
- Primary replacement at instance level.
  Only affects failed instances.
- Clients assigned to instances to prevent duplication.

Theorem (Wait-free parallelization of consensus)

Non-faulty instances can continuously order requests independently.

Conclusion

More information

https://jhellings.nl

Paper: https://doi.org/10.4230/LIPIcs.DISC.2019.44.

Technical Report: https://arxiv.org/abs/1908.01458.