UC Santa Barbara
Computer Science Department

CAPER: A Cross-Application Permissioned Blockchain

Mohammad Javad Amiri, Divyakant Agrawal, Amr El Abbadi

The 45th International Conference on Very Large Data Bases (VLDB)

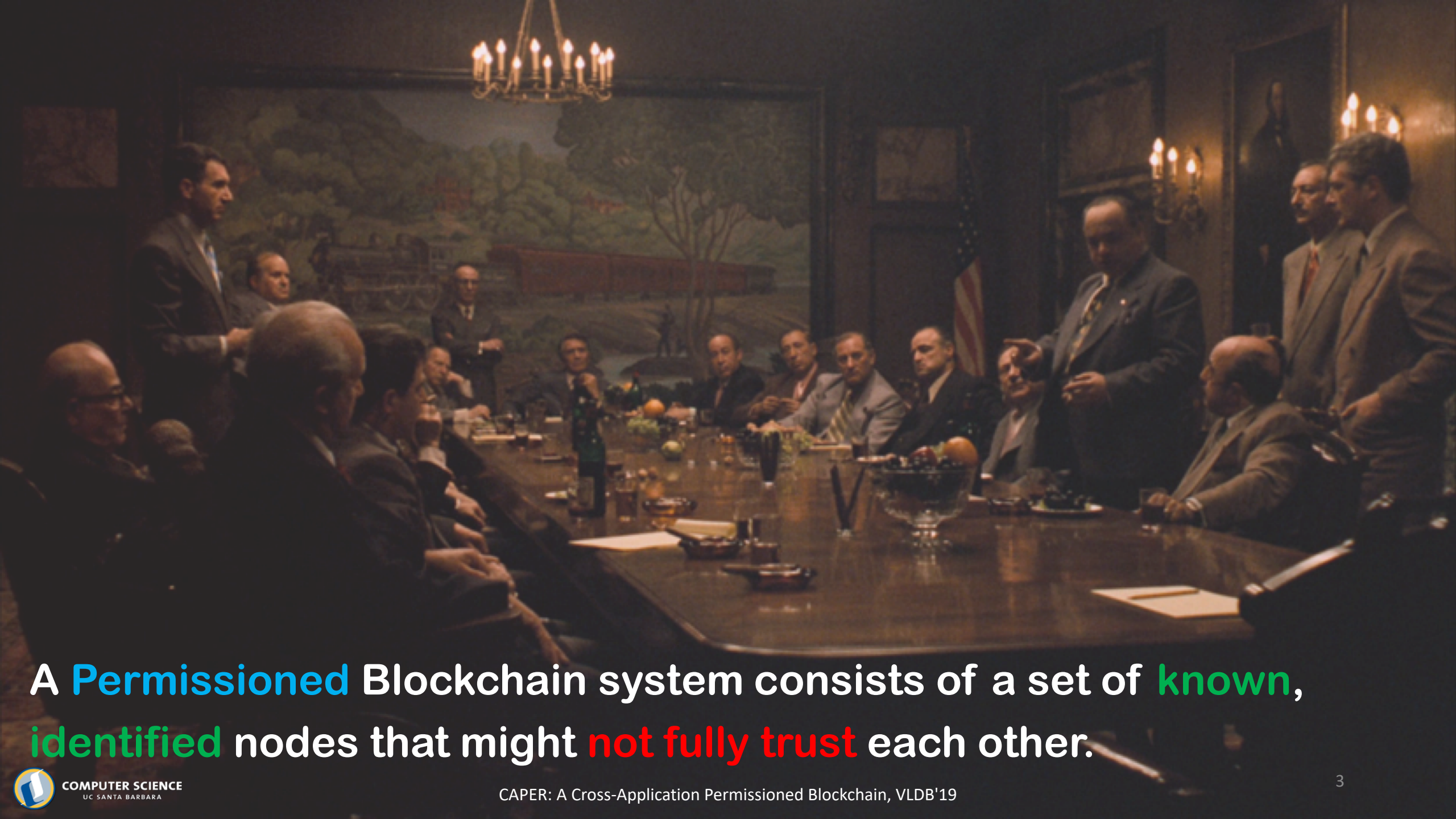
VLDB 2019

Anyone can participate without a specific identity

Participants are known and Identified

Permissionless Blockchain

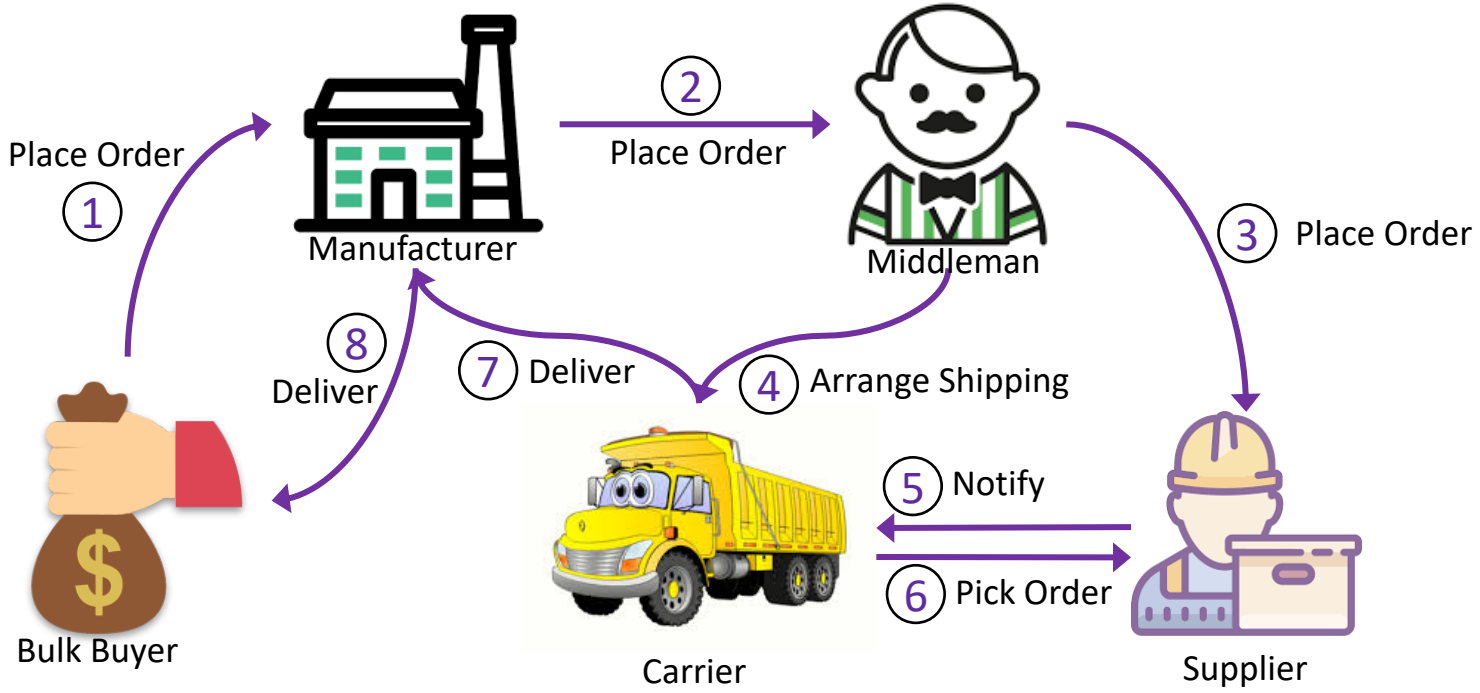
Permissioned Blockchain



A Permissioned Blockchain system consists of a set of known, identified nodes that might not fully trust each other.

Collaborative Workflows: Supply Chain Management

- Different parties (applications) communicate across organizations
- Communication follows *Service Level Agreements* (agreed upon by all participants)
- Applications *do not trust* each other
- Support *both* cross-application and internal transactions
- Internal data of each party is *confidential*



Collaborative Workflows using Blockchain

First Solution: Deploy all applications on the same blockchain system

- Similar to (single-channel) Hyperledger Fabric
- Transactions data and blockchain ledger are replicated on every application

Confidentiality issue

Second Solution: Deploy each application on a separate blockchain system

- Use another blockchain system for the cross-application transactions

Data Integrity issue

Third Solution: Deploy each application on a separate blockchain system

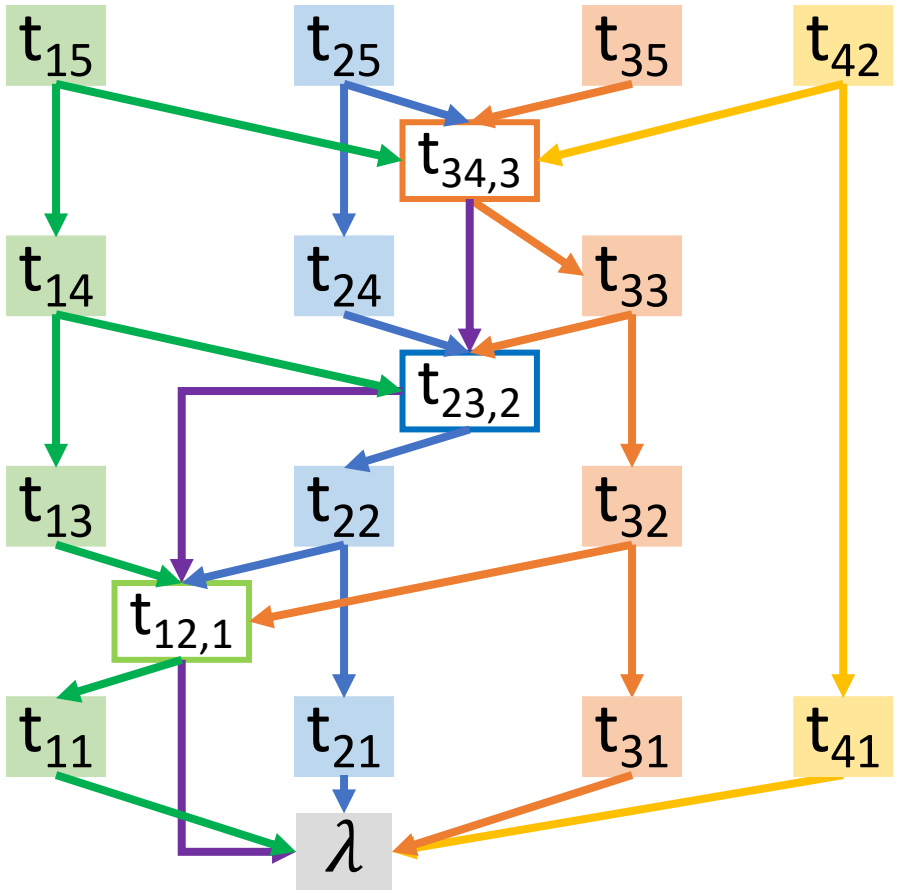
- Use cross-chain operation

Performance issue

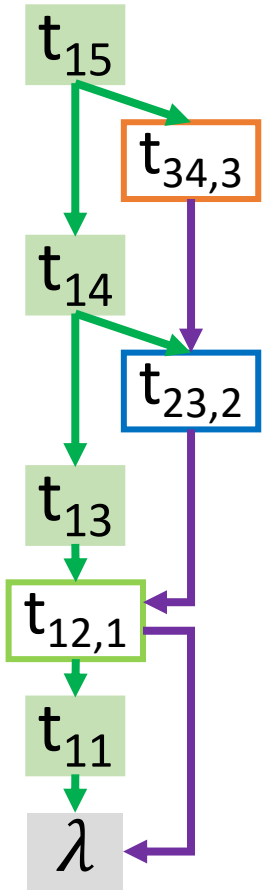
CAPER: A Cross-Application Permissioned Blockchain

- Supports collaborative distributed applications
- Two types of transactions: *internal* and *cross-application*
- Internal transactions of each application are *confidential*
  Cross-application transactions are *visible to all* applications
- The blockchain ledger is formed as a *directed acyclic graph*
- Each application maintains *only* its own view of the ledger including its internal and all cross-application transactions

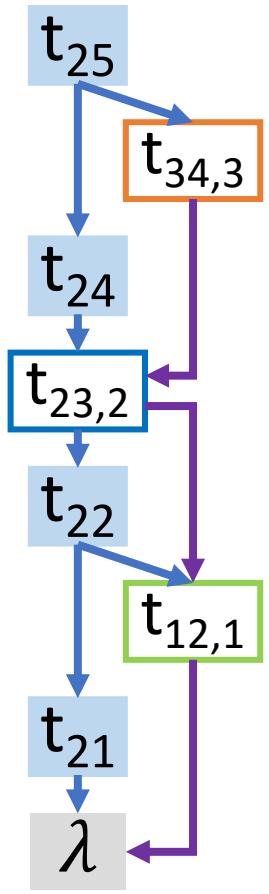
The Blockchain Ledger of CAPER



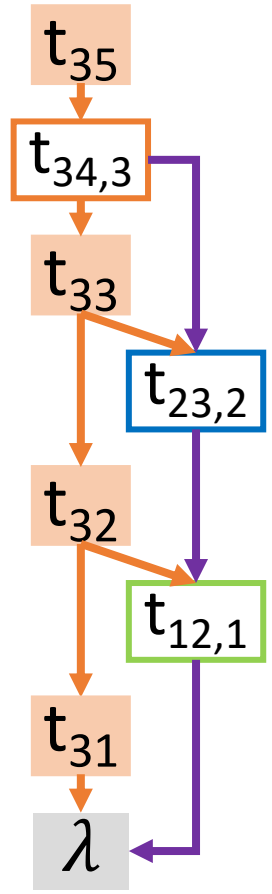
The Blockchain Ledger



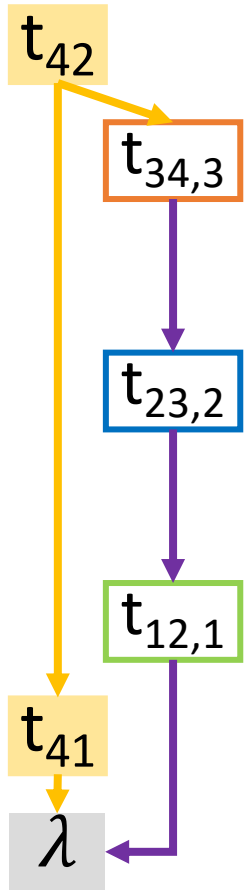
Application 1



Application 2



Application 3



Application 4

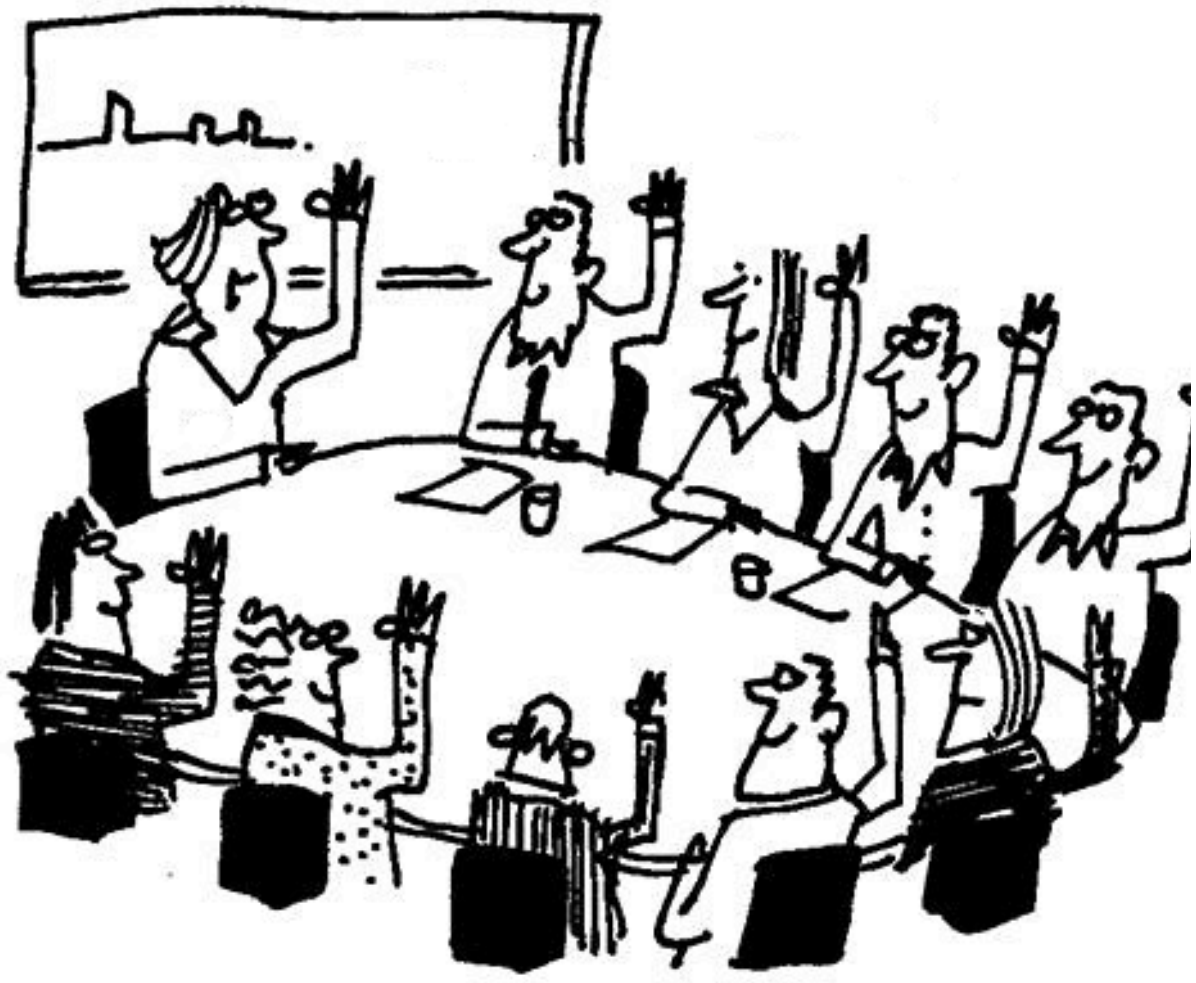
Local and Global Consensus

Local Consensus: pluggable and depends on the failure model of nodes

- Crash-only failure, e.g., Paxos
- Byzantine failure, e.g., PBFT

Global Consensus: needs the participation of all applications

1. Global Consensus Using Orderer nodes
2. Hierarchical Global Consensus
3. One-Level Global Consensus



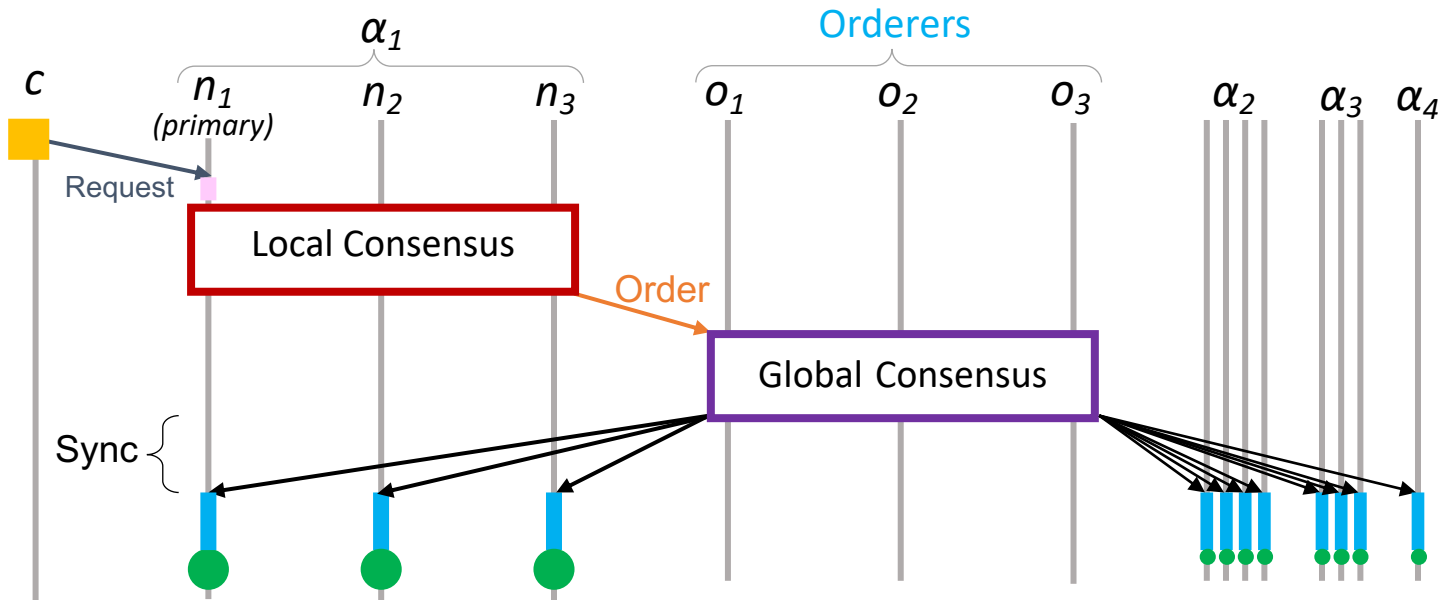
1. Global Consensus using a Separate Set of Orderers

- A disjoint set of nodes, called orderers, globally orders cross-application transactions
  - Similar to Hyperledger Fabric
- Cross-application transactions are first ordered locally and then ordered globally

<Request, tx, τ_c, c>σ_c

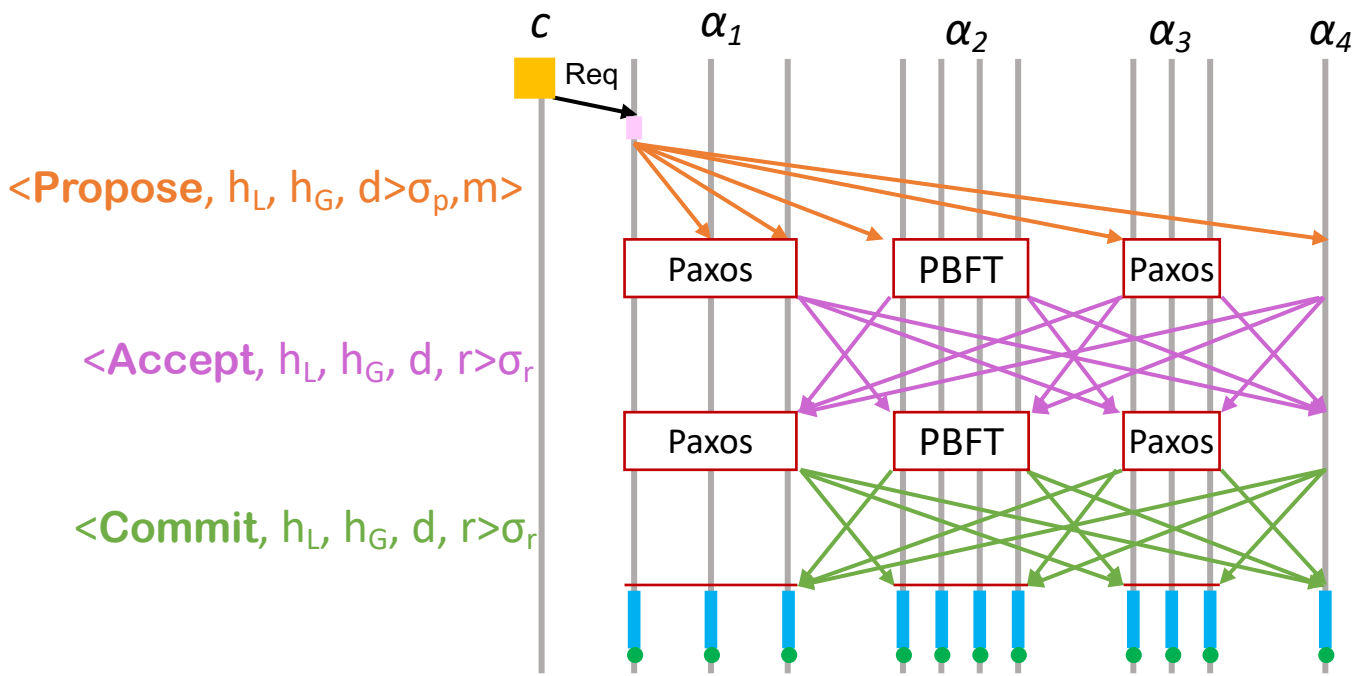
<Order, h_L, d,r>σ_r,m>

<Sync, h_L, h_G, d, o>σ_o,m>



2. Hierarchical Global Consensus

- Using orderers comes with an extra cost of adding orderers to the system
- Applications do not trust each other: we run PBFT among applications
- In each phase of global consensus, every application runs its local consensus
- CAPER ensures that the initiator application agrees with the ordering



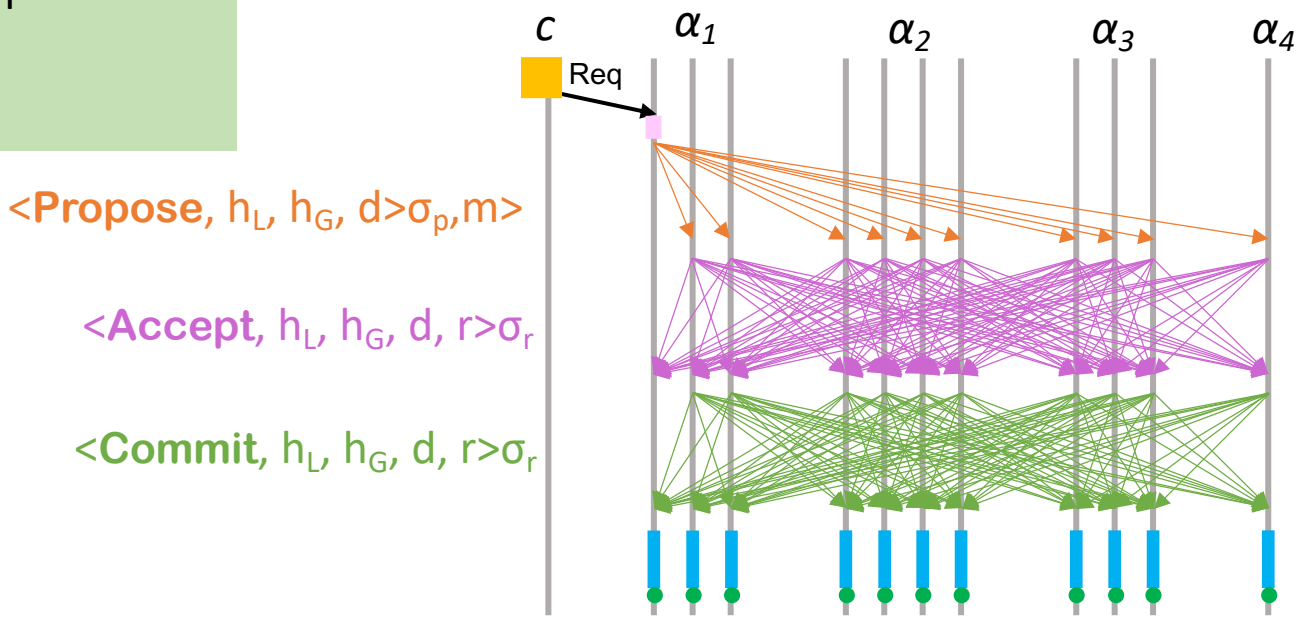
3. One-Level Global Consensus

- Hierarchical consensus requires an expensive two-level consensus protocol
  - Each step of the global consensus needs local consensus within each application
- One-Level Consensus: all nodes of all applications talk to each other

Local-majority: the required number of matching messages from the agents of an application

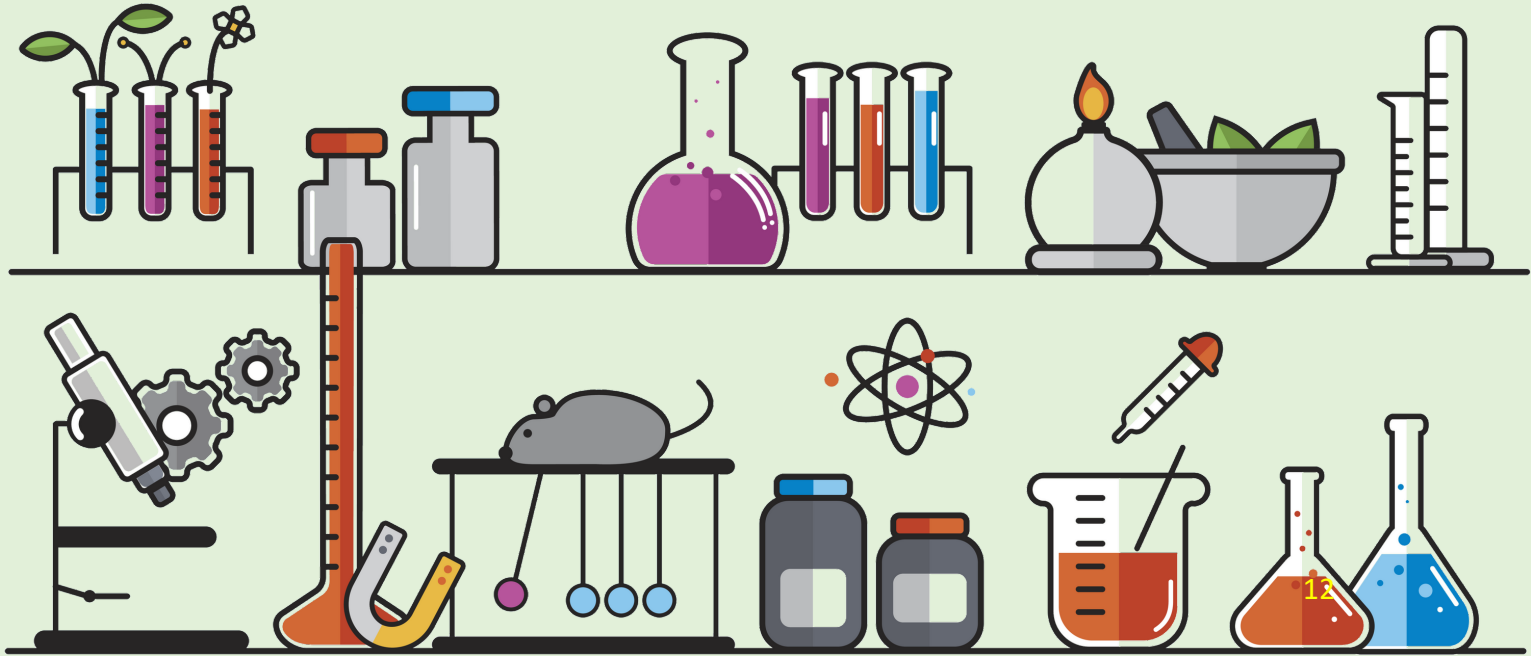
PBFT: 2f+1

Paxos: f+1



Experimental Settings

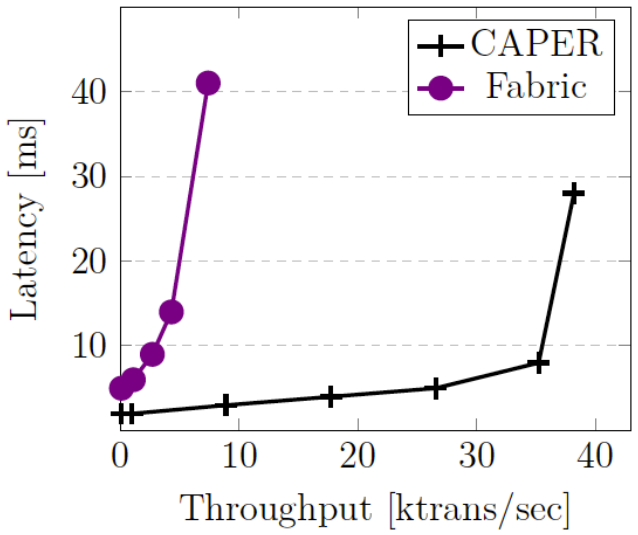
- Systems:
  - Fabric (single-channel, does not preserve confidentiality)
  - CAPER
    - Orderers
    - Hierarchical
    - One-level
- Applications: Accounting
- Platform: Amazon EC2
- Measuring performance
  - Throughput
  - Latency



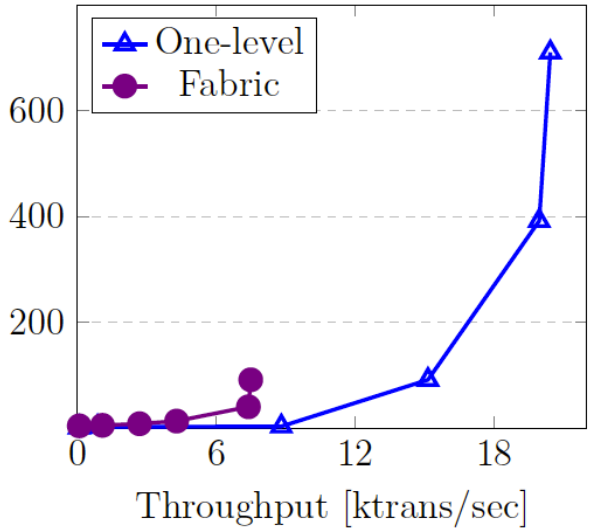
Workloads with Cross-Application transactions

4 Applications

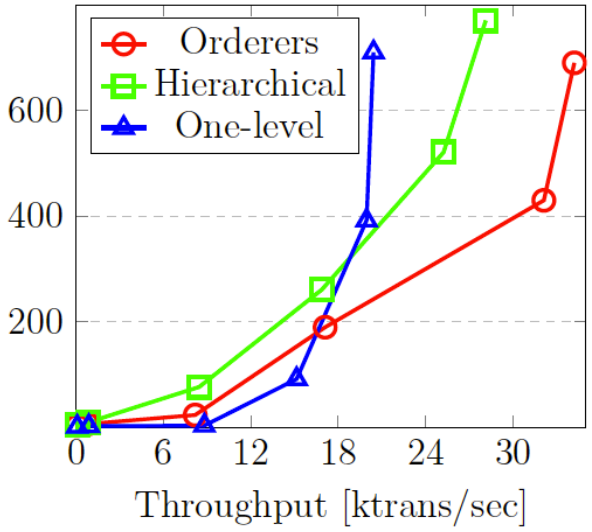
No cross-application



20% cross-application



20% cross-application



CAPER processes more than 36000 transactions (9000 transactions per application)

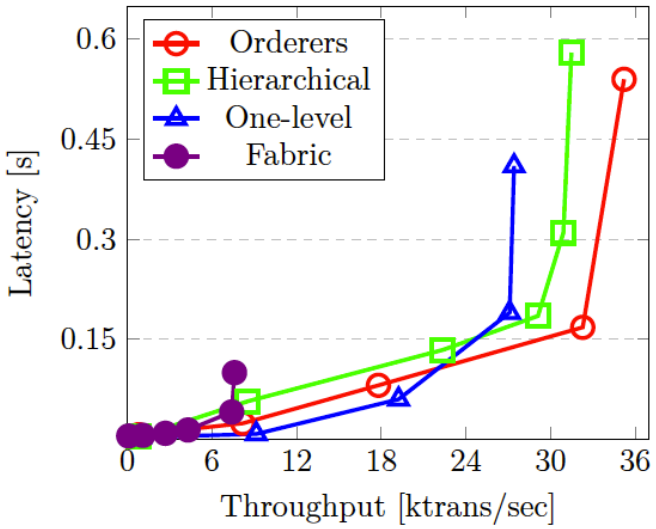
One-Level protocol has better performance in lightly loaded workloads

In a highly loaded workloads the orderers approach has better performance

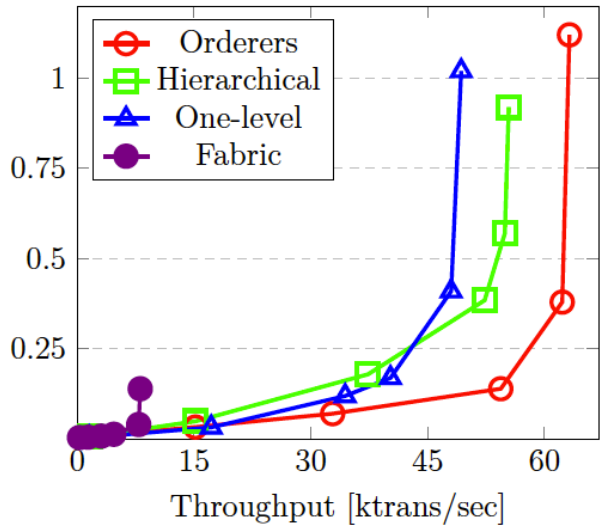
Performance with Multiple Applications

10% Cross-Application

4 Applications

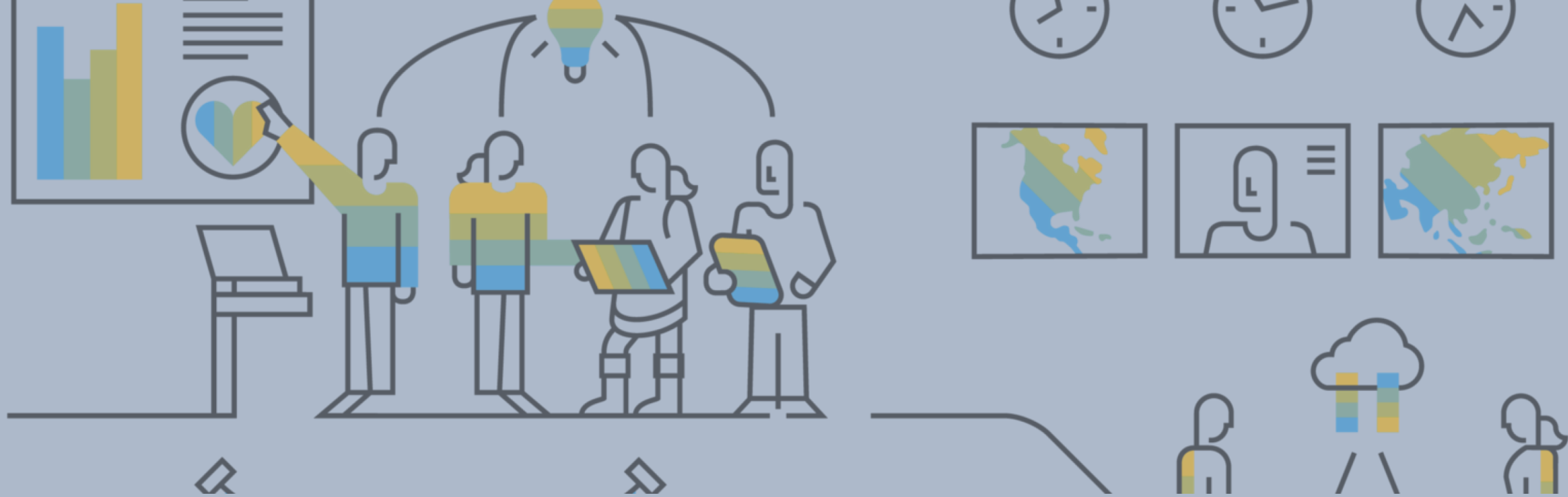


8 applications



The overall throughput of CAPER improves near-linearly

The performance of Fabric does not improve significantly



Future Work

- Supporting transactions among subsets of applications
- Supporting high contention workloads
- Enhancing the scalability of blockchains using sharding

THANK YOU!

Questions?!

amiri@cs.ucsb.edu