The Cloud, the Edge and Blockchains: Unifying Themes and Challenges.

Amr El Abbadi  
University of California, Santa Barbara

Divy Agrawal, Mohammad Amiri, Sujaya Maiyya, Faisal Nawab (UCSC), Victor Zakhary
Distributed Systems are Common Place: The Cloud

Scalability
Availability & Fault Tolerance
Consistency
Protocols Supporting the Cloud

• Scalability:
  • Shard or Partition the Data.
  ➔ Commit Protocols
  ➔ Eg: 2PC
Protocols Supporting the Cloud

• Fault-tolerance and fast access:
  • Replicate the Data
    ➔ State Machine Replication and Consensus Protocols
    ➔ Eg: Paxos
Google’s Spanner [OSDI 2012]

Application Access Tier

Application Execution Tier

Transactions
2PL+2PC

Storage Tier
Abstract Replication
PAXOS

Datacenter A

Datacenter B

Datacenter Z

Berlin 2019
A Path for Unification
Consensus or Byzantine Agreement

• A General wants to send a binary value to n-1 participants such that:
  • Agreement: All correct participants agree on same value
  • Validity: If general is correct, every participant agrees on the value general sends
Consensus

What to do?
We are making the world a better place through Paxos algorithms.
Paxos: No failure Case

- **Leader Election**: Initially, a leader is elected by a majority quorum.
- **Replication**: Leader replicates new updates to a majority quorum.
- **Decision**: Propagate decision to all asynchronously
Paxos: Failure Case

• **Leader Election**: If the leader fails, a new leader is elected.

Also, **Value Discovery** in case agreement has been reached.
Atomic Commitment
Atomic Commitment

• A coordinator wants to commit/abort a transaction that is executed on n-1 participants such that:

  • Agreement: All processes reach same decision
  • Atomicity: Commit only if all processes vote Yes.
  • Validity: If there are no failures and all processes vote Yes, decision will be commit.
Idea: Getting Married over the NW

Will you ... ?

Yes!

Married!

Will you .. ?

Yes!

CS271
Two Phase Commit: No Failure Case:

- **Leader:** Initially, a Coordinator is chosen by transaction manager.
- **Value Discovery:** Coordinator collects votes from **ALL** cohorts
  - If all yes, Decision=Commit, if any (no or failed) Decision=Abort
- **Fault-Tolerance:** Make Decision persistent on disk
- **Decision:** Send Decision to all cohorts

![Diagram of Two Phase Commit](https://example.com/diagram.png)
Three Phase Commit

- 2PC has possibility of **Blocking**
- Solution: 3 Phase Commit.
  - Replicate decision to other cohorts (like Paxos) to avoid site failure blocking.
Three Phase Commit: Termination

- If leader fails or partitioned → Elect new leader and execute termination protocol
Common phases observed?

• Paxos and 2PC/3PC are leader based protocols

• Agreement on a single value is the main goal

• Both protocols ensure fault tolerance on the decided value

• Disseminate the decision, typically asynchronously
Consensus & Commitment (C&C) Framework [VLDB 2019]
Paxos Atomic Commitment (PAC)

- Any processes can terminate a transaction: leader election
- No separate termination case (like Paxos)
2PC/State Machine Replication (SMR)

- Alternative approach to achieve fault-tolerance
  - Replicate state of each process for persistence
  - Spanner and Gray and Lamport 2006

- Layered architecture: 2PC on top of SMR
  - 2PC among coordinator and cohorts
  - SMR among shard leaders and replicas
2PC/State Machine Replication (SMR)

• Alternative approach to achieve fault-tolerance
  • Replicate state of each process for persistence
  • Spanner and Gray and Lamport 2006

• Layered architecture: 2PC on top of SMR
  • 2PC among leaders of coordinator and cohorts
  • SMR among shard leader and replicas

22
Generalized-PAC (G-PAC)

- Follows the abstractions of C&C
- **Flattened** architecture:
  - No notion of cohort leader and replica Coordinator → all identical replicas
- Reduces one round-trip communication
- Related to other consolidating consensus and commitment like TAPIR [Zhang SOSP 2015] and Janus [Mu OSDI 2016]
  - Restrictive assumptions
Consensus & Commitment (C&C) Framework

- Useful in modeling many existing data management protocols as well as propose new protocols
Consensus for Edge Data Management
The future of web/cloud applications

- Emerging technologies
  - Business Analytics
  - Virtual/Augmented Reality
  - Data Science
  - Sensors/IoT
Is there a principled approach to decentralize the cloud for large scale replication?
Edge-Awareness: Last-Mile Latency

The cloud beyond the datacenter:
- Independent Micro datacenters
- Edge Datacenters attached to edge infrastructure
- private/personal clouds

Infrastructure directions: Edge computing, fog computing, and cloudlets

Less than 10 milliseconds

Up to 100 milliseconds
(Multi-)Paxos Execution Example
A lot of nodes are needed to cover users globally

• BUT Majority (or super majorities) are A LOT of machines
Flexible Paxos [Howard et. al. @VMWare OPODIS 2016]

• Majority quorums for **BOTH** Leader Election AND Replication are too conservative
Flexible Paxos

• Generalized Quorum Condition: Only Leader Election Quorums and Replication Quorums must intersect.
  • Decouple Leader Election Quorums from Replication Quorums
  • Arbitrarily small replication quorums as long as Leader Election Quorums intersect with every Replication Quorum

• No changes to Paxos algorithms
- **A zone:**
  - Mutually exclusive set of nodes
  - Datacenter + edge nodes
  - Or Edge nodes
An edge-aware Paxos

- Direct application of Flexible Paxos to zones.
- Elect a leader zone rather than a leader node

**Paxos**
- Replicate updates to majority of all nodes
- Leader election: majority of all nodes

**Edge Paxos**
- Replicate updates to majority of nodes in the leader zone
- Leader election: majority from within all zones.
An edge-aware, mobile Paxos

Zone 1

Zone 2

Zone 3

Zone 4

Leader Election

Local replication

Local replication

Local replication
Great for normal-case operations! But..
CAN WE DO BETTER???
Observation: Leader Election quorum must only intersect with current Replication quorums
Expanding Quorums

- Dynamic Expanding Leader Election Quorums:
  - A leader announces the Replication Quorum it will use
  - Future leader election quorums need intersect only announced quorums

- Implementation
  - Intent Replication Quorums are piggybacked in the leader election phase
  - To detect Intents, leader election quorums must intersect
  - If an announcement is detected, the Leader Election Quorum expands to intersect the announced Intent Replication Quorums
Expanding Quorums example

Leader Election
{intent: zone 1}

Leader Election
{intent: zone 5}

Local replication

Leader Election expansion

Local replication

Berlin 2019
Zone Fault-Tolerant Expanding Quorums

Tolerating One Zone Failure

Leader Election {intent: zone 1}

Leader Election expansion

Leader Election {intent: zone 5}

2-zone replication

Berlin 2019
Leader Zone Expanding Quorums

• Can we design smaller Leader Election quorums?

• Leader Zone: Assign one zone as Leader Manager Zone
• Leader Election quorums: Majority of nodes in the Leader Manager Zone
  • All Leader Election quorums intersect
  • `Use Intent Quorums to expand Leader Election Quorums.

• Especially useful if the aspiring leaders are close to each other
Blockchains

• Many interesting (controversial?) problems in new guises.
  • **Distributed Systems**: Consensus, replication, etc
  • **Data Management**: Transactions, replication, commitment, etc

Berlin 2019
Origins of Blockchain: Traditional Banking Systems
Bitcoin
Traditional Banking Systems

• From Database and Distributed Computing Perspective

• Identities and Signatures
  • You are your signature: IDENTITY ➔ Private and Public Digital signatures

• Ledger
  • The balance of each identity (saved in a DB) ➔ Blockchain (basically a linked list!)

• Transactions
  • Move money from one identity to another
  • Concurrency control to serialize transactions ➔ Mining and Proof of Work
  • Typically backed by a transactions log
    • Log is persistent ➔ Replication to the whole world
    • Log is immutable and tamper-free (end-users trust this) ➔ HashPointers
A Bitcoin Big Picture

- A bitcoin is a chain of digital signatures
  - Coin owners digitally sign their coins to transfer them to other recipients
  - Alice gives a bitcoin to Bob, Bob gives it to Diana, etc.
Double Spending

• Spending the same digital cash asset more than once
• Impossible to do in physical cash
• Prevented in traditional banking systems through concurrency control
Double Spending Prevention

• Classic Approach: Centralization, Concurrency Control, etc

• Blockchain Approach: State Machine Replication (SMR)
  • A network of nodes maintains a ledger (or log)
  • Network nodes work to agree on transaction order
    • Serializing transactions on every coin prevents double spending
The Ledger (or log)

• Where is the ledger stored?
  • Each network node maintains its copy of the ledger

• Transactions are grouped into blocks

• How is the ledger tamper-free?
  Blocks are connected through **hash-pointers (SHA-256)**
  • Each block contains the hash of the previous block
Making Progress

• The ledger is fully replicated to all network nodes

• To make progress:
  • Network nodes **validate** new transactions to make sure that:
    • Transactions on the new block **do not conflict** with each other
    • Transactions on the new block **do not conflict** with previous blocks transactions
  • Network nodes need to agree on the next block to be added to the blockchain
Mining Details: Block Creation
Consensus Protocols

All participants should be known a priori

• Permissioned vs Permissionless settings

• Permissionless setting:
  • Network nodes freely join or leave at anytime
Nakamoto’s Consensus: Proof of Work (PoW)

• Intuitively, network nodes race to solve a puzzle: A Lottery

• This puzzle is computationally expensive

• Once a network node finds (mines) a solution:
  • It adds its block of transactions to the blockchain
  • It multi-casts the solution to other network nodes
  • Other network nodes accept and verify the solution
Mining Details: Mining

SHA256(Version, Previous Block Hash, Merkle Tree Root Hash, Time Stamp, Current Target Bits, Nonce) < D

Transactions
Mining Big Picture
Forks: Double Spending

• Transactions in the forked blocks might have conflicts.
• Forks have to be eliminated.
• Transactions in the forked blocks have to be resubmitted.

Bob tries to double spend the same coin twice in two transactions.
Parting Thoughts

• Building **global-scale data management systems**

- Distributed Systems
- Data Management
- Security and Privacy
- Economics