Overlay (and P2P) Networks

Part II

Samu Varjonen

Ashwin Rao

Outline for this lecture

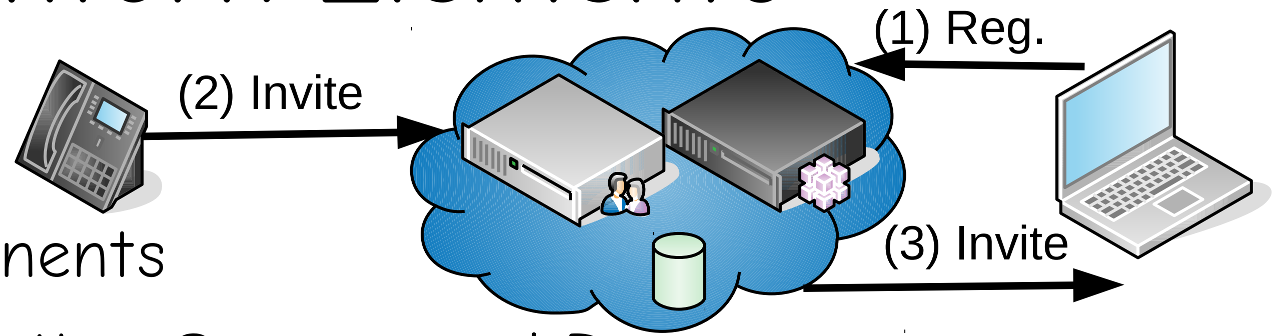
- P2P SIP
- Amazon Dynamo

SIP

- Session Initiation Protocol
  - An Application-layer control (signaling) protocol for creating, modifying and terminating sessions with one or more participants
  - Sessions include Internet multimedia conferences, Internet telephone calls and multimedia distribution
  - Members can communicate via multicast or mesh of unicast relations, or a combination of the two
  - Text based, model similar to HTTP

J. Rosenberg et al. **"RFC 3261: SIP: session initiation protocol."** (2003).

Network Elements

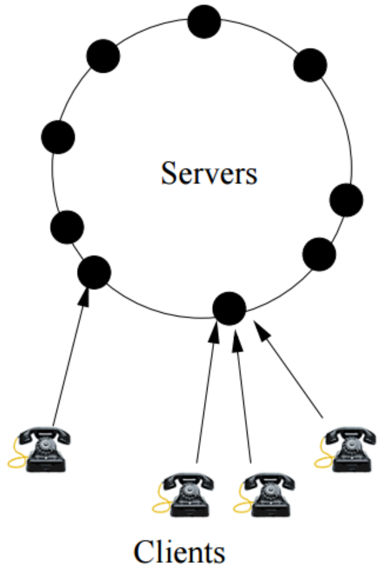


- User-agent
  - End-point components
- SIP Registrar, Location Server, and Proxy
  - Helps users resolve the IP address of each other
- Feature Servers
  - Value added services (call forwarding, recording, etc.)
- Session Border Controller
  - Protect a SIP sub-network from attacks
- Gateways: Signalling Gateway and Media Gateway
  - Transcode Media
  - Support interaction with non-SIP clients

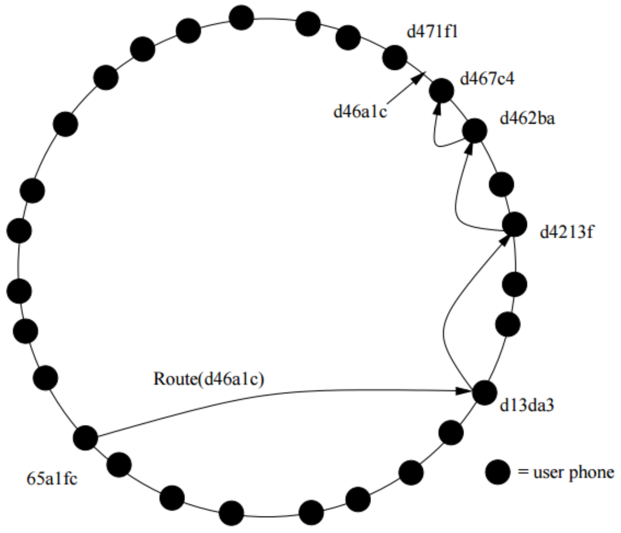
http://sipsense.com/

P2P SIP

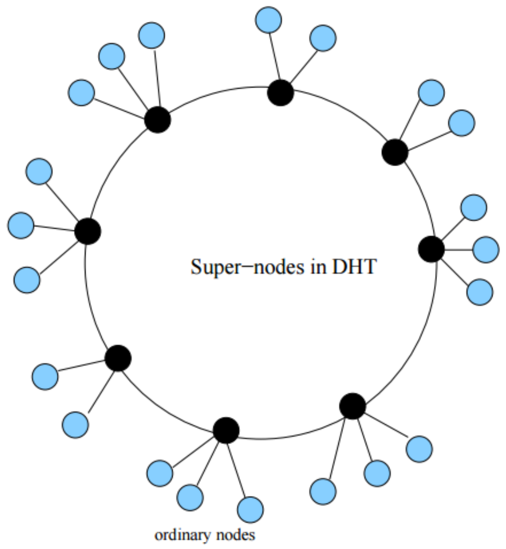
- Can we leverage P2P technologies to implement SIP?
  - Do away with central servers



Only Servers in DHT
Unmodified Clients



All Clients in DHT
Requires modification of clients



Super-nodes in DHT

Kundan Singh et al. **"Peer-to-peer internet telephony using SIP."** In NOSSDAV 2005

ACID
(Recap)

- Atomicity
  - All or nothing
- Consistency
  - Successful transaction commits only legal results
- Isolation
  - Events within a transaction must be hidden from other transactions running concurrently
- Durability
  - Once a transaction has been committed its results, the system must guarantee the results survive subsequent malfunctions

Theo Haerder et al. **"Principles of transaction-oriented database recovery."** ACM Computing Surveys. 1983.

CAP Theorem

- **C**: Strong Consistency (single-copy ACID consistency)
- **A**: High Availability (available at all times)
- **P**: Partition Resilience (survive partition between replicas)

PICK ANY TWO

- C A without P
- C P without A
- A P without C

Popular work-around – reduced consistency (eventual consistency) or reduced availability

A. Fox et al. **"Harvest, yield, and scalable tolerant systems."** HotOS. 1999.

Two Phase Perspective of CAP

- Two phase commit
  - P1: Coordinator asks databases to perform a pre-commit and asks them if commit is possible. If all DBs agree then proceed to P2
  - P2: Coordinator asks DBs to commit
- Two phase commit supports consistency and partitioning. How is availability violated?
  - *Availability of any system is the product of the availability of the components required for the operation*
- ACID provides Consitency. Partion Tolerance is essential. How do you achieve Availability?
  - BASE

Dan Pritchett. **“BASE: An Acid Alternative.”** ACM Queue. 2008

BASE

- Basically available, Soft state, Eventually consistent
- Strong vs Eventual (informal comparison)
  - Strong: Every replica sees every update in the same order (atomic updates)
  - Eventual: every replica will eventually see updates and eventually agree on all values (non-atomic updates)
- Eventual Consistency
  - Database consistency will be in a state of flux but eventually it will be consistent
  - Reads might not return the results of the latest update

Dan Pritchett. **“BASE: An Acid Alternative.”** ACM Queue. 2008

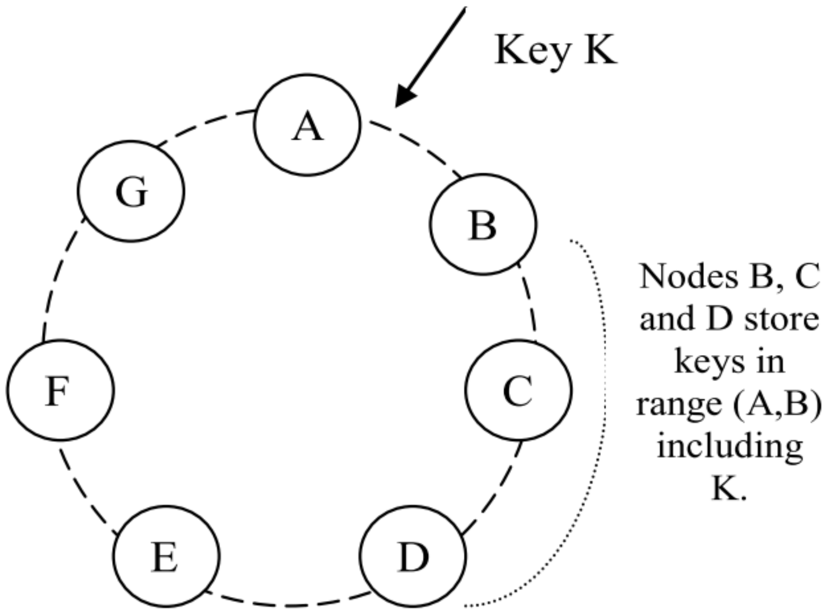
Requirements from Dynamo

- Key-value store
  - shopping carts, seller lists, preferences, product catalog
- System built using off-the-shelf hardware.
- Platform must scale to support continuous growth
- Address tradeoff of high-availability, guaranteed performance, cost-effectiveness, and performance
  - ”The system needs to have scalable and robust solutions for load balancing, membership and failure detection, failure recovery, replica synchronization, overload handling, state transfer, concurrency and job scheduling, request marshalling, request routing, system monitoring and alarming, and configuration management”

G. DeCandia et al. **“Dynamo: Amazon’s Highly Available Key-value Store,”** In SOSP 2007.

Partitioning and Replication in Dynamo

- Consistent Hashing DHT
  - Virtual nodes in DHT
  - Each physical node added as multiple virtual nodes
- Each data-item replicated in N nodes
  - Each virtual node responsible for the region between it and its Nth predecessor
  - Preference List: list of nodes (in (multiple datacenters) storing a key



G. DeCandia et al. **“Dynamo: Amazon’s Highly Available Key-value Store,”** In SOSP 2007.

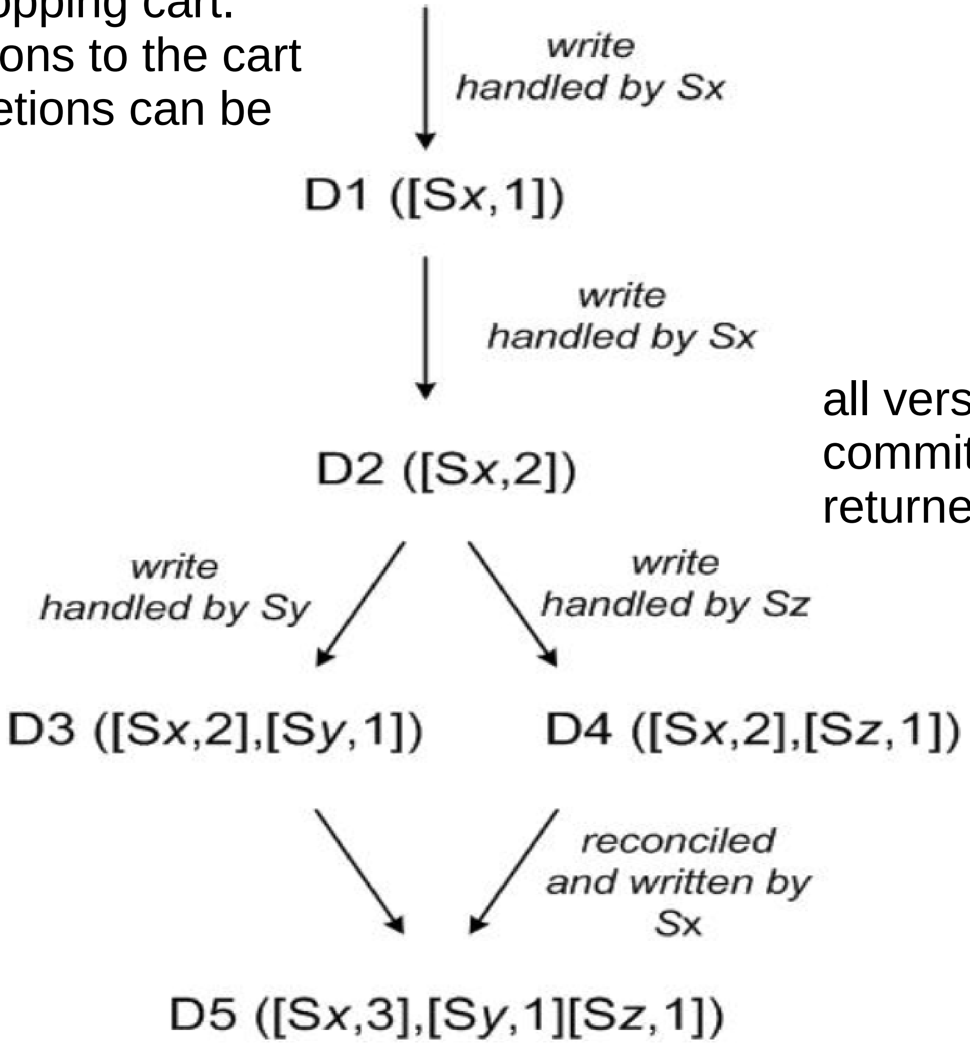
API

- get (key)
  - may return many versions of the same object
- put(key, context, object)
  - Context: encodes system metadata and includes information such as the version of the object
  - may return to its caller before the update has been applied at all the replicas
  - An object may have different version sub-histories
- Vector clock based versioning
  - One vector clock associated with every version of objects

Data Versioning

Objects versions: D1, D2, D3, ...

Assume object is shopping cart. Requirements: additions to the cart don't get lost but deletions can be lost



all versions of the object committed to the system are returned when read

G. DeCandia et al. **"Dynamo: Amazon's Highly Available Key-value Store,"** In SOSP 2007.

Sloppy Quorum

- Read + Write involves N nodes from the preference list
  - R: minimum number of nodes for Read
  - W: minimum number of nodes for Write
- R + W > N
  - R = W = 5 → high consistency but system is vulnerable to network partitions
  - R = W = 1 → weak consistency with failure
  - Typical values of (N, R, W) = (3,2,2) → balance between performance and consistency

Read and Write Operations

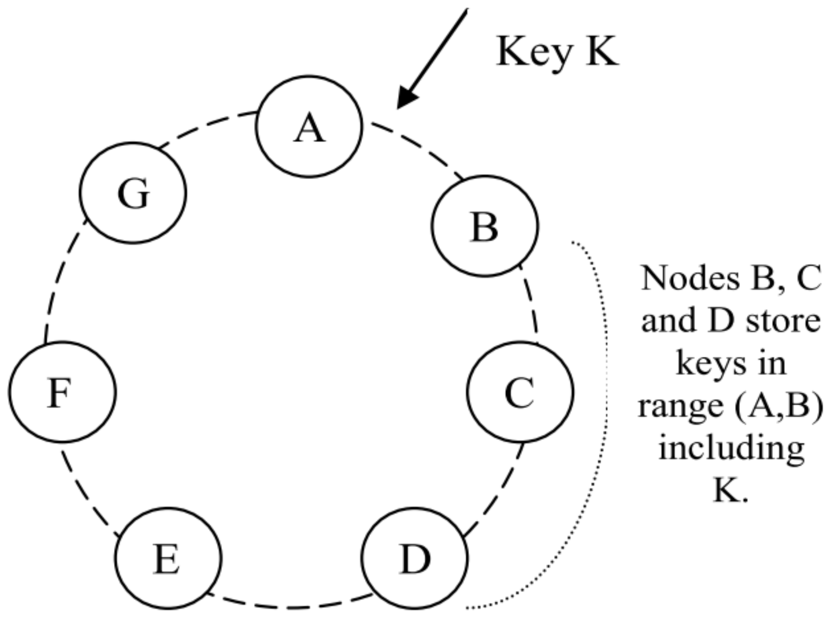
- Coordinator
  - Node responsible for read/writes
  - First node in the preference list
- Write Operation
  - New vector clock from coordinator
  - Write locally and forward to N-1 nodes, if W-1 nodes respond then write was successful
- Read Operation
  - Forward request to N-1 nodes, if R-1 nodes respond then forward to user
  - User resolves conflicts and writes back result

Membership Changes

- Gossip-based Protocol to propagate membership changes
  - Each node contacts a peer chosen at random every second and the two nodes efficiently reconcile their persisted membership change histories
- Each node is aware of the key ranges handled by its peers

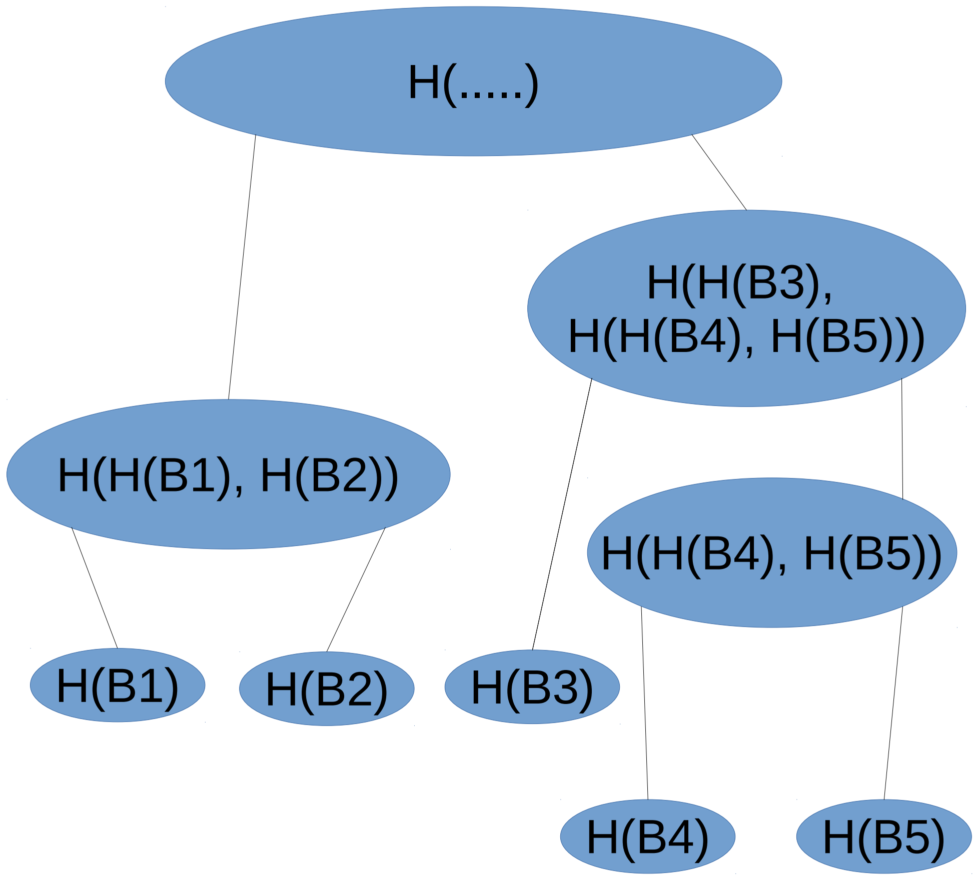
Handling Failures: Hinted Handoff

- Imagine A goes down and N=3
- Keys stored by A will now be stored by D
- D is hinted in the metadata that it is storing keys meant for A
- When A recovers, the keys at D are now copied to A

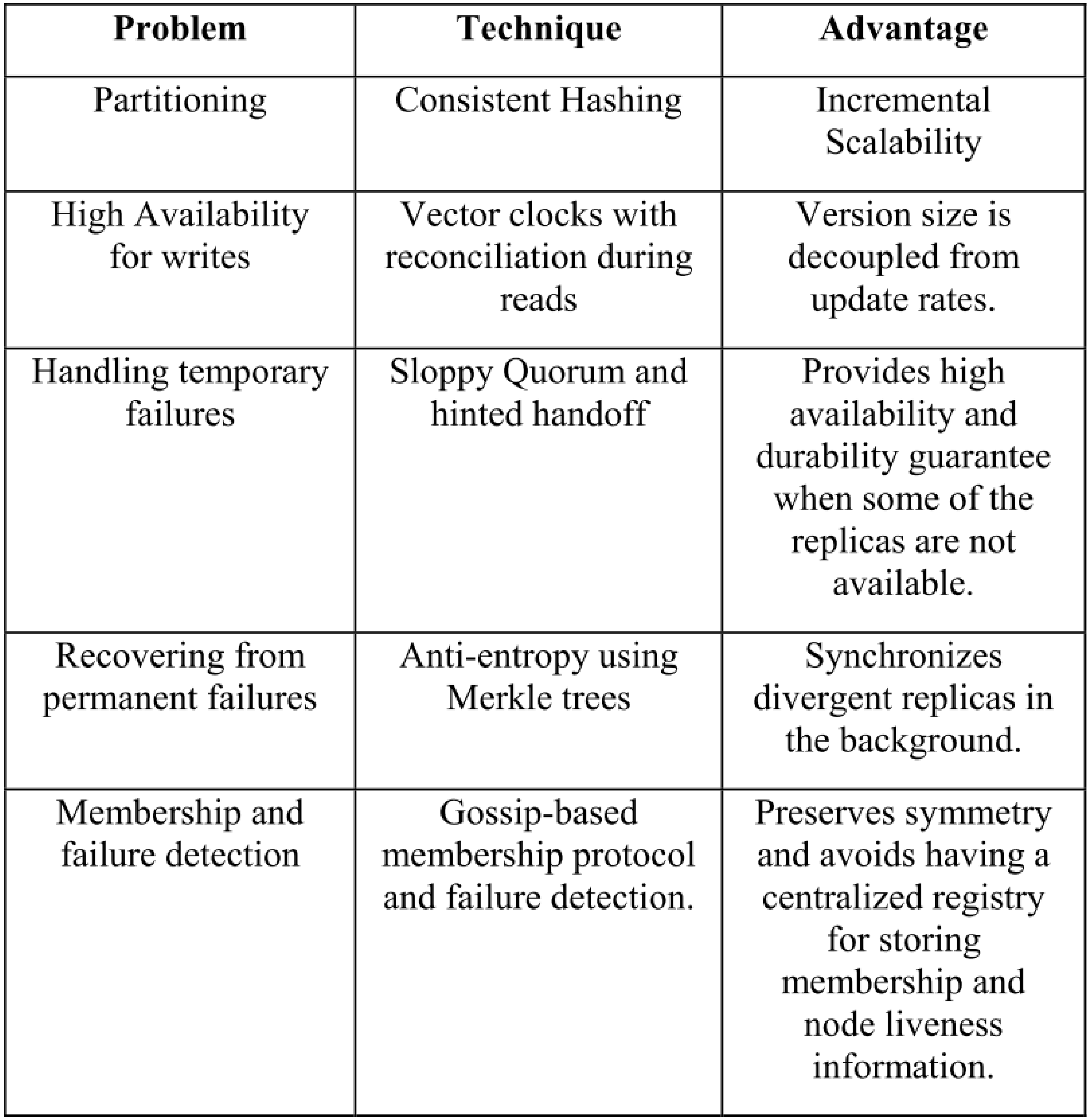


Handling Failures: Merkle Trees

- Minimize the amount of transferred data
- Merkle Tree:
  - Leaves are hashes of keys
  - Parents are hashes of children
- Each node maintains seperate Merkle tree for each key-range



Summary



Problem	Technique	Advantage
Partitioning	Consistent Hashing	Incremental Scalability
High Availability for writes	Vector clocks with reconciliation during reads	Version size is decoupled from update rates.
Handling temporary failures	Sloppy Quorum and hinted handoff	Provides high availability and durability guarantee when some of the replicas are not available.
Recovering from permanent failures	Anti-entropy using Merkle trees	Synchronizes divergent replicas in the background.
Membership and failure detection	Gossip-based membership protocol and failure detection.	Preserves symmetry and avoids having a centralized registry for storing membership and node liveness information.

G. DeCandia et al. **"Dynamo: Amazon's Highly Available Key-value Store,"** In SOSP 2007.

Important References

- http://sipsense.com/
- A. Fox et al. **“Harvest, yield, and scalable tolerant systems.”** HotOS. 1999.
- Theo Haerder et al. **"Principles of transaction-oriented database recovery."** ACM Computing Surveys. 1983.
- Dan Pritchett. **“BASE: An Acid Alternative.”** ACM Queue. 2008.
- G. DeCandia et al. **“Dynamo: Amazon’s Highly Available Key-value Store,”** In SOSP 2007.