Bringing Global-Scale Data Management Closer to the Users.

Amr El Abbadi
University of California, Santa Barbara

Divy Agrawal, Sujaya Maiyya, Faisal Nawab (UCSC), and Victor Zakhary.
The Cloud is an enabler for Networked Applications

Scalability
Availability & Fault Tolerance
Consistency
The future of web/cloud applications

- Emerging technologies
  - Business Analytics
  - Virtual/Augmented Reality
  - Data Science
  - Sensors/IoT
.. But, it is still limited

• Big potential, but bigger challenges
• Requirements:
  • Real-time (low latency)
  • Continuous data flows (high throughput)

• Challenge 1: The cloud is far away

100 of milliseconds to seconds
Single Datacenter: Access Latency Variance
Challenge 2: Datacenter-scale outages

Sutter Health cancels surgeries at Alta Bates amid wider outage
Is there a principled approach to decentralize the cloud?
Globally-distributed EdgeStore Model

• **EdgeStore** deployment:
  - Includes datacenters and edge datacenter nodes.

• **Datacenters:**
  - Typically, 10,000s of machines
  - Constraints: physical space, power availability, and cooling infrastructure

• **Edge Datacenters:**
  - In general, 100s of machines
  - Much easier deployment options especially in dense regions
  - Small footprint ➔ large-scale deployment

• **Problem:**
  - Explosion in the number of centers and replicas
Edge-Awareness: Last-Mile Latency

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

Infrastructure directions: Edge computing, fog computing, and cloudlets

![Diagram showing cloud architecture and latency ranges](image-url)
Edge Caching
Big Data and Data Management Systems

Commodity Machines

Timely & Personalized

Today's Architecture

100s of Millions of end-users

Page-load and page-update stream (millions/sec)

Stateless front-end server

Billions of key lookups per second

Persistent Storage

Cloud Core Datacenters

Overloaded

BIRTE 2018
Todays Architecture

100s of Millions of end-users

Page-load and page-update stream (millions/sec)

Thousands of Stateless Front-end Servers

Load Balancer

Billions of key lookups per second

Persistent Storage

Overloaded

Cloud Core Datacenters

BIRTE 2018
Today's Architecture

100s of Millions of end-users

Page-load and page-update stream (millions/sec)

Thousands of Stateless Front-end Servers

Billions of key lookups per second

Persistent Storage

Cloud Core Datacenters

Load Balancer

High Latency

Supported operations

Consistency

Partitioned and replicated

BIRTE 2018
Today’s Architecture

- **100s of Millions of end-users**
- **Page-load and page-update stream (millions/sec)**
- **Thousands of Stateless Front-end Servers**
- **Billions of key lookups per second**
- **Persistent Storage**
- **Cloud Core Datacenters**
- **100s of Caching Servers**

Cloud Core Datacenters

- **Partitioned and replicated**
- **Load Balancer**
- **Persistent Storage**

BIRTE 2018

UCSB
Edge-based Architecture

100s of Millions of end-users

Page-loads, page-updates (millions/sec)

Edge Datacenters

Thousands of Stateless Front-end Servers

Cloud Core Datacenters

Billions of key lookups per second

Persistent Storage

100s of Caching Servers

Partitioned and replicated

BIRTE 2018
Workload Skewness

- Skewness causes load-imbalance
- Load-imbalance:
  - Increases average latency and long tail latency
  - Decreases the system throughput
Load Monitoring

Centralized

Distributed (server-side)

Distributed (client-side)

Slicer

SPORE

Small Cache Big Effect

Hong, Yu-Ju, and Mithuna Thottethodi. "Understanding and mitigating the impact of load imbalance in the memory caching tier." SoCC’13. Purdue
Small Cache Big Effect

- Caching heavy hitters limits skewness
- Fixed cache size for all front-end
- Assuming perfect caching
  - All accesses for the hottest c keys always hit cache
  - Other accesses always miss cache
Cache on the Edge

100s of Millions of end-users

Page-load and page-update stream (millions/sec)

Edge Datacenters

Thousands of Stateless Front-end Servers

Small Cache

Small Cache

Small Cache

Cloud Core Datacenters

Billions of key lookups per second

hit

miss

Persistent Storage

100s of Caching Servers

Partitioned and replicated

BIRTE 2018
Cache on Track: CoT

- Track approximate top K hot keys (metadata approx key distribution)
- Cache top C keys and their values of K tracked keys
- Prevent cold keys from populating the cache
CoT: Caching Algorithm

Tracking minheap of size K
(meta-data)

Caching minheap of size C
(meta-data + values)

If \( h_k > h_{\text{min}} \) insert

\[ h_k = n_r \cdot r_w - n_u \cdot u_w \]
CoT vs LFU vs LRU

- Key Space 1M
- Zipfian $s = 0.90$
- Tracker size = 1023
- Memory overhead = 8K
- Value size = 1K
- CoT with 39KB (31 cache + 8 tracker) surpasses LRU and LFU with 511KB
CoT: Adaptive Resizing Algorithm

- Cache and Trackers sizes depend on the local key access distribution
- Adapt cache and tracker sizes
  - To cache hot keys above a given skew threshold
  - Limit skew of requests served at the caching servers
  - Limit load imbalance among caching servers
Adaptive Resizing of Front-end Caches

- Different front-ends are deployed in different region edge datacenters
- Distributions differ across regions
- Hot keys in LA is different from SB
CoT: Adaptive Resizing Approach

- Time is divided into epochs
- At each epoch:
  - Evaluate average hotness of cached keys
  - Evaluate average hotness of tracked but not cached keys
- Double cache and tracker sizes if both averages > skew threshold
CoT: Adaptive Resizing Approach

- Time is divided into epochs
- At each epoch:
  - Evaluate average hotness of cached keys
  - Evaluate average hotness of tracked but not cached keys
- Double cache and tracker sizes if both averages > skew threshold
CoT: Adaptive Resizing Approach

- Time is divided into epochs
- At each epoch:
  - Evaluate average hotness of cached keys
  - Evaluate average hotness of tracked but not cached keys
- Double cache and tracker sizes if both averages > skew threshold
CoT: Adaptive Resizing Approach

- **Stop when** average hotness of keys in cache are above skew threshold and average hotness of keys in tracker but not cached are below skew threshold
- Similar algorithm to shrink tracker and cache

---

**Tracked but not cached**

**Cached**

**Hotspot Workload**

**Number of accesses**

**Keys**

**Skew threshold**
Caching Thoughts

• Edge Cloud facilitates caching closer to users
• Cloud offers potential for flexible resizing of cache.
• On-going work.
Edge Replication
Global-Scale Replication (solves Access Latency & DC outages)

Data Consistency?
Geo-Replication

• Asynchronous replication:
  • Widely acknowledged that it results in data losses
  • Recognized as a serious shortcoming due to the variety of internet applications

• Synchronous replication has emerged as an acceptable solution:
  • Industry solutions: Megastore and Spanner
  • Academic proposals: Scatter, MDCC, Megastore’, COPS, Transaction Chains, Replicated Commit, Helios, and others.
  • Main Challenge: how to handle conflicts between copies?
We are making the world a better place through Paxos algorithms
(Multi-) Paxos

- Paxos is currently used to manage local data in global-scale systems
  - Spanner [OSDI’12, SIGMOD’17], Megastore [CIDR’11], etc.

- Multi Paxos, simplified:
  - Initially, a leader is elected by a majority quorum.
  - Replication: Leader replicates new updates to a majority quorum.
  - Leader Election: If the leader fails, a new leader is elected
Paxos Execution Example
Geo-replication Solutions: Google’s Spanner

Application Access Tier

Application Execution Tier
Transactions
2PL+2PC

Storage Tier
Abstract Replication
PAXOS

Datacenter A

Datacenter B

Datacenter Z

BIRTE 2018
Round Trip Time Communication Overhead
The Laws of PHYSICS

Speed of light: $3 \times 10^8$ meters/second

How to address latency bottlenecks given this physical constraint?
“Is there a lower-bound on transaction latency?”

[SIGMOD’15]
A T1 requests to commit

B Events can affect outcome of T1

Transaction T2

Events can be affected by T1

Commit latency of T1 + Commit latency of T2 must be greater than or equal the Round-Trip Time between them
Cloud Computing Infrastructures: The Emerging Landscape
World Population: 7 billion
Cloud Computing Limitations

• *Connectivity* to the Cloud is a pre-requisite for IoT infrastructures and global scale analytics.

• Cloud computing assumes that there is enough *bandwidth* for data transfer.

• Cloud computing centralizes analytics resulting in *large reaction times* for controlling the physical environment.
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. 2016]

- **Optimization opportunity:** there is a trade-off between leader election and replication quorum sizes

- **Flexible Paxos:**
  - 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
Edge Data Management

• *Edge persistence*: edge datacenters to store copies of data

• *Storage offloading*: data may be placed in the edge near users
Paxos for the Edge: Design Goals

• **Access Locality:**
  • Requests are served from a nearby edge node.

• **Data Mobility:**
  • Data shards follow user location in real-time

• **Fault-tolerance redefined:**
  • Edge nodes (Edge Datacenters) can be relied upon for failure-recovery
  • Failure model:
    • Zone failures
    • Datacenter (Core or Edge) failures
Zone Model & Abstraction

• Divide all edge nodes to zones
• A zone:
  • Mutually exclusive set of neighboring nodes
  • Optimize for geographic locality
• A zone:
  • 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

<table>
<thead>
<tr>
<th>Paxos</th>
<th>Edge Paxos</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replicate updates to <strong>majority</strong> of all nodes</td>
<td>Replicate updates to a <strong>majority</strong> of nodes in the leader zone</td>
</tr>
<tr>
<td>Leader election by getting the votes of a <strong>majority</strong> of all nodes</td>
<td>Leader election across <strong>all zones</strong>. (votes from a <strong>majority</strong> in each zone)</td>
</tr>
</tbody>
</table>
An edge-aware, mobile Paxos [SIGMOD 2018]
Great for normal-case operations! But..
Dynamic Paxos [SIGMOD 2018]

Observation: Leader Election quorum must only intersect with currently used Replication quorums by other leaders.
Expanding Quorums [SIGMOD 2018]

- 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

Zone 1
Zone 2
Zone 3
Zone 4
Zone 5

Leader Election {intent: zone 1}
Leader Election expansion

Leader Election {intent: zone 5}

Local replication

Local replication

Local replication
Leader Zone Expanding Quorums

• Can we design Leader Election quorums to be even smaller?

• Leader Zone quorums: assign one zone as Leader Manager Zone
  • Leader Election quorums is a majority of the nodes in the Leader Manager Zone
  • All Leader Election quorums intersect
  • Each Intent Quorums to expand Leader Election Quorums.

• Especially useful if the aspiring leaders are close to each other
Leader Zone Quorums Example

Leader Manager Zone

Zone 1

Zone 2

Zone 3

Zone 4

Zone 5

Leader Election {intent: zone 1}

Leader Election {intent: zone 3}

Leader Election expansion

Local replication

Local replication

Local replication
Changing the location of the Leader Manager Zone

• What happens if the aspiring leaders are no longer close to the Leader Manager Zone?

• Treat leadership management as a logical role instead of physical
  • Relinquish leadership to another node when users moves

```java
relinquish()
- current state
- slots []
```
Bringing it all together

• Analytics and IoT applications generate and consume data at the edge
• Need a data management layer to integrate the resources at the edge
• D-Paxos provides an edge data infrastructure:
  • Consistency & Fault-tolerance
  • Application Development is agnostic to the data in the cloud vs the edge
Parting Thoughts

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

  - Data Analytics
  - Replication and Consistency
  - Data Ingestion and IOT

• What about Data Privacy?
• Integrating Blockchain?