EdgeCons: Achieving Efficient Consensus in Edge Computing Networks

Zijiang Hao
Shanhe Yi
Qun Li

College of William and Mary
Williamsburg, VA
Achieve fast event ordering for delay-sensitive edge applications
Achieve fast event ordering for delay-sensitive edge applications
• Achieve fast event ordering for **delay-sensitive** edge applications
Use the Paxos-based approach on the edge:

- Achieve a consistent view across a distributed system
- Low latency
- Modest communication cost
The Paxos algorithm:
- Phase 1: The distributed system determines a leader node
- Phase 2: The distributed system accepts a value published by the leader node

The ordering protocol:
- Run a sequence of Paxos instances on the edge
- In each Paxos instance, either an event is published, or no event is published
EXISTING WORK (1)

- **Multi-Paxos:**
  - All Paxos instances use the same node as the leader node
  - Bottleneck

- **Mencius:**
  - The leadership of the Paxos instances is distributed on the nodes in a round-robin way (1, 2, ..., N, 1, 2, ..., N, ...)
  - The leadership distribution method is fixed, inflexible
EXISTING WORK (2)

- E-Paxos:
  - The leader nodes of the Paxos instances are not pre-assigned, flexible
  - More communication cost
**EdgeCons: Design Philosophy**

- Determine the leader node of each Paxos instance dynamically
  - Based on the running history of the protocol
  - Take advantage of the temporal locality of the workload

- Use the cloud to resolve contentions on the edge
  - When two edge nodes intend to publish an event simultaneously, a contention may happen
**EDGECONS: APPROACH (1)**

Edge Node A  
Edge Node B  
Edge Node C  
Cloud

- : Effective
- : Skipped
- : Scheduled

Epoch (N – 1)  
Epoch N

---

A B C A B C A B C A C A C A B A C A C A B

---
Some details:
- The leadership distribution algorithm is deterministic
  - The edge nodes do not need to communicate with each other to determine the leadership distribution for the next epoch
- Edge nodes proactively skip the Paxos instances belonging to others
  - The cloud instance cannot be skipped
- The length of epoch should be carefully chosen
  - To well capture the temporal locality of the workload
A simulation experiment:
- RTT between the client and the edge: 10 ms
- RTT between the edge nodes: 10 ms, expect a slow one: 40 ms
- RTT between the edge and the cloud: 60 ms
- Length of epoch: 120 Paxos instances
- Event rate: No more than 10,000 events/s on each edge node
EdgeCons: Evaluation (2)
How to better adapt to the workload change?
- Look at more previous epochs
- More recent, more weight

How to work around an unreliable cloud?
- Build a reliable arbitrator using lease-based local edge nodes
- The arbitrator should not become a bottleneck
Thank you!

Suggestions & Questions?