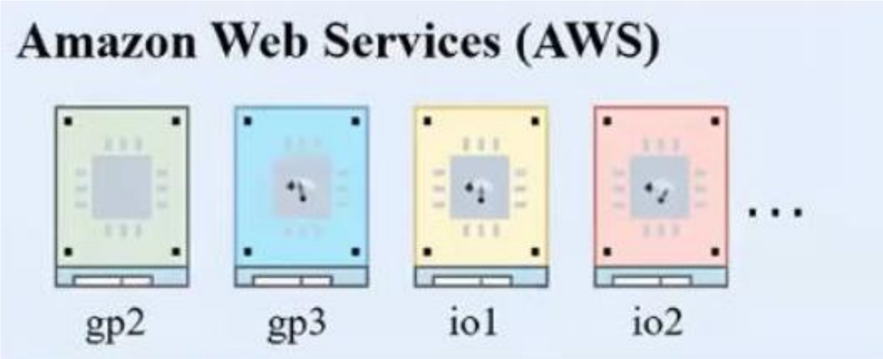
ATC'23_Calcspar:
A Contract-Aware LSM Store for Cloud Storage with Low Latency Spikes

2023/11/01

Cloud Storage Based on LSM Stores

Cloud storage is gaining popularity

- pay-as-you-go reduces storage costs
- not explored its contract model and latency characteristics

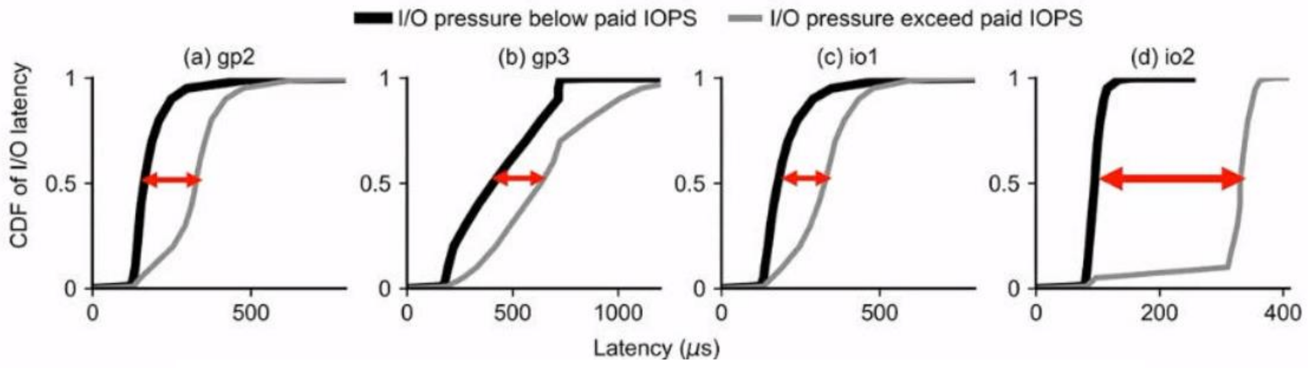


LSM stores become the building block for many cloud applications

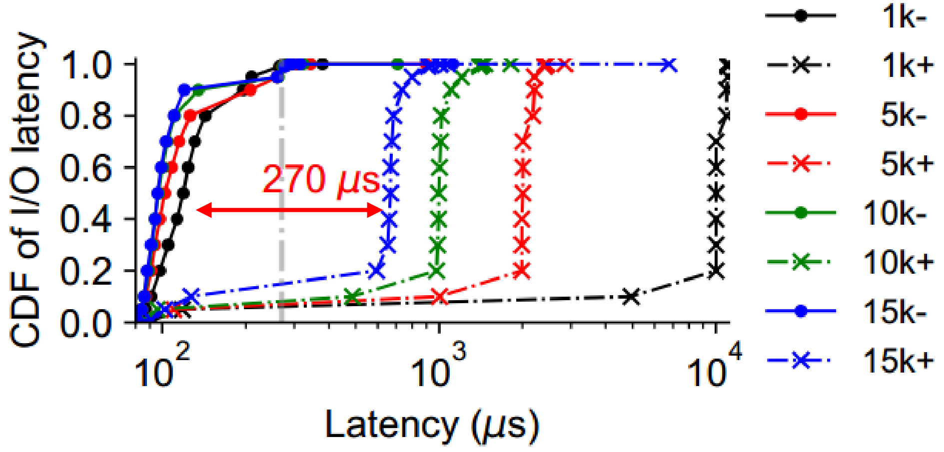
- none is optimized for cloud storage to eliminate long-tail latency

Latency Spike of Cloud Storage

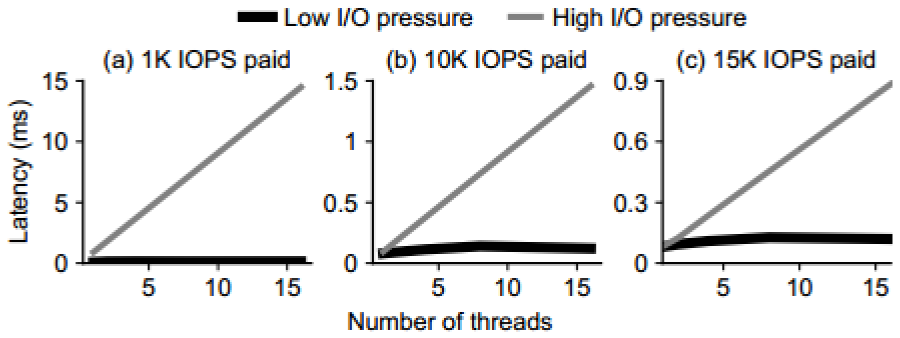
- When the I/O pressure exceeds the paid IOPS, the latency increases.



- The higher the paid IOPS, the lower the latency.



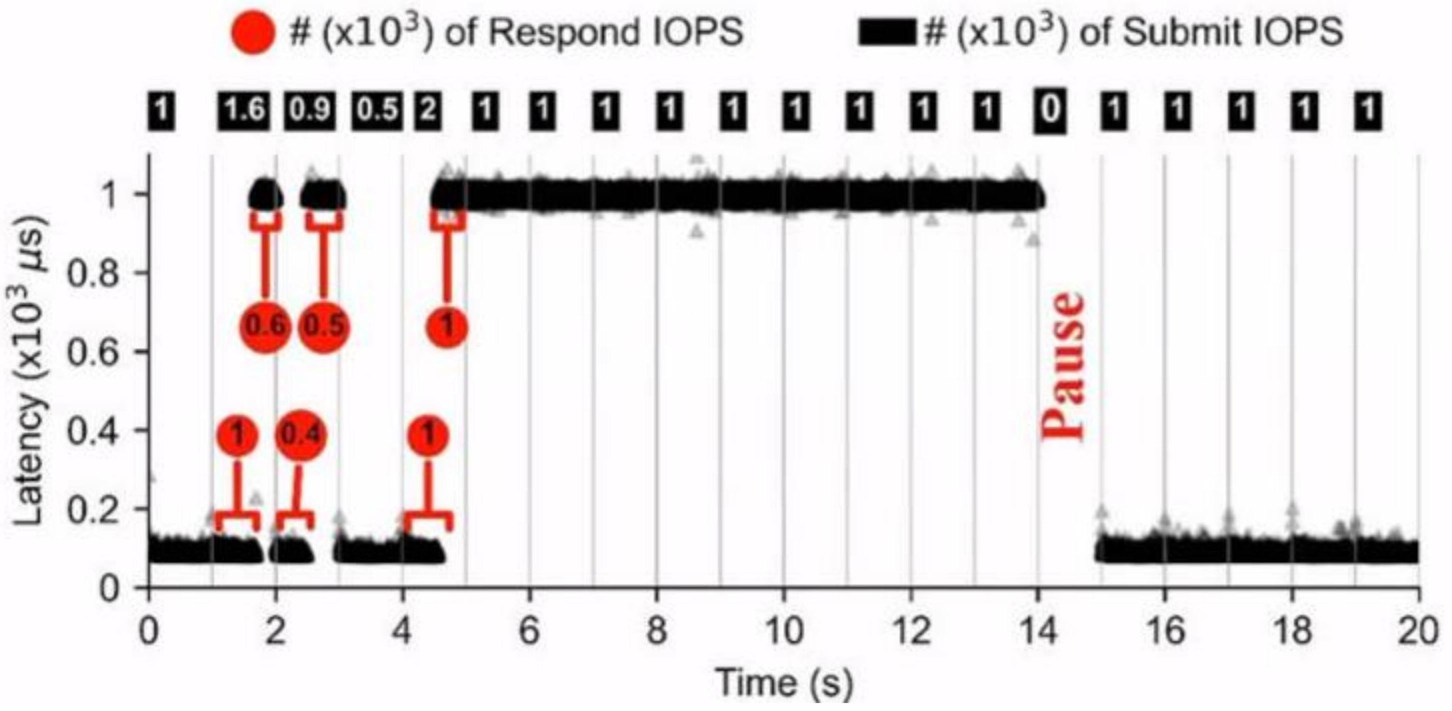
- Response time = # Threads / Paid IOPS on the high I/O pressure



Essential Reason of Latency Skipe

Quantifying the change of latency

- Overdraw: overdrawing the IOPS of the next 1 second.
- Punish: punitively increase latency to 1/IOPS.
- Defense: prevent continually responding I/Os beyond the payment.



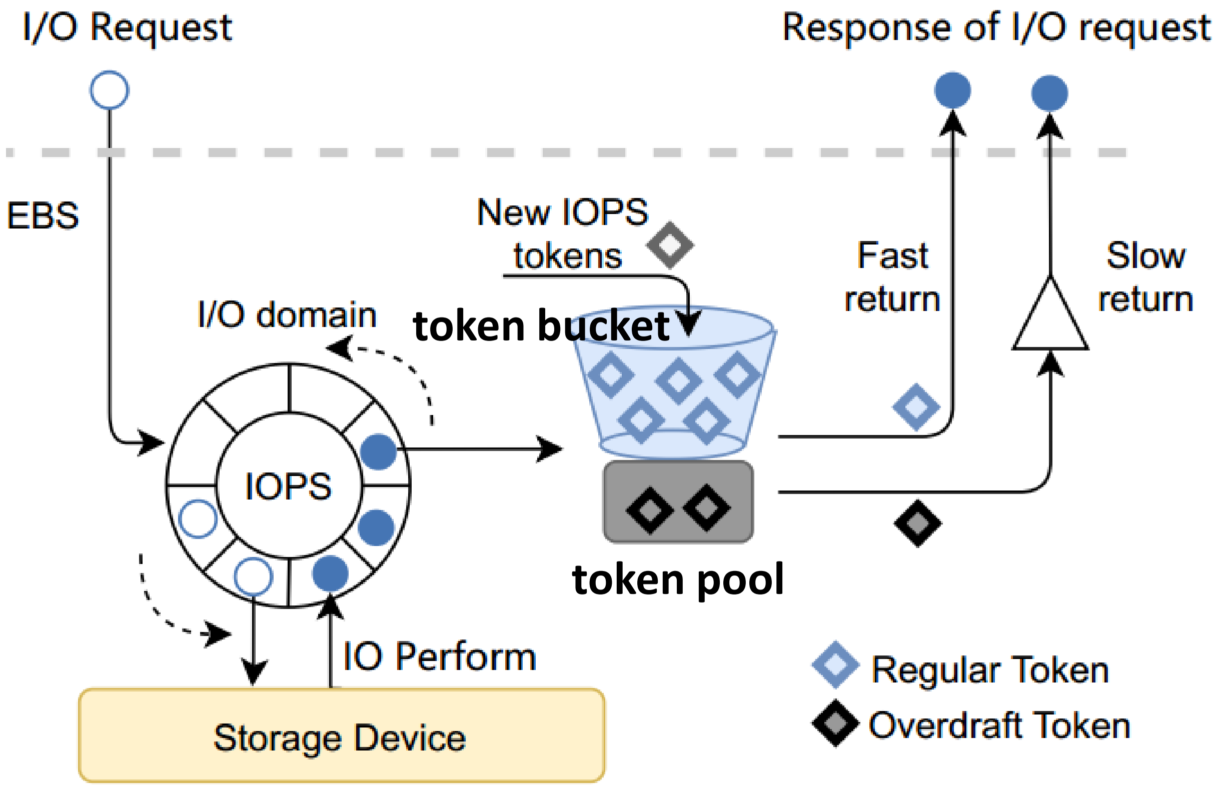
latency Model

Submit IOPS < Paid IOPS

- -> Regular token with low latency

Submit IOPS > Paid IOPS

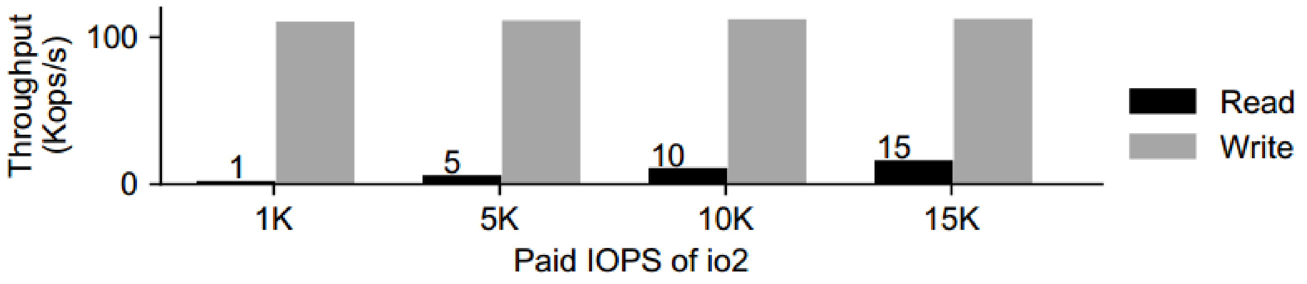
- -> Overdraft token with high latency
- -> I/O domain slots filled
  - -> Persistent high latency

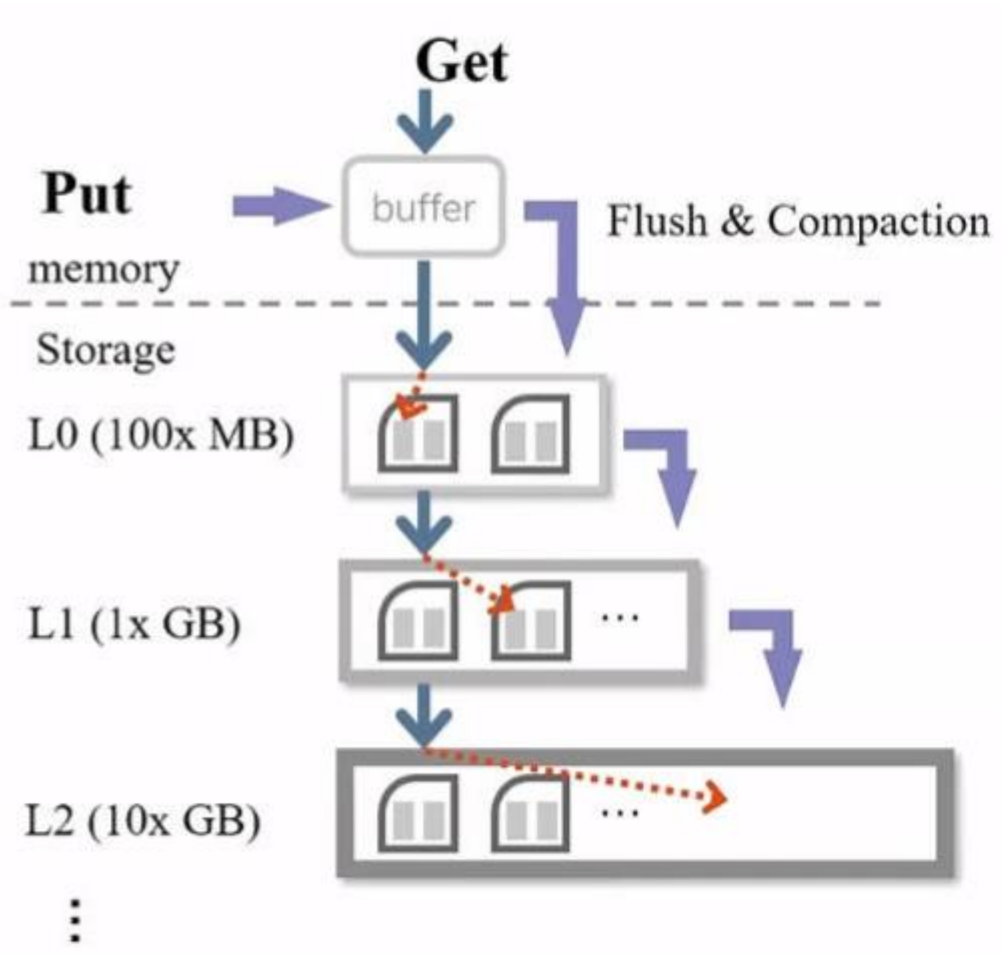


LSM-Tree

The existing LSM stores for cloud storage has long-tail latency

- Read amplification

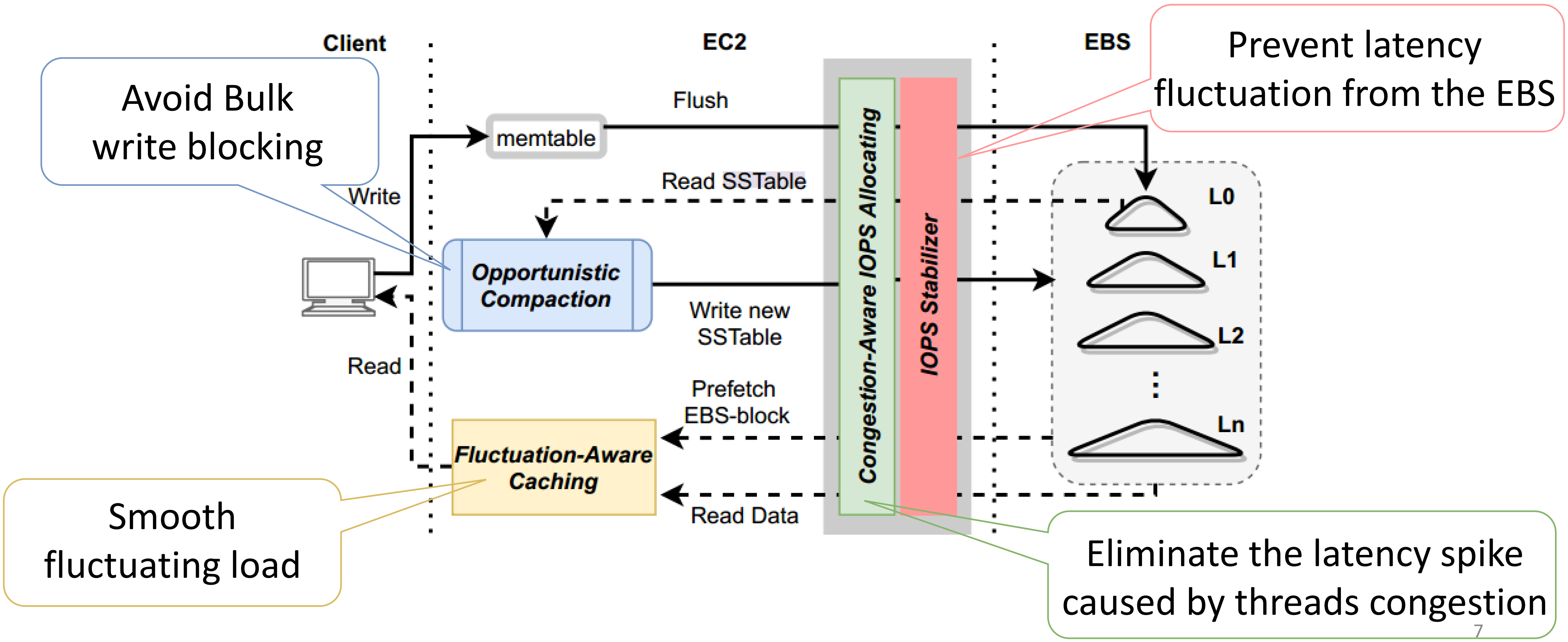




Read amplification

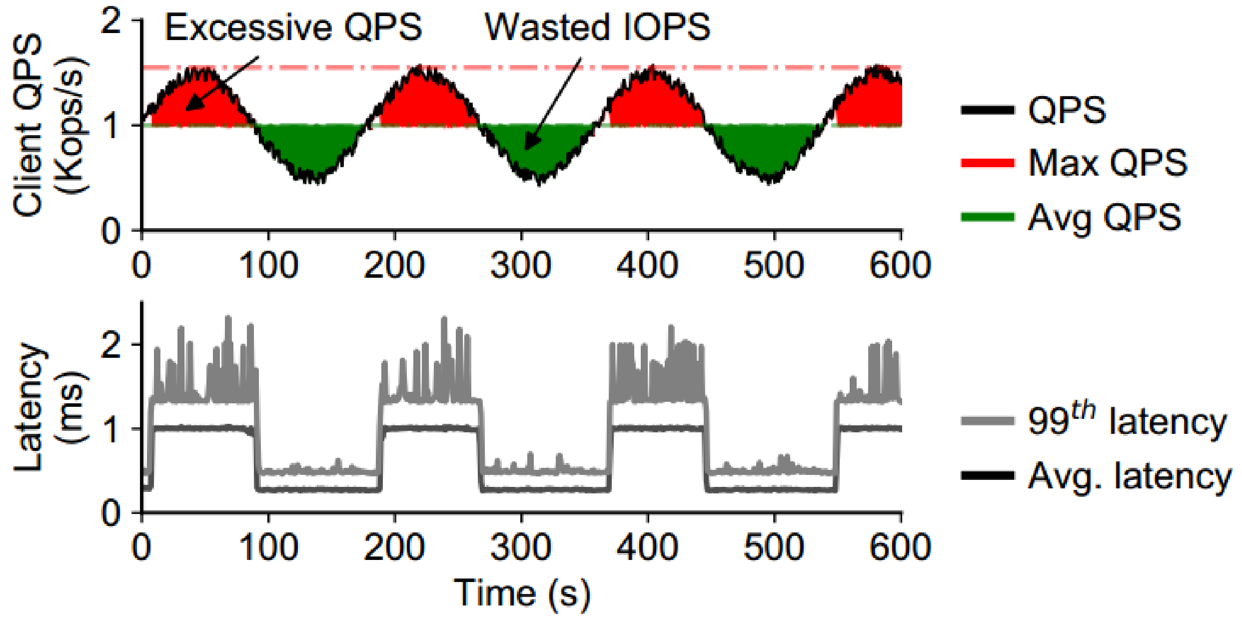
Calcspar Architecture

Problem: How to avoid read latency spikes in LSM stores?

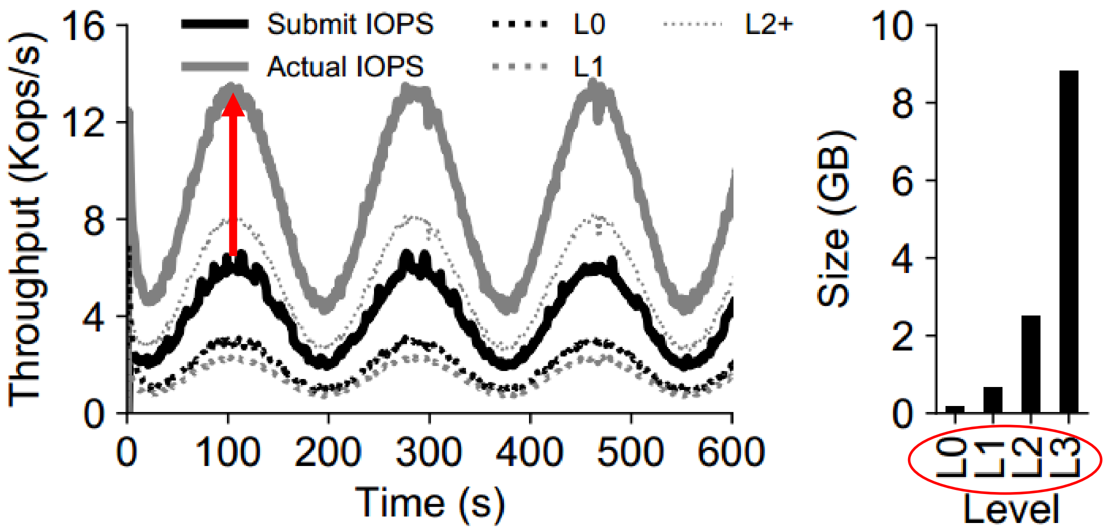


C1: Fluctuating request numbers

Challenge: Fluctuating request numbers



Workload Bursts



Read amplification

C1-D1: Fluctuation-Aware Caching

Challenge: Fluctuating request numbers

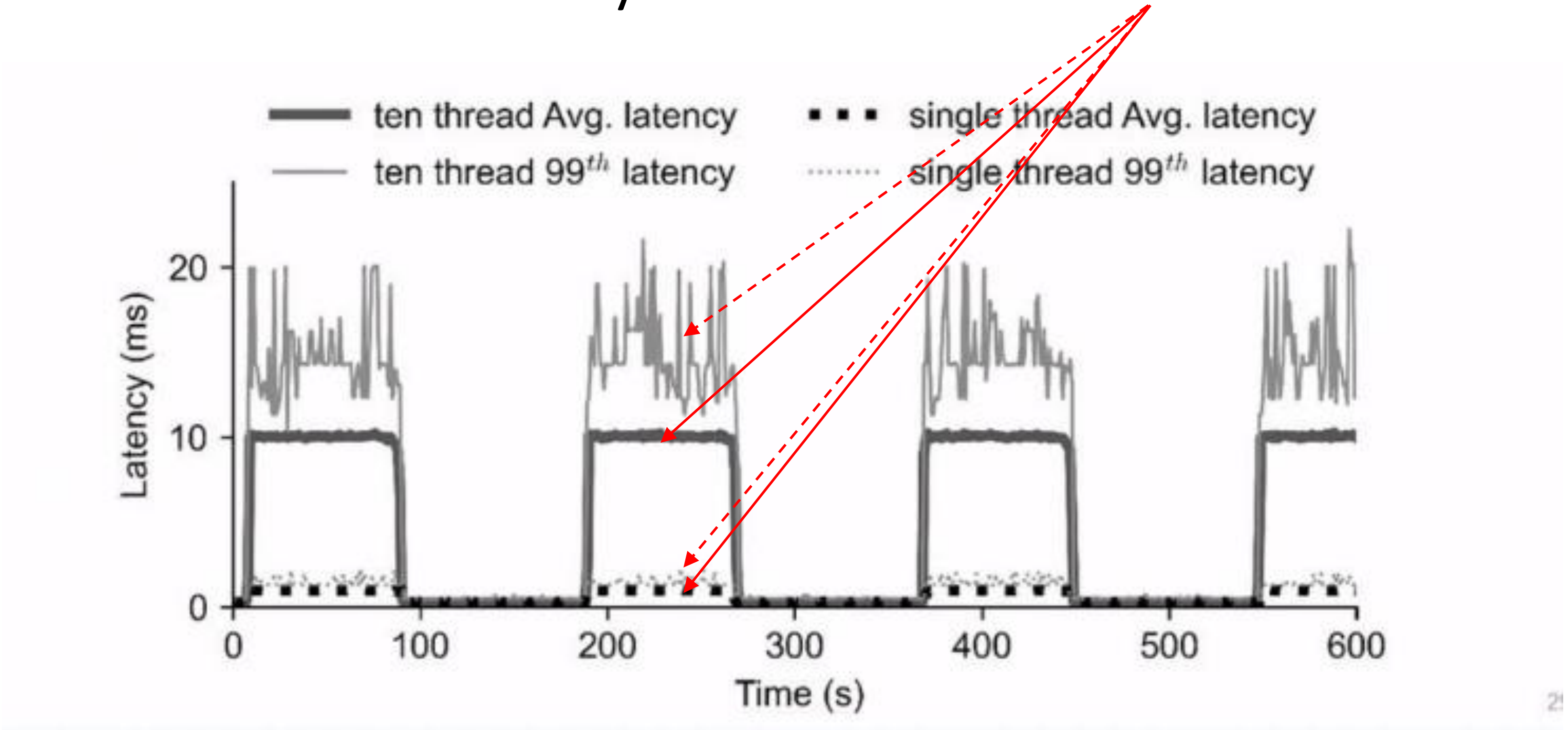
Solution: Fluctuation-Aware Caching

- Hotspot-Aware Proactive Prefetching
  - When the workload is light, prefetch SSTable
- Shift-Aware Passive Caching
  - When the workload is heavy, use LRU evicte data
- Cache Integration
  - Switch them

C2: Thread I/O competition

Challenge: Thread I/O competition

Requests between multiple threads during workload bursts are congested in the I/O domain, resulting in an exponential increase in tail latency.

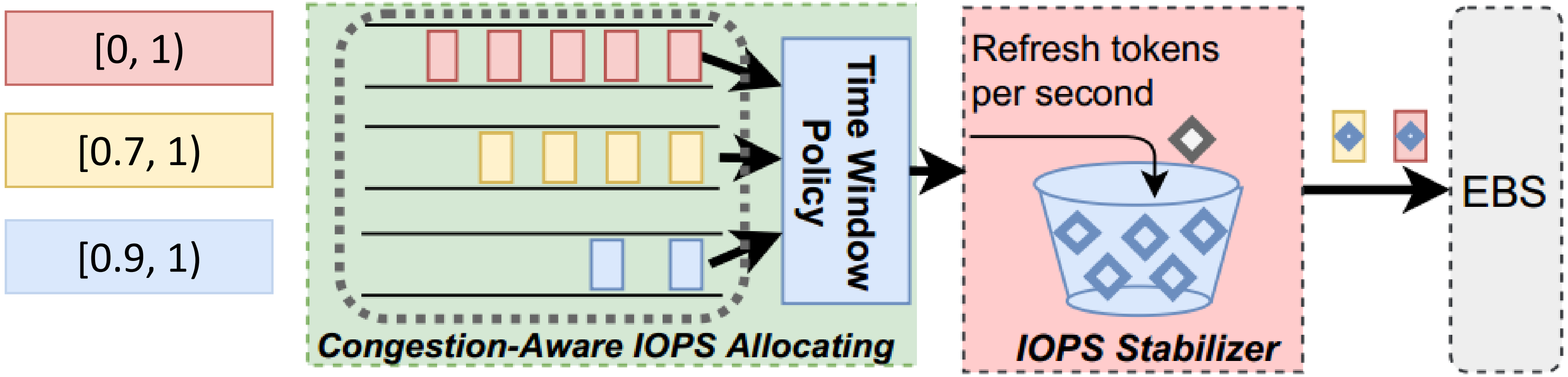


C2-D2/3: Congestion-Aware IOPS Allocating & IOPS Stabilizer

Challenge: Thread I/O competition

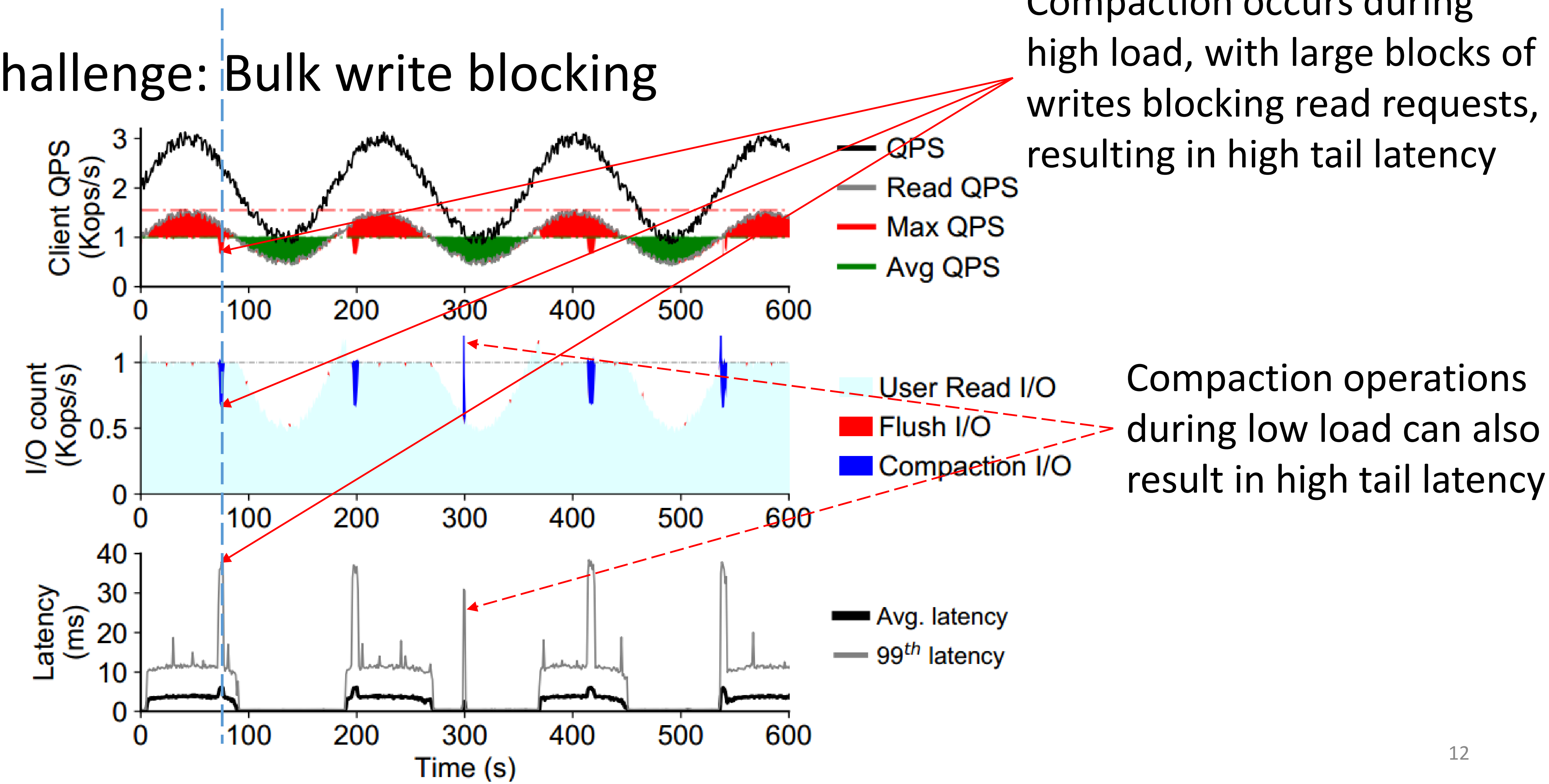
Solution:

- Congestion-Aware IOPS Allocating
  - Multi-Priority Queues
  - Dynamic Time Window Policy
- IOPS Stabilizer
  - Control Submit IOPS not exceed by using token bucket



C3: Bulk write blocking

Challenge: Bulk write blocking



C3-D4: Opportunistic Compaction

Challenge: Bulk write blocking

Solution: Opportunistic Compaction

- L0 SSTables -> the high priority queue
- L1 and L2 SSTables -> the medium priority queue
- SSTables levels below L2 -> low priority queue

Overall Performance

Calcspar can significantly reduce tail latency while maintaining regular read and write performance, keeping the 99th percentile latency under 550µs and reducing average latency by 66%.

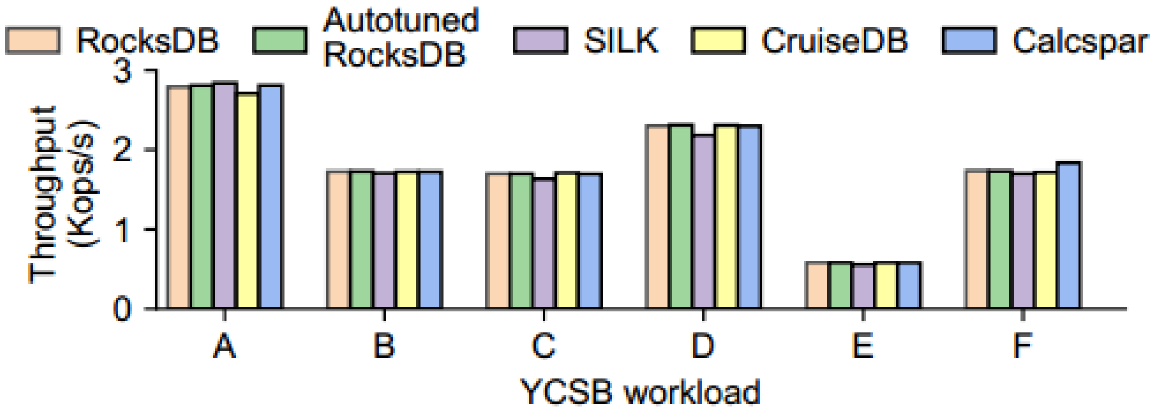


Figure 15: Throughput under YCSB workload.

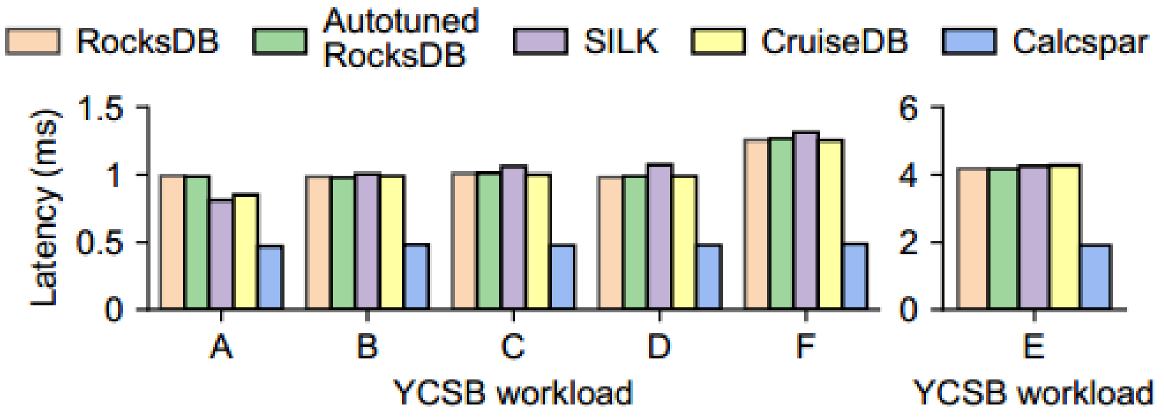
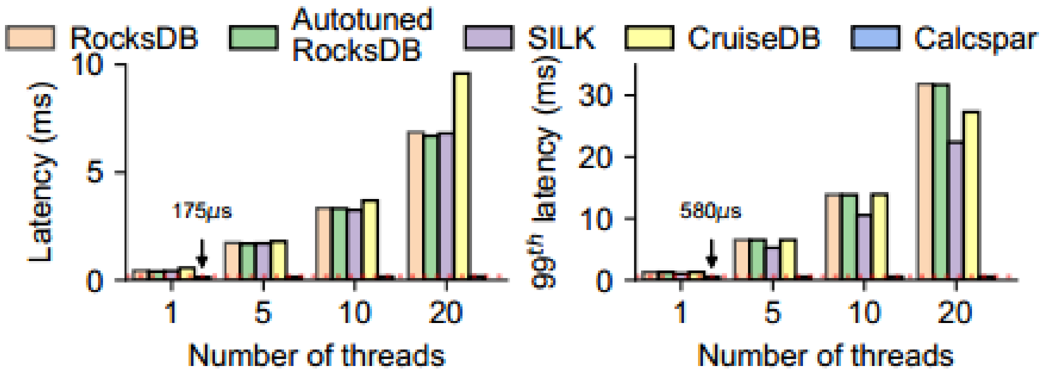


Figure 16: The 99th percentile latency under YCSB workload.

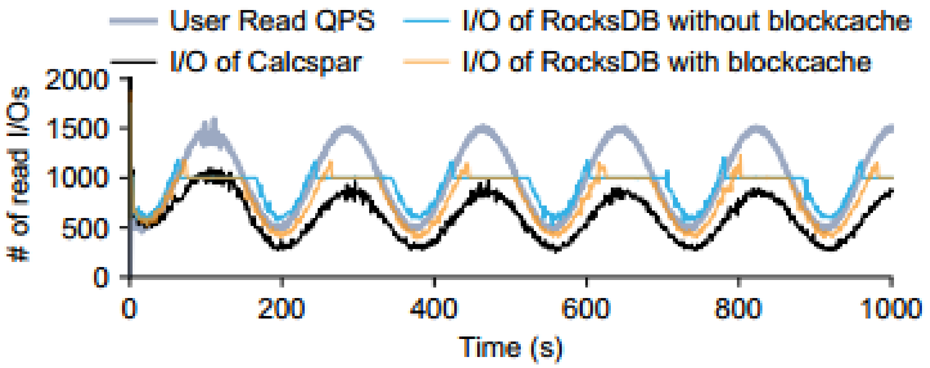
Congestion Mitigation Effectiveness

Avoid multi-thread congestion.

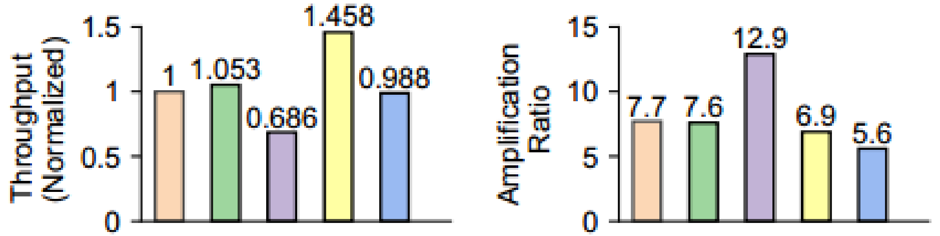


(a) Average latency. (b) 99^{th} percentile latency.

Cache effectiveness with cache



Impact of Opportunistic Compaction.



(a) Throughput of random write. (b) Write amplification.

Paper Summary

Contract model and latency characteristics of cloud storage

↓

LSM stores latency perforance

↓

Three challenges and solutions

Fluctuating request numbers

-> Fluctuation-Aware Caching

- Hotspot-Aware Proactive Prefetching
- Shift-Aware Passive Caching
- Cache Integration

Thread I/O competition

-> Congestion-Aware IOPS Allocating

- Multi-Priority Queues
- Dynamic Time Window Policy

Bulk write blocking

-> Opportunistic Compaction