MittOS: Supporting Millisecond Tail Tolerance with Fast Rejecting SLO-Aware OS Interface

Mingzhe Hao, Huaicheng Li, Michael Hao Tong, Chrisma Pakha,
Riza O. Suminto, Cesar A. Stuardo, Andrew A. Chien, and Haryadi S. Gunawi

ucare.cs.uchicago.edu
ceres.cs.uchicago.edu

No Millisecond TT (Tail Tolerance)

Nowadays low and stable latency is a critical key to success of many services. Unfortunately, most NoSQL systems serve requests with millisecond-level SLOs, but **none is tail tolerant** at this granularity.

	Def. TT	TO Val.	Fail-over	Clone	Hedged/ Tied
Cassandra	×	12s	✓	×	×
Couchbase	×	75s	×	×	×
HBase	×	60s	✓	✓	×
MongoDB	×	30s	×	×	×
Riak	×	10s	×	×	×
Voldemort	×	5s	✓	✓	×

Table: Tail tolerance in NoSQL.

Ineffectiveness of Current TT Methods

Wait-then-speculate (e.g. Hadoop MapReduce)
- Focuses on coarse-grained jobs (tens to hundreds of seconds)
- Reacts too late for millisecond-level tails

Cloning (e.g. Riak)
- Doubles/Triples IO intensity (cloning)
- Has to implement complicated revocation logics (tied requests)
- Must wait before retrying slow requests (hedged requests)

Snitching (e.g. Cassandra)
- May pick wrong metrics
- Does not work when noise is bursty

MittOS' Principle & Use-Case

MittOS provides operating system support that helps data-parallel applications **cut millisecond-level tail latencies**.
- **Accurately predicts** the latency of an IO based on **white-box knowledge** of resource managements
- **Promptly** returns EBUSY when IO SLO cannot be met
- Allows the application to failover to less-busy node **without waiting**

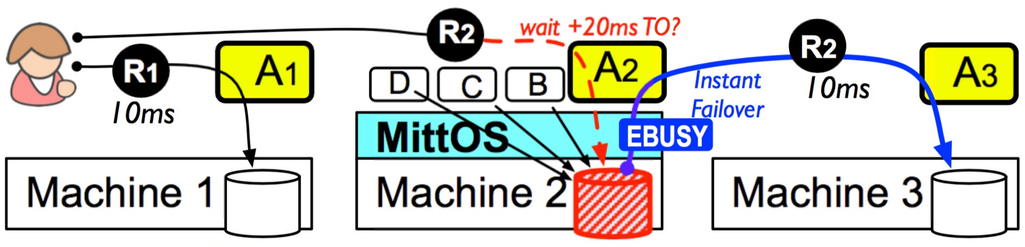

Figure: MittOS Deployment Model

Leveraging MittOS interface is easy and only requires applications to **add tens of LOC.**

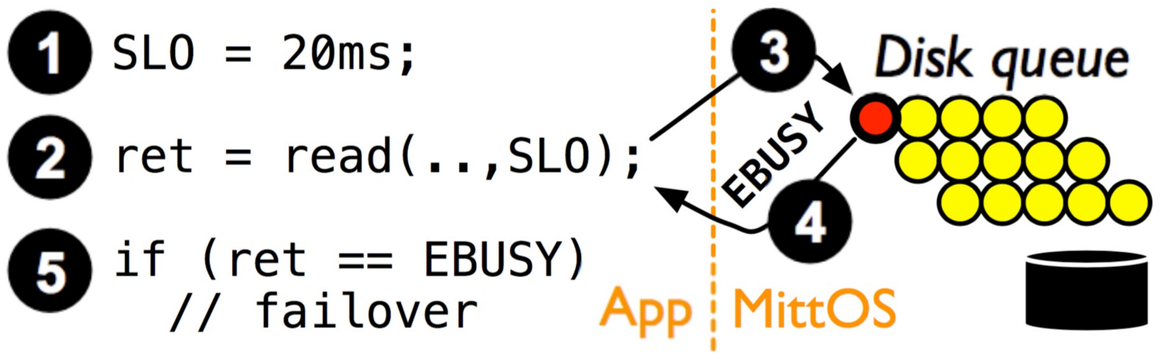

Figure: MittOS use-case illustration

Implementation & Experiment Results

We build MittOS within the storage stack:
- Disk: MittNOOP (noop scheduler) + MittCFQ (CFQ scheduler)
- SSD: MittSSD (Open-Channel SSD)
- Cache: MittCache (OS Cache)

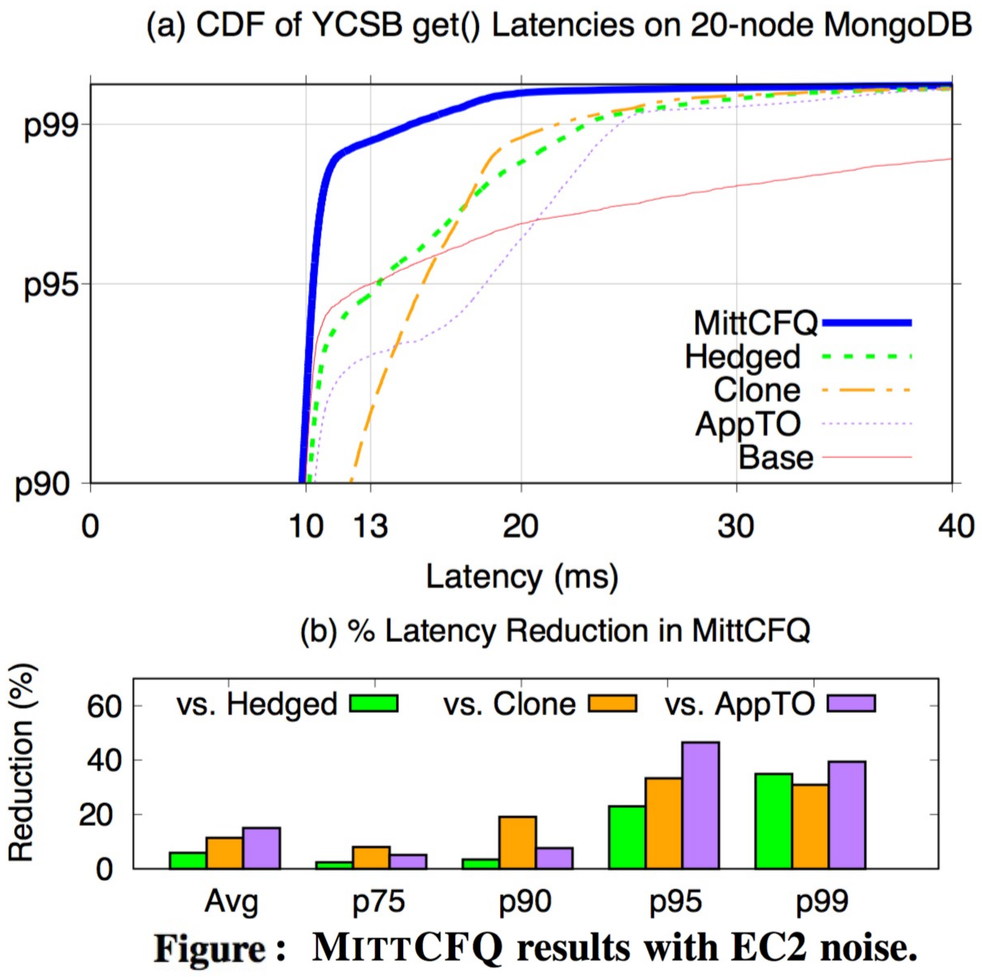

Figure: MittCFQ results with EC2 noise.

MittOS' **no-wait** approach helps reduce IO completion time up to **35%** compared to existing approaches.

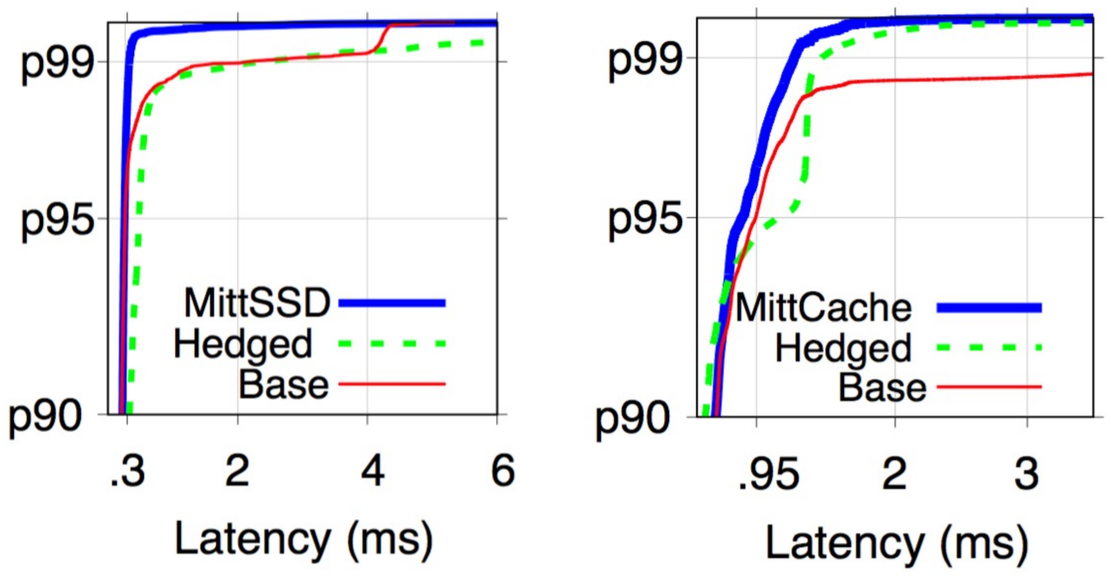

Figure: MittSSD vs. Hedged. & MittCache vs. Hedged.

Future Work

- Automatic adoption of storage devices via ML/DL techniques
- Incorporating settings of certainty/confidence for SLO
- Providing hints for applications to setup appropriate SLO deadline
- Extension of MittOS' principle to CPU, VM and runtime memory management, SMR drives, etc.

Appeared in CERES summit & the 26th ACM Symposium on Operating Systems Principles (SOSP), 2017