UNIVERSITY OF WATERLOO
FACULTY OF MATHEMATICS
David R. Cheriton School
of Computer Science

MicroFuge: A Middleware Approach to Providing Performance Isolation in Cloud Storage Systems [1]

Akshay Singh, Xu Cui, Benjamin Cassell, Bernard Wong and Khuzaima Daudjee
David R. Cheriton School of Computer Science

Background

A Cloud Scenario

- Cloud computing allows sharing of resource at the cost of reduced isolation.
- Storage systems are highly sensitive to performance interference.
- Lack of perform isolation will lead to unpredictable latencies.
- In worst case, a particular HTTP request may require 35 database lookups. [2]
- Amazon reported 100ms of latency cost them 1% in sales. [3]
- Google found an extra .5 seconds delay caused 20% drop in search traffic. [3]

Performance Isolation

- One can give clients dedicated resources, however this eliminates cost reduction.
- **MicroFuge** aims to meet clients' response time requirement in the shared environment.
- Represent response time requirements with **request deadlines**.
- Provide performance isolation by meeting **request deadlines**.

MicroFuge

A distributed caching and scheduling middleware that provides performance isolation.

Deadline Cache (DLC)

- Builds a performance model.
- Uses multiple LRU queues for deadline-ware evictions

Deadline Scheduler (DLS)

- Performs intelligent replica selection.
- Implements feedback-driven deadline-aware scheduling.
- Optionally performs admission control.

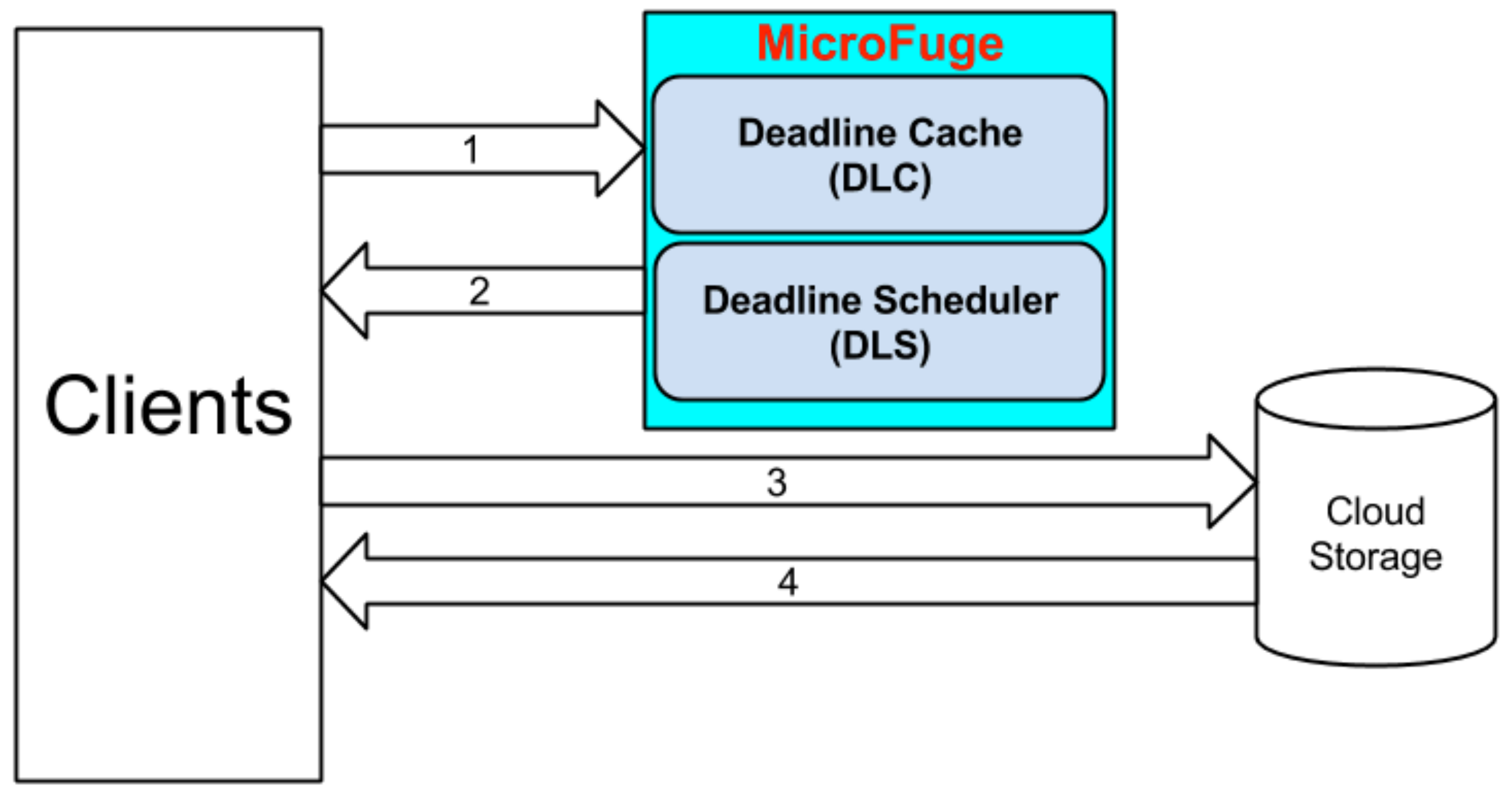

Figure 1: System Overview

Deadline Cache

DLC offers adaptive deadline-aware caching.

- Multiple LRU queues enable DLC to perform deadline-aware evictions.
- Each eviction victim is selected by computing the *Modified Recency Value*

$$\frac{Current\_Timestamp - Stored\_Timestamp}{Queue\_Specific\_Divisor} \tag{1}$$

- **DLC** uses an adaptive policy that considers both the client request rate for each deadline range and the underlying system's performance to update the adaptive divisors.

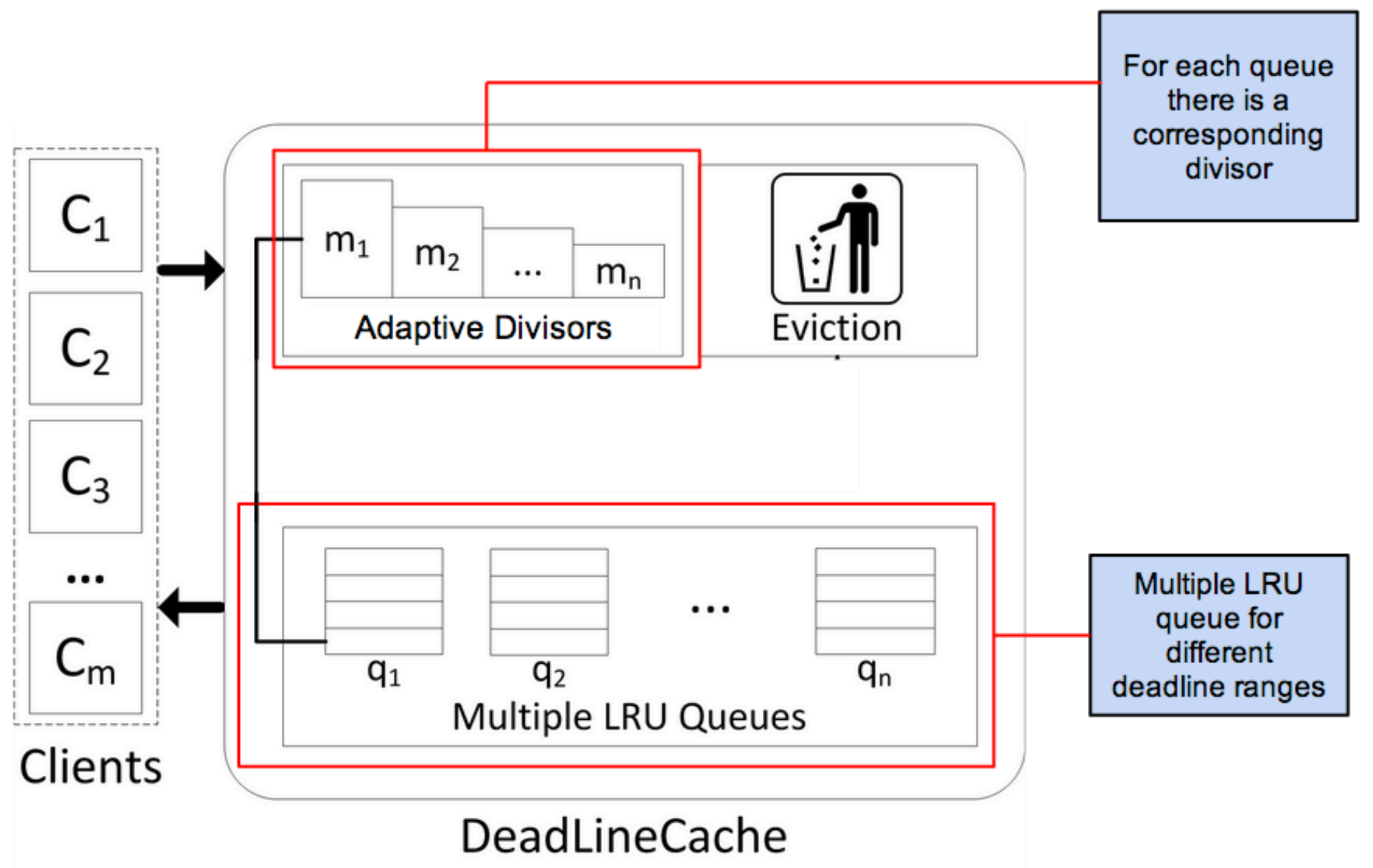

Figure 2: Deadline Cache

Deadline Scheduler

With **MicroFuge**'s distributed design, each **DLS** is responsible for scheduling client access to a subset of distributed data servers. Each scheduler performs three tasks to provide performance isolation in the scheduling layer.

- **DLS** will select the replica which is most likely to meet request's deadline.
- In order to make the selection, **DLS** relies on the latency modeling component which uses previous request latencies to predict incoming request's latency.
- There are cases where server load just exceeds its capacity. **DLS** additionally provides an optional admission control mechanism which performs early rejection of requests which are likely to miss their deadlines.

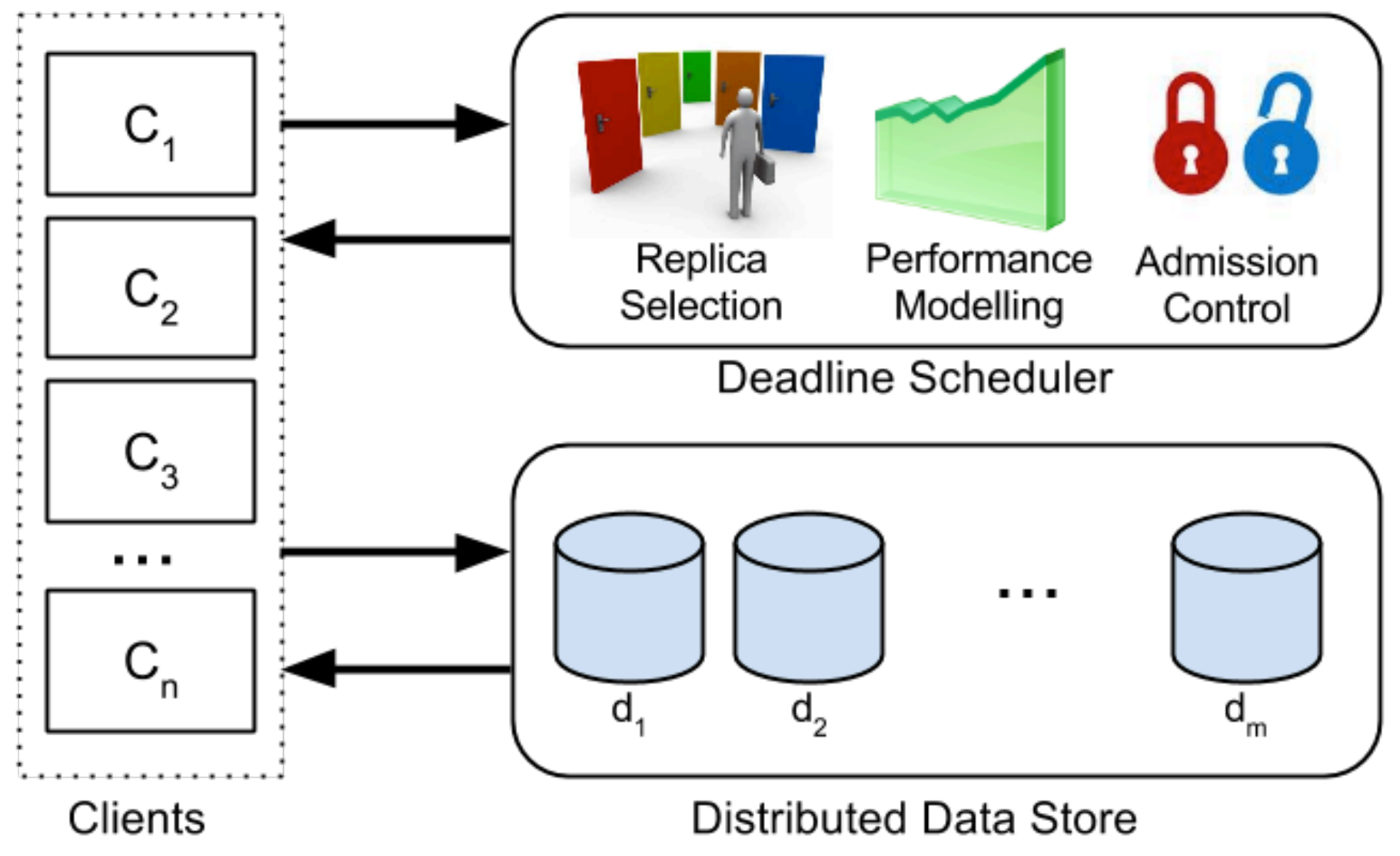

Figure 3: Deadline Scheduler

References

[1] Akshay K. Singh, Xu Cui, Benjamin Cassell, Bernard Wong, and Khuzaima Daudjee. MicroFuge: A Middleware Approach to Providing Performance Isolation in Cloud Storage Systems. In *International Conference on Distributed Computing Systems (ICDCS)*, Madrid, Spain, July 2014.

[2] N. Farrington and A Andreyev. Facebook's data center network architecture. In *Optical Interconnects Conference, 2013 IEEE*, May 2013.

[3] Greg Linden. Make Data Useful. http://www.scribd.com/doc/4970486/Make-Data-Useful-by-Greg-Linden-Amazon-com.

Performance Evaluation

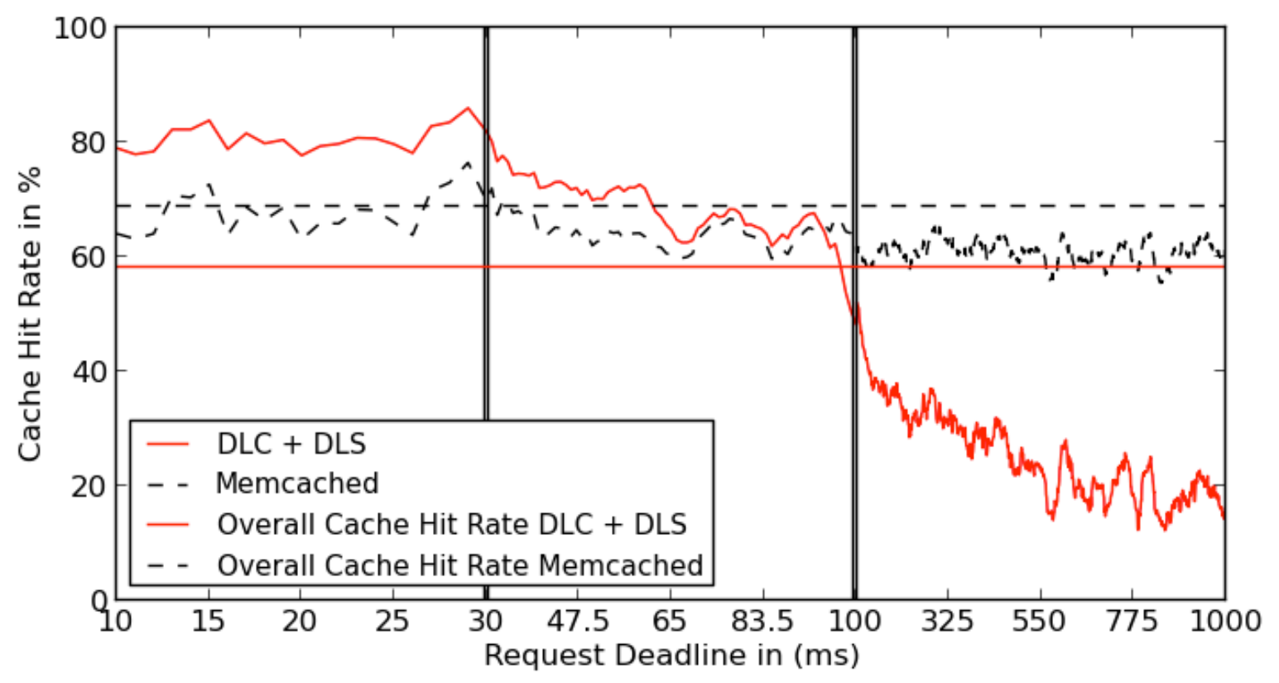

Figure 4: Cache Hit Rate

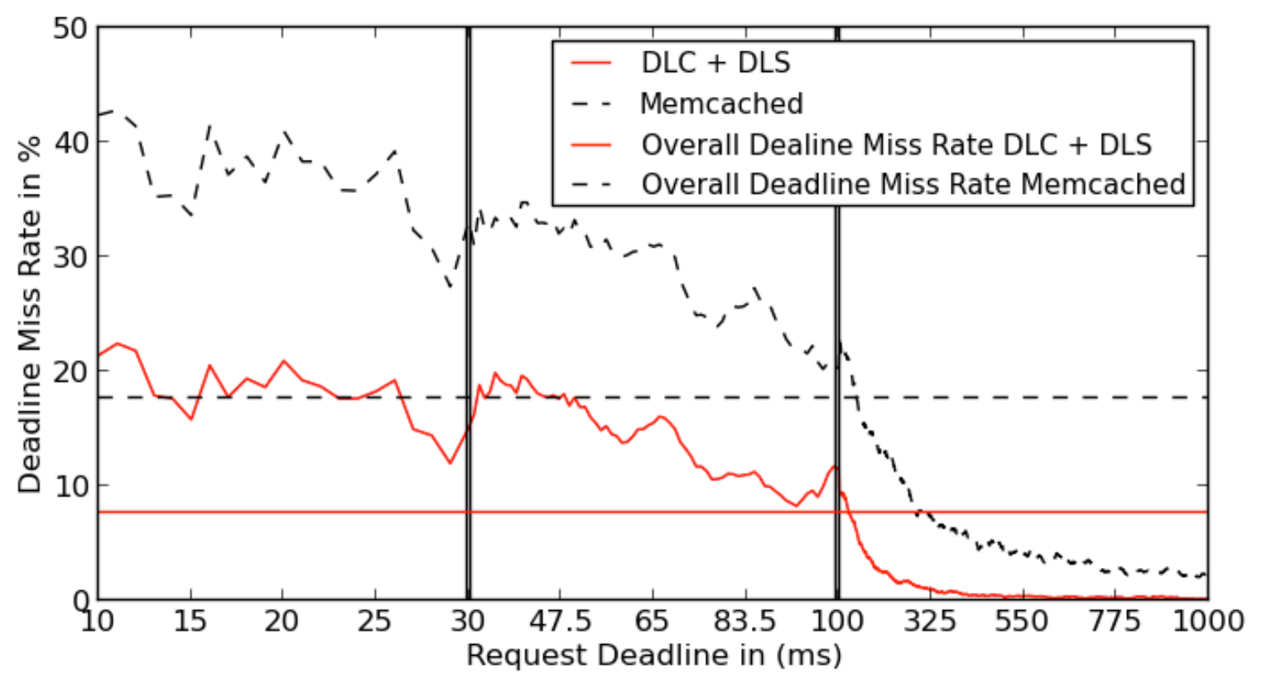

Figure 5: Deadline Miss Rate

Experimental Setup

- Twenty-node test cluster on AWS. Each cluster node is an m1.medium EC2 instance.
- Benchmarking System: Yahoo! Cloud Serving Benchmark (YCSB). Modified to assign different ranges of deadlines to each key.

Conclusion

- Predictable performance is necessary in multi-tenant environments.
- MicroFuge tackles the performance isolation problem with its deadline-aware caching and scheduling middleware.
- MicroFuge reduces deadline miss rate from 17.5% to 7.7% and it can be as low as 4.7% if we turn on the admission control.