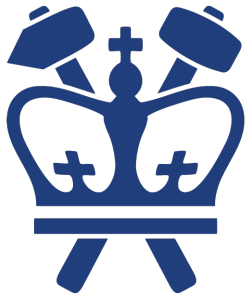
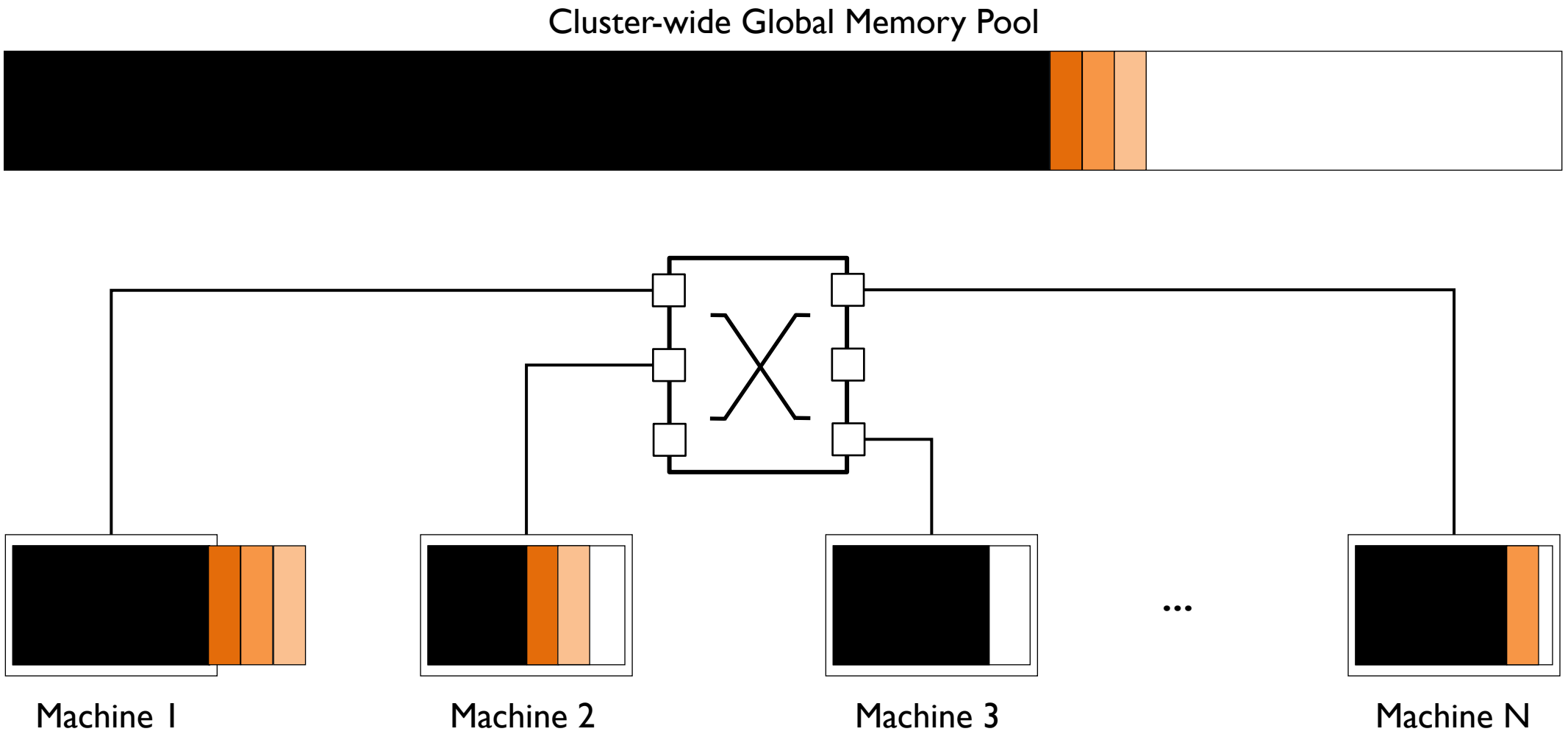
Hydra : Resilient and Highly Available Remote Memory

Youngmoon Lee*, Hasan Al Maruf*, Mosharaf Chowdhury, Asaf Cidon, Kang G. Shin

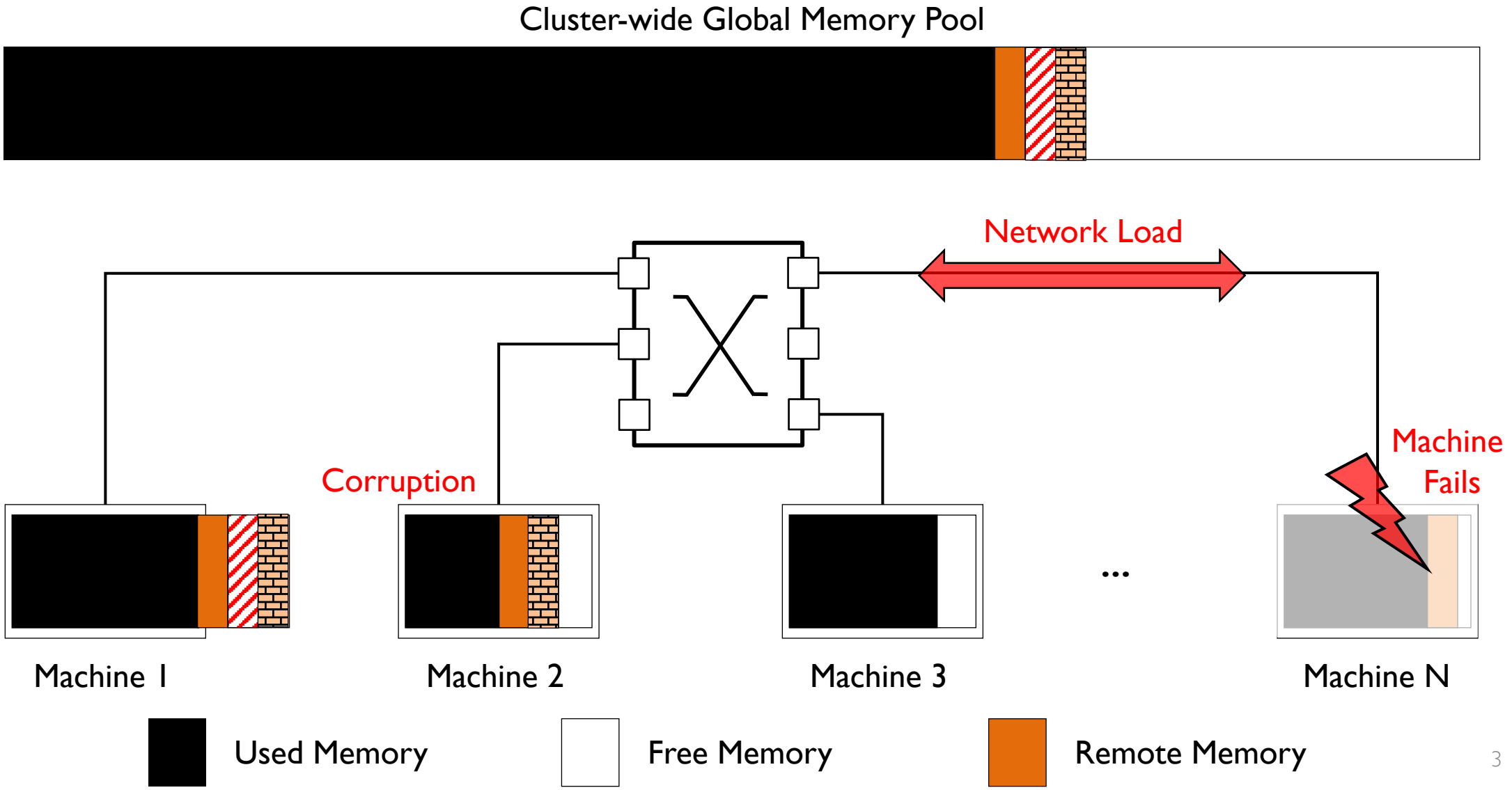


* equal contribution

Remote Memory



Remote Memory is Vulnerable



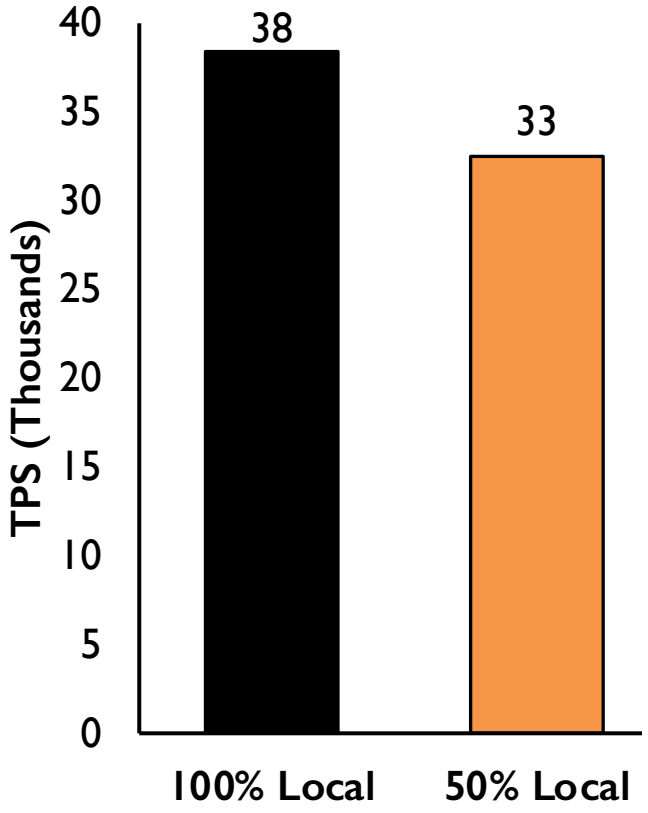
µs-scale Access Requirement

Latency requirement for preferable performance[1]

3-5 µs

[1] P. X. Gao et al. "Network requirements for resource disaggregation" OSDI'16.

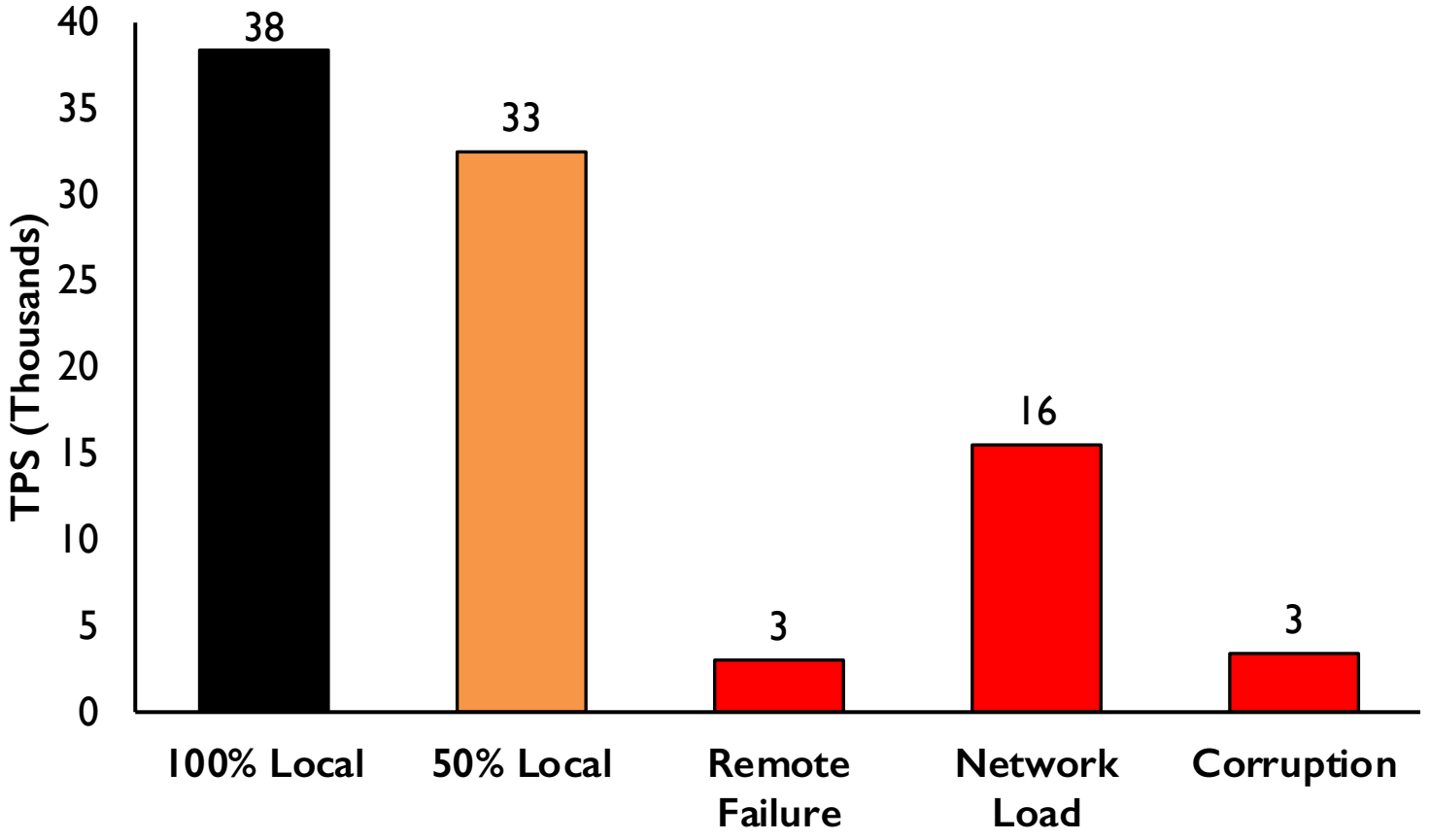
Remote Memory Performs Great!



Throughput drops by
13%
at **50%** less memory

TPC-C on VoltDB w/ Infiniswap

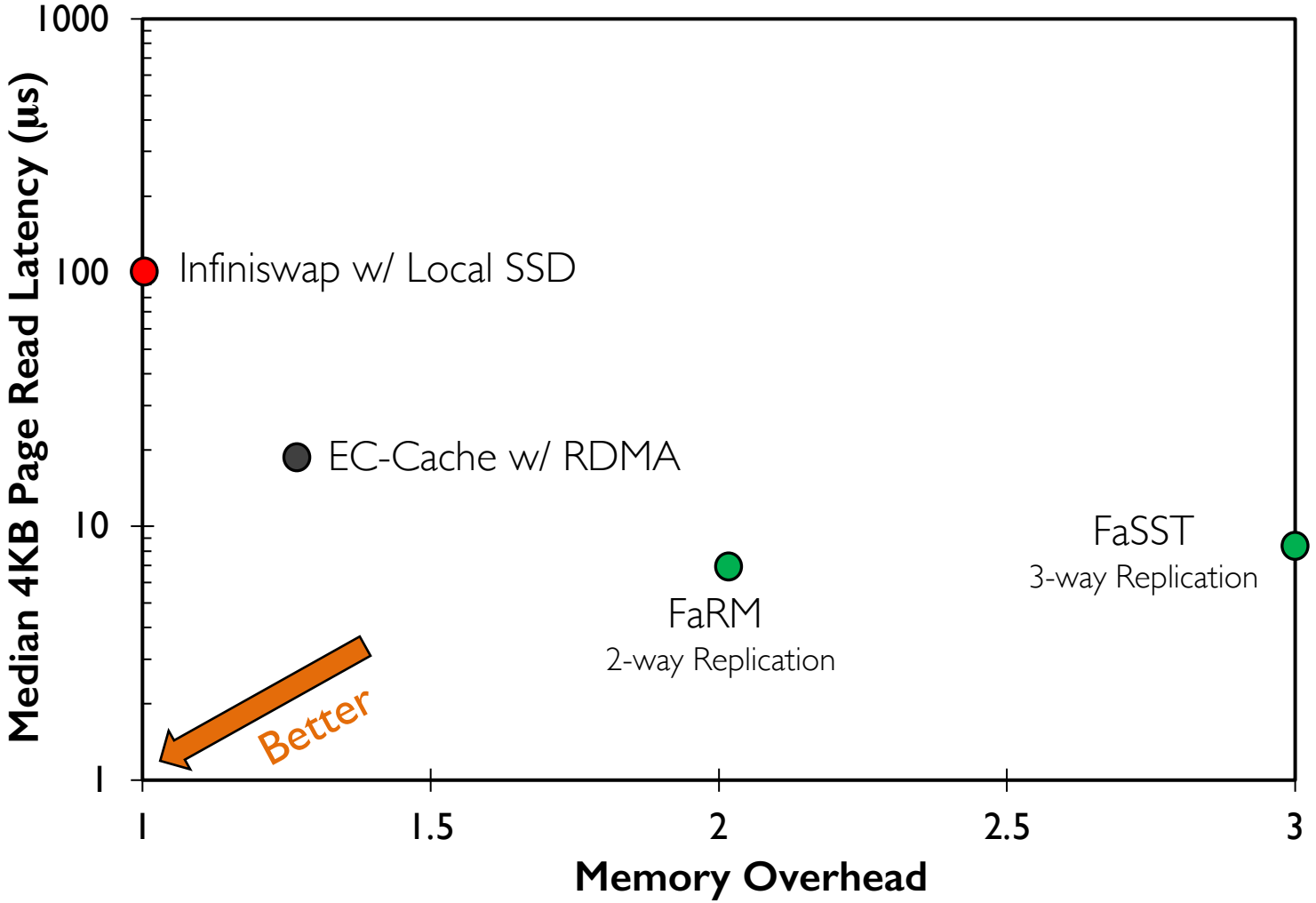
Performs Great w/o Uncertainties



TPC-C on VoltDB w/ Infiniswap

Throughput drops by
50-90%

Performance Tradeoff for Resiliency



Disk backup-based resiliency
100 µs–3 ms latency

In-memory replication
2–3X memory overhead

In-memory erasure coding
>**10 µs** avg. latency

Challenges in Erasure-Coded Memory

1. **High coding latency**
   - *data size*
   - *stragglers and errors*
   - *interruptions*
   - *copy overhead*

2. **Availability under simultaneous failures**
   - *data placement*
   - *load balancing*

Hydra

Low-latency and Highly Available Resilient Remote Memory

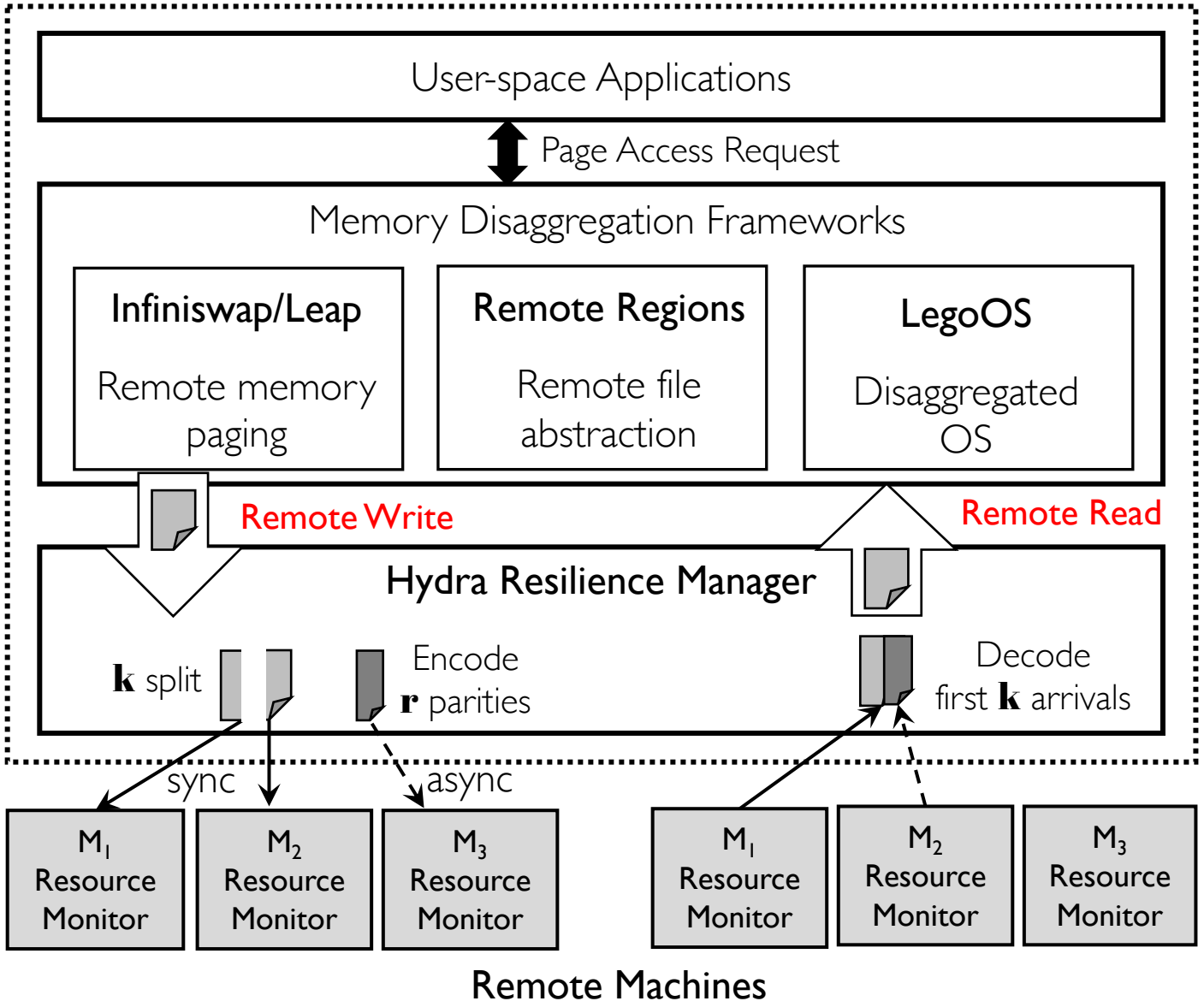
online erasure coded remote memory

- single-digit µs tail memory access latency,
- CodingSets for better load balancing and availability,
- pluggable to existing frameworks

without modifying any

- applications, or
- hardware

Remote Memory w/ Hydra



Hydra Components

Data Plane

Asynchronous encoded write

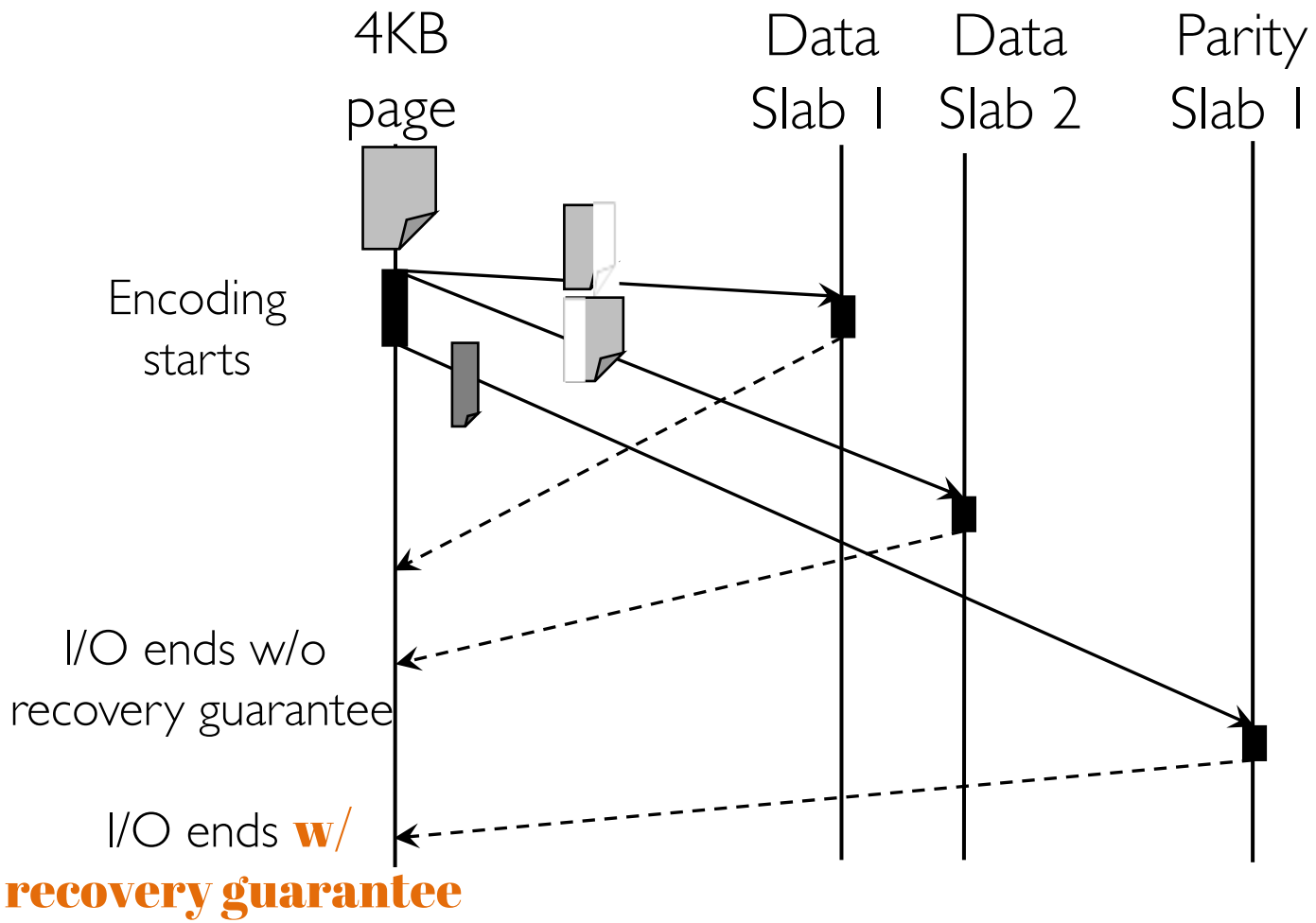
Late-binding resilient read

Run-to-completion

In-place coding

Control Plane

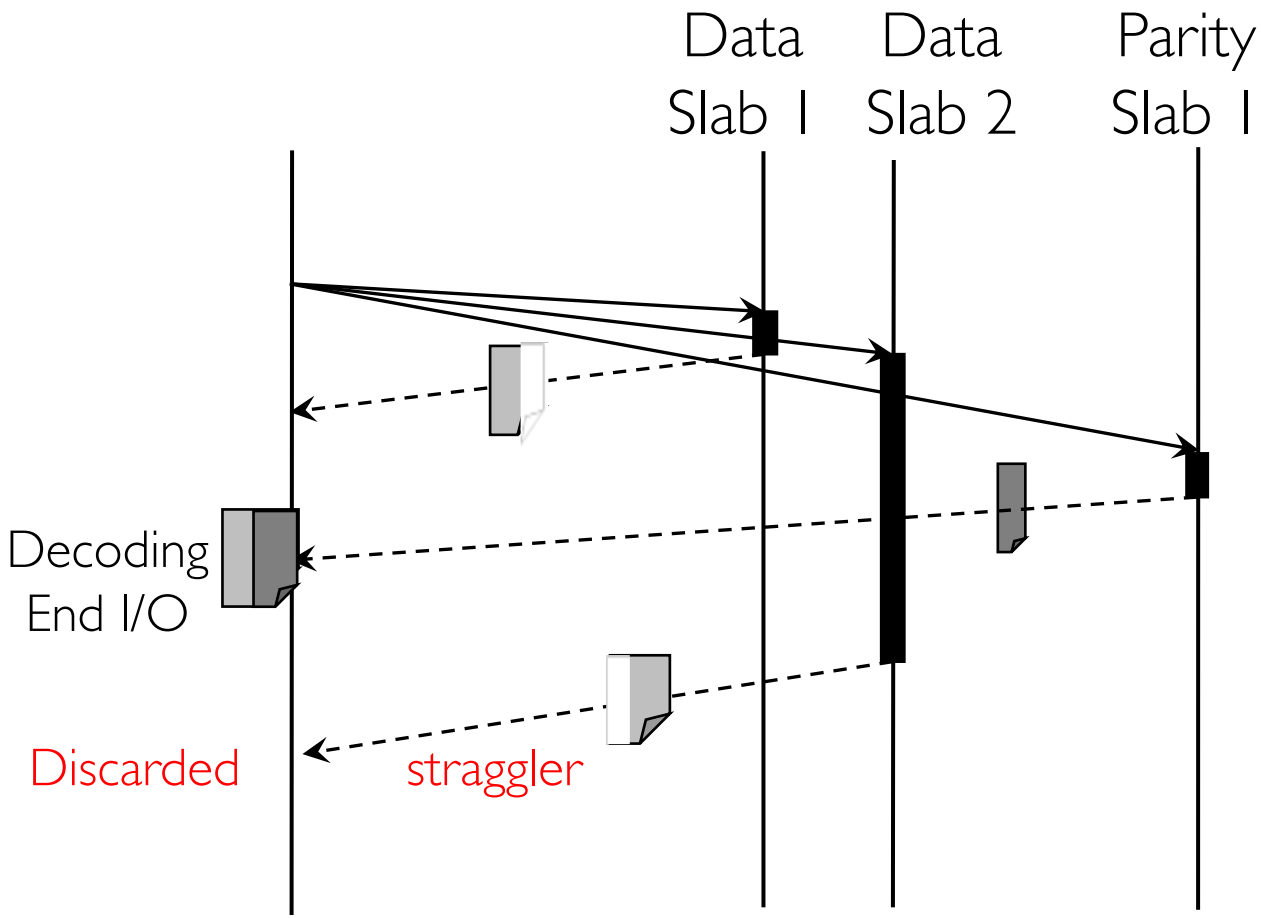
Hydra Datapath – Asynchronously Encoded Write



Encodes **4KB** page

- divides into **k** data splits
- async. data split write and encoding
- writes **r** parity split after encoding
- remote I/O ends with **k**+**r** writes

Hydra Datapath – Late-binding Read



k splits are enough to decode a page

- **k**+**Δ** reads to avoid straggler
- decodes when **k** splits arrives
- extra read avoids latency spike
- **Δ** = **1** is often enough

Hydra Components

Data Plane

Asynchronous encoded write

Late-binding resilient read

Run-to-completion

In-place coding

Control Plane

CodingSets for high availability

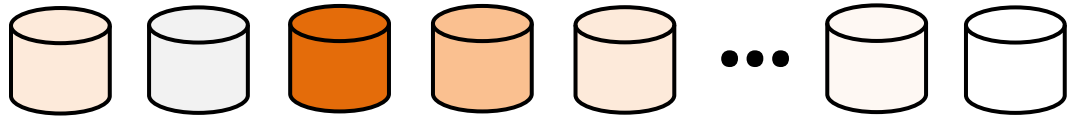
Adaptive slab allocation

Decentralized eviction

Background slab regeneration

CodingSets for Load Balancing and Availability

Distribute **k**+**r** splits across **N** memory nodes w/ varied loads



k+**r**+**l** nodes form
a unique failure domain

pick **k**+**r** least loaded nodes

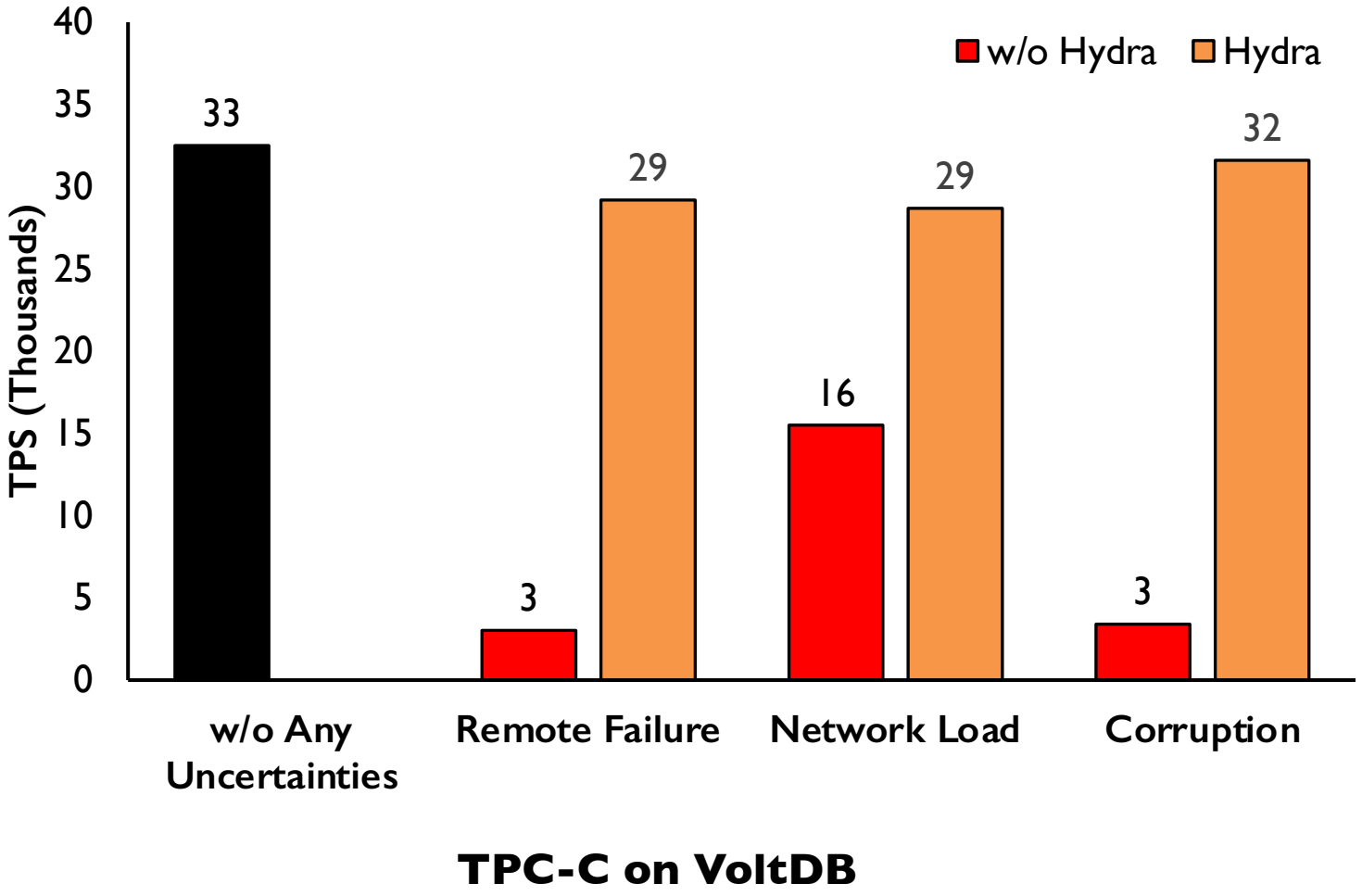
Evaluation

Deploy and evaluate over a 50-machine cluster w/
56 Gbps InfiniBand network

Memory Disaggregation Frameworks

- *Disaggregated VMM:* **Infiniswap, Leap**
- *Disaggregated VFS:* **Remote Regions**

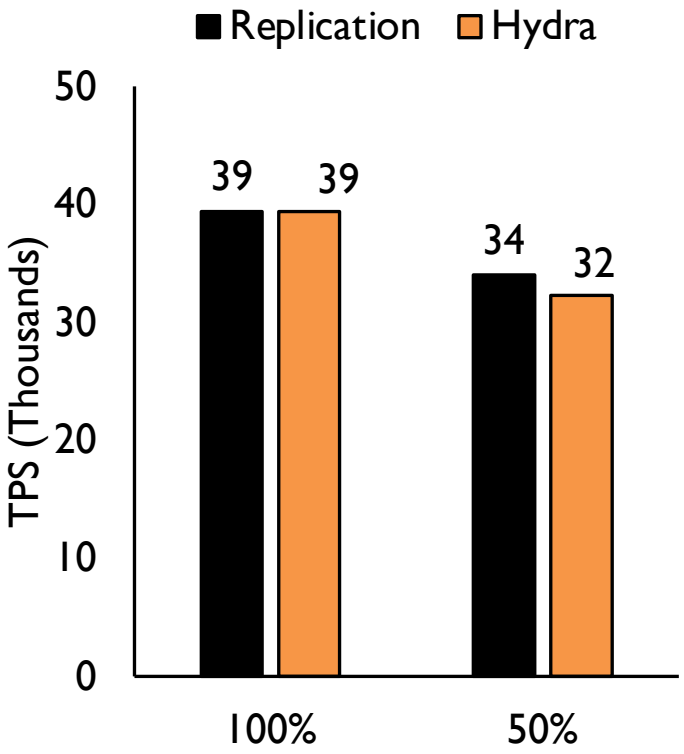
Perform Great During Uncertainties



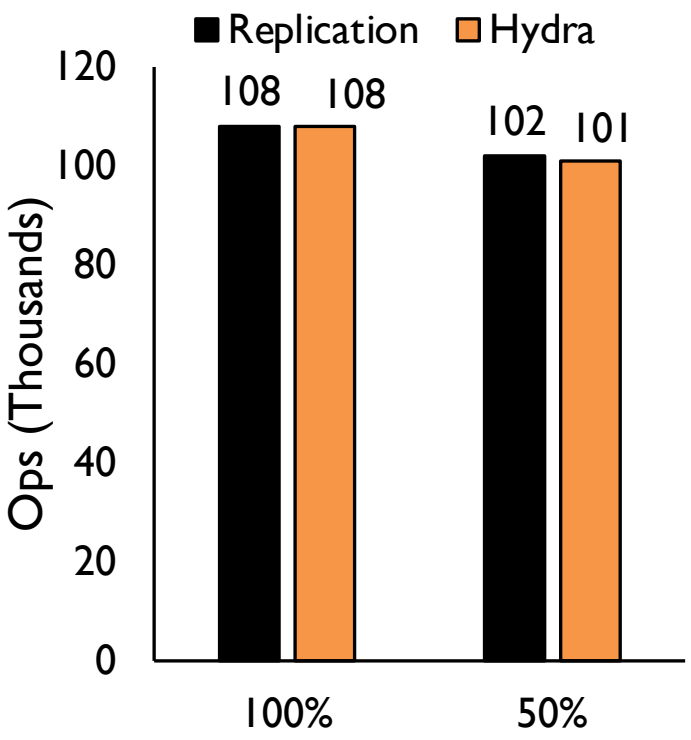
TPC-C on VoltDB

similar performance even in the presence of **uncertainties**

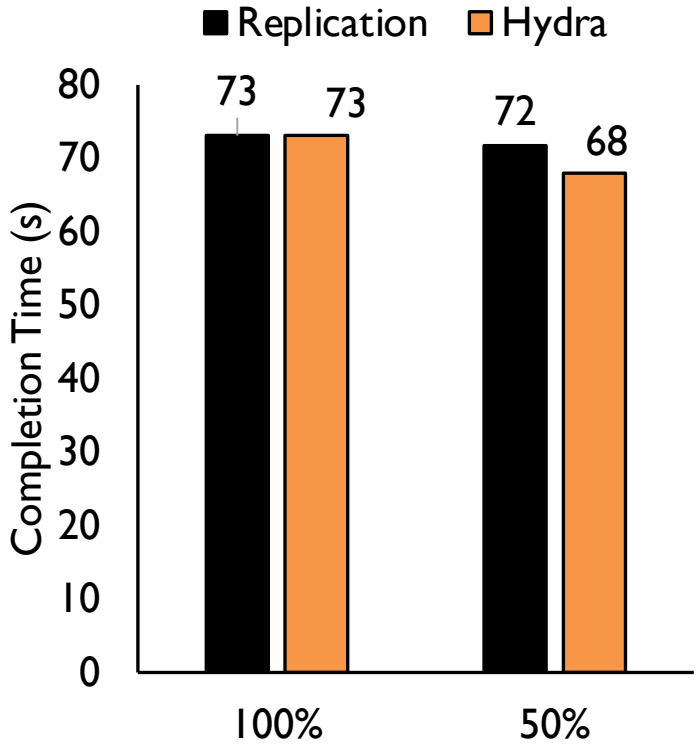
Replication-like Performance at Lower Overhead



TPC-C on VoltDB



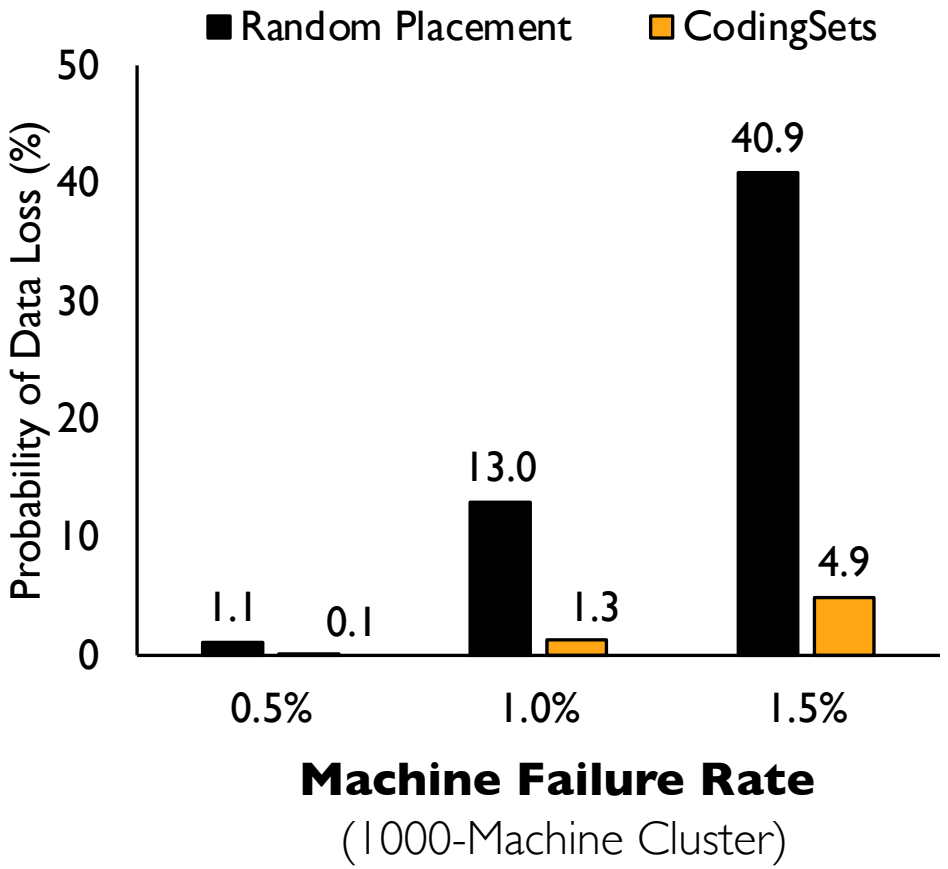
FB Workload on Memcached

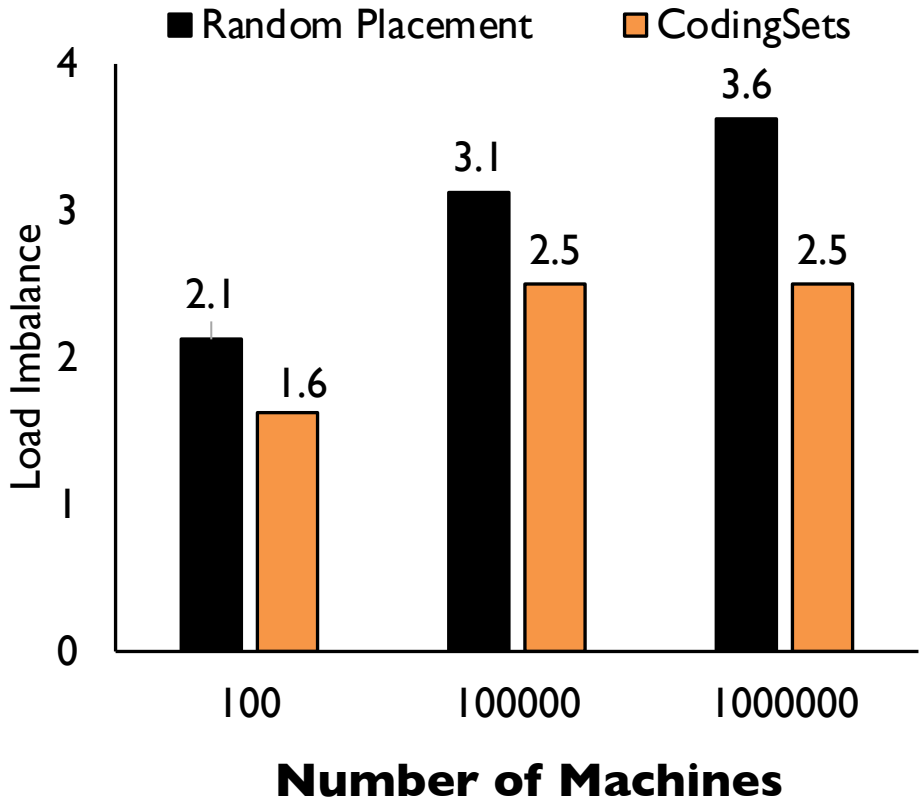


PageRank on PowerGraph

≈1X performance at 1.6X lower memory overhead

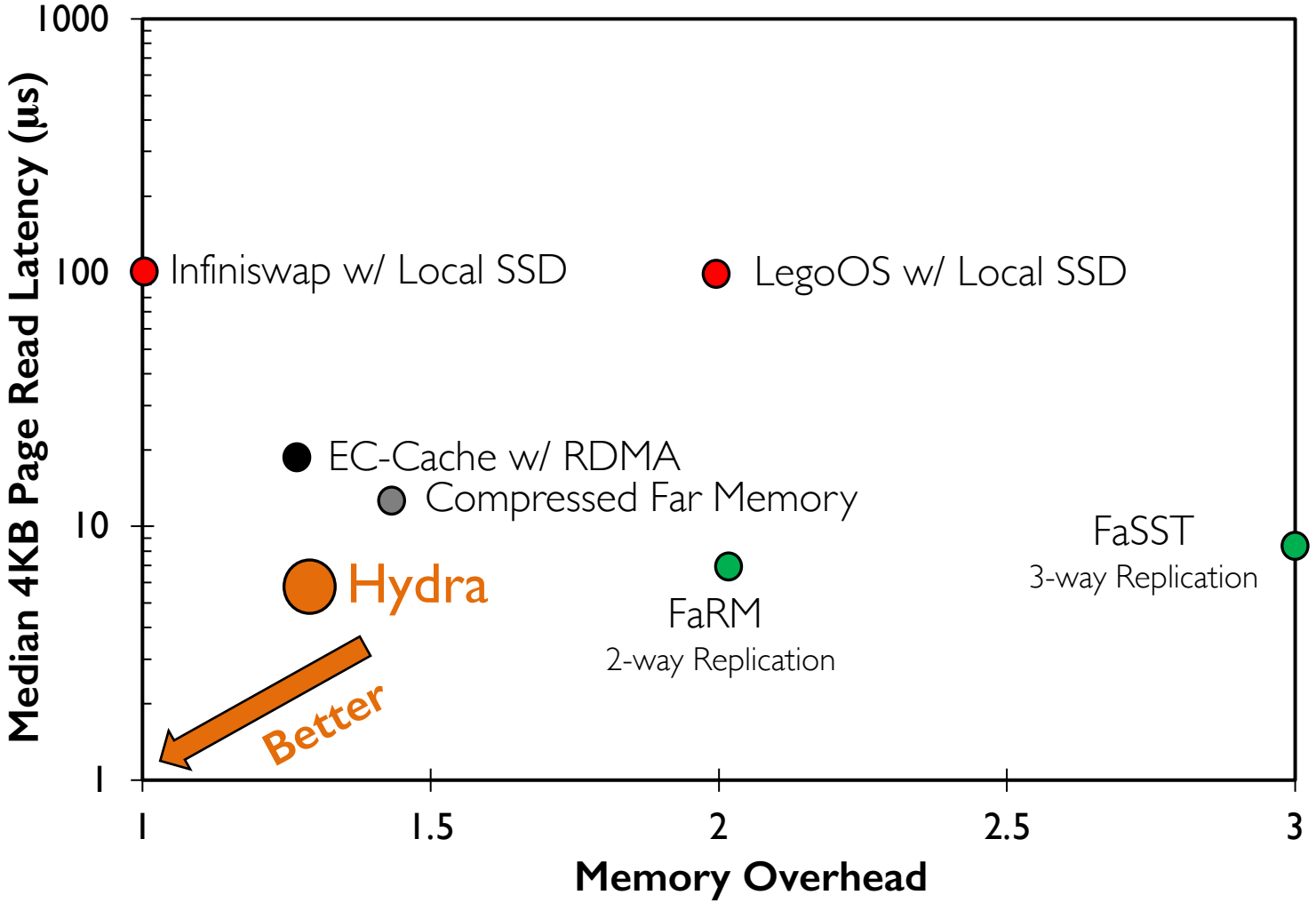
Better Availability and Load Balancing w/ CodingSets





10X higher availability at **1.5X** better load balancing

Performance Tradeoff w/ Hydra



Hydra

1.25X memory overhead

<10 µs p^{99th} latency

Hydra

Low-overhead & Highly Available Resilient Remote Memory

source code available at
https://github.com/SymbioticLab/Hydra

Drop-in solution for online erasure coded memory

- optimized data path for **<10µs** tail memory access latency
- CodingSets featured data-placement for **10X** higher availability
- replication-like performance at **1.6X** lower memory overhead

without modifying any

- application, or
- hardware