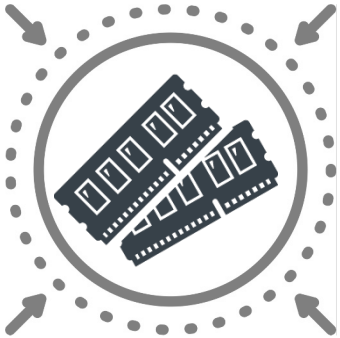
UCLA Samueli School of Engineering

Canvas: Isolated and Adaptive Swapping for Multi-Applications on Remote Memory

Chenxi Wang*, **Yifan Qiao***(co-first author), Haoran Ma, Shi Liu, Yiying Zhang, Wenguang Chen, Ravi Netravali, Miryung Kim, Guoqing Harry Xu

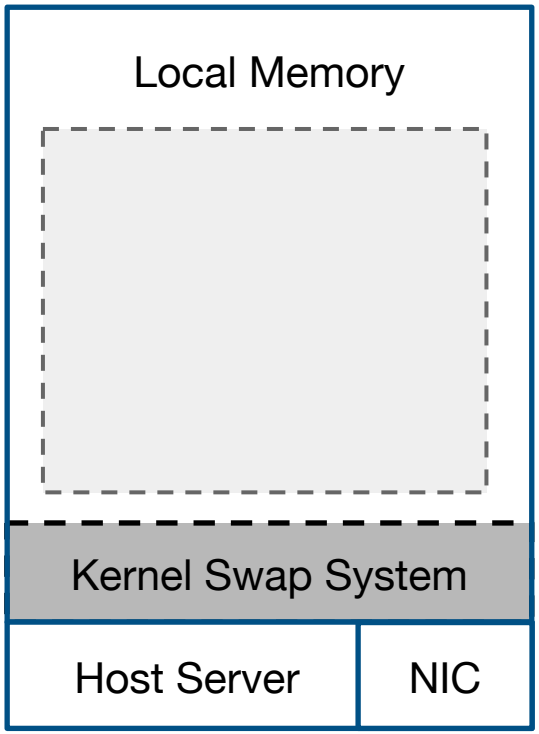
Memory Challenge in Datacenters



“Memory capacity wall”

Memory underutilization in datacenters

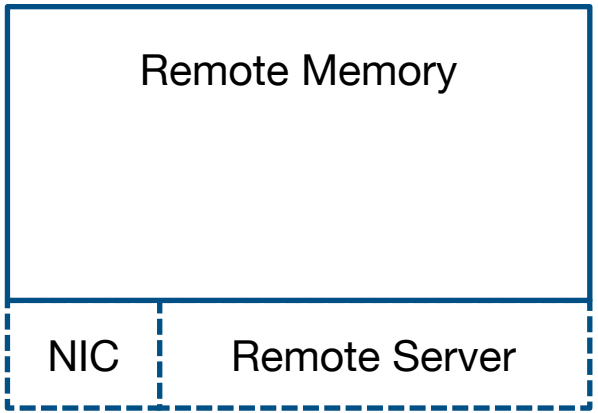
Remote Memory Systems



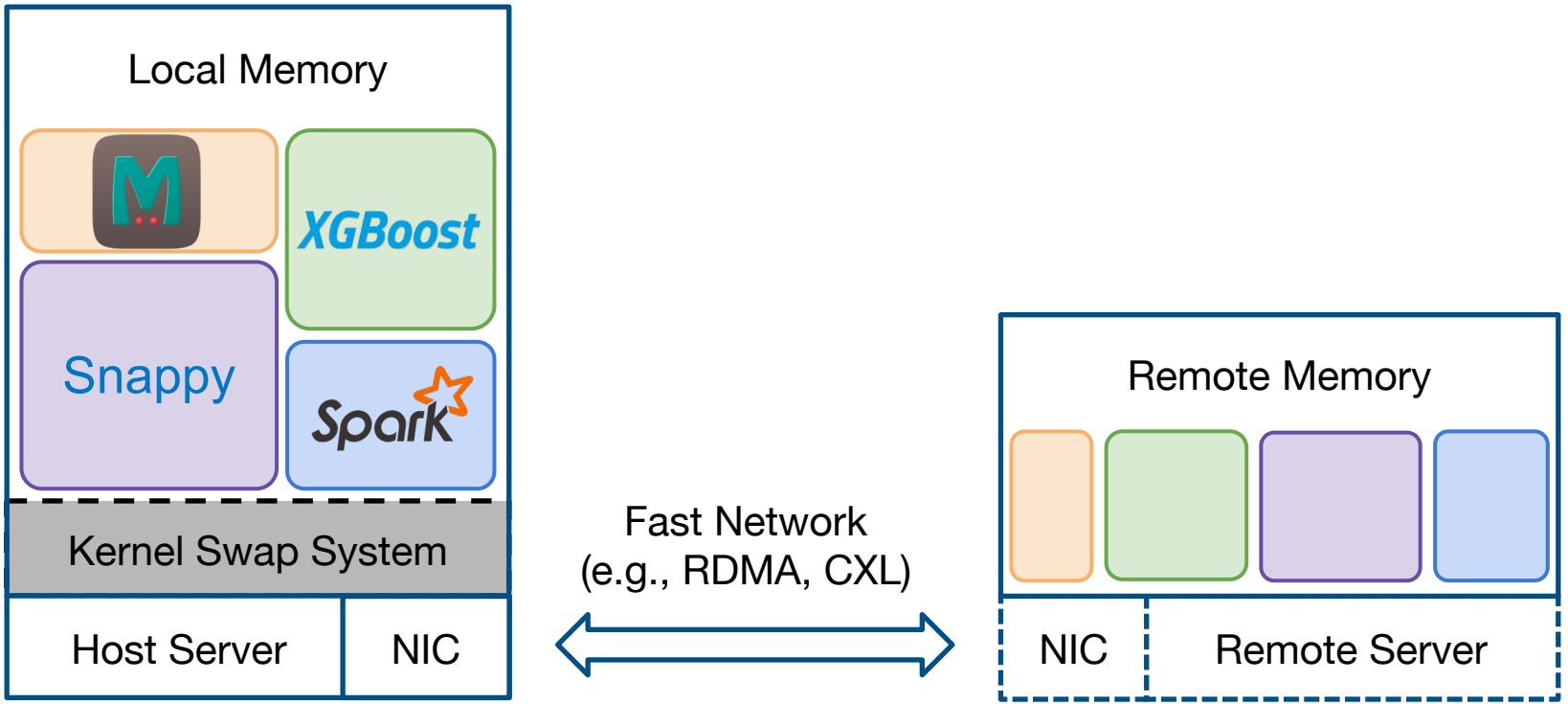
Efficiency: Fastswap [EuroSys'20], AIFM [OSDI'20]

Reliability: Hydra [Fast'22], Carbink [OSDI'22]

Multi-Tenancy ?



Multi-Tenant Cloud on Remote Memory



Kernel Swap Performs Poorly in Shared Settings

Experiment with four real-world cloud applications.

State of the art: Fastswap [EuroSys'20]

Google's file compression service

Memcached

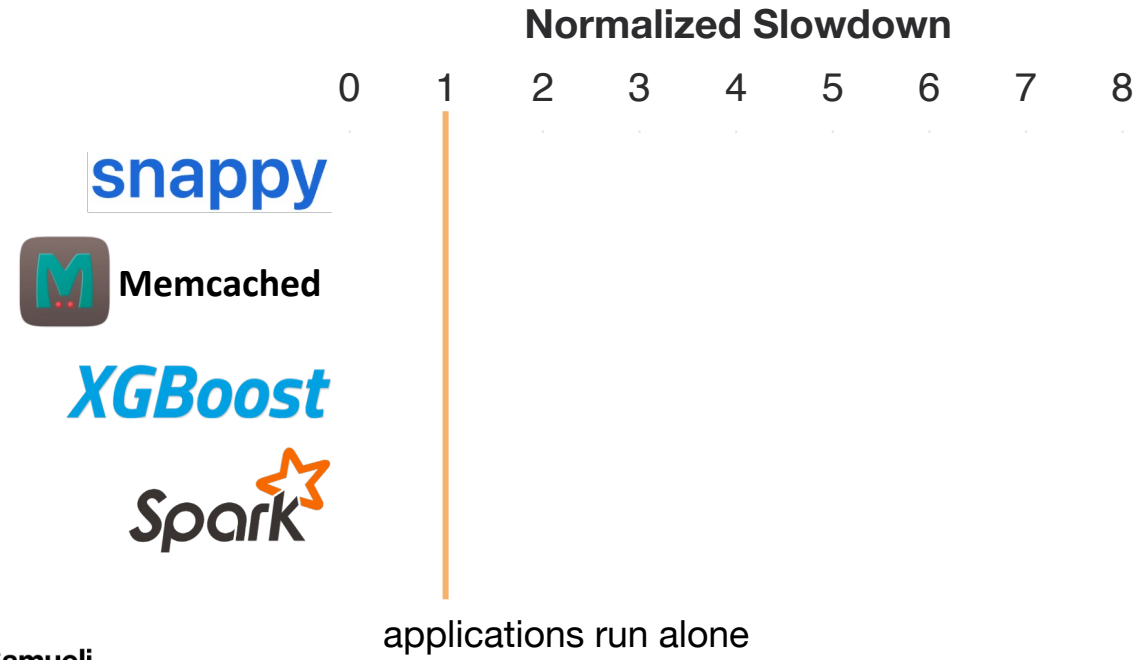
In memory key-value cache

CPU-based ML framework

Data-processing framework

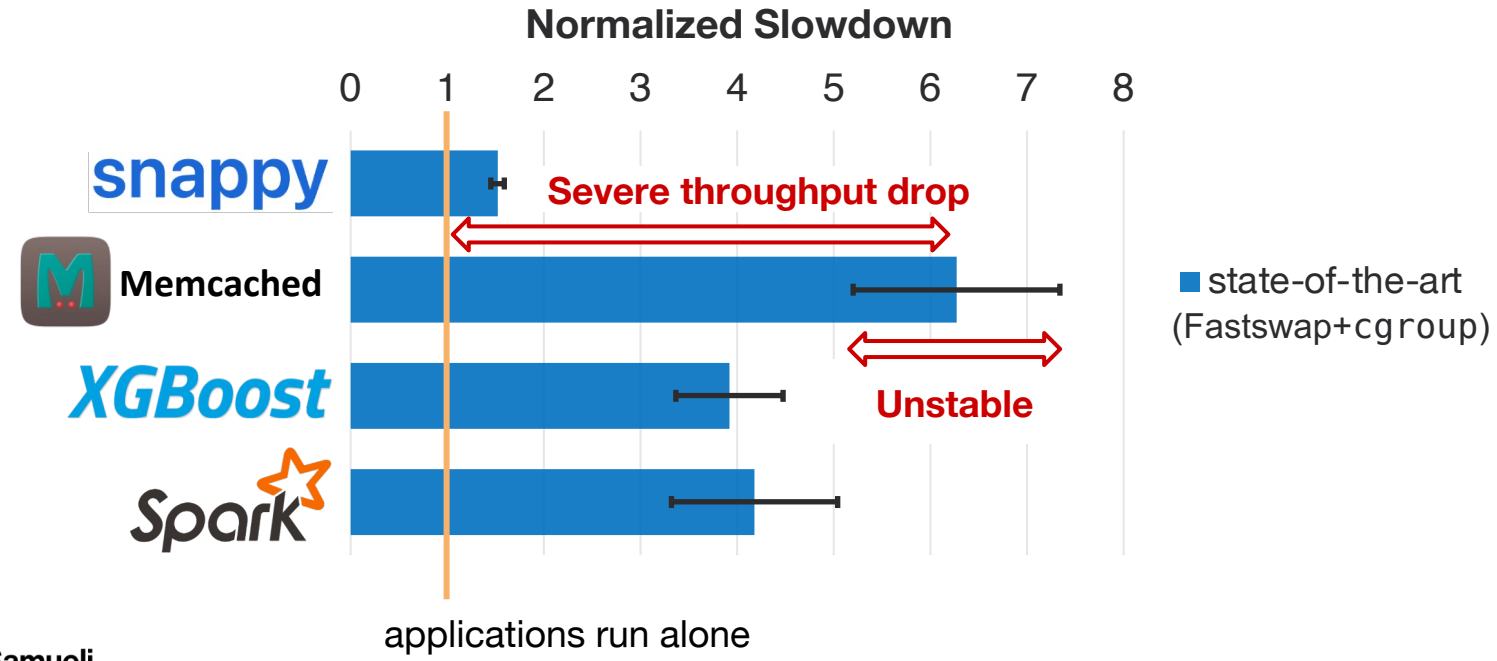
Kernel Swap Performs Poorly in Shared Settings

Run each application alone with 25% of their working sets cached in local memory.

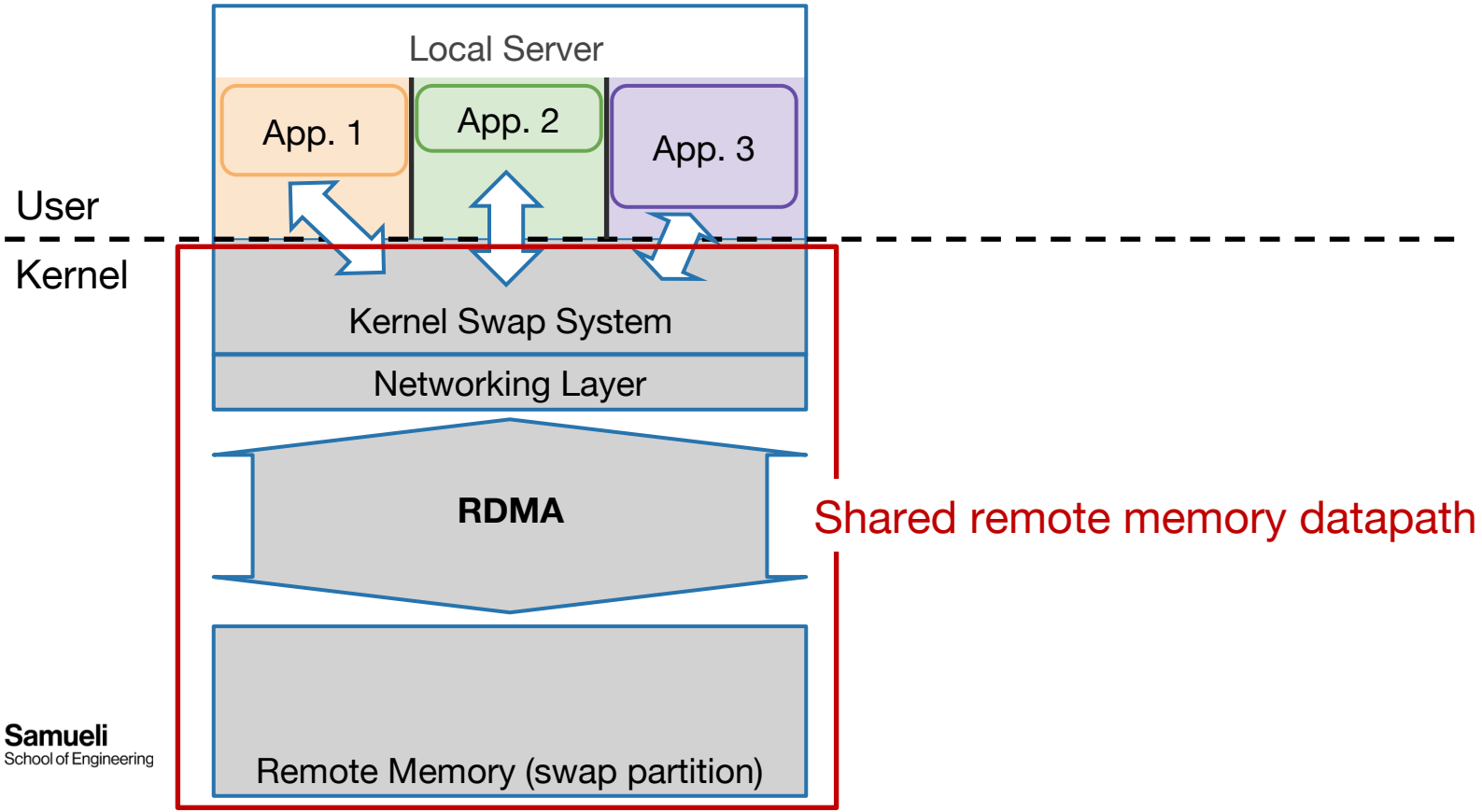


Kernel Swap Performs Poorly in Shared Settings

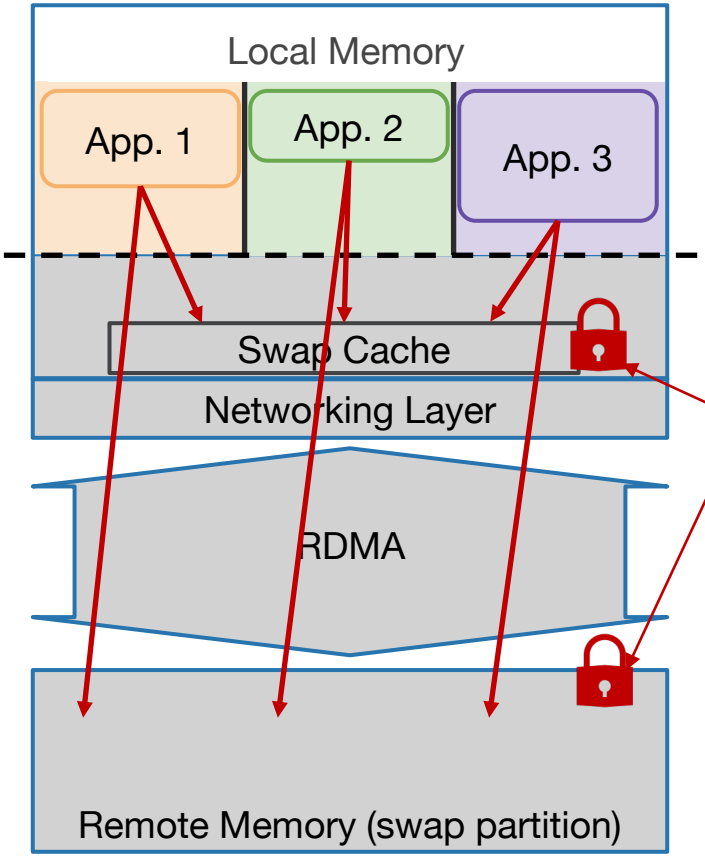
Co-run four real-world applications with 25% of their working sets cached in local memory.



Where Does The Overhead Come From?



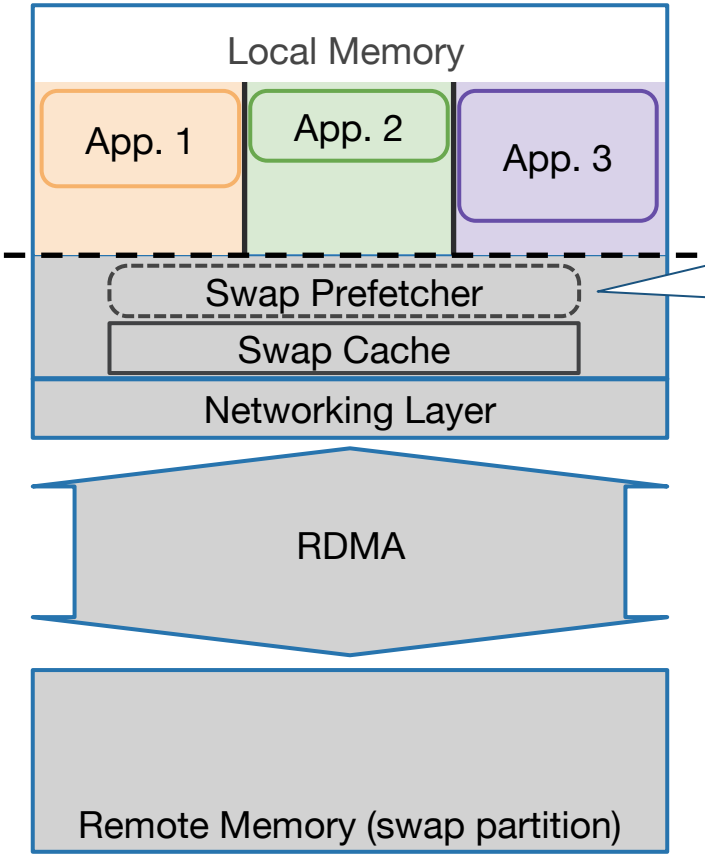
Interference #1: Shared Swap Resources

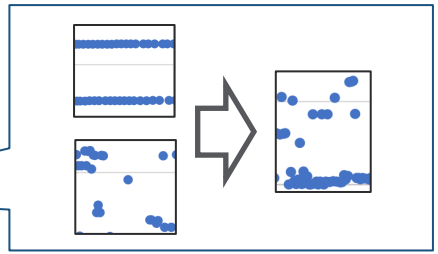


Lock contention when alloc./free

Swap throughput drops by *56%*

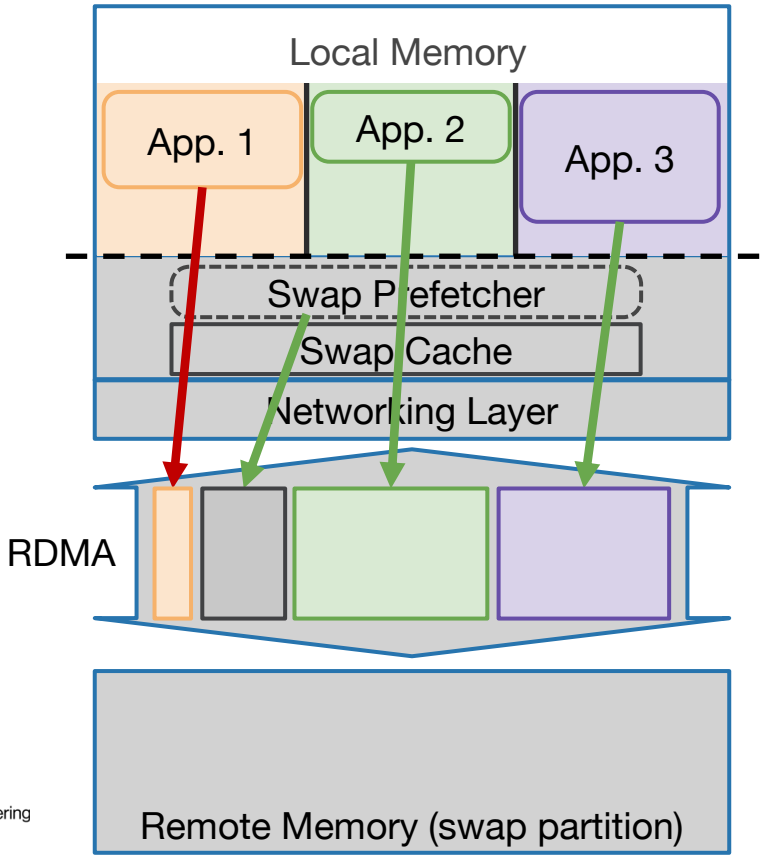
Interference #2: Mixed Access Patterns





Prefetch contribution drops by *33%*

Interference #3: RDMA Bandwidth Competition

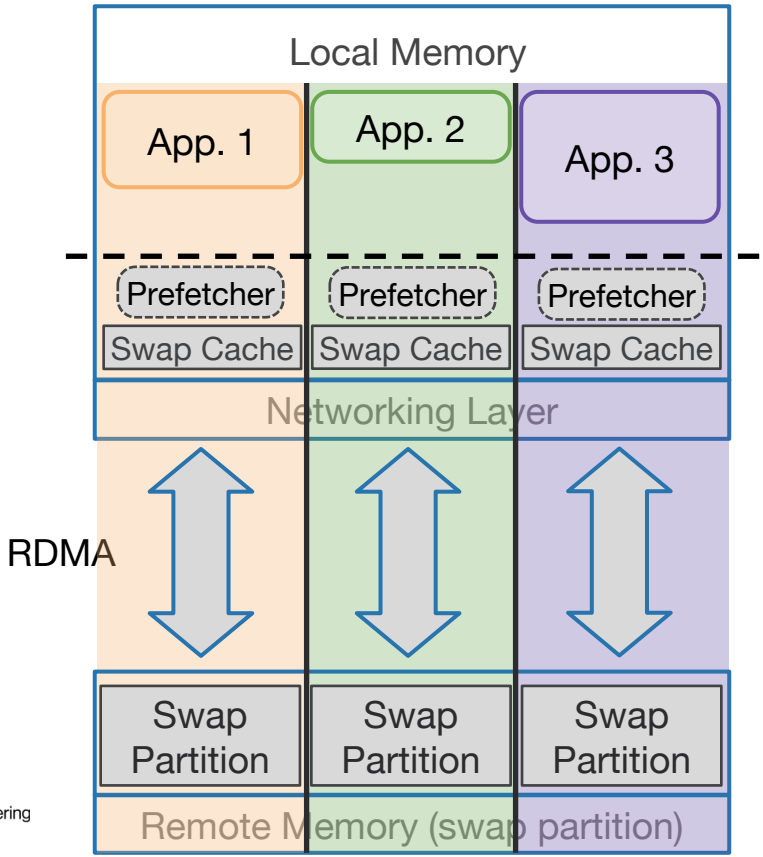


Competition occurs:

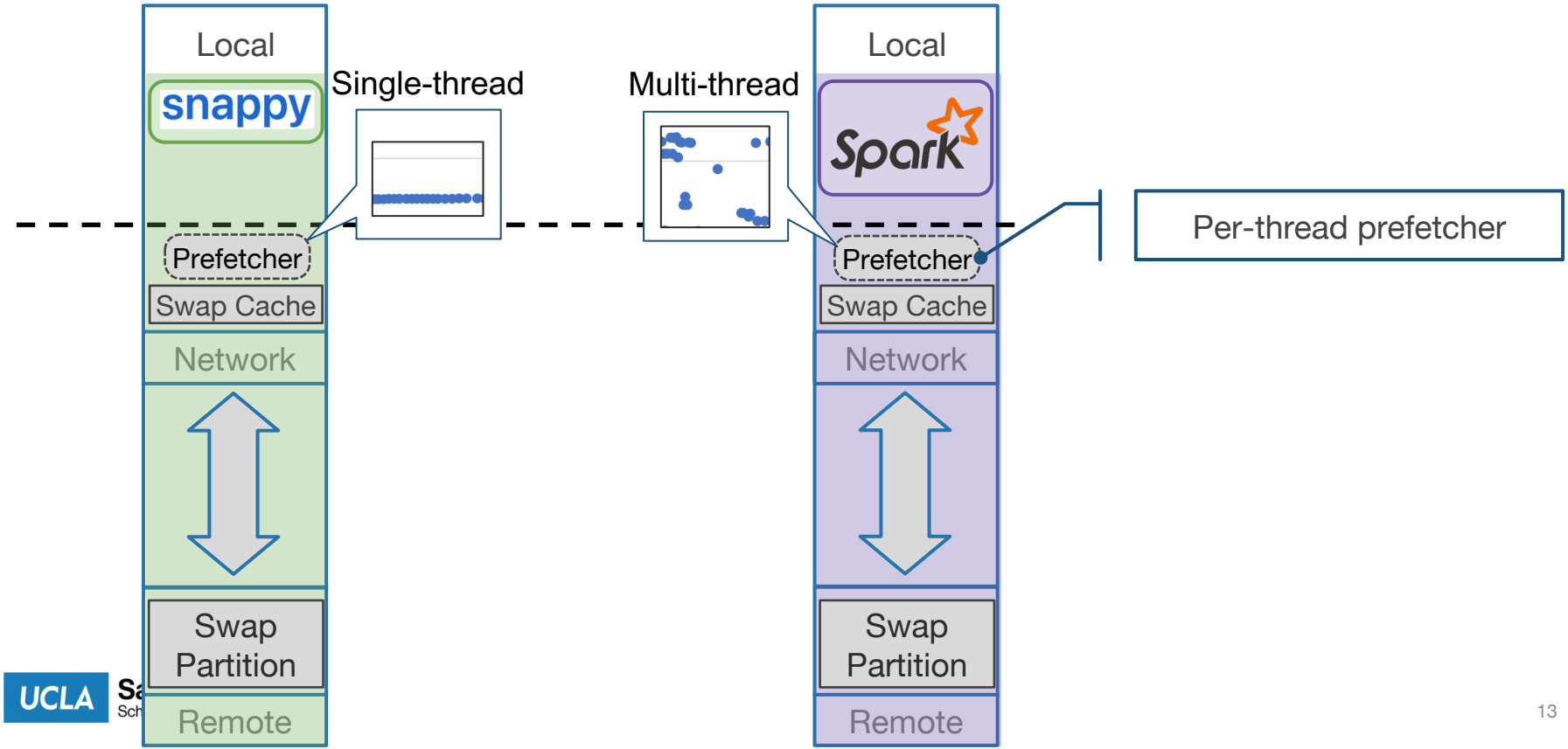
- among applications
- between application and prefetcher

P99 round-trip latency increases by *2.1x*

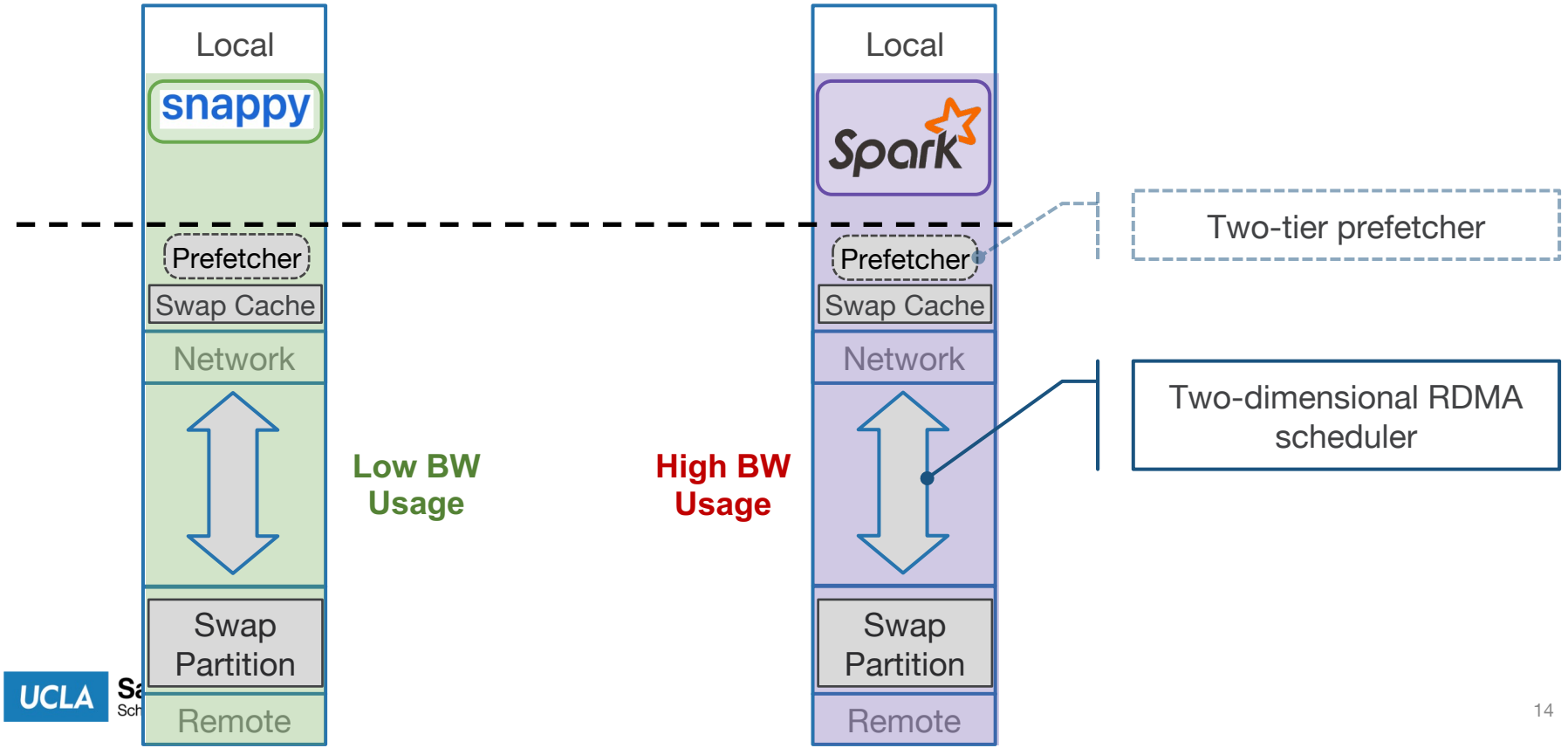
Canvas Design: Holistic Isolation



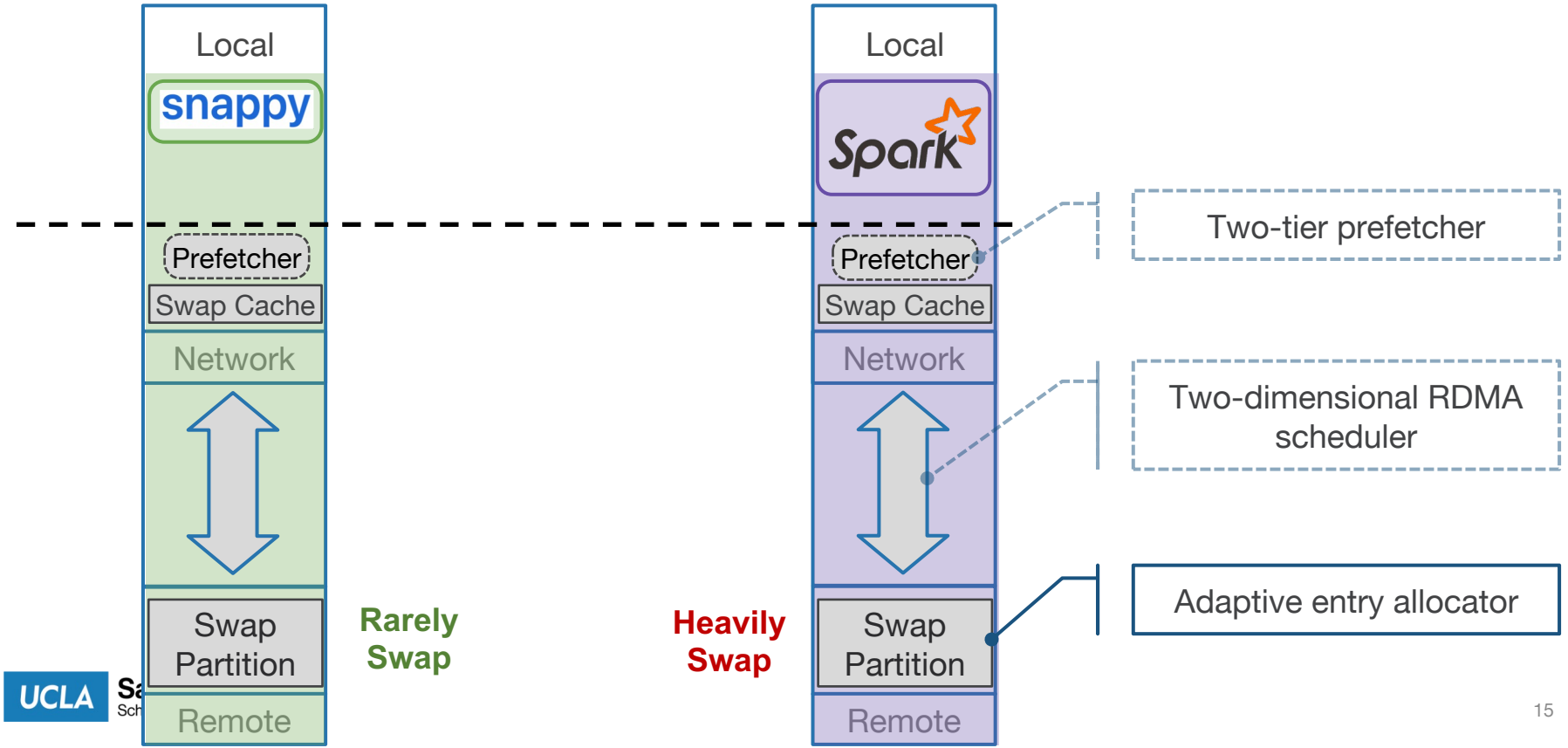
Isolation Enabled Adaptive Optimizations



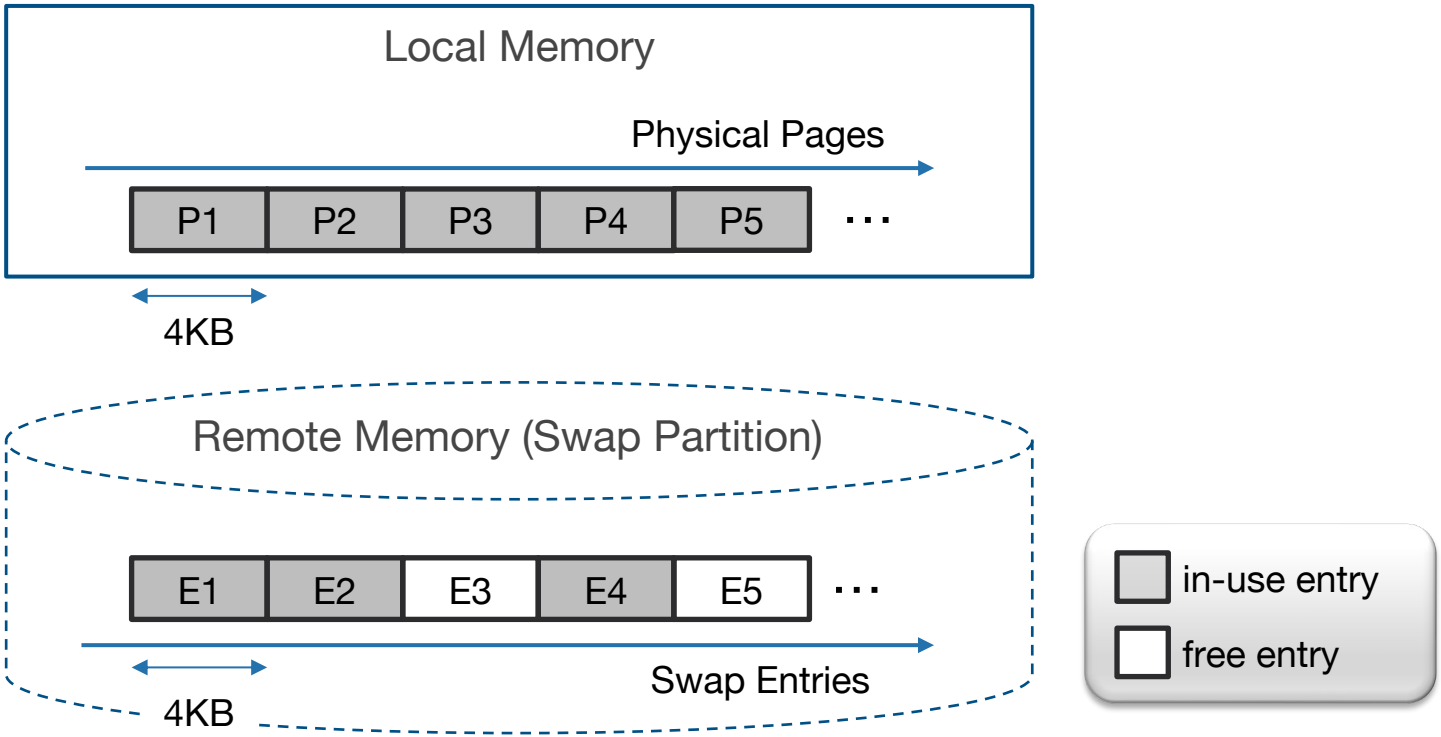
Isolation Enabled Adaptive Optimizations



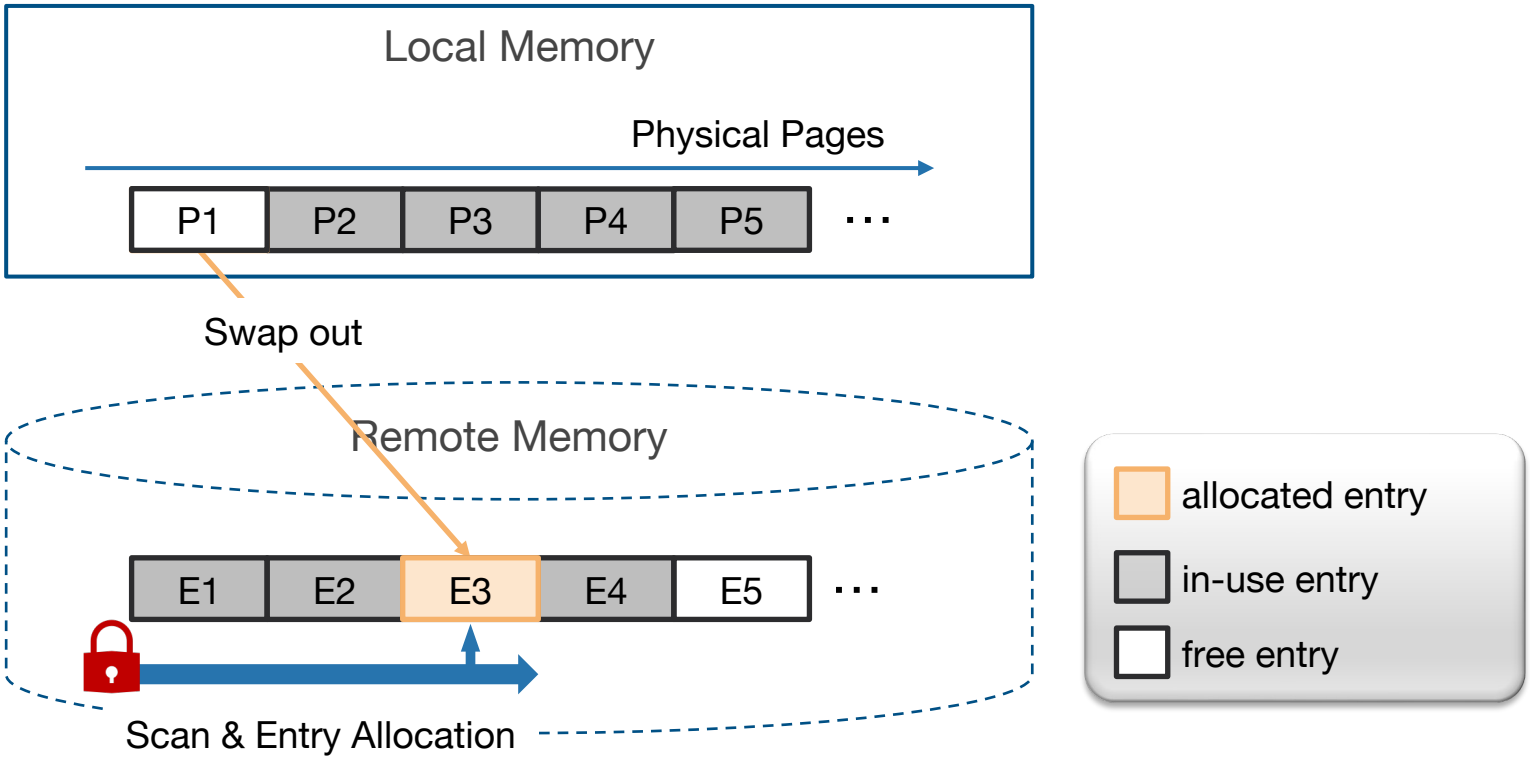
Isolation Enabled Adaptive Optimizations



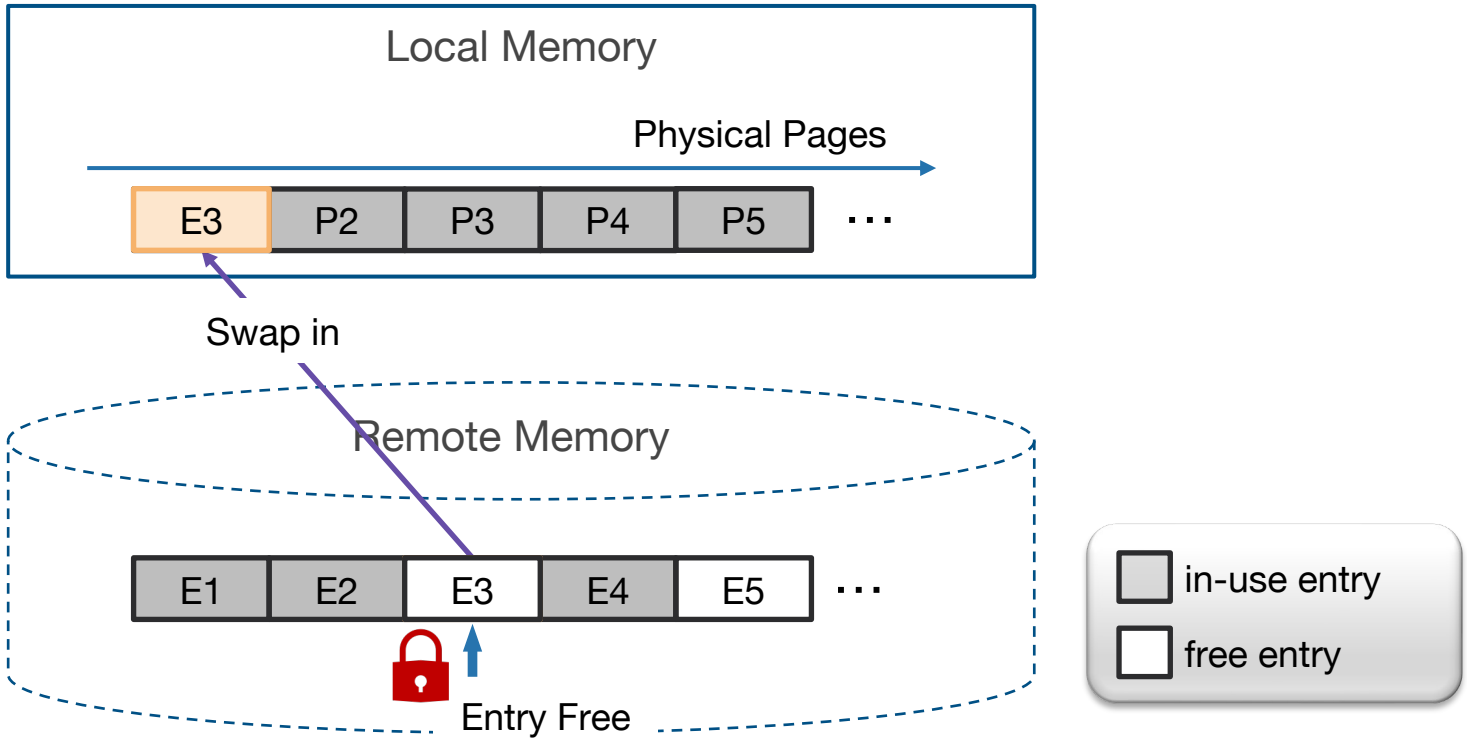
Remote Memory Management



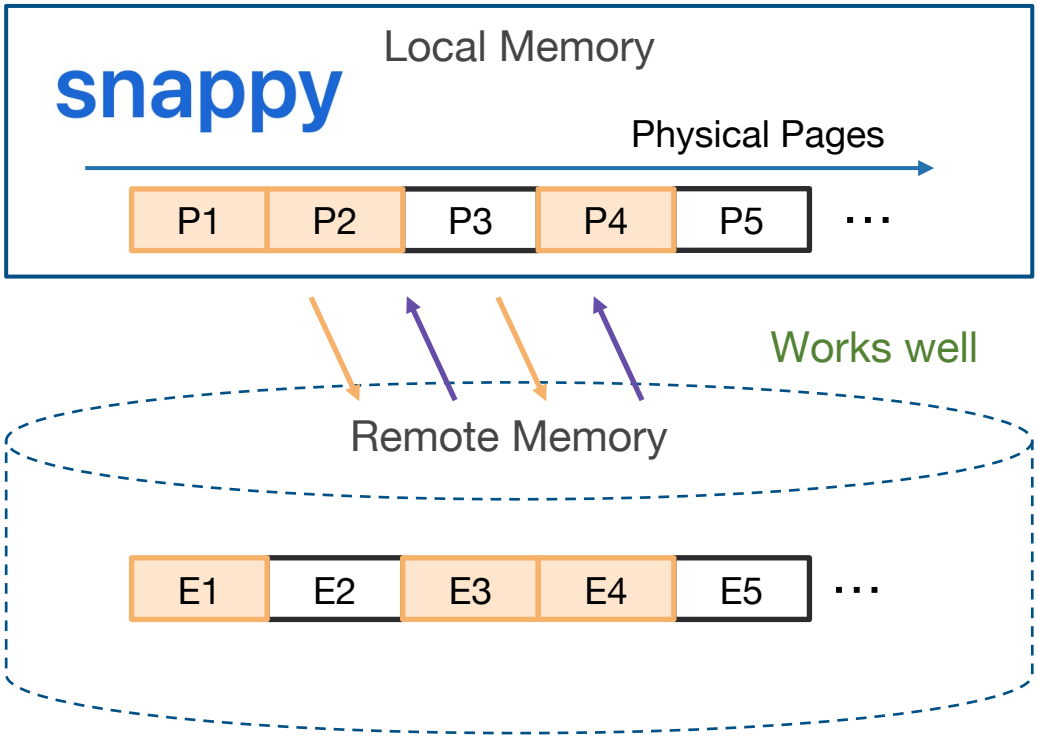
Remote Memory Management: Swap Out



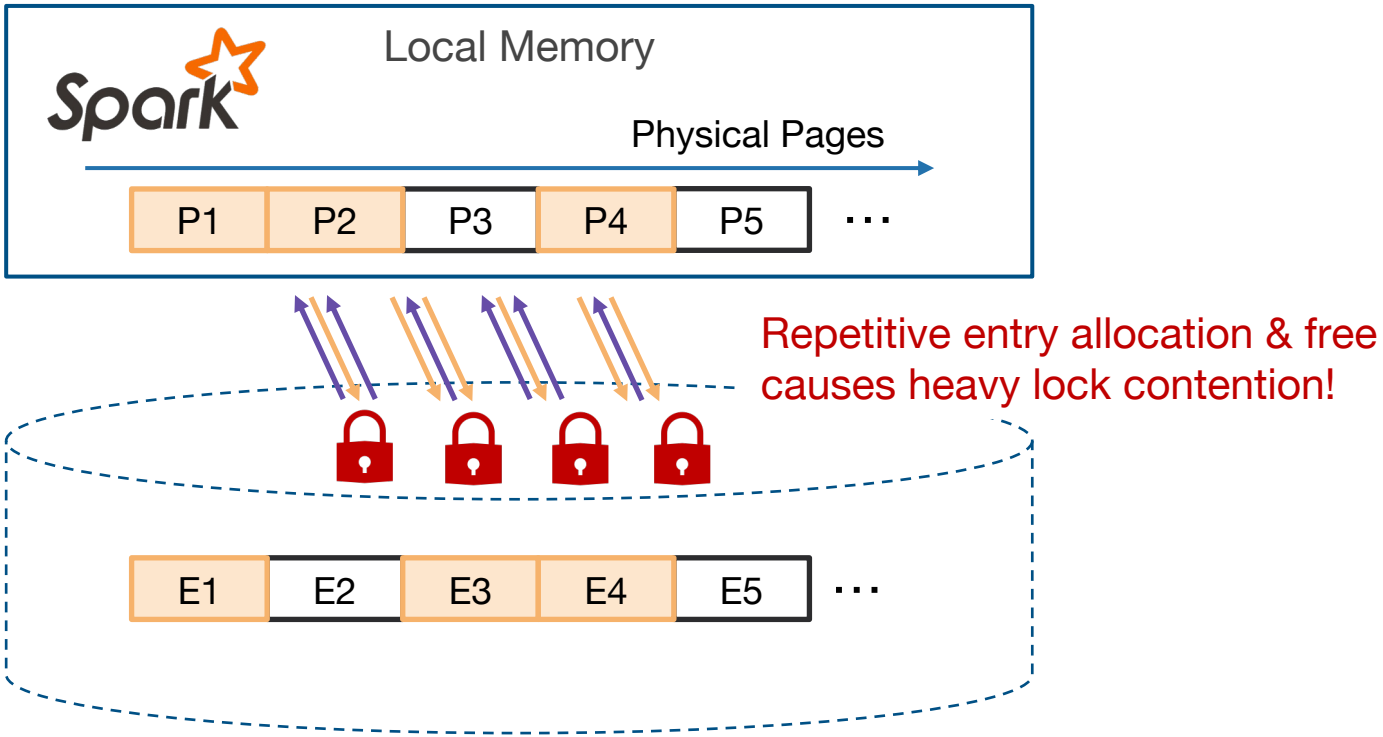
Remote Memory Management: Swap In



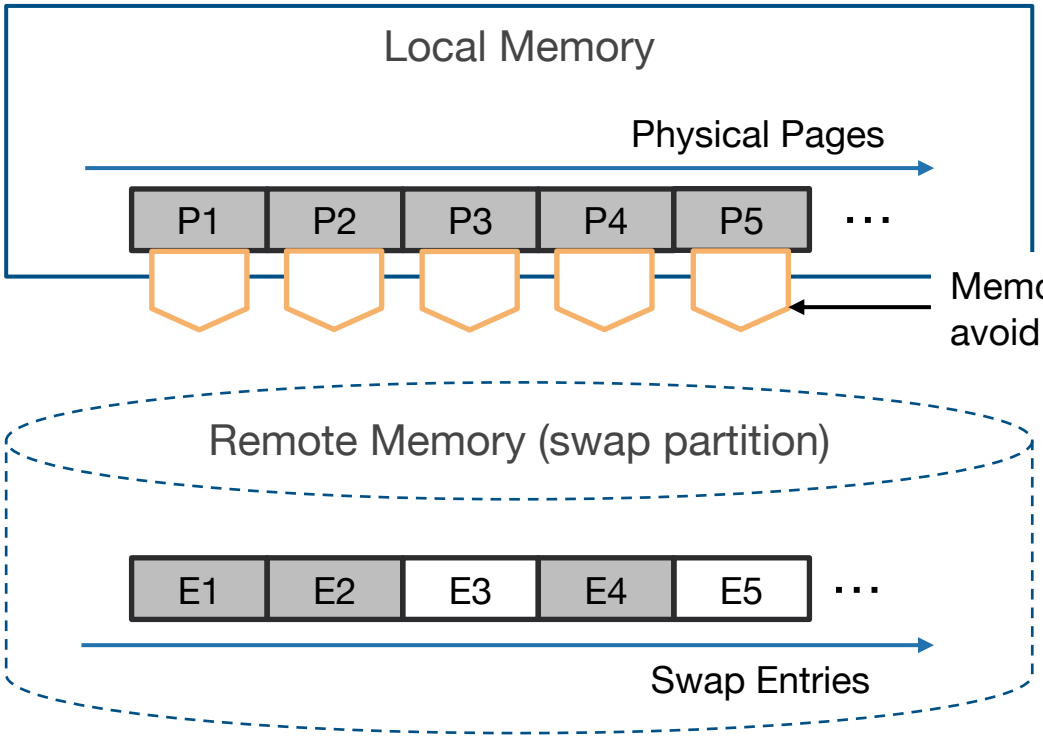
Efficient When Swap is Rare



Inefficient When Swap Is Intensive

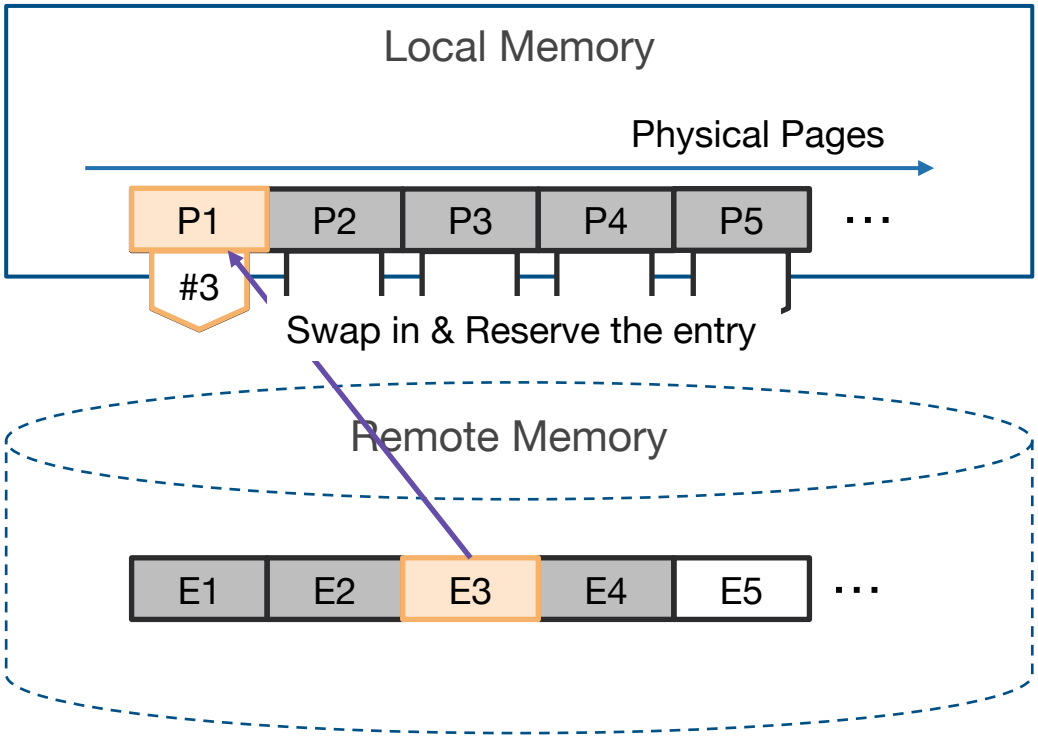


Adaptive Entry Allocator

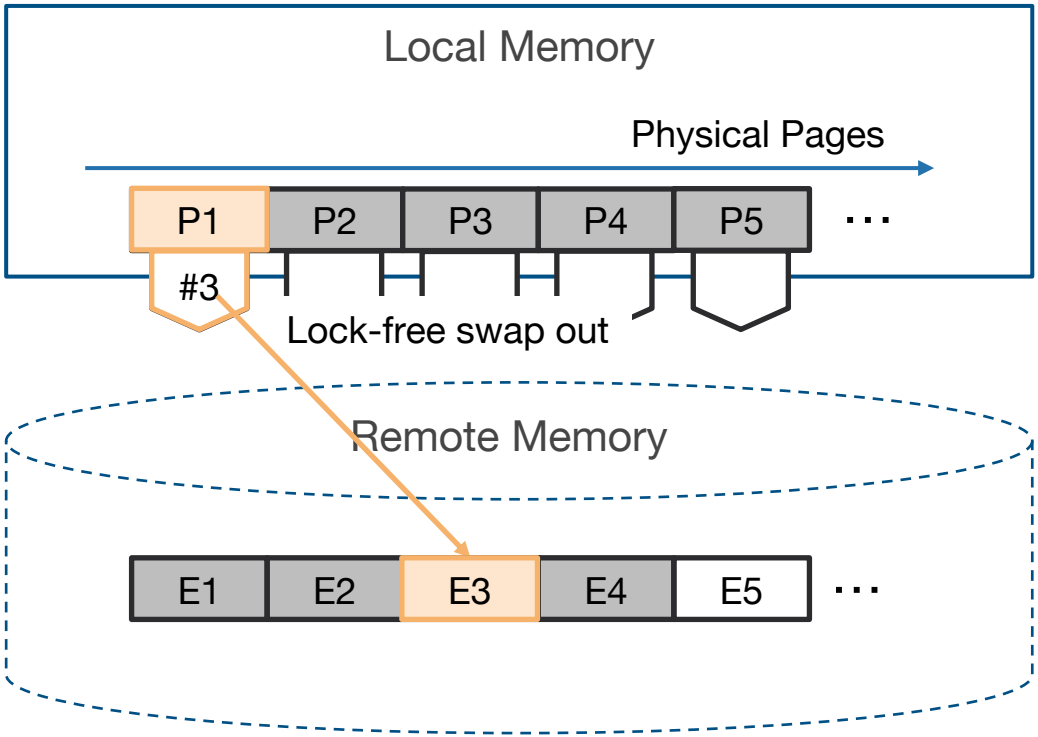


Sufficient remote memory:
Trade-off space for efficiency

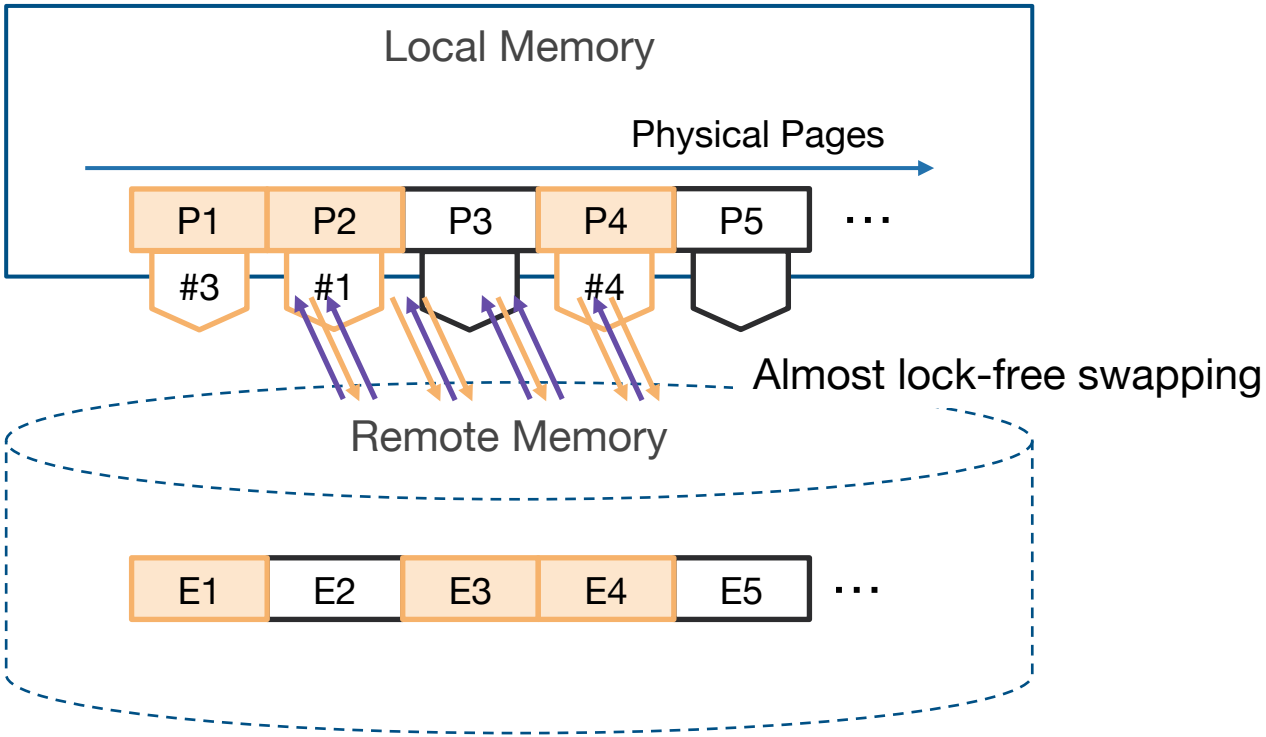
Adaptive Entry Allocator: Swap In & Reserve



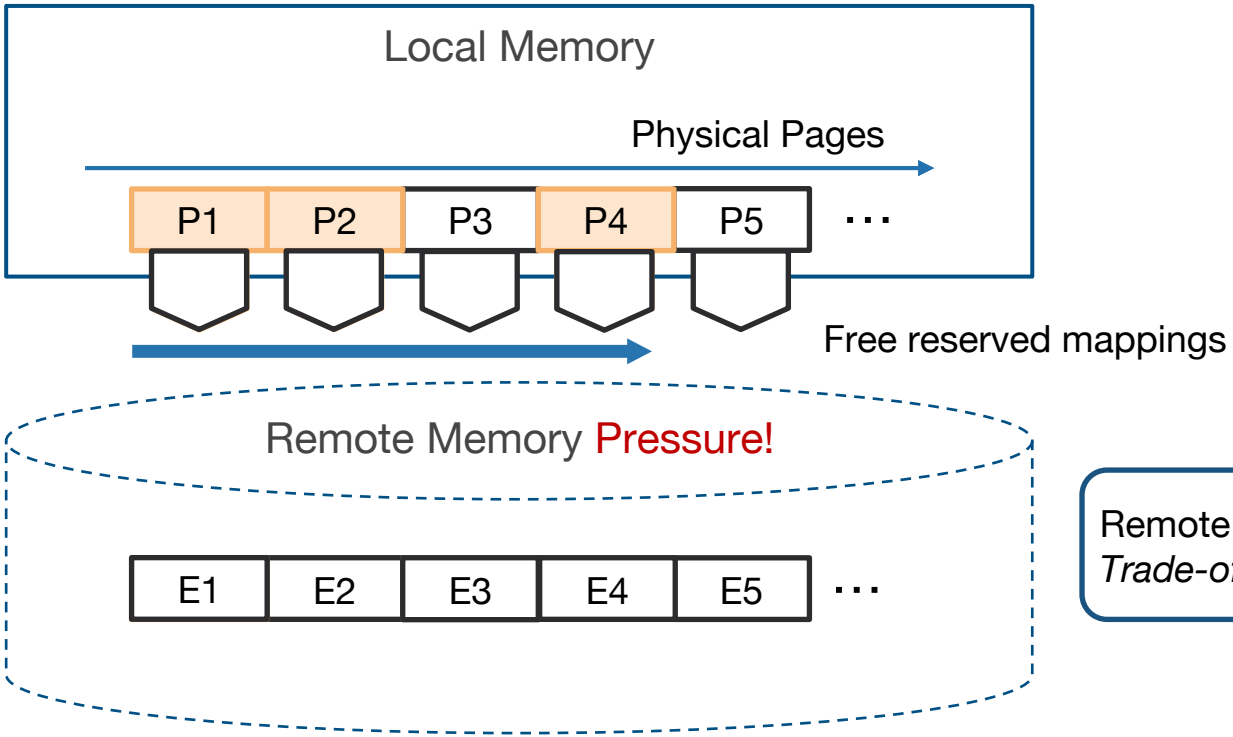
Adaptive Entry Allocator: Lock-Free Swap Out



Adaptive Entry Allocator: Intensive Swap



Adaptive Entry Allocator: Free Mappings



Remote memory pressure:
Trade-off time for space

Evaluation

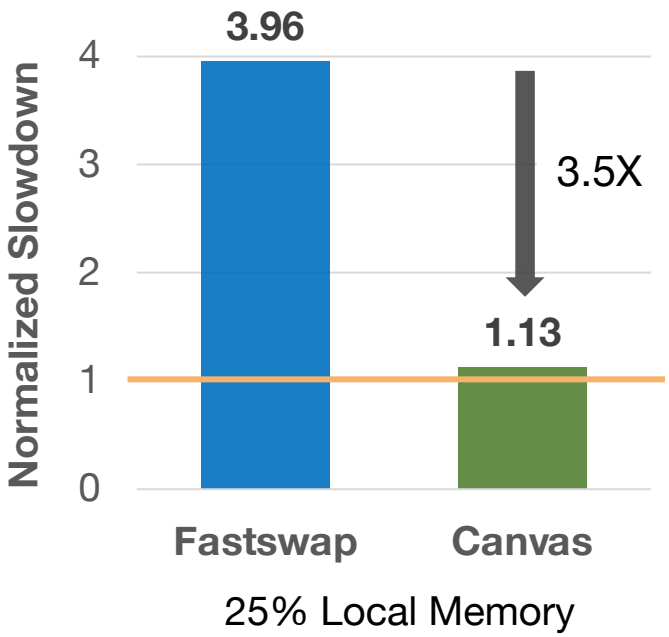
Evaluated on 6 real-world cloud applications with 11 workload combinations

State of the art: Linux `cgroup` on Fastswap [EuroSys'20]

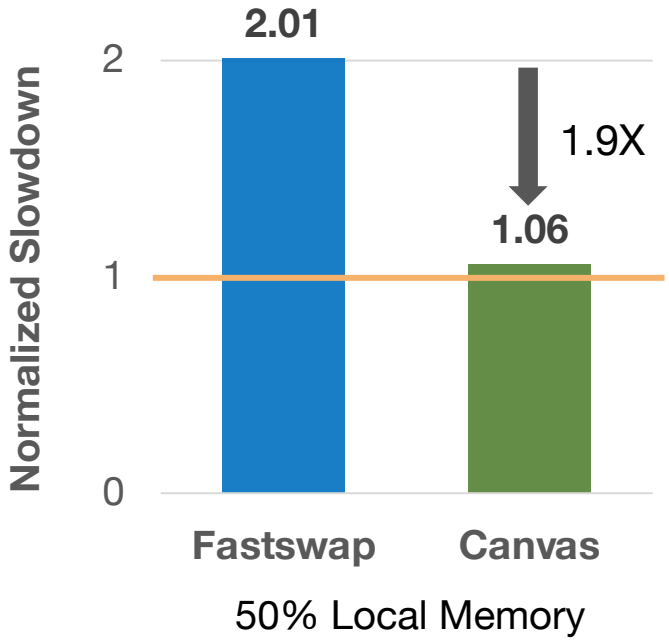
- How does Canvas improve throughput for co-running applications?
- How does Canvas reduce performance variation?

Results: Improved Throughput

Co-run Snappy, Memcached, XGBoost with Spark, Cassandra, and Neo4j, respectively



25% Local Memory

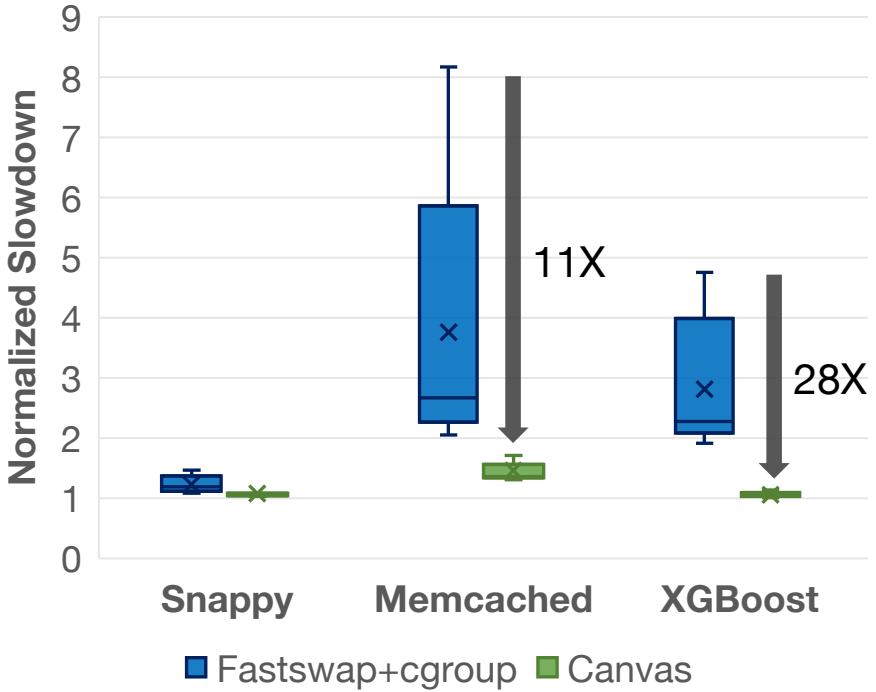


50% Local Memory

Results: Reduced Performance Variation

Fix Snappy, Memcached, and XGBoost and co-run them with another Java application

- 11 workload combinations in total
- Under 25% local memory
- On average 7.4x variation reduction



Conclusion

Canvas: holistic isolation + adaptive optimizations

- Isolation is necessary for real deployment of remote memory
- Isolation improves performance and QoS for shared remote memory
- Isolation enables adaptive optimizations for further performance boosts

- Canvas offers co-running applications 6.2x speedup and 7.4x variation improvement

https://github.com/uclasystem/canvas

Thank You!