SOFTWARE-DEFINED CACHING: MANAGING CACHES IN MULTI-TENANT DATA CENTERS

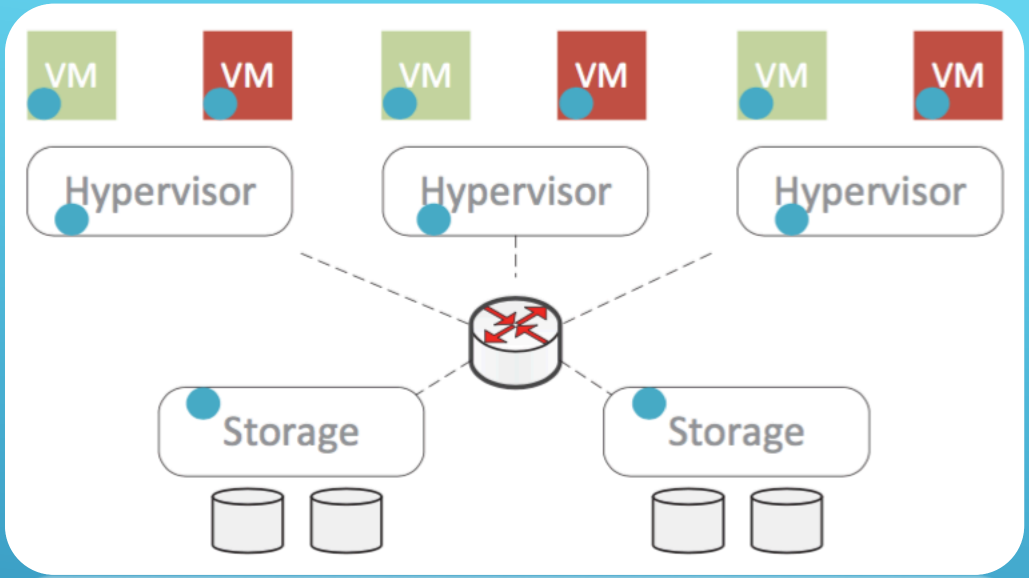
IOAN STEFANOVICI, ENO THERESKA, GREG O'SHEA, BIANCA SCHROEDER,
HITESH BALLANI, THOMAS KARAGIANNIS, ANTONY ROWSTRON, TOM TALPEY

Presentation by:

Neda Paryab

OVERVIEW

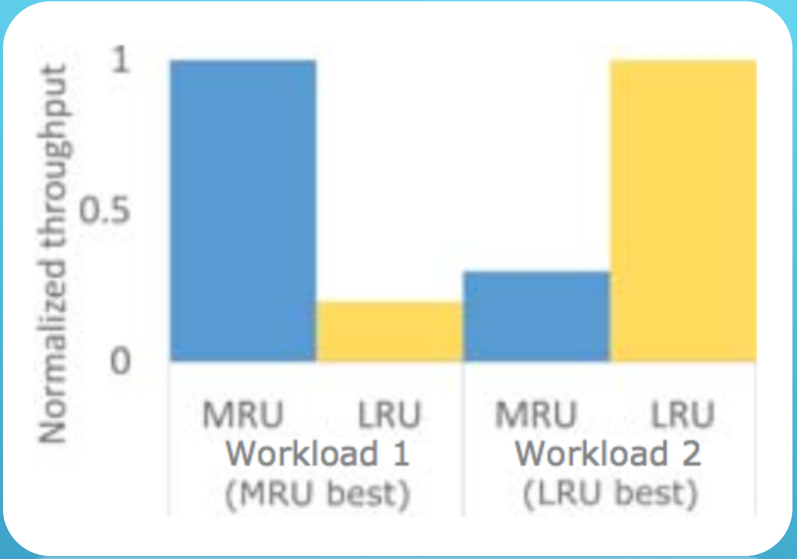
- **Why cache matters?**
  - I/O latency
  - Back-end load
  - **Problem:** Multi-tenancy
- **How got worse?**
  - un-coordinated caches on the IO data plane
- **The main problem:**
  - storage caches cannot provide workload-awareness
  - lack of vision in control plane

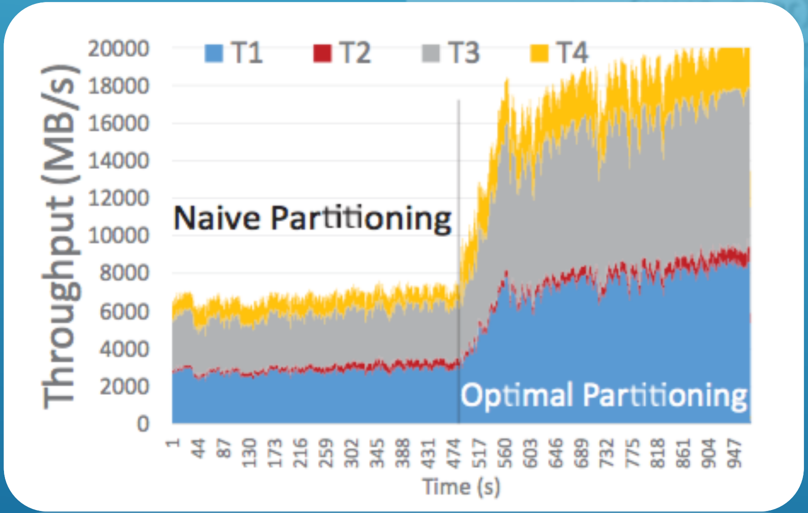


- Lack of performance isolation
- Lack of customization

- Lack of coordination
- Lack of adaptability

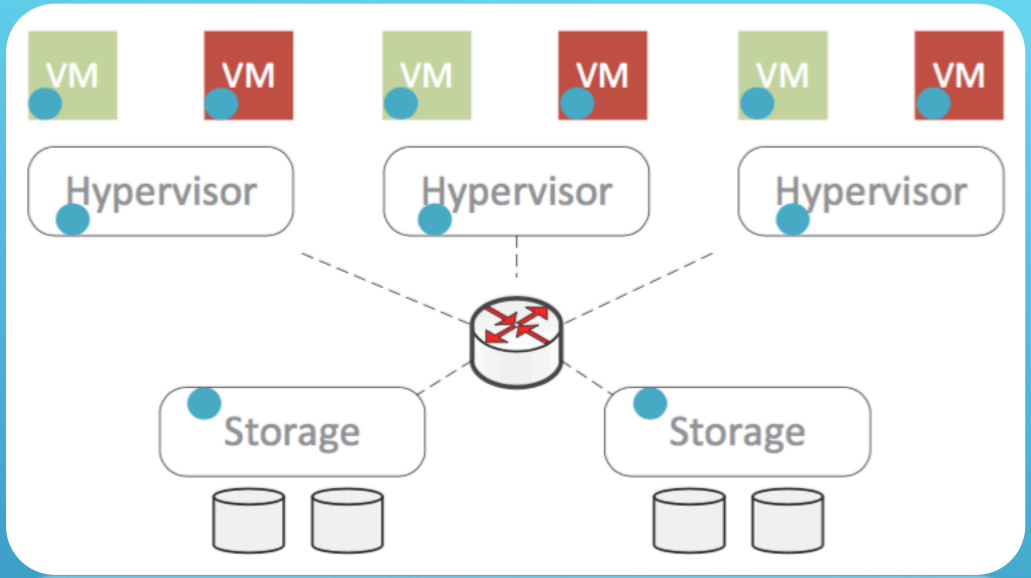
- Waste of system resources



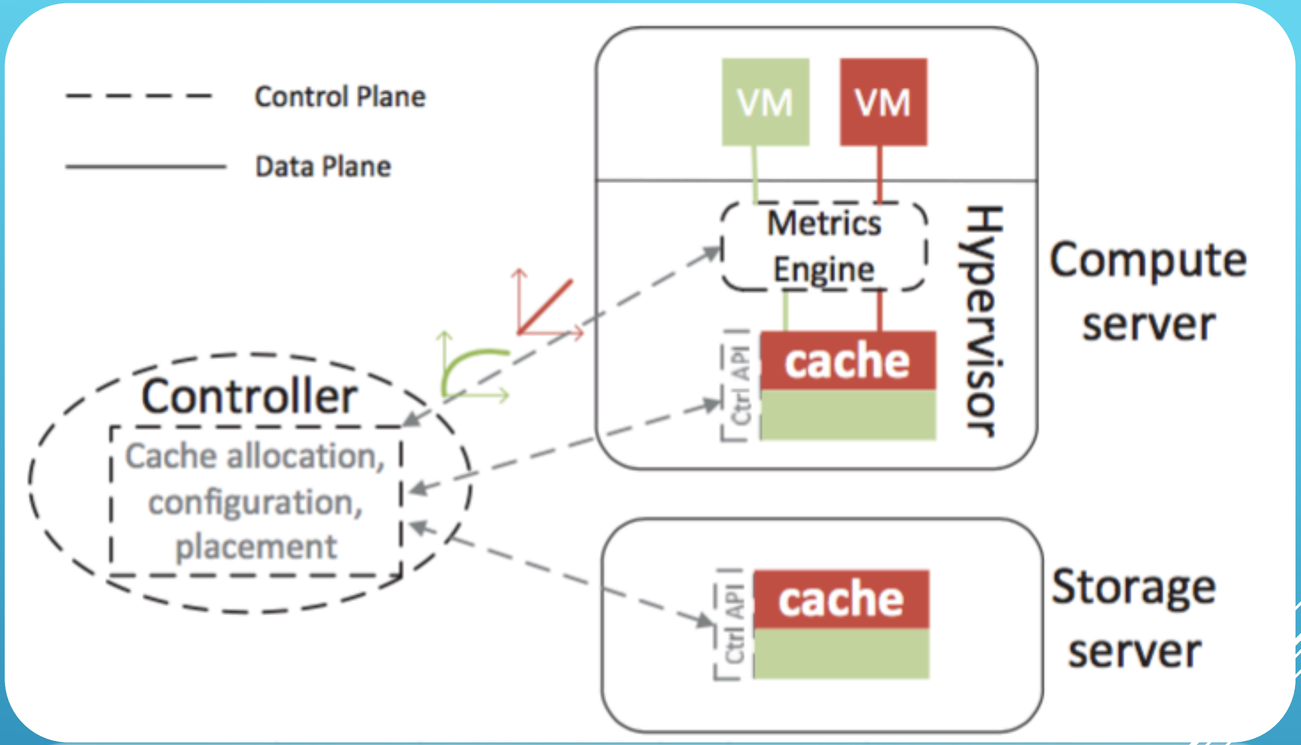


OVERVIEW

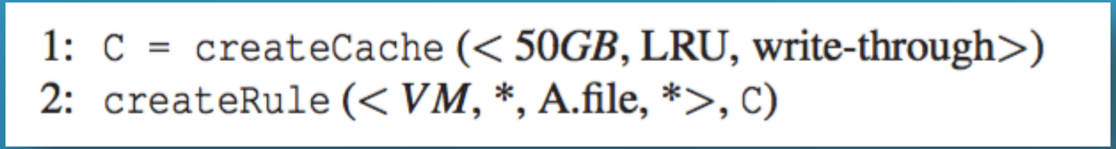
- Generally **What we need?**
  - Cache infrastructure management
    - Resource utilization
    - Tenant isolation
    - QoS guarantee
    - Transparency



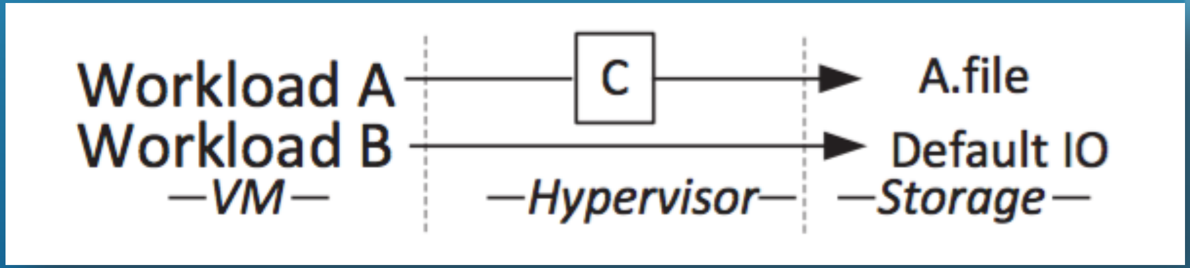
- ❑ **Controller**
  - ❑ Logically-centralized
- ❑ **Metrics engine**
  - ❑ Maintains workload characteristics
    - ❑ Throughput
    - ❑ Reads vs. writes
    - ❑ Hit ratio curves
- ❑ **Programmable caches**
  - ❑ Maintains provider objectives
    - ❑ Create cache at proper position
    - ❑ Workload-aware cache (rule)
    - ❑ Later on cache configurability
    - ❑ Performance monitoring

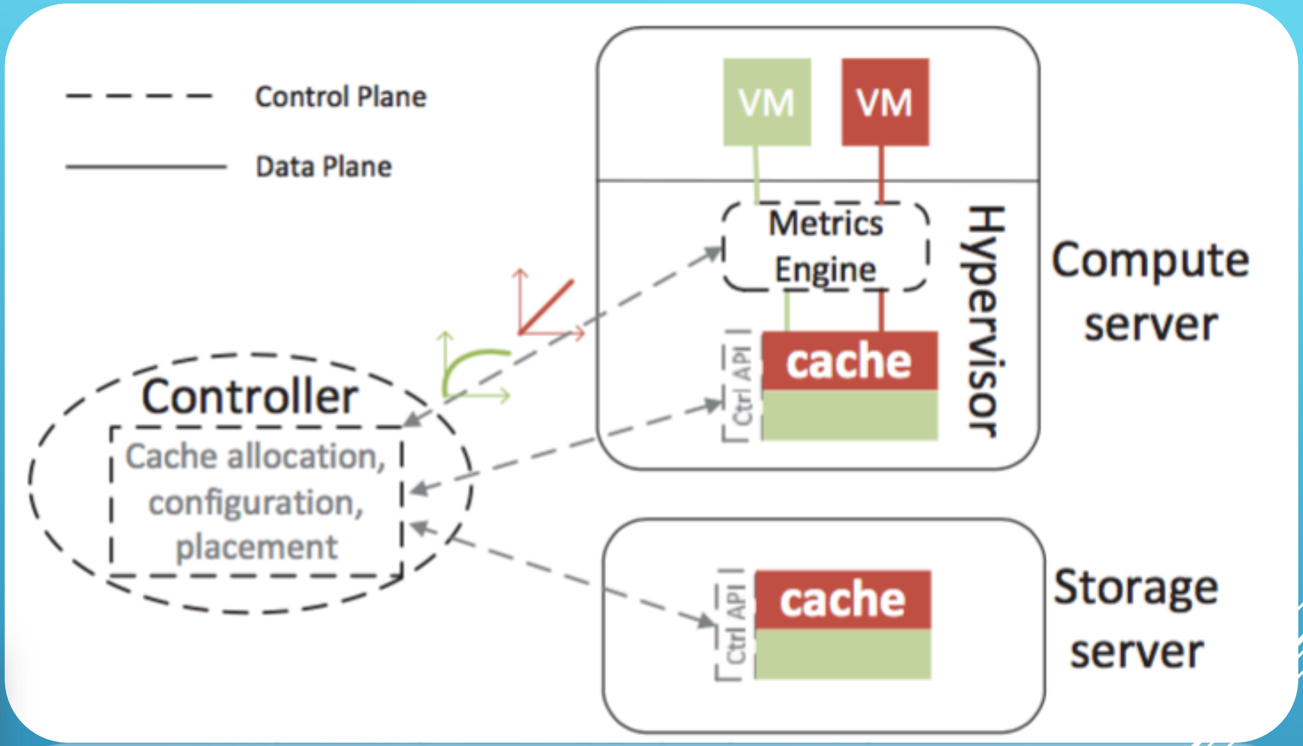


- **Data plane transformation**
    - Prioritizing workload



```
1:  C = createCache (< 50GB, LRU, write-through>)
2:  createRule (< VM, *, A.file, *>, C)
```

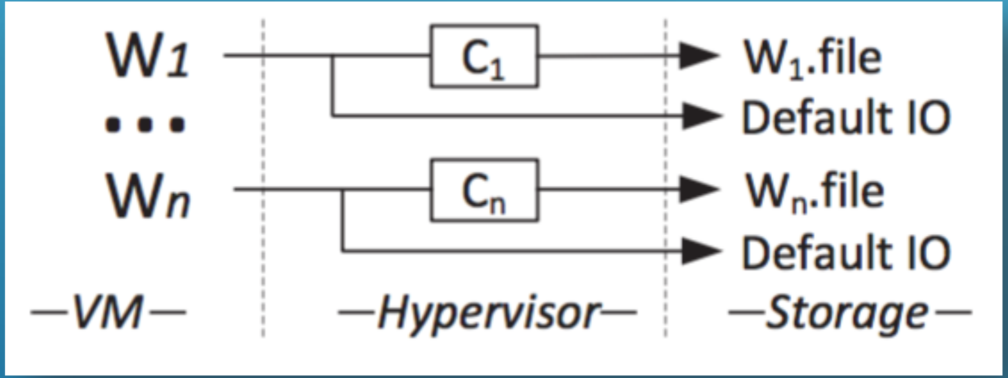


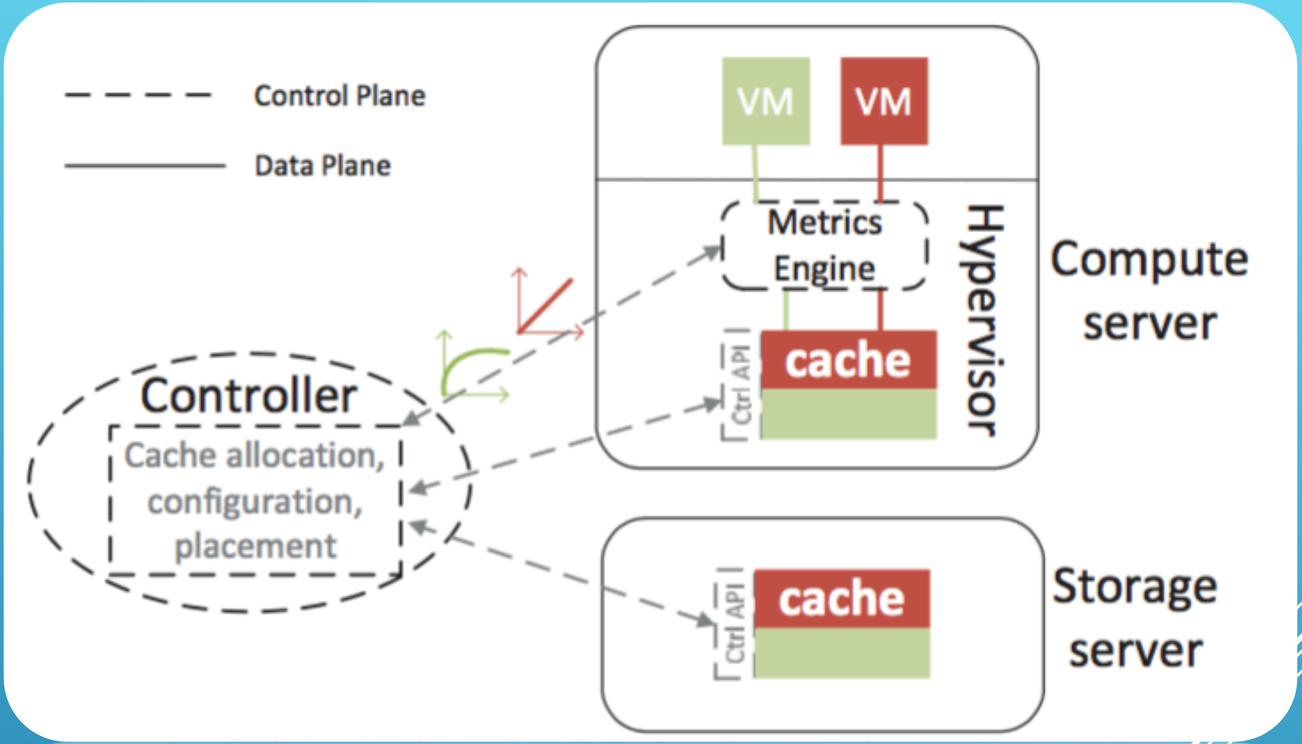


MOIRAI

- ❑ **Per-Worklaod bandwidth guarantee**
  - ❑ Hit_i^{cache}
  - ❑ Determine bandwidth

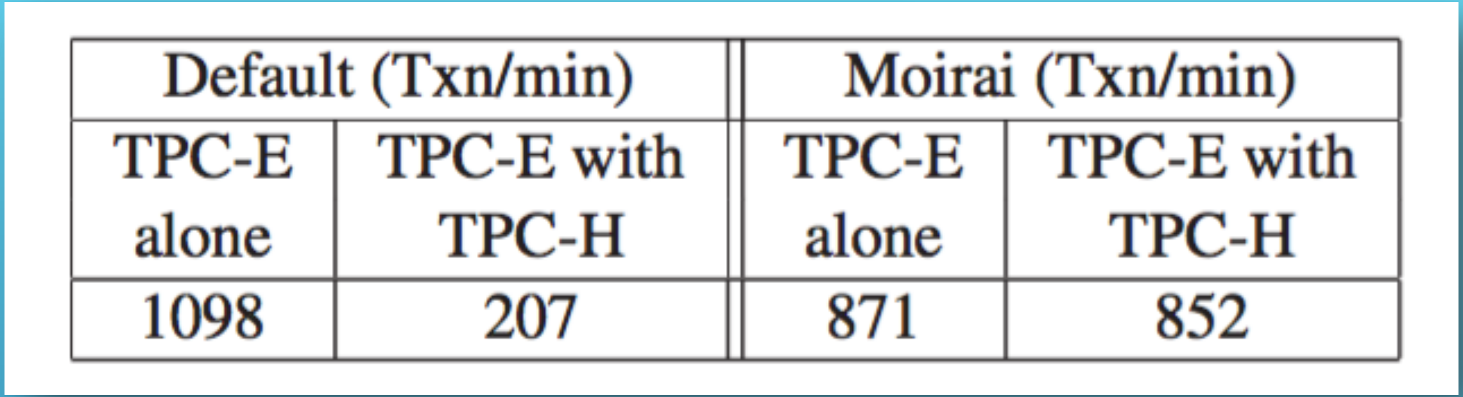
$$SLA_i^{BW} \leq Hit_i^{cache} \times BW^{memory} + (1 - Hit_i^{cache}) \times BW_i^{storage}$$





- ❑ **Maximize global workload utility**

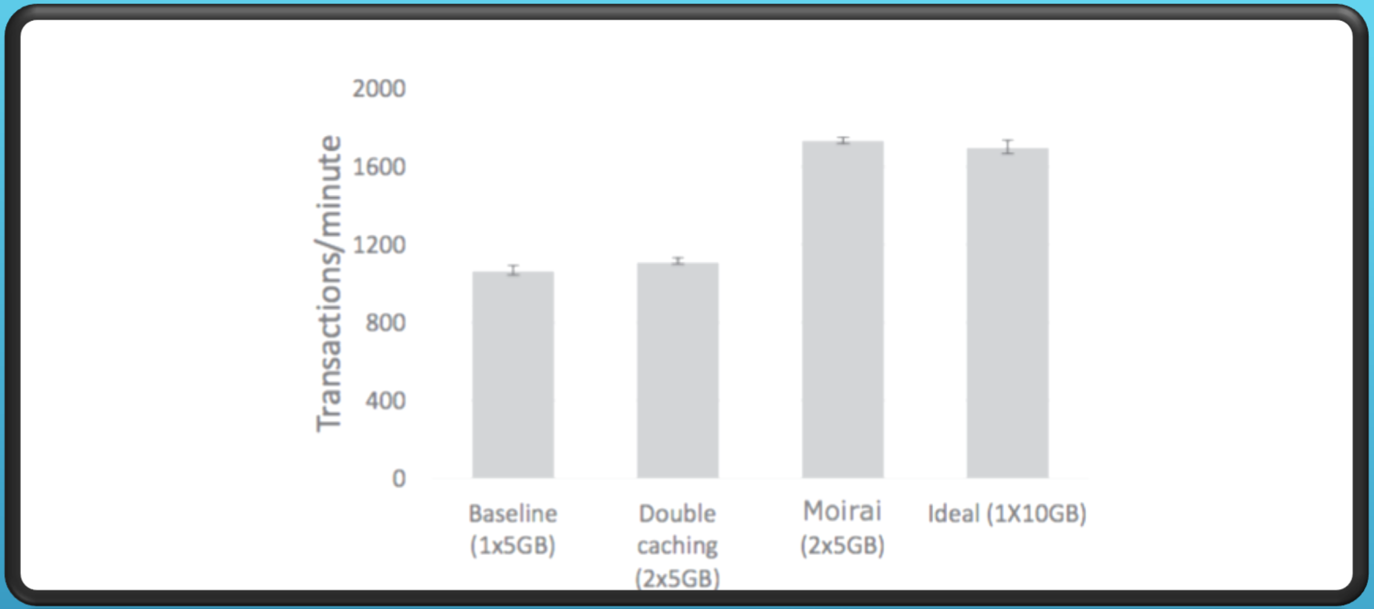
MOIRAI



Default (Txn/min)		Moirai (Txn/min)	
TPC-E alone	TPC-E with TPC-H	TPC-E alone	TPC-E with TPC-H
1098	207	871	852

- Enforcing Priorities
- Experimentations:
  - default caching for TPC-E per se
  - TPC-E & TPC-H together; how they affect each other
  - Throughput: “transaction per minute”

EVALUATION



- TPC-E on a low-memory machine
- Without Moirai:
  - Little improvement by increasing caches, because of "double caching"
- With Moirai:
  - Elimination of "double caching"

SCALING OUT CACHES

RELATED WORK

- Application caches
  - focus on system-level caches vs. specialized *application* caches
- System caches
  - Moirai focuses on the caches, beneath the VM abstractions
- Cache replacement policies
  - workload- and tenant-aware
- Inefficiencies in cache hierarchies
  - Moirai is Independent of VM support, or protocol
- Software defined storage
  - Architecture is controller-based (separation between data and control plane)
  - Moirai supports traffic classification

Thanks!