ECI-Cache: A High-Endurance and Cost-Efficient I/O Caching Scheme for Virtualized Platforms

Saba Ahmadian[1,2], Onur Mutlu[3,4], Hossein Asadi[1,2]

1 

2 

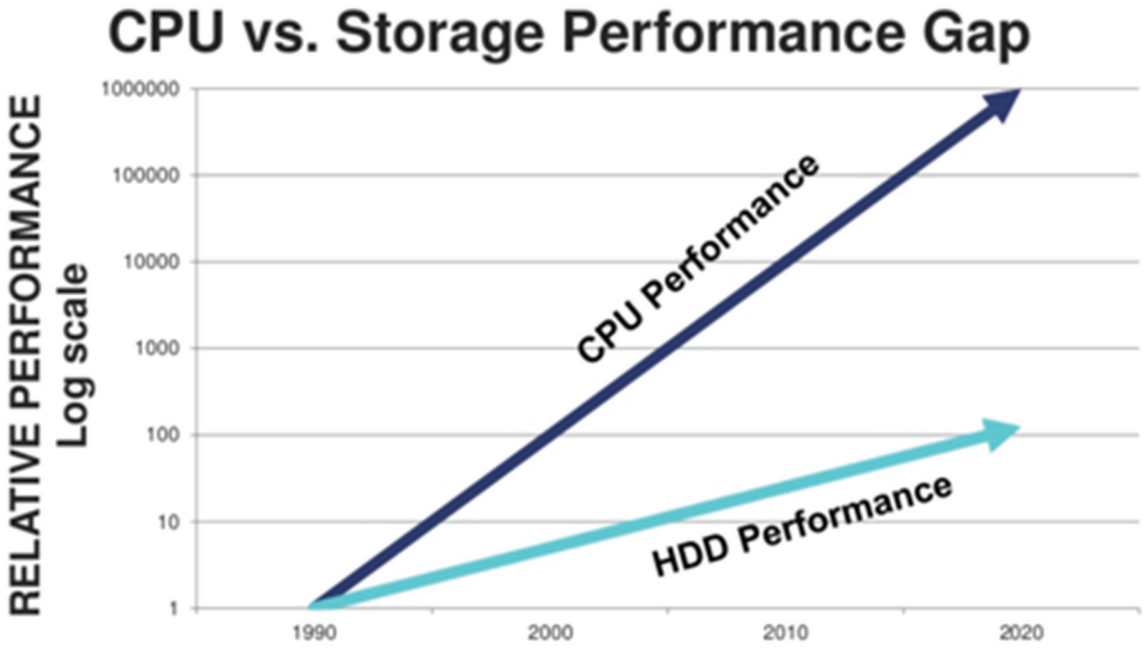
3 

4 ETH Zürich

Presenter: Saugata Ghose (Carnegie Mellon University)

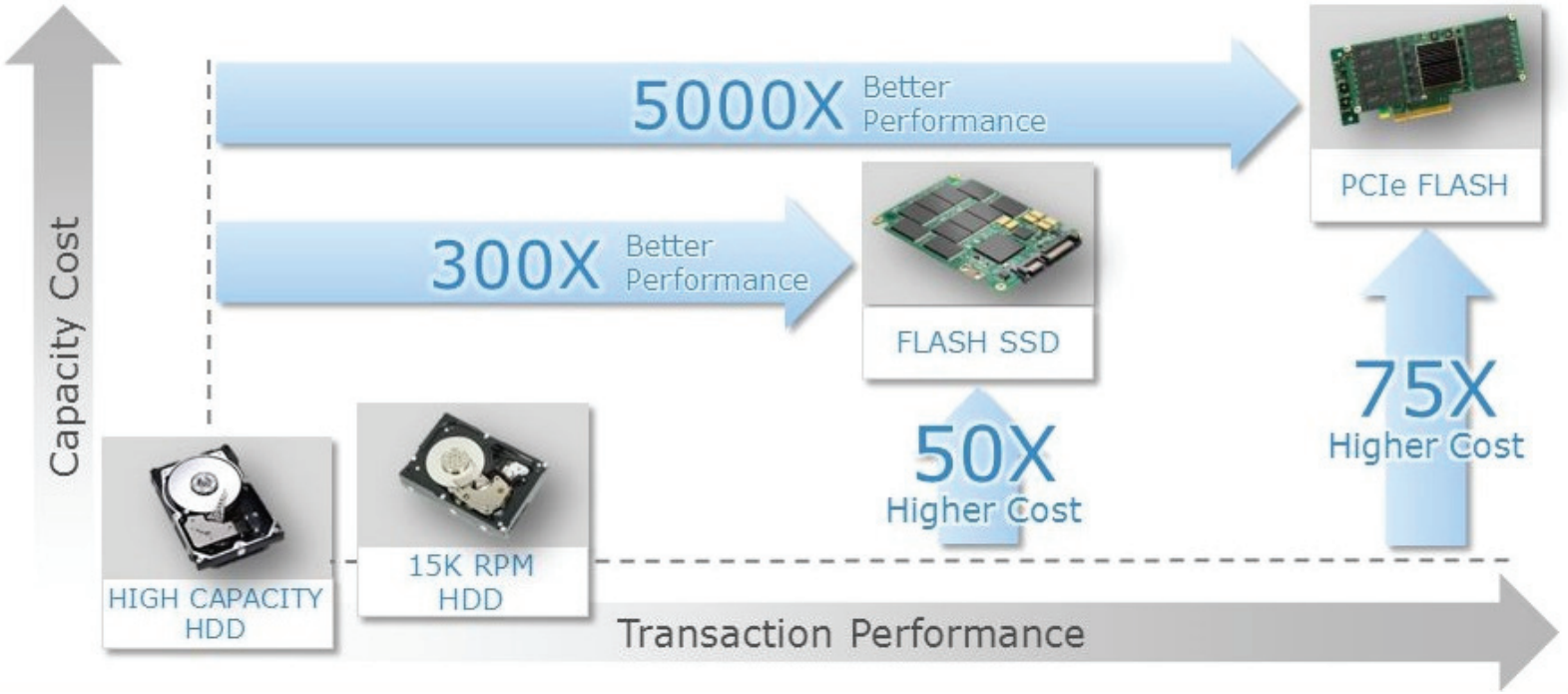
Introduction: Performance Gap Between CPUs and I/O Devices

- Data centers run an increasing number of **data-intensive applications**
  - Facebook in 2014: 600TB data per day
- Magnetic hard drive (HDD) latency not improving



Storage subsystem → performance bottleneck

Trade-offs in Modern Storage Devices



EMC Corp., 2013

Magnetic HDDs:

Low Performance, Low Cost, Larger Capacity

Solid-State Drives (SSDs):

High Performance, High Cost, Smaller Capacity

Writes Slower than Reads, Limited Endurance

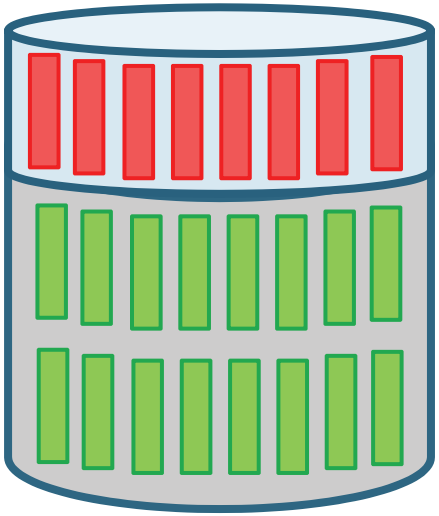
Hybrid Storage Systems:
Combine HDDs and SSDs for Best of Both Worlds

Cache Hot (Active) Data on High-Performance SSD

Store All Other Data in Low Cost HDD

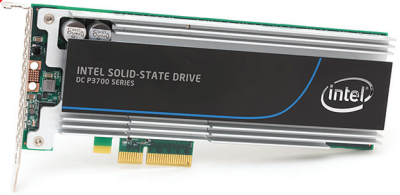
COLD

HDD



HOT

SSD



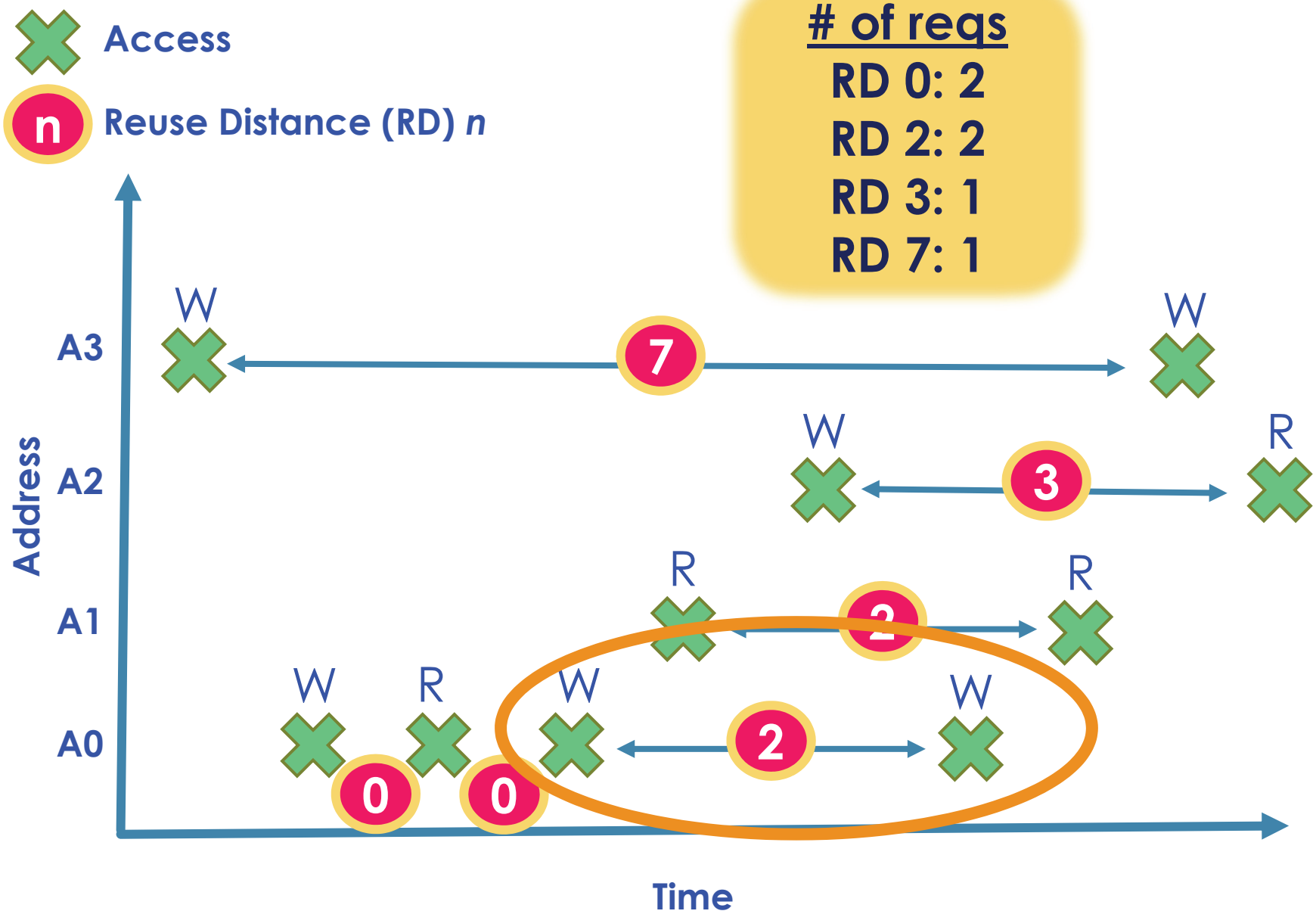
How much of the SSD cache should we allocate to each application in a data center machine?

Calculating SSD Cache Size

- Today: use **reuse distance** to determine how much cache to allocate to each VM
  - Time between two accesses to the same block
  - Achieves a balance between allocated cache size and cache hit rate
  - **Used by state-of-the-art I/O cache managers**

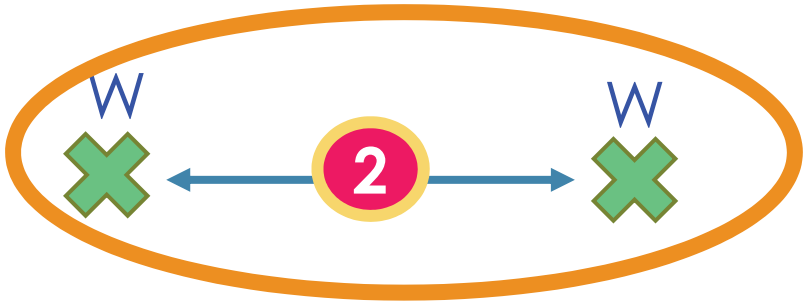
- Challenge: SSD write behavior
  - Much slower than reads
  - More writes decrease the SSD lifetime

Traditional Reuse Distance (TRD)



The Problem with Reuse Distance

- Write requests do **not** fall on the critical path
- In SSDs, read requests (on the critical path) can be delayed by long-latency write requests
- TRD does not consider the **request type**
  - **Write After Write (WAW)**
    first write not used, but incurs latency/endurance cost



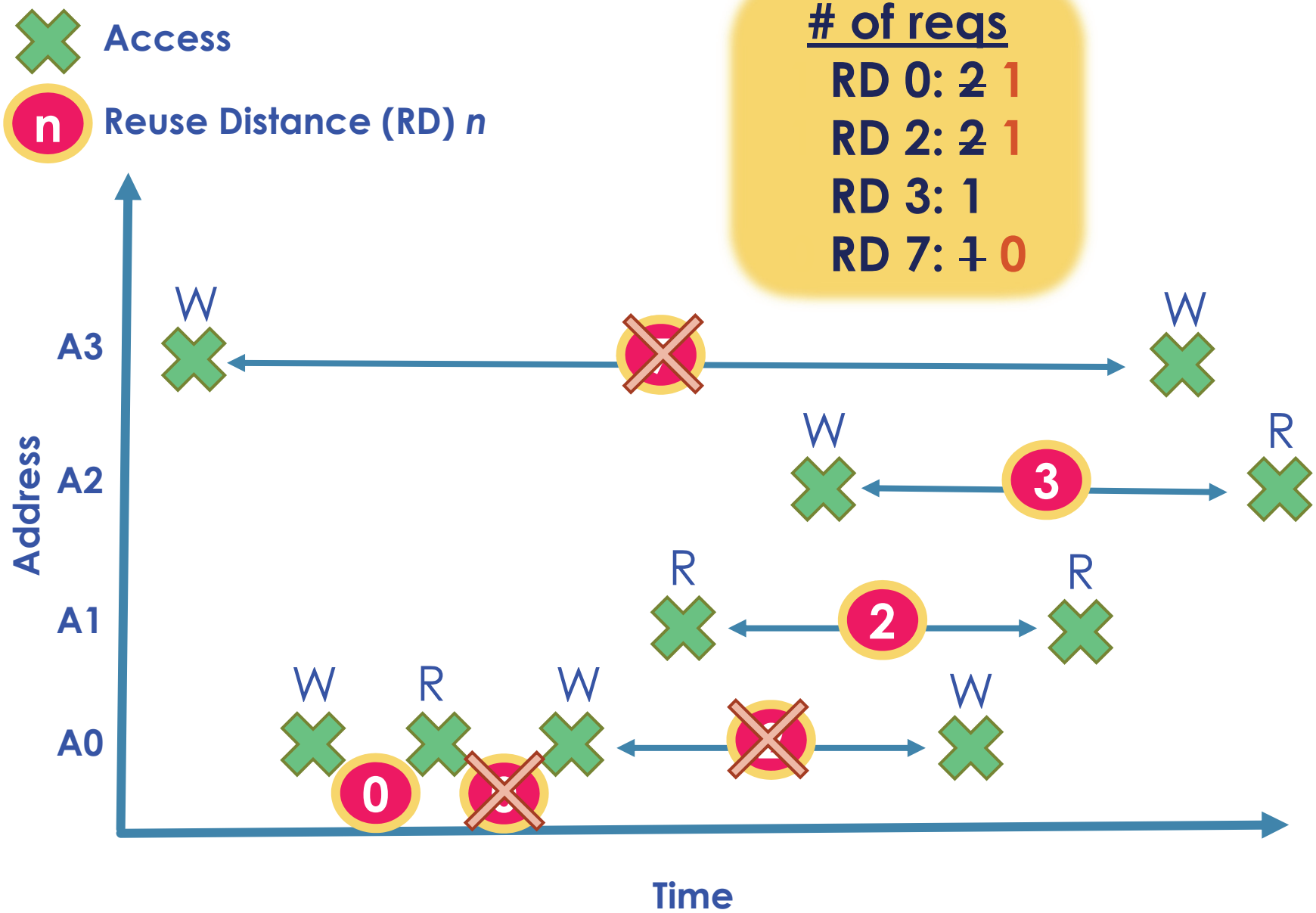
The Problem with Reuse Distance

- Write requests do **not** fall on the critical path
- In SSDs, read requests (on the critical path) can be delayed by long-latency write requests
- TRD does not consider the **request type**
  - **Write After Write (WAW)**
    first write not used, but incurs latency/endurance cost
  - **Write After Read (WAR)**
    unclear if newly-written data will be reused later
  - **Read After Read (RAR)/Read After Write (RAW)**
    data is needed by the application → useful

New Metric: Useful Reuse Distance

Consider only requests that we know will be reused

Useful Reuse Distance (URD)

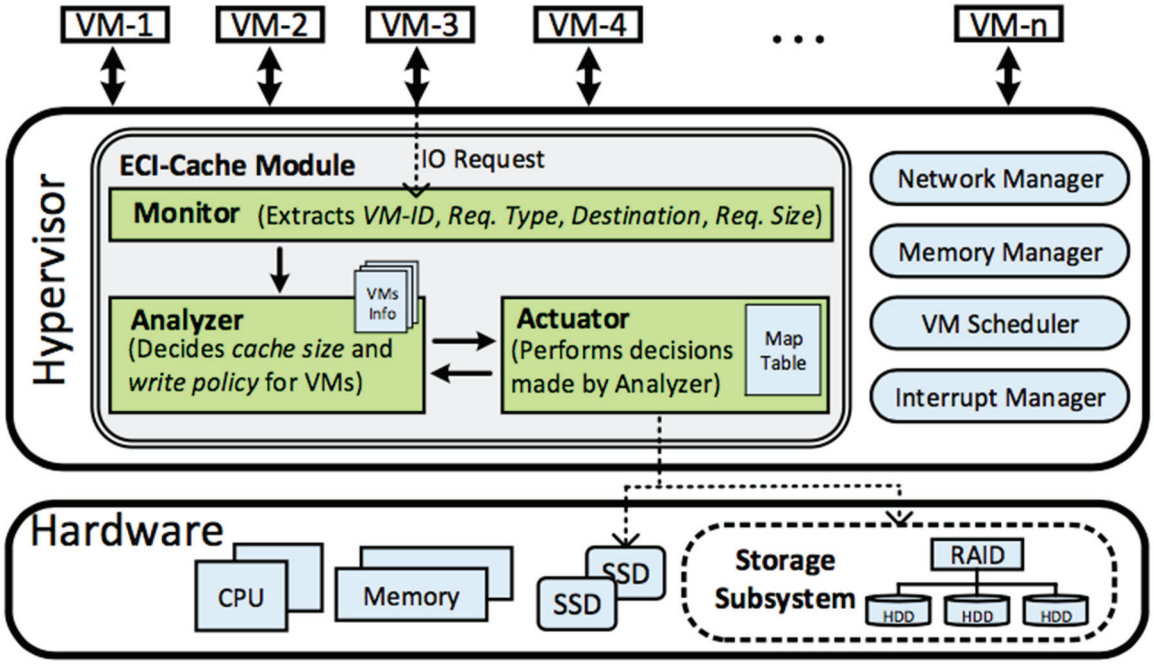


ECI-Cache

High-**E**ndurance, **C**ost-Efficient **I**/O **Cache**

- Hypervisor-based SSD cache manager
- Implemented as a working QEMU module

- Allocates per-VM cache space dynamically using URD
- Assigns per-VM write policies adaptively



ECI-Cache Management

- For each VM
  - Collect storage traces from prior interval
  - Calculate **fraction of reuse for each type**: RAR, RAW, WAR, WAW
  - Determine write policy
    - **If WAW + WAR > threshold, disable cache writes**
    - **Otherwise, cache uses writebacks**
  - Calculate Useful Reuse Distance
- Adjust cache sizes (see the paper)
- Assign policies

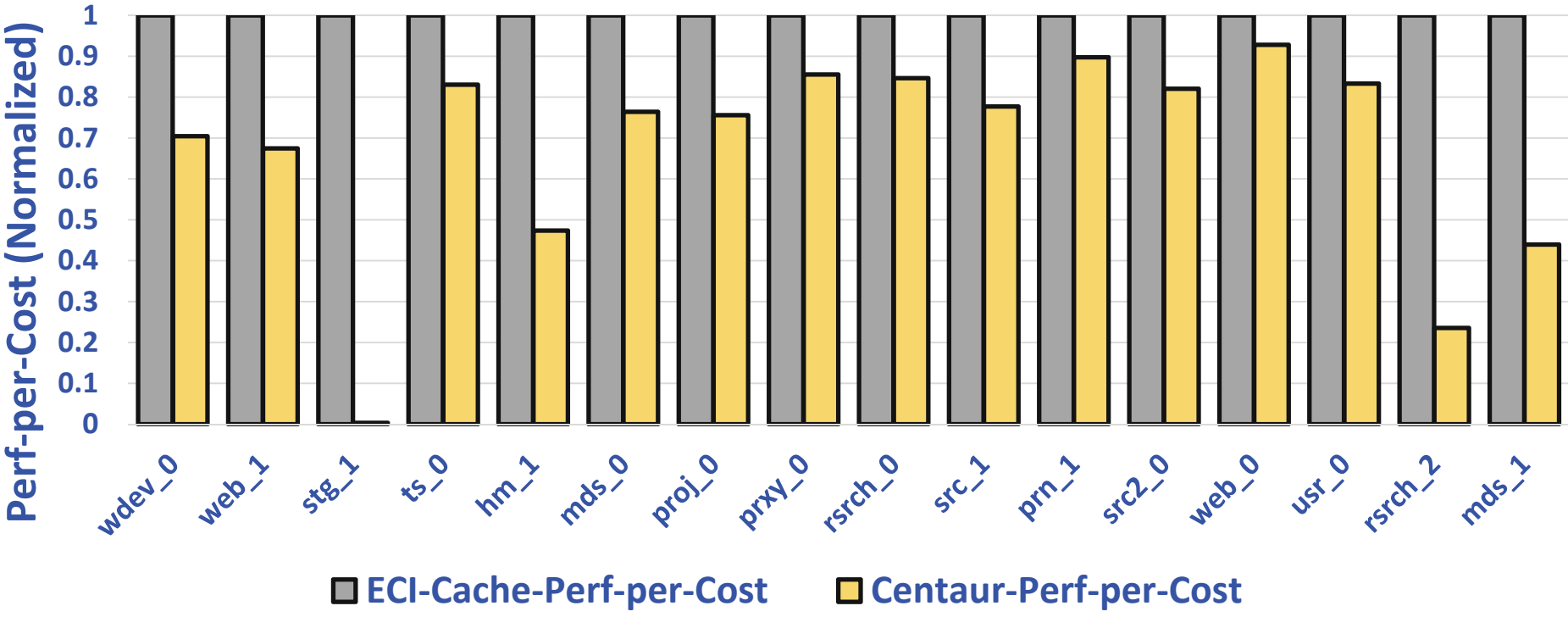
Evaluation: Experimental Setup

- All experiments run on a **real platform**
  - HP Proliant DL380 Gen. 5, 8 Xeon CPUs, 16GB RAM
  - 4x HDD: 146GB SAS 10K HP → RAID5 (3+1)
  - SSD: 128GB SATA Samsung 850 Pro

- **QEMU Hypervisor** on Centos 7 (Kernel: 3.10.0-327)
  - 16 running VMs (OS: Ubuntu 15.04 and Centos 7)
  - Applications: MSR traces from SNIA

- Compare ECI-Cache to **Centaur** [Koller+ ICAC '15], best prior I/O cache manager

Experiments: Performance-Per-Cost Improvement

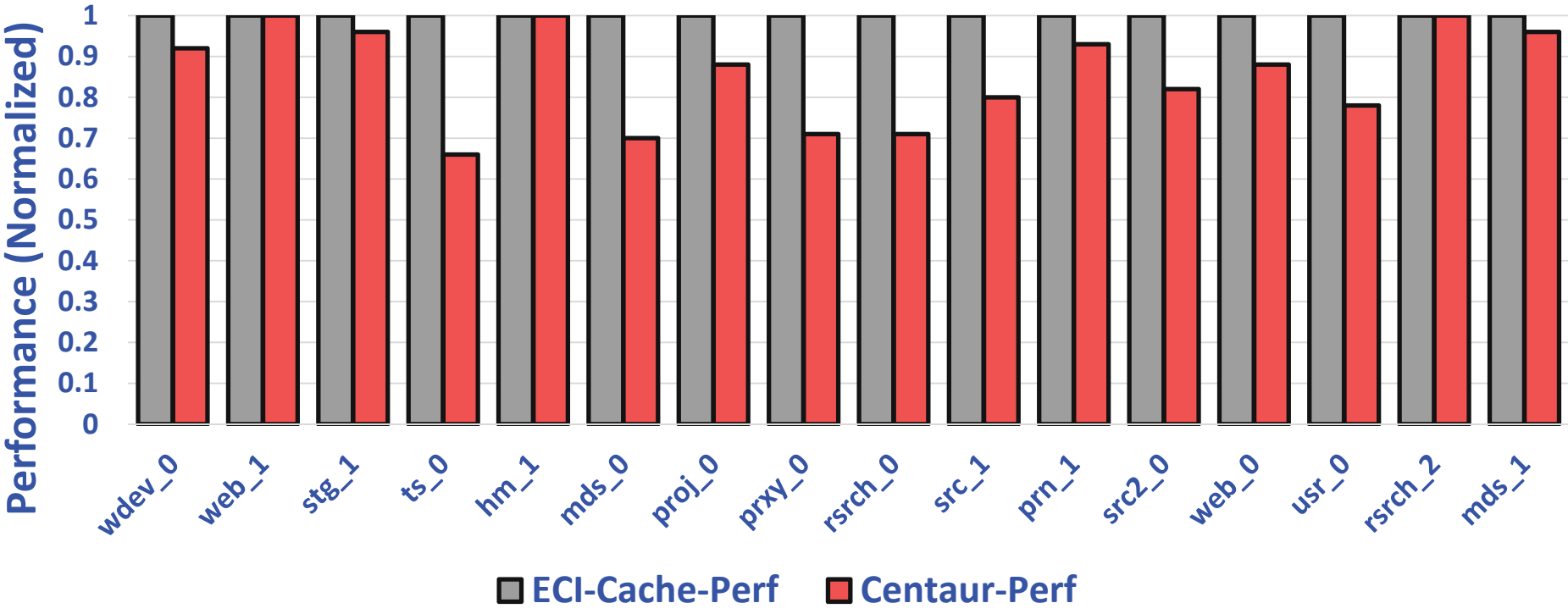


➔ ECI-Cache improves perf-per-cost by more than 30%

✓ Much smaller cache space allocation by ECI-Cache

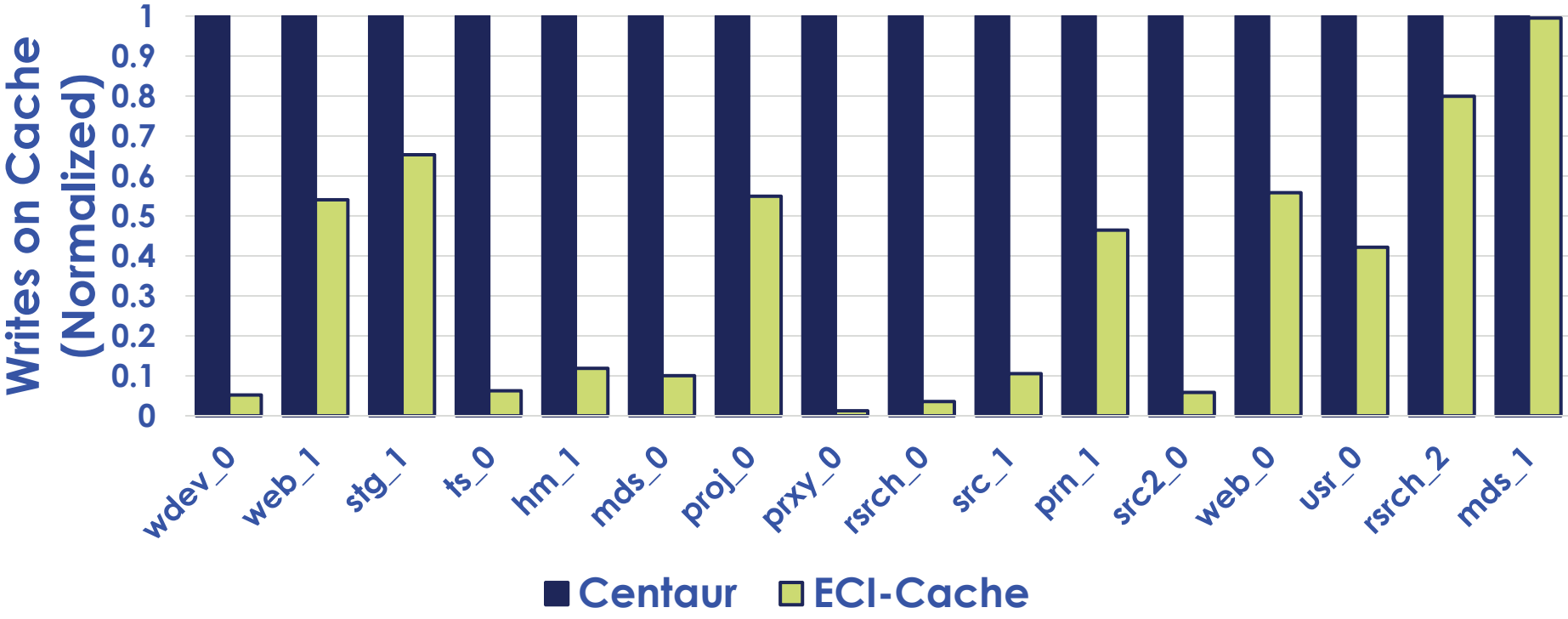
✓ Higher hit rate for ECI-Cache

Experiments: Performance Improvement



➔ ECI-Cache improves performance by 17%

Experiments:
Endurance (Lifetime) Improvement



➔ ECI-Cache improves SSD lifetime by 65%

Conclusion

I/O is a **performance bottleneck** in virtualized platforms

→ Improve I/O performance using an **SSD cache**

→ SSD writes introduce **long latencies, lower endurance**

URD: Useful Reuse Distance

New metric: count reuse only when next access is read

ECI-Cache: Hypervisor-based I/O cache

- Per-VM URD-based cache sizing
- Per-VM write policy

Results on real system → Improved Performance (**17%**), Performance-per-Cost (**30%**), and Endurance (**65%**)

ECI-Cache: A High-Endurance and Cost-Efficient I/O Caching Scheme for Virtualized Platforms

Saba Ahmadian[1,2], Onur Mutlu[3,4], Hossein Asadi[1,2]

1 

2 

3 

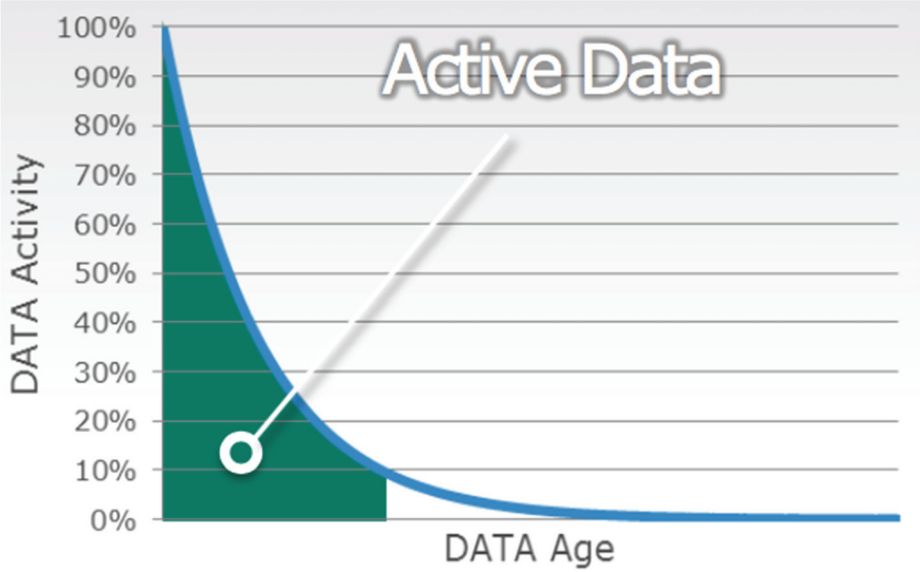
4 ETH Zürich

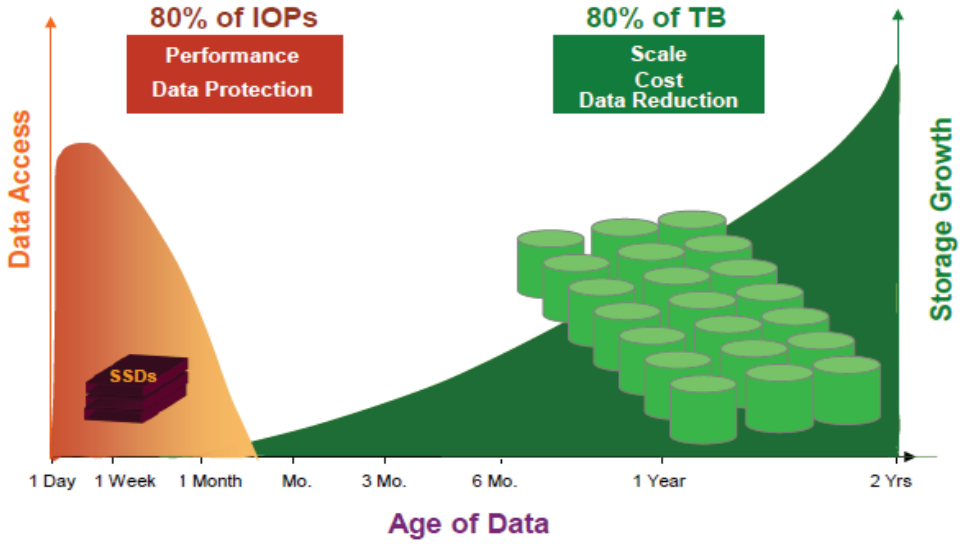
Presenter: Saugata Ghose (Carnegie Mellon University)

Backup Slides

Challenge for Hybrid Storage: Closing the Performance Gap

- Data is Highly Skewed
- At Any Given Time, Only a Small Percentage of Data is Active



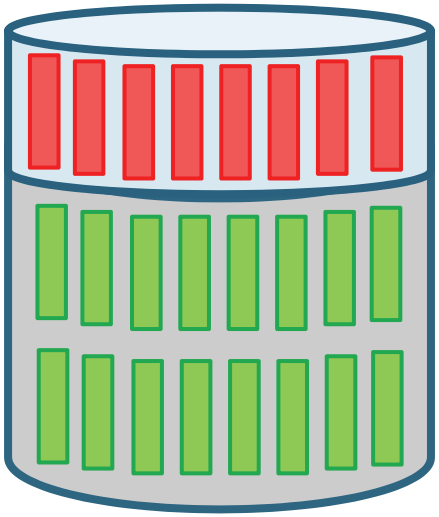


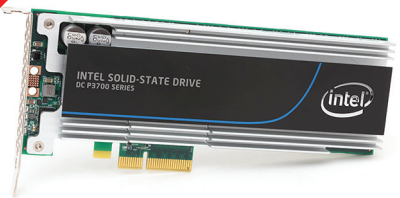
EMC Corp., 2015.
IMEX Research Cloud Infrastructure Report 2009-2011.

Challenge for Hybrid Storage: Closing the Performance Gap

Active Data on
High Performance SSD

Hot Data Cold Data





ACTIVE
SSD

Inactive Data on
Low Cost HDD

INACTIVE
HDD

Small Size Hot Data → SSD
Large Size Cold Data → HDD

Previous Cache Size Estimation Approaches

- Reuse Intensity
  - Mostly capable for CPU caches

- WSS (Working Set Size)
  - Doesn't measure data reuse
  - Indicates required space for acceptable performance

- Reuse Distance Based
  - Trade-off between allocated size and miss ratio
  - Miss Ratio Curves (MRCs)
    - Miss ratio as a function of cache size → Predict future Perf.
  - Locality
    - Spatial
    - Temporal (in selected time intervals)

Previous Cache Size Estimation Approaches

- Reuse Intensity
  - Mostly capable for CPU caches

- WSS (Working Set Size)
  - Doesn't measure data reuse

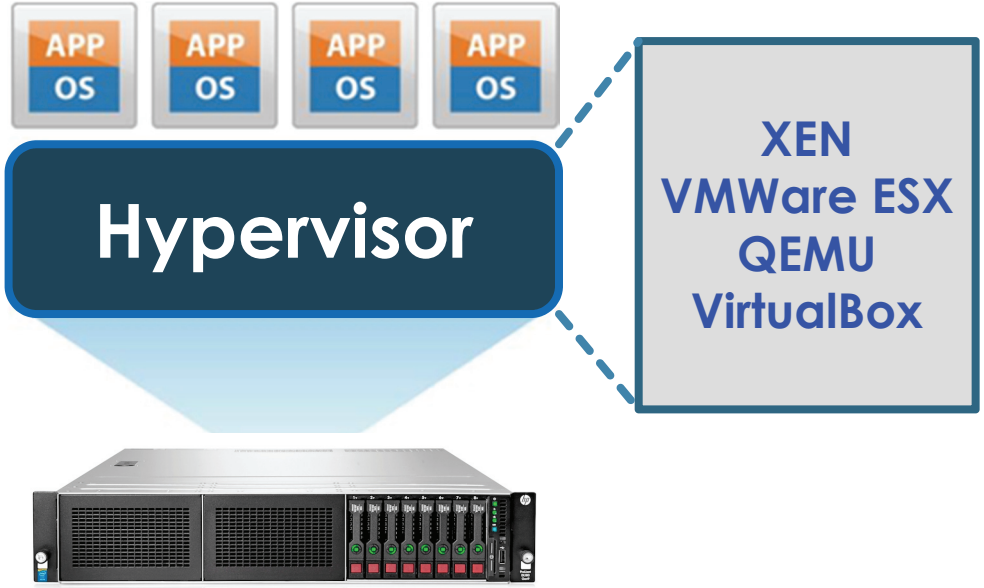
Compared to Previous Approaches, Reuse Distance Based Provides Most Effective Cache Size Estimation

Existing Reuse Distance Calculation Approaches Not Consider Request Type

Our Goal: Propose a Novel Reuse Distance Calculation Scheme that Considers: 1) Distance and 2) Type of Requests

Introduction: Why Virtualization?

- High Resource Utilization ☺
  - Delivering high performance for each VM
- Providing System Isolation ☺
  - Each VM limited to its own resources
- Providing Abstraction ☺
  - Dynamically adjustable
    - Storage
    - Network
    - Computing Resources



Available in: http://www.arumtec.net/fr/outils-virtualisation/outils-de-virtualisation/vmware-esxi-5/presentation-de-vmware-esxi-5, Accessed: Oct, 2015.

Motivation: HDDs vs. SSDs

WD Black SATA HDD (4TB)

Slow (180 MB/s, IOPS: 140) ☹
Power Hungry (Active: 9w, Idle: 8w) ☹
Heavy (780g) ☹
Noisy ☹
MTBF < 1 Mh ☹
Inexpensive (0.06 $/G) ☺
Unlimited Writes ☺

Samsung 850 PRO SATA SSD (2TB)

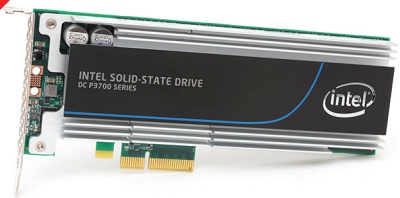
Fast (550 MB/s, IOPS > 50K) ☺
Low Power (Active: 3w, Idle: 60mw) ☺
Light (66g) ☺
Very Low Noise ☺
MTBF: 2 Mh ☺
Expensive (0.1 $/G) ☹
Limited Writes ☹

http://www.storagereview.com/, Accessed: Oct, 2015

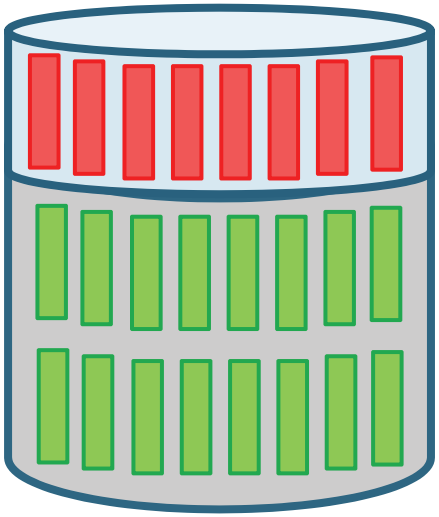
Challenge for Hybrid Storages: IO Caching

- ✓ Balance Workload
- ✓ Handle IO Bursts

Active Data on
High Performance SSD



ACTIVE
SSD



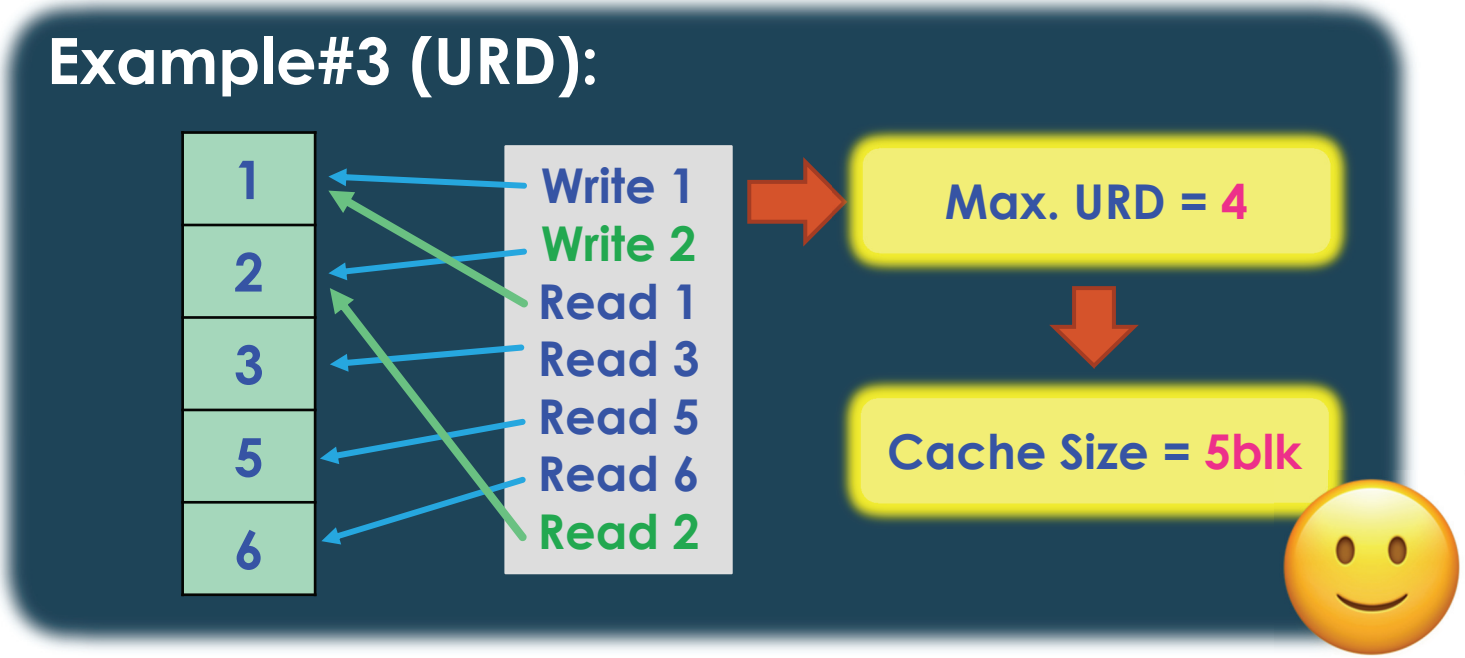
INACTIVE
HDD

Inactive Data on
Low Cost HDD

Hot Data

Cold Data

How URD based cache works? (cont.)



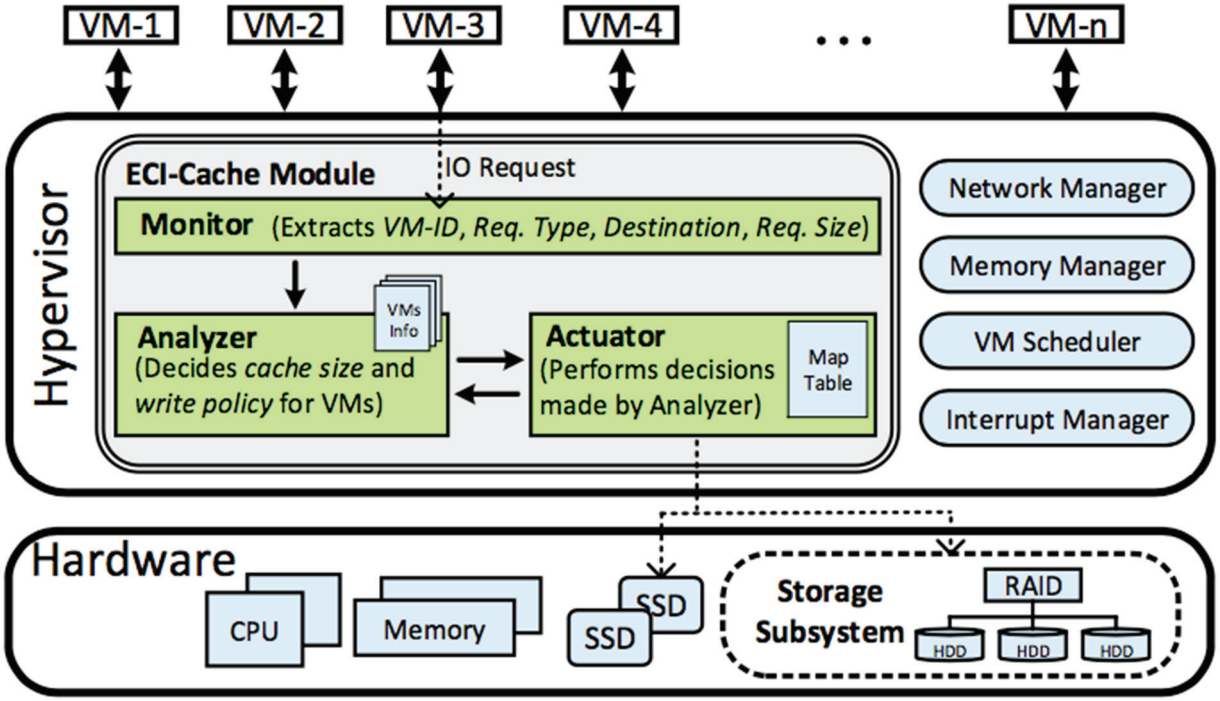
Access ←
Hit ←

1. RD(WA*) < RD(RA*) → TRD α RD(RA*), URD α RD(RA*) ➔ TRD = URD
2. RD(WA*) > RD(RA*) → TRD α RD(WA*), URD α RD(RA*) ➔ TRD > URD

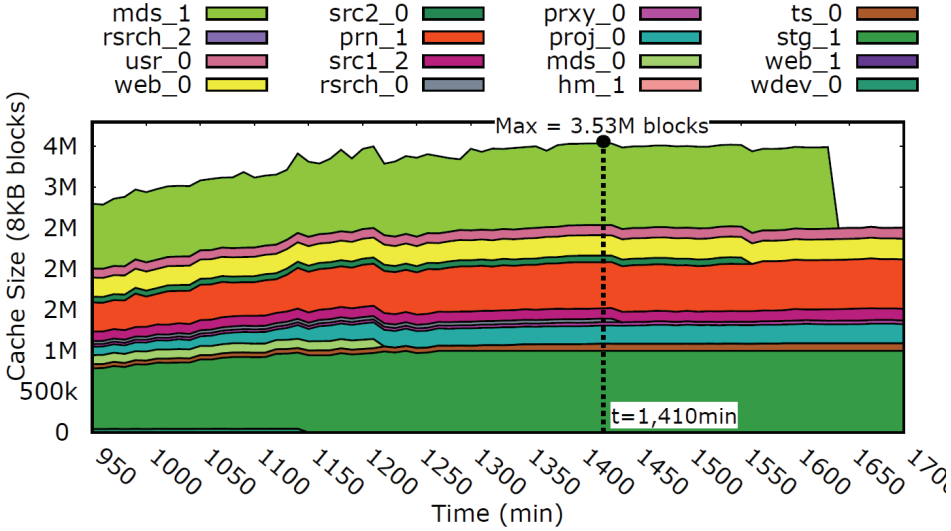
Architecture of ECI-Cache

- Resides in Path of VMs IO Requests
- As a Module of Hypervisor Kernel
- Full Control on VMs IO Requests

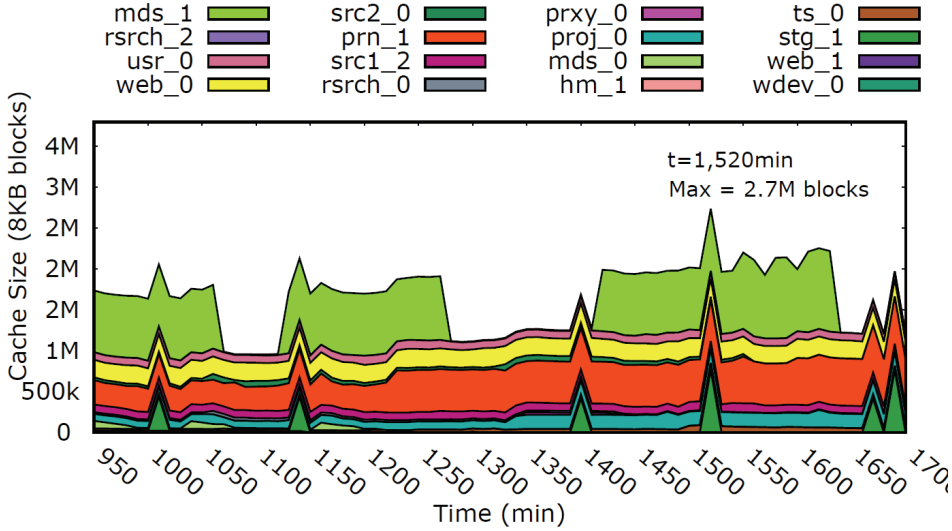
- ✓ Monitor
  - ✓ Extracts IO traces
  - ✓ Device block layer
  - ✓ blktrace
- ✓ Analyzer
  - ✓ Estimates size (URD)
  - ✓ Assigns write policy
- ✓ Actuator
  - ✓ Performs configs.



Experiments: Cache Allocation (Unlimited SSD Space)



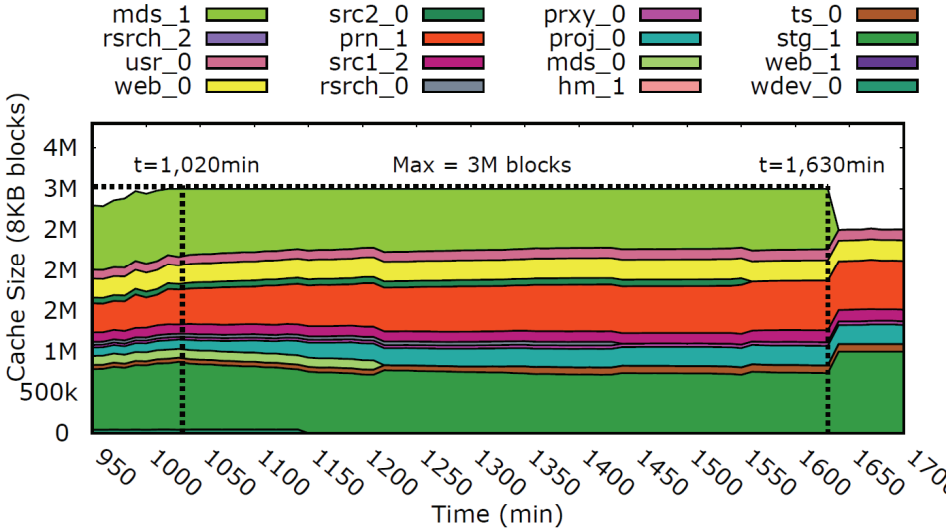
Cache Allocation for VMs by Centaur (TRD)



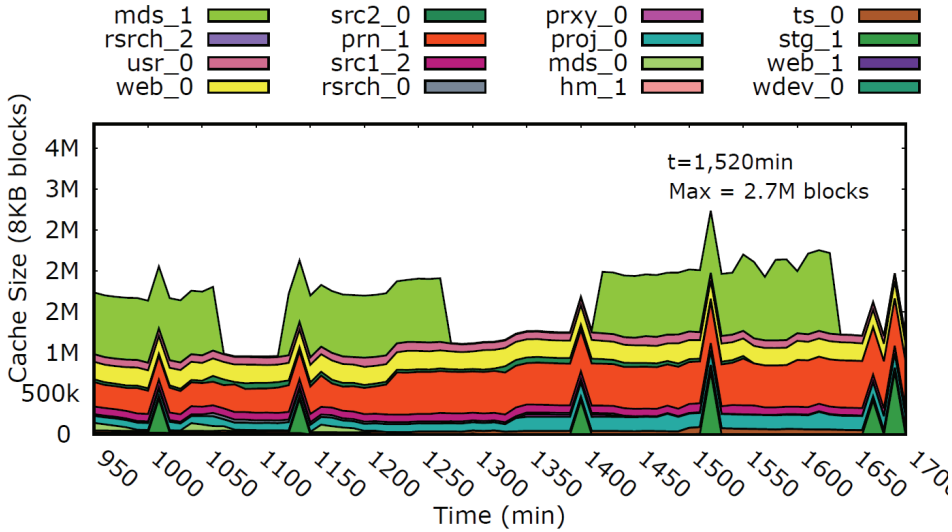
Cache Allocation for VMs by ECI-Cache (URD)

- ✓ **Much Smaller Cache Space Allocation by ECI-Cache**
- ✓ **Similar Hit Ratio by ECI-Cache**
- ✓ **ECI-Cache Improves Perf-Per-Cost by 30%**

Experiments: Cache Allocation (Limited SSD Space → Infeasible)



Cache Allocation for VMs by Centaur (TRD)

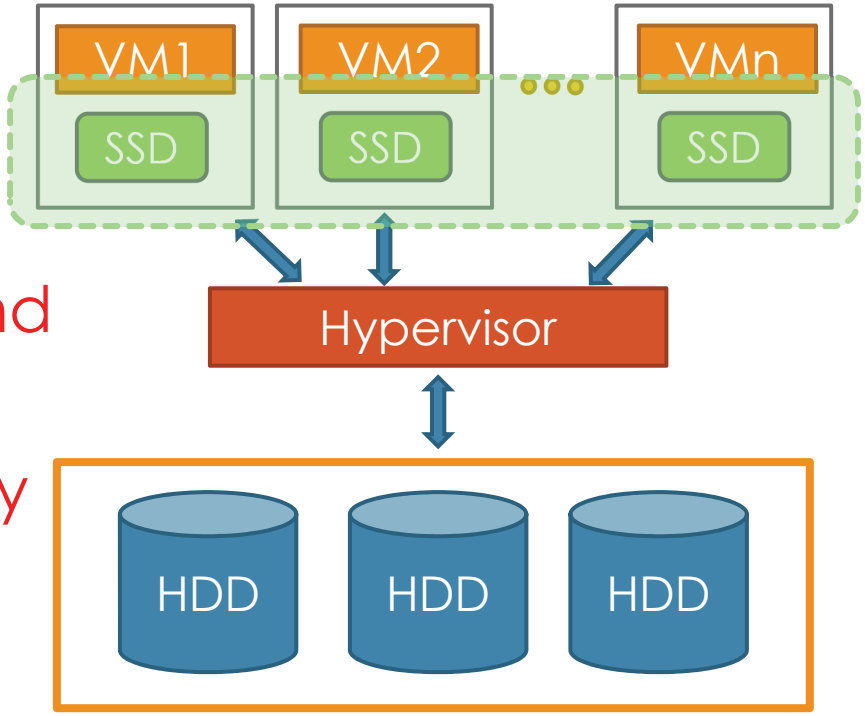


Cache Allocation for VMs by ECI-Cache (URD)

- ✓ Much Smaller Cache Space Allocation by ECI-Cache
- ✓ Higher Hit Ratio by ECI-Cache
- → ECI-Cache Improves Perf-Per-Cost by More than 30%
- → ECI-Cache Improves Performance by 17%

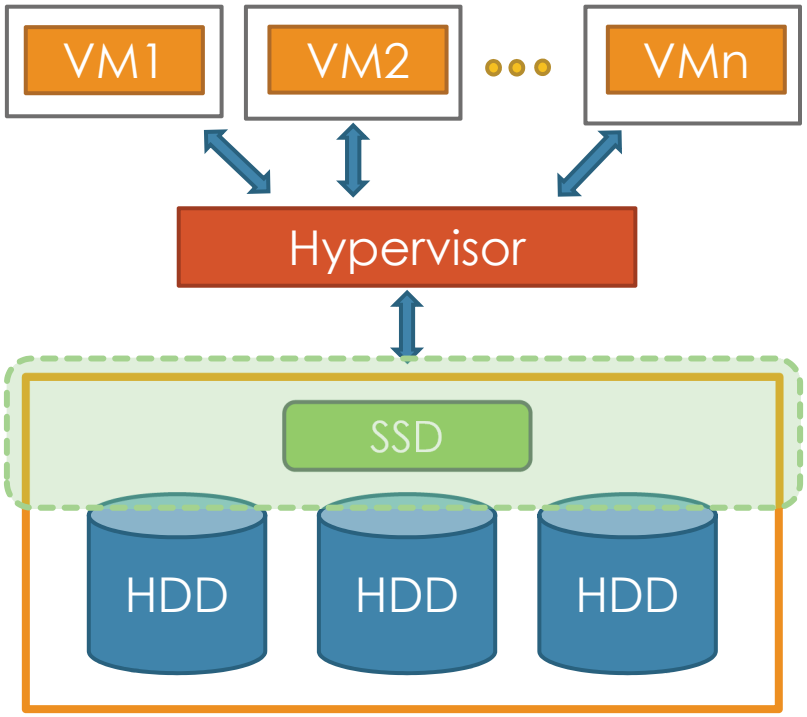
Alternatives For SSD Caching (cont.)

- VM-Based SSD Caching
  - VM has best knowledge and full control of Cache ☺
  - Isolation ☺
  - Guest OS should modify and create cache ☹
  - Cache size not dynamically adjustable ☹



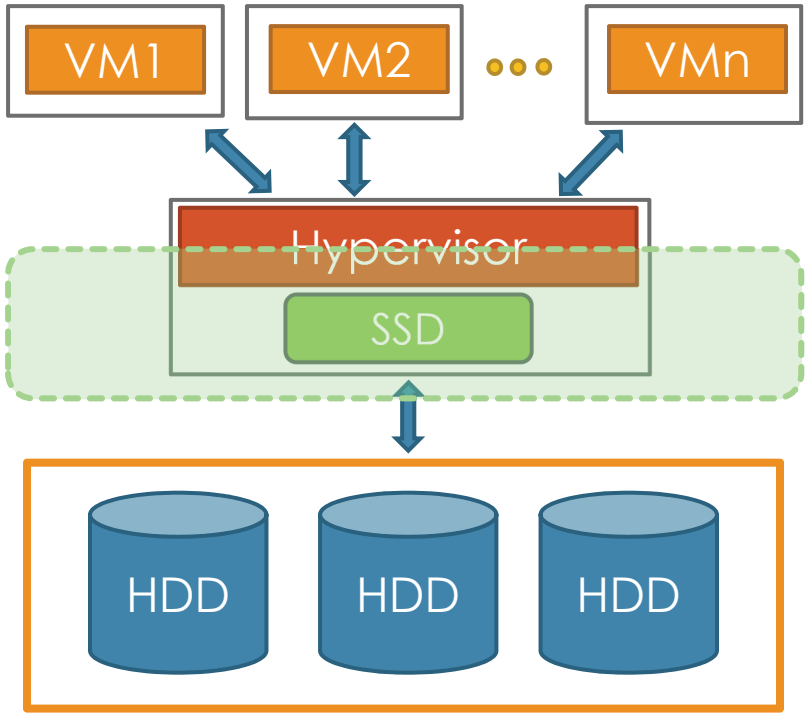
Alternatives For SSD Caching (cont.)

- Storage System-Level SSD Caching
  - Easy to use ☺
  - Not transparent for VMs ☹
  - Performance management cannot be guaranteed ☹



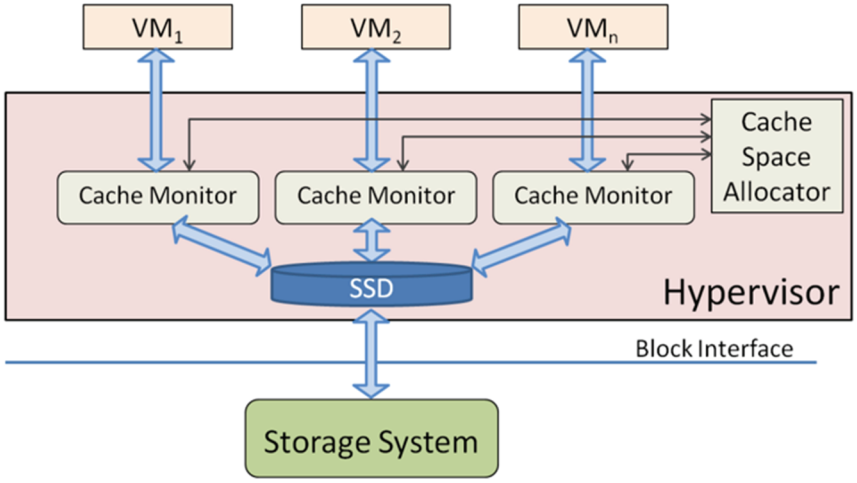
Alternatives For SSD Caching (cont.)

- Hypervisor-Based SSD Caching
  - Transparent for VMs ☺
  - Hypervisor full control ☺
  - Hypervisor is aware of VMs workload ☺



Previous Work: S-CAVE

- Hypervisor-Based SSD Cache
- Granularity: Fixed-Sized
- Policy: Write Through
- Dynamic Size Management
- Estimate Cache Demand For Each VM:
  - Working Set Size (WSS)



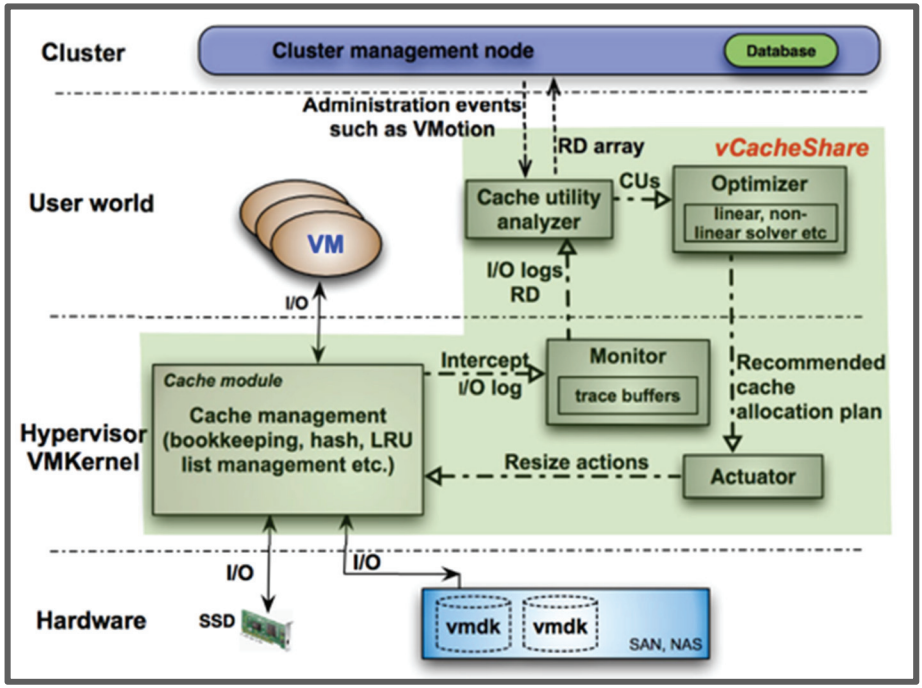
Luo, Tian, et al. "S-cave: Effective ssd caching to improve virtual machine storage performance." *Proceedings of the 22nd international conference on Parallel Architectures and Compilation Techniques (PACT)*, 2013.

Previous Work: vCacheShare

- Hypervisor-Based SSD Cache
- Granularity: Fixed-Sized
- Policy: Write Around (RO)
- Dynamic Cache Size Management
- Estimate Cache Demand for each VM:
  - Cache Utility
    - $CU = I \times RR \times (H(c) + \alpha RI)$

Previous Work: vCacheShare

- Hypervisor-Based SSD Cache
- Granularity: Fixed-Sized
- Policy: Write Around (RO)
- Dynamic Cache Size Management
- Estimate Cache Demand for each VM:
  - Cache Utility
    - $CU = I \times RR \times (H(c) + \alpha RI)$



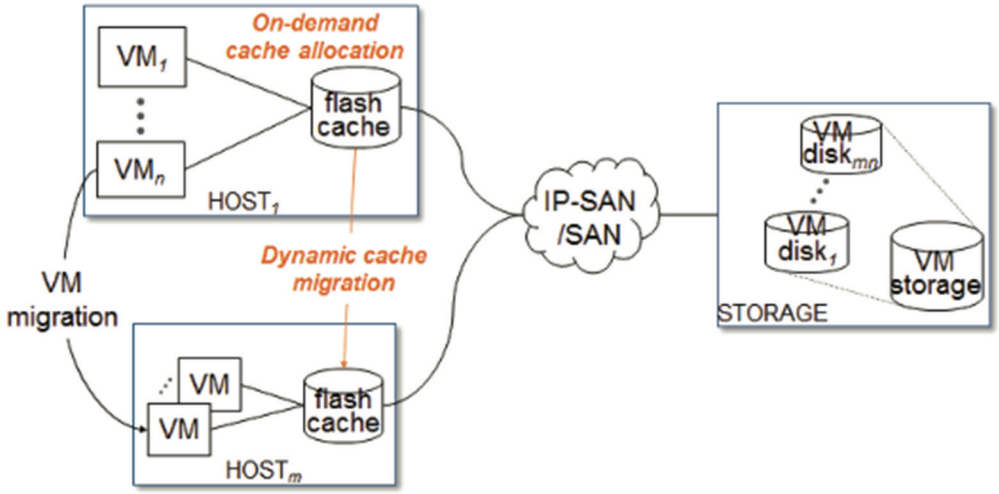
Meng, Fei, et al. "vCacheShare: automated server flash cache space management in a virtualization environment." *USENIX ATC*. 2014.

Previous Work: Centaur

- Hypervisor-Based SSD Cache
- Granularity: Fixed-Sized
- Policy: Write Back
- Dynamic Cache Size Management
- Estimate Cache Demand for each VM:
  - Reuse Distance (RD)
  - MRCs

Previous Work: CloudCache

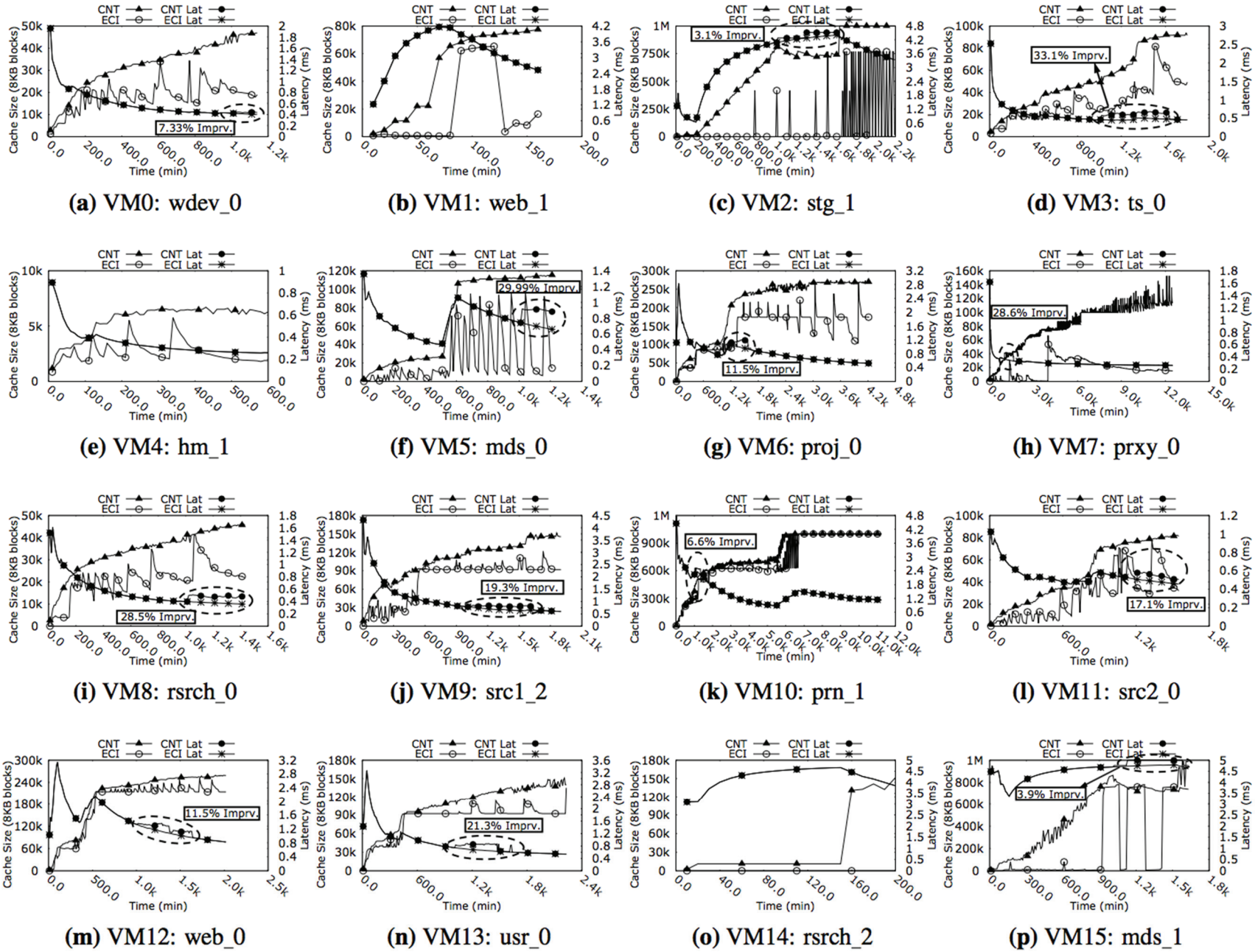
- Hypervisor-Based SSD Cache
- Policy: Write Back
- VMs Migration
  - Performance
  - Endurance



- Estimate Cache Demand for each VM:
  - Reuse Working Set Size (RWSS)
    - Temporal Locality

Arteaga, Dulcardo, et al. "CloudCache: On-demand Flash Cache Management for Cloud Computing." *FAST*. 2016.

Latency Improvement



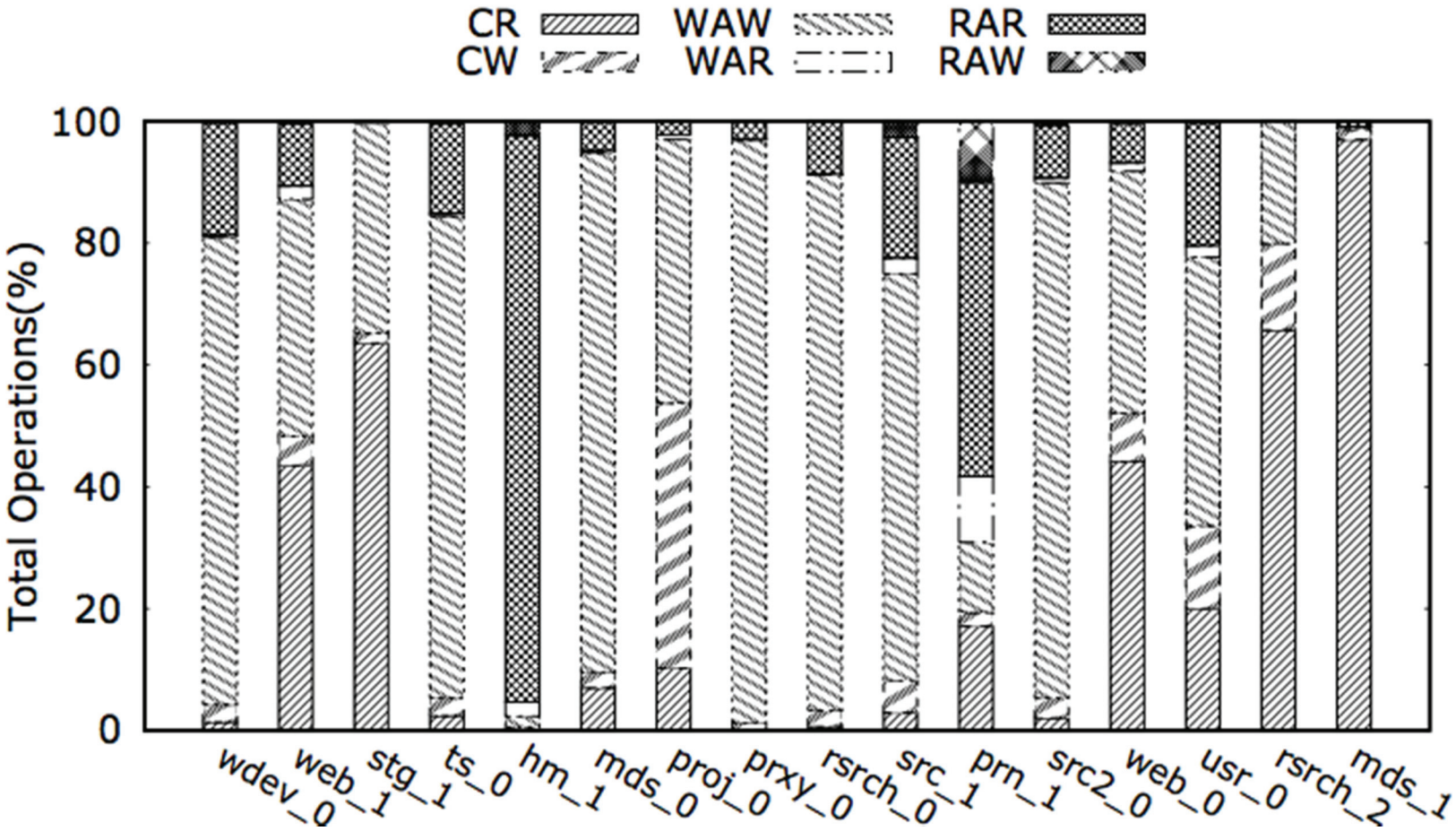
(a) VM0: wdev_0 (b) VM1: web_1 (c) VM2: stg_1 (d) VM3: ts_0

(e) VM4: hm_1 (f) VM5: mds_0 (g) VM6: proj_0 (h) VM7: prxy_0

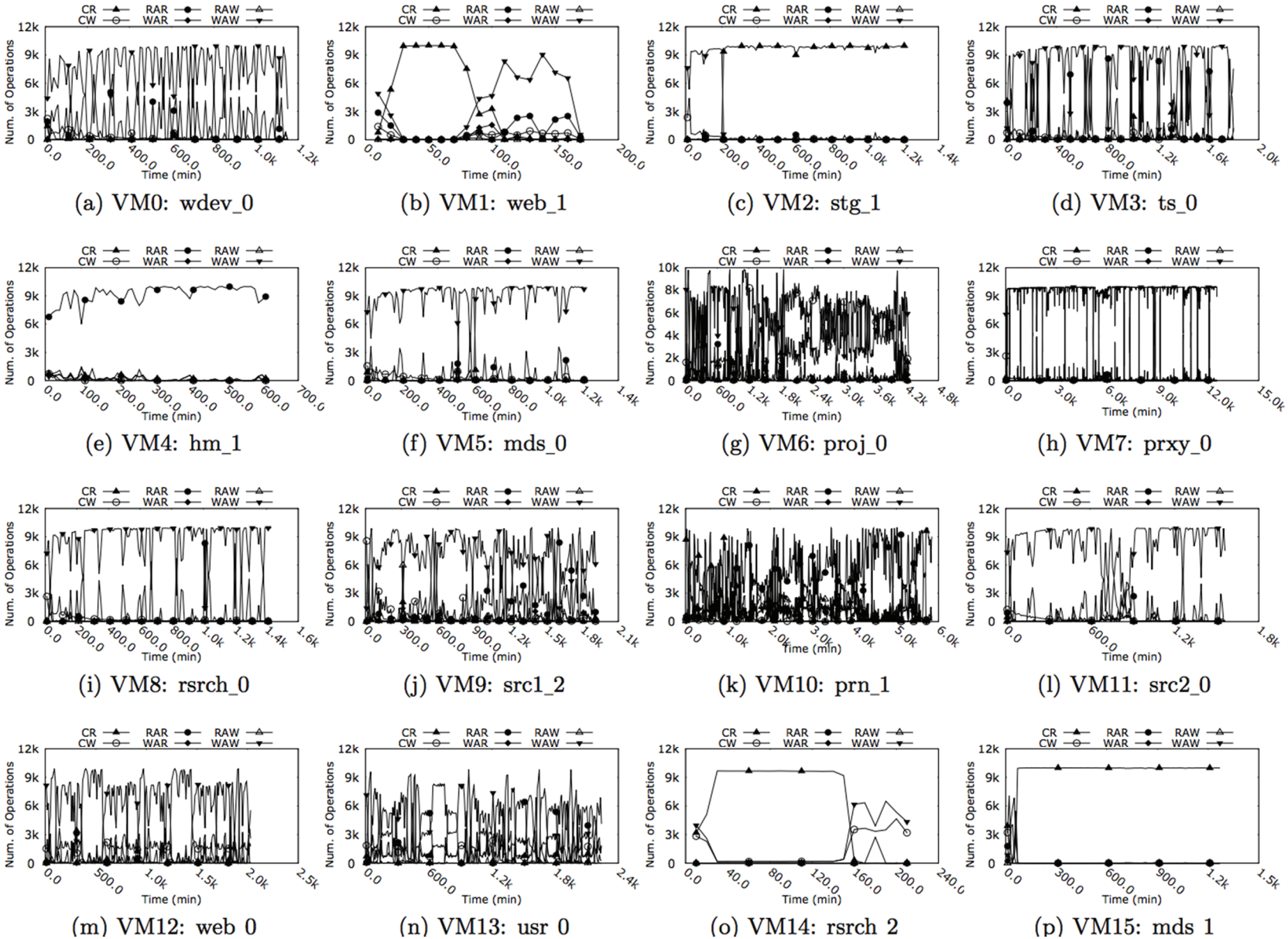
(i) VM8: rsrch_0 (j) VM9: src1_2 (k) VM10: prn_1 (l) VM11: src2_0

(m) VM12: web_0 (n) VM13: usr_0 (o) VM14: rsrch_2 (p) VM15: mds_1

Operations on VMs Workload



Operations on VMs Workload



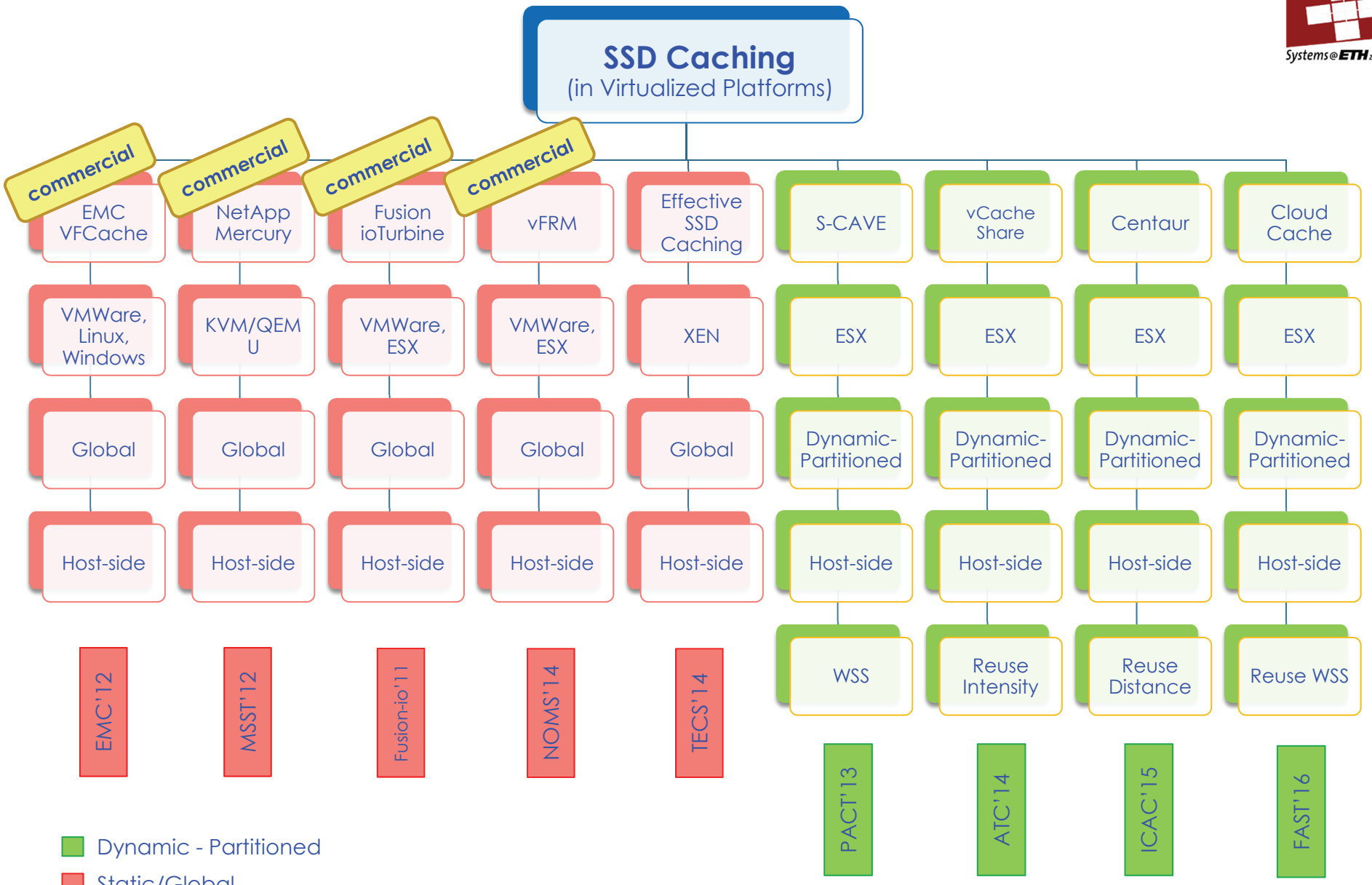
(a) VM0: wdev_0 (b) VM1: web_1 (c) VM2: stg_1 (d) VM3: ts_0

(e) VM4: hm_1 (f) VM5: mds_0 (g) VM6: proj_0 (h) VM7: prxy_0

(i) VM8: rsrch_0 (j) VM9: src1_2 (k) VM10: prn_1 (l) VM11: src2_0

(m) VM12: web 0 (n) VM13: usr 0 (o) VM14: rsrch 2 (p) VM15: mds 1

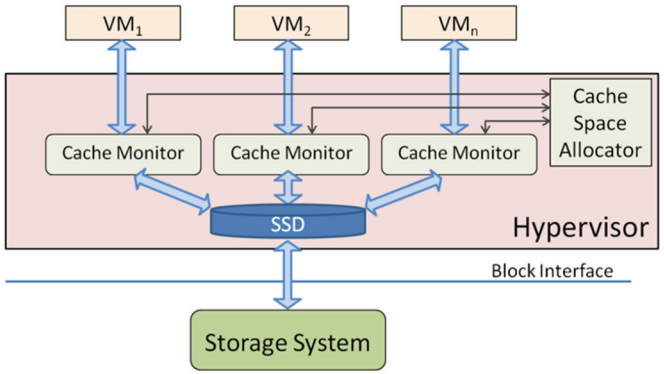
Previous Work

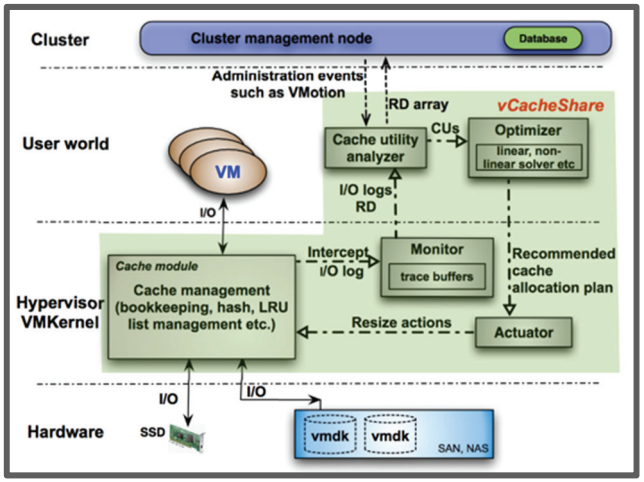


Previous Work:

Dynamic Cache Space Allocation

- S-CAVE [PACT'13]
  - Hypervisor-based SSD cache
  - Policy: Write Through
  - Size estimation:
    - Working Set Size (WSS)
- vCacheShare [ATC'14]
  - Hypervisor-based SSD cache
  - Policy: Write Around (RO)
  - Size estimation:
    - Reuse Intensity → $CU = I \times RR \times (H(c) + \alpha RI)$

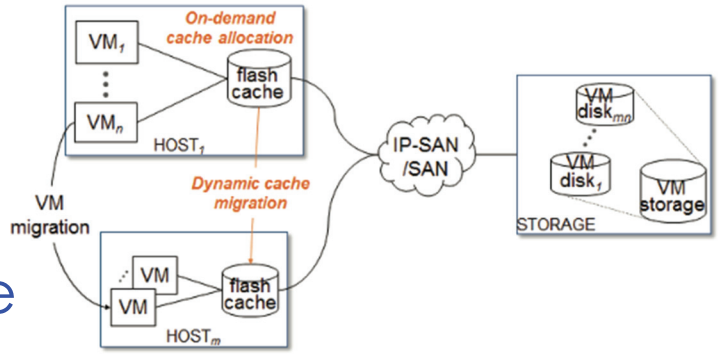




[1] Luo, Tian, et al. "S-cave: Effective ssd caching to improve virtual machine storage performance." *Proceedings of the 22nd international conference on Parallel Architectures and Compilation Techniques (PACT)*, 2013.
[2] Meng, Fei, et al. "vCacheShare: automated server flash cache space management in a virtualization environment." *USENIX ATC*. 2014.

Previous Work (Cont.):
Dynamic Cache Space Allocation

- Centaur [ICAC'15]
  - Hypervisor-based SSD cache
  - Policy: Write Back
  - Size estimation:
    - Reuse Distance (RD)
    - MRCs
- CloudCache [FAST'16]
  - Hypervisor-based SSD cache
  - Policy: Write Back
  - VMs migration → Perf. + Endurance
  - Size estimation:
    - Reuse Working Set Size (RWSS)
      - Temporal locality



Arteaga, Dulcardo, et al. "CloudCache: On-demand Flash Cache Management for Cloud Computing." *FAST*. 2016.