Caching less for better performance: Balancing cache size and update cost of flash memory cache in hybrid storage systems

Yongseok Oh Jongmoo Choi Donghee Lee Sam H. Noh

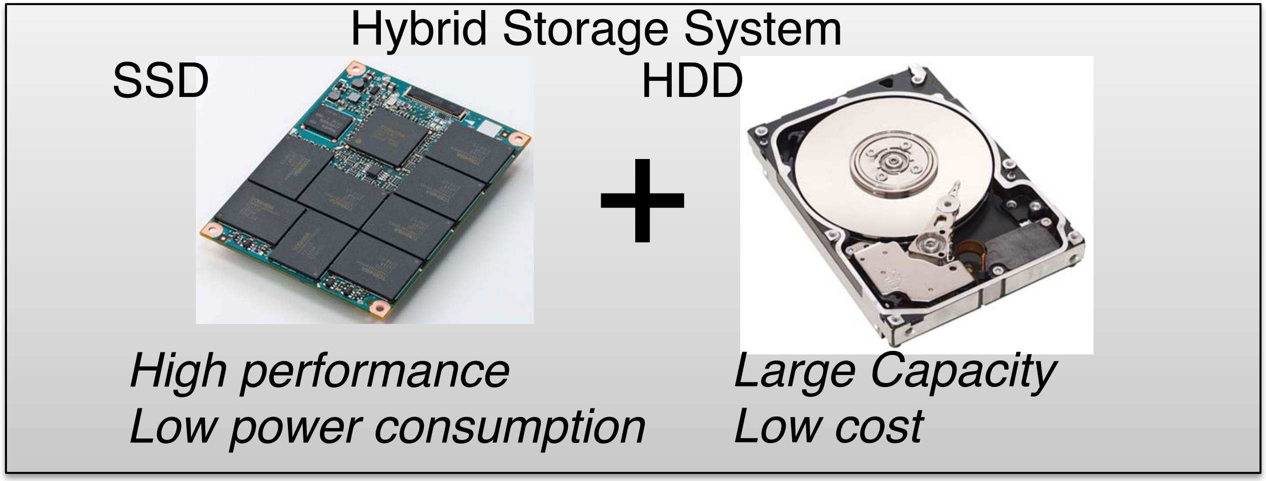
University of Seoul
{ysoh,dhl_express}@uos.ac.kr

Dankook University
choijm@dankook.ac.kr

Hongik University
samhnoh@hongik.ac.kr

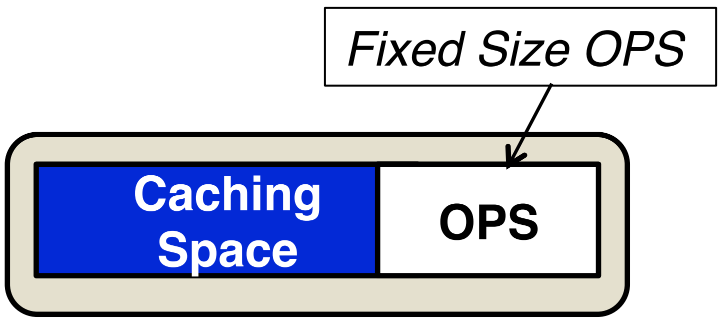
Hybrid Storage Systems

- Harness benefits of SSDs and HDDs
  - High performance, large capacity, affordable cost
- SSDs used as flash cache (NVCache)
  - Seagate Momentus XT(SLC 4GB), OCZ RevoDrive Hybrid (MLC 100GB)
- *Our focus: issue of managing flash cache*



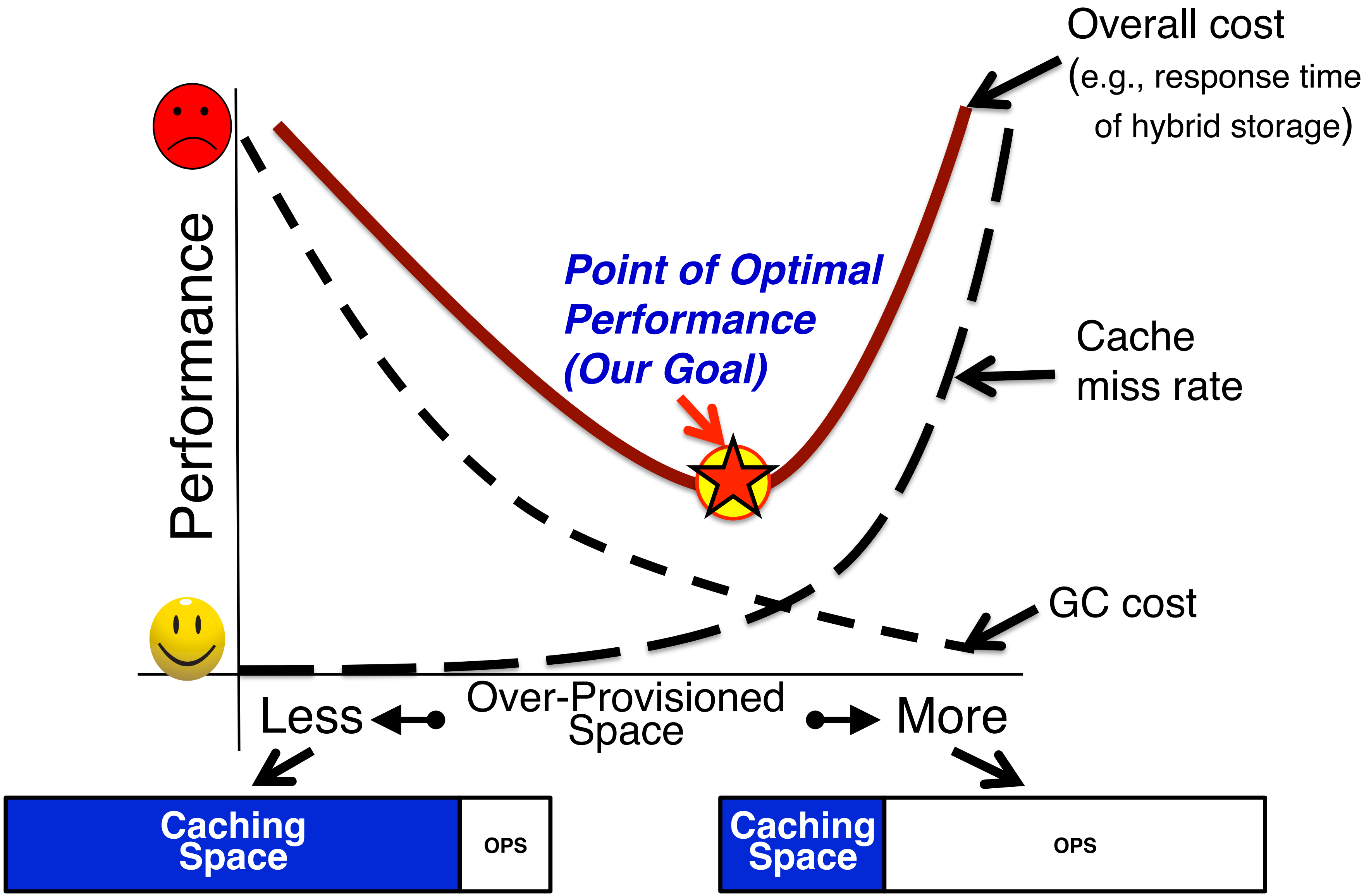
Important Characteristics of Flash based SSDs

- Maintain *Over-Provisioned Space* (OPS)
  - Reserved space for Garbage Collection (GC)
  - Greatly influence *GC performance*
- Typical SSDs
  - OPS size is *fixed*
  - Optimal size is *unknown*
  - Cannot adapt to workload changes

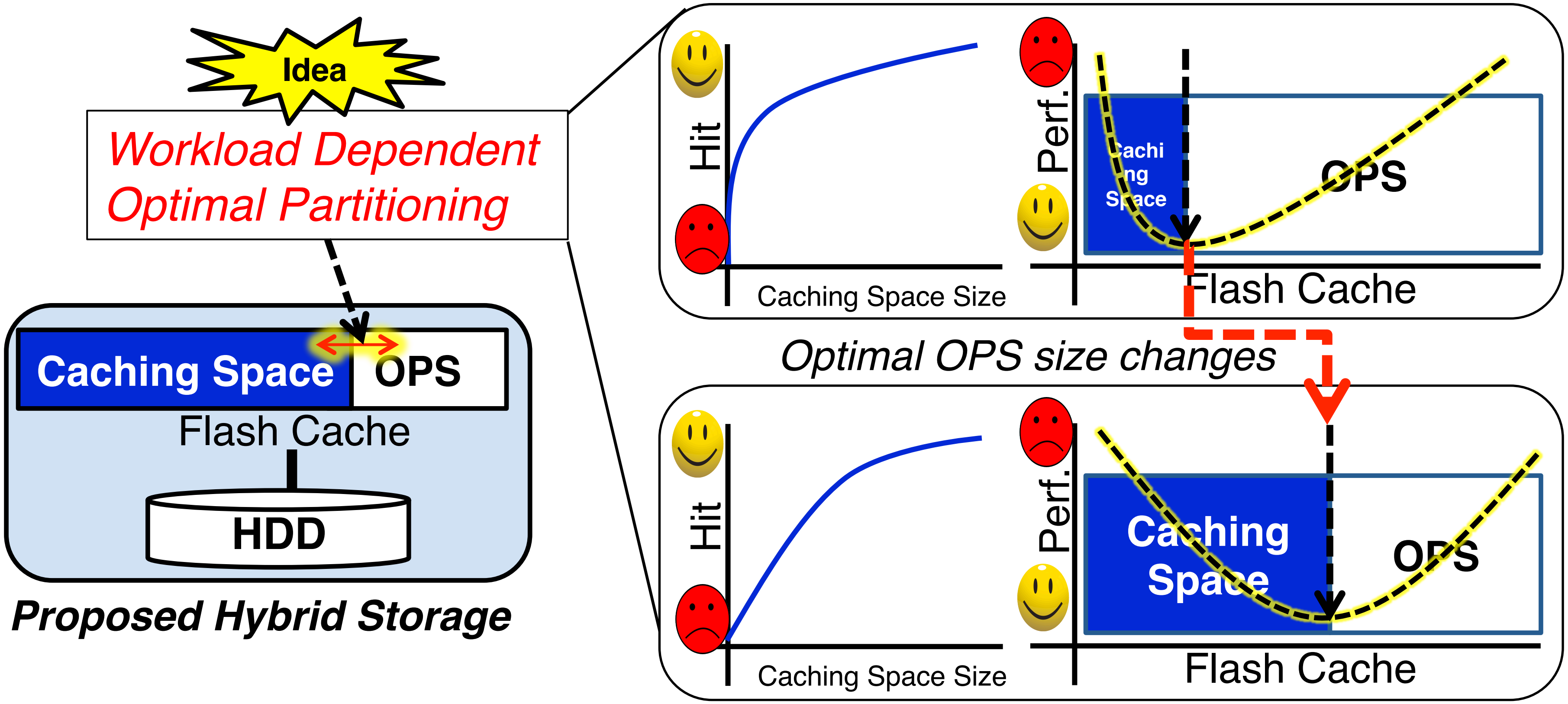


Flash based SSD

Our Goal: Find Optimal OPS Size



Workload Dependent Optimal Partitioning



- Periodically adjust OPS size to maximize the performance
  - Based on hit ratio and garbage collection cost
- Question: ***how to find optimal OPS size?***
  - Solution: ***Hybrid Storage Cost Model*** *(Dynamically adjusted according to workload)*

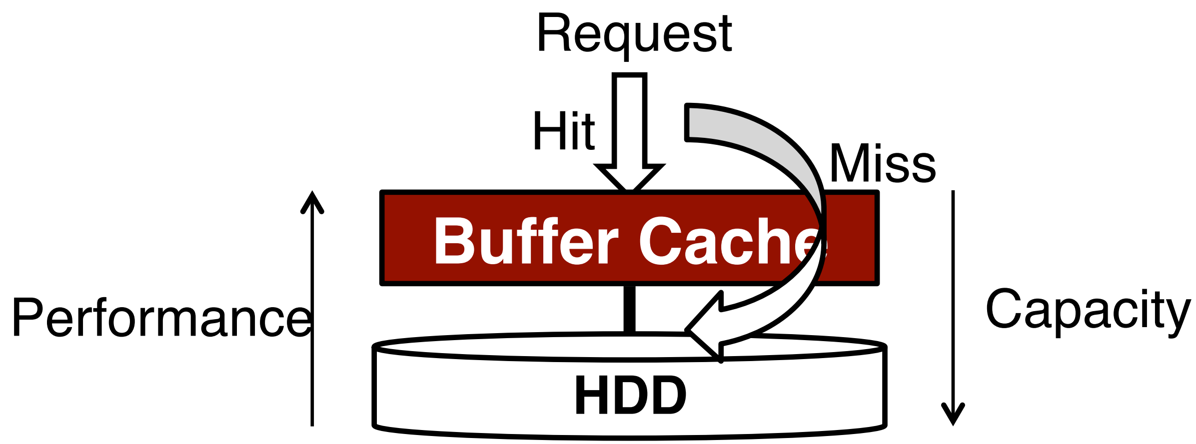
Outline

- Introduction
- Hybrid Cost Model
- Implementation
- Evaluation
- Conclusion

OS 101: Access Cost Model (ACM)

C_{ACM} = Hit Rate x Cache Cost + (1-Hit Rate) x Miss Penalty

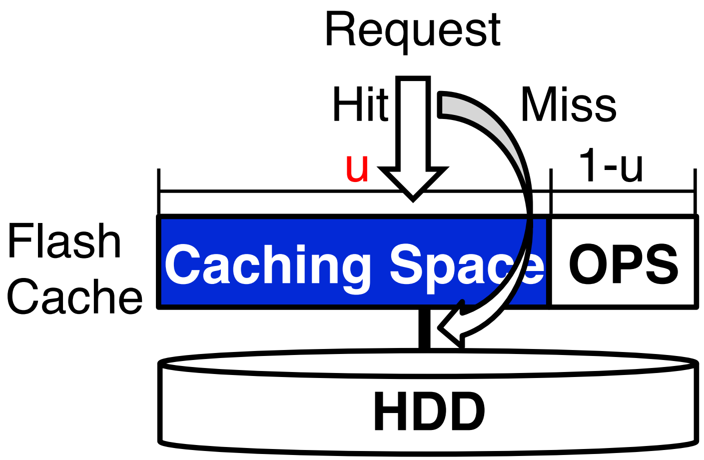
→ Expected I/O cost



Storage Hierarchy

Hybrid Storage: Access Cost Model

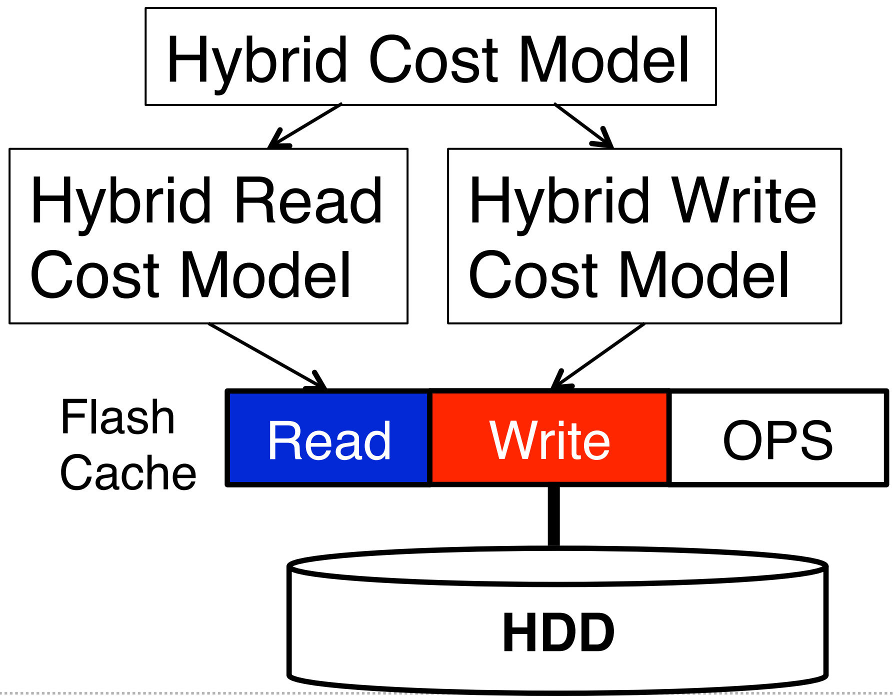
$C_{ACM}(u)$= Hit Rate(u) x Flash Cache Cost(u) + (1-Hit Rate(u)) x Miss Penalty(u)



- $C_{ACM}(u)$ represents expected I/O cost based on *u*
  - Incorporating *u* into the access cost model
- Flash Cache is divided based on *u* (tunable)
  - *u* is fraction of caching space in flash cache (e.g., $0 \leq u \leq 1.0$)
  - *u* influences hit ratio and access cost of flash cache

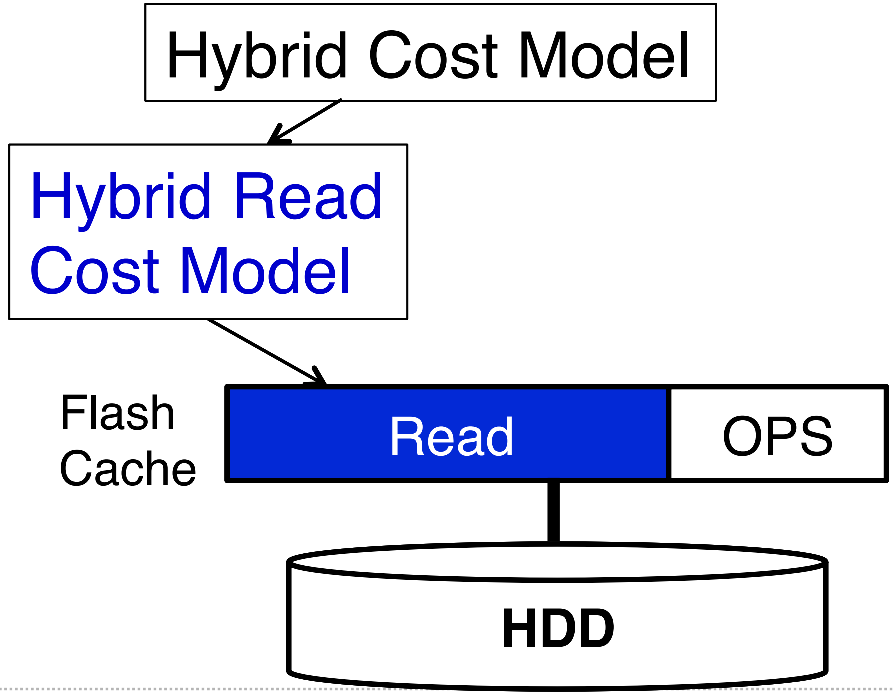
Overview of Hybrid Cost Model

- Hybrid cost model represents expected I/O cost
  - Combines *hybrid read cost model* and *hybrid write cost model*
  - Caching space divided into read and write spaces
- For this talk we derive hybrid read cost model



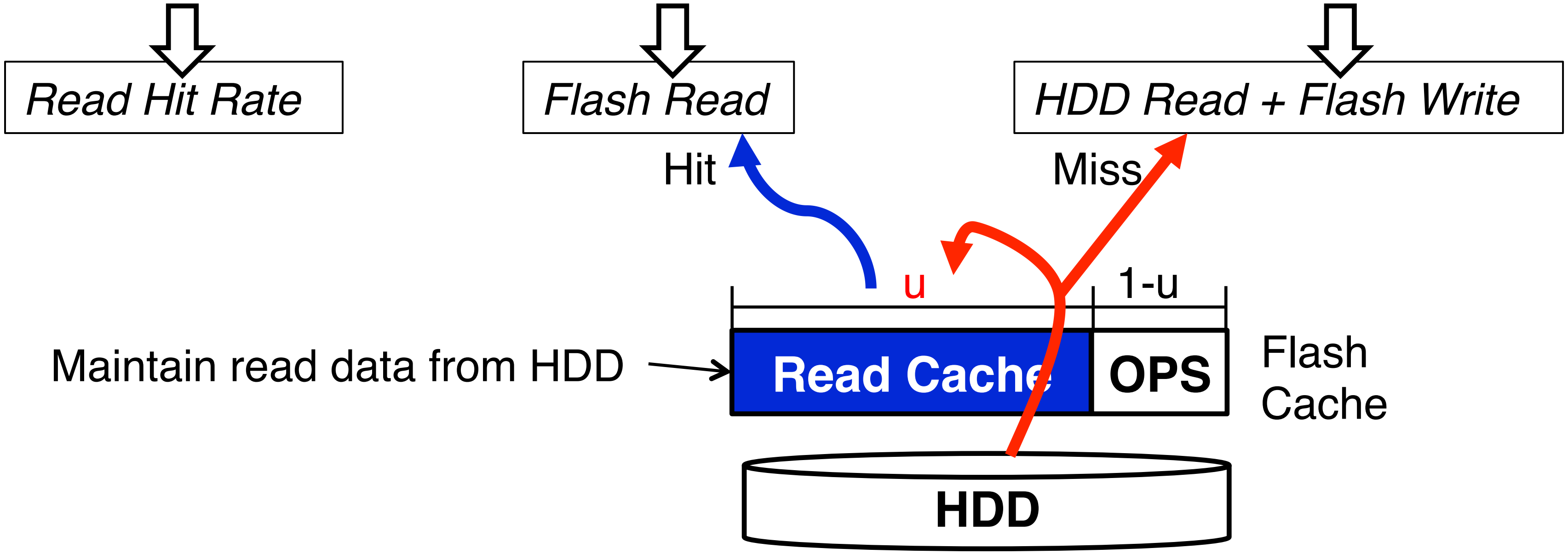
Overview of Hybrid Cost Model

- Hybrid cost model represents expected I/O cost
  - Combines *hybrid read cost model* and *hybrid write cost model*
  - Caching space divided into read and write spaces
- *For this talk we derive hybrid read cost model*



OPS Aware Hybrid ***Read*** Cost Model

$C_{HR}(u)$ = Hit Rate(u) x Flash Cache Cost(u) + (1-Hit Rate(u)) x Miss Penalty(u)



- Requirements for derivation
  - Read Hit Rate Function
  - HDD Cost Model
  - Flash Cache Cost Model

Hybrid Read Cost Model

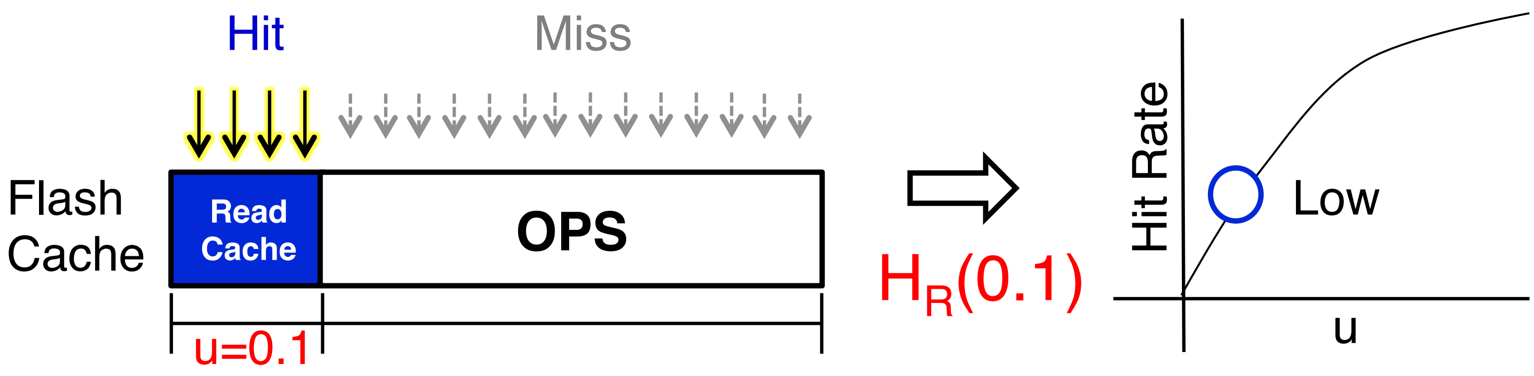
$C_{HR}(u)$ = Hit Rate(u) x Flash Cache Cost(u) + (1-Hit Rate(u)) x Miss Penalty(u)

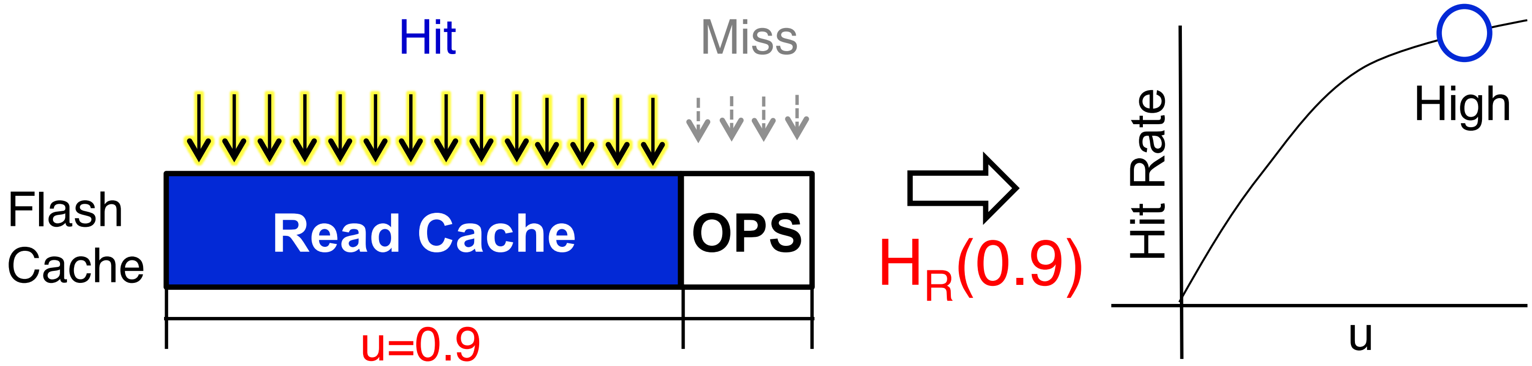
Read Hit Rate | Flash Cache Read | HDD Read + Flash Cache Write

- Read Hit Rate Function
- HDD Cost Model
- Flash Cache Cost Model
- Finding Optimal Point

Read Hit Rate Function

- Read Hit rate function: $H_R(u)$, miss rate: $1-H_R(u)$
  - Related to workload pattern
  - Depends on u





Hybrid Read Cost Model

$C_{HR}(u)$ = Hit Rate(u) x Flash Cache Cost(u) + (1-Hit Rate(u)) x Miss Penalty(u)

⇩ $H_R(u)$

⇩ Flash Cache Read

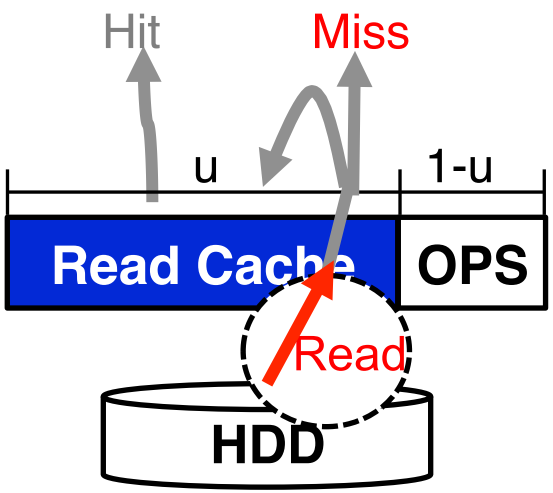
⇩ HDD Read + Flash Cache Write

- Read Hit Rate Function
- HDD Cost Model
- Flash Cache Cost Model
- Finding Optimal Point

HDD Cost Model

- HDD I/O requires positioning cost + bus transfer cost [Hylog]
  - HDD Read: $C_{DR} = C_{D\_RPOS} + P/B$
  - HDD Write: $C_{DW} = C_{D\_WPOS} + P/B$
- Independent from u

Notation	Description
C_{D_RPOS}	Read positioning Cost
C_{D_WPOS}	Write positioning Cost
P	Page size (in bytes)
B	Bandwidth



Hybrid Read Cost Model

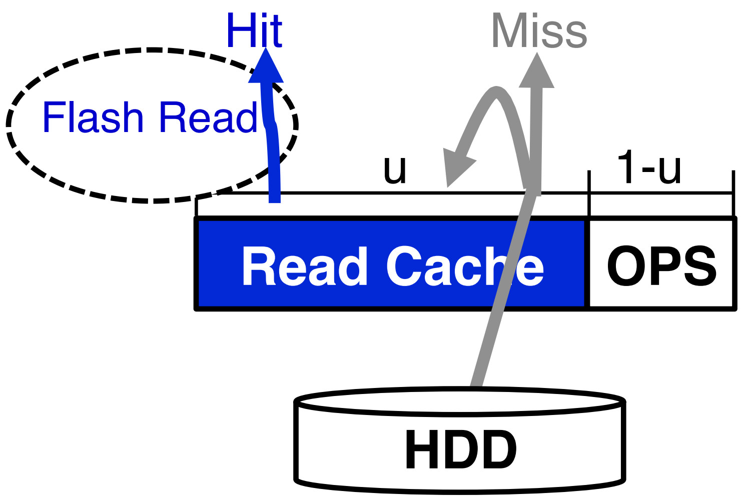
$C_{HR}(u)$ = Hit Rate(u) x Flash Cache Cost(u) + (1-Hit Rate(u)) x Miss Penalty(u)

- Hit Rate(u) → $H_R(u)$
- Flash Cache Cost(u) → Flash Cache Read
- Miss Penalty(u) → C_{DR} + Flash Cache Write

- Read Hit Rate Function
- HDD Cost Model
- Flash Cache Cost Model
  - Read Cost Model
  - Write Cost Model
- Finding Optimal Point

Flash Cache *Read* Cost Model

- *Hit request* requires flash page read: C_{PR}
  - Near constant cost (e.g., 25us)
  - Regardless of garbage collection cost
  - Independent from *u*



Hybrid Read Cost Model

$C_{HR}(u)$ = Hit Rate(u) x Flash Cache Cost(u) + (1-Hit Rate(u)) x Miss Penalty(u)

Hit Rate(u) ⇩ $H_R(u)$

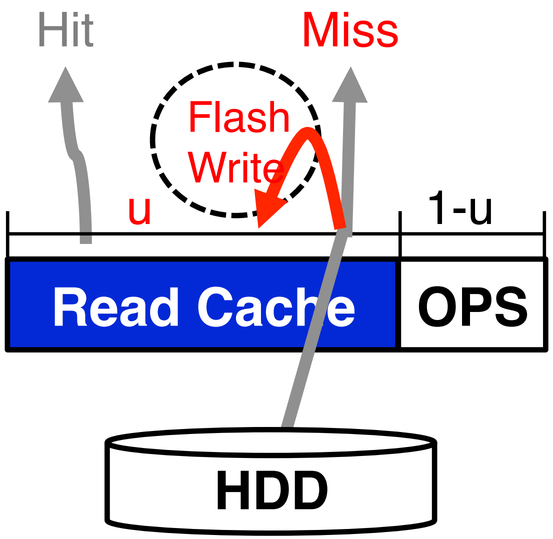
Flash Cache Cost(u) ⇩ C_{PR}

Miss Penalty(u) ⇩ C_{DR} + Flash Cache Write

- Read Hit Rate Function
- HDD Cost Model
- Flash Cache Cost Model
  - Read Cost Model
  - Write Cost Model
- Finding Optimal Point

Flash Cache *Write* Cost Model

- *Miss request* requires flash page write: $C_{PW}(u)$
  - Write cost + GC cost(u)
  - GC cost(u) varies depending on ***u*** [LFS, Janus-FTL]
  - As *u* increases, GC cost(u) increases → $C_{PW}(u)$ increases



↓ Detailed Derivation

$$C_{GC}(u) = u \cdot N_{NP} \cdot C_{CP} + C_E$$

$$C_{PW}(u) = \frac{C_{GC}(u)}{(1-u) \cdot N_P} + C_{PROG}$$

See the paper for derivation

Hybrid Read Cost Model

$C_{HR}(u)$ = Hit Rate(u) x Flash Cache Cost(u) + (1-Hit Rate(u)) x Miss Penalty(u)

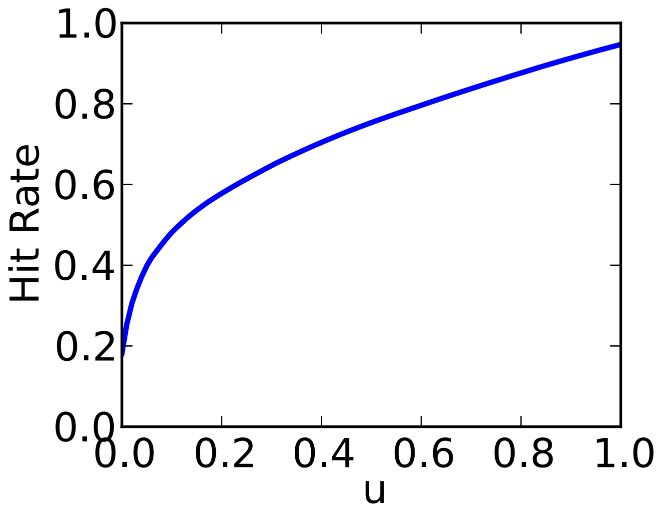
⇩ *Derive*

$$C_{HR}(u) = H_R(u) * C_{PR} + (1-H_R(u)) * (C_{DR}+C_{PW}(u))$$

- Read Hit Rate Function
- HDD Cost Model
- Flash Cache Cost Model
- Finding Optimal Point

Finding Optimal Point

1. Observe Hit Ratio

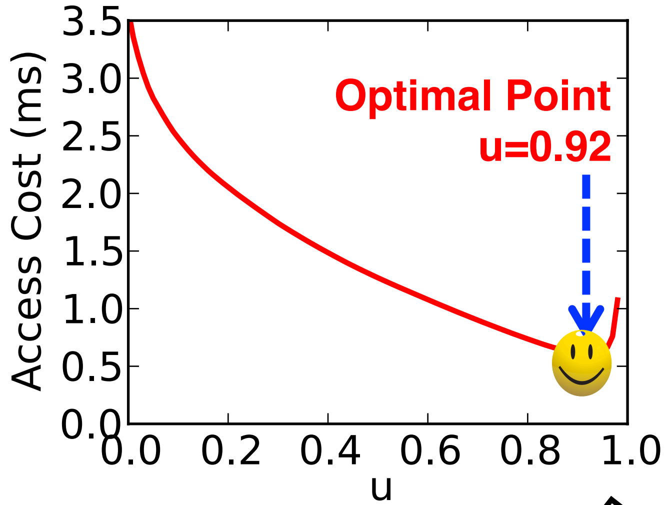


(a) Read hit ratio

2. Calculate for all values of u

$$C_{HR}(u) = H_R(u) * C_{PR} + (1-H_R(u)) * (C_{DR}+C_{PW}(u))$$

3. Find Optimal Point



(b) Read access cost

4. Adjust

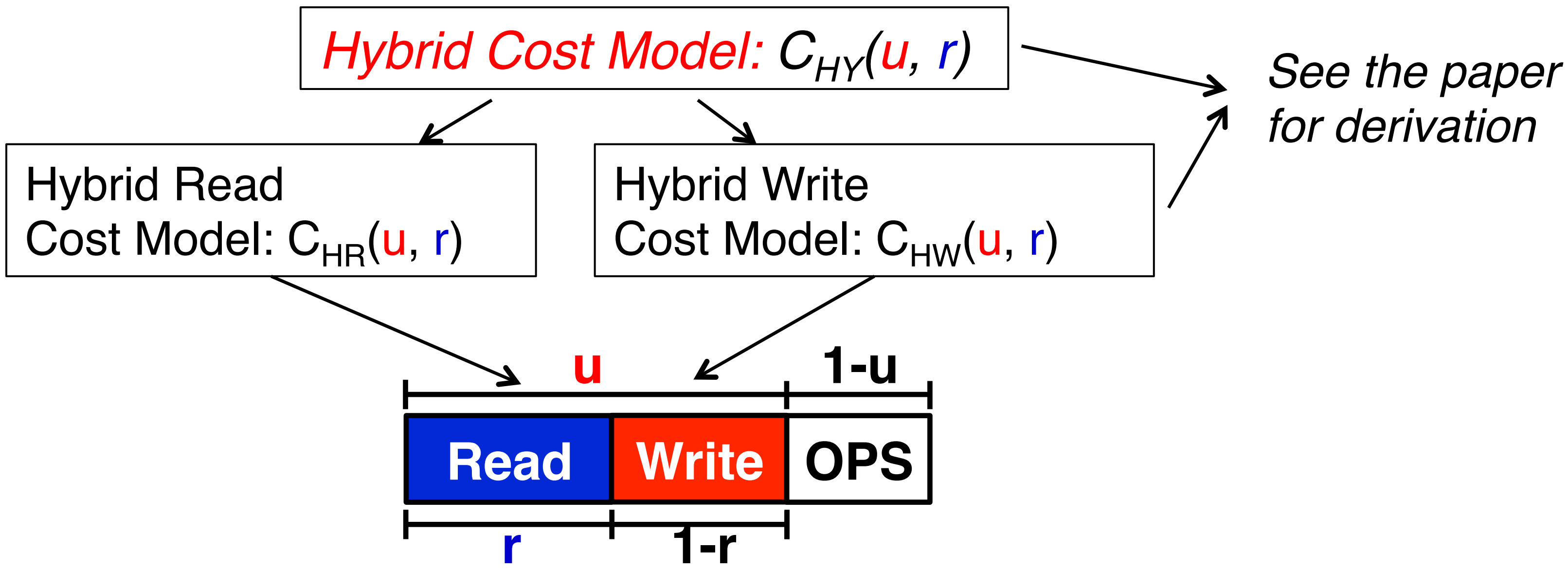
Flash Cache: e.g., 4GB

Partition based on optimal u = 0.92

Caching Space 3.68GB	OPS 0.32GB

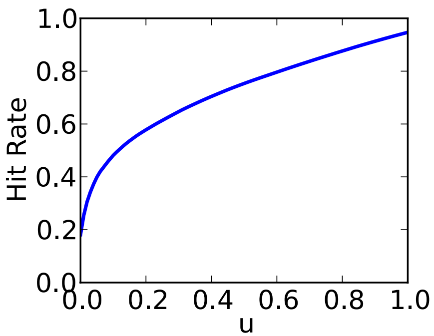
Hybrid Cost Model: Distinguishing Read and Write

- $C_{HY}(u, r)$ represents expected I/O cost based on u and r
  - Caching space divided into read and write spaces based on r
  - r is fraction of read space in caching space (e.g., $0 \le r \le 1.0$)
  - Modification: $C_{HR}(u) \rightarrow C_{HR}(u, r)$, $C_{HW}(u) \rightarrow C_{HW}(u, r)$
- Used to find optimal values: u and r

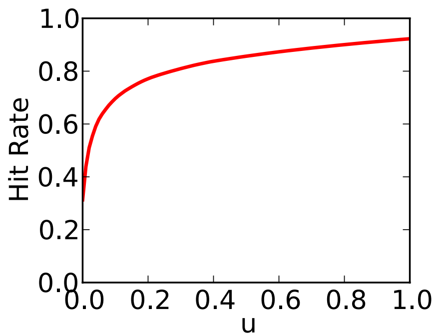


Calculate Hybrid Cost Model

1. Observe Hit Ratio

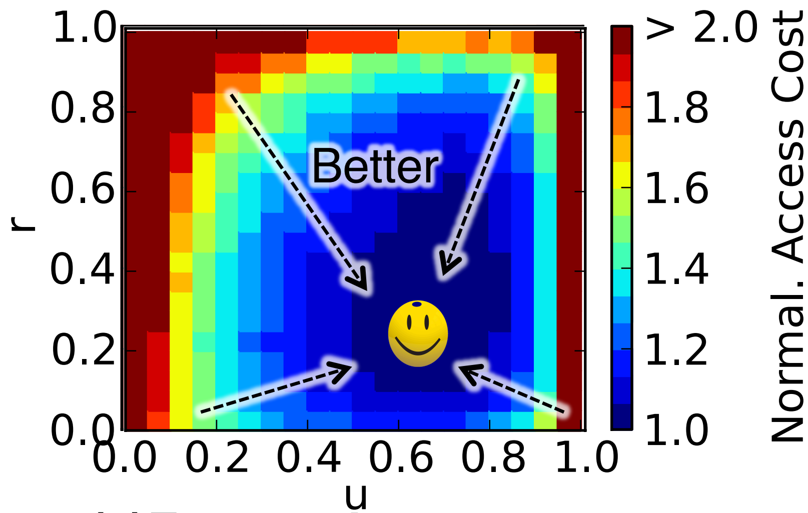


(a) Read hit ratio



(b) Write hit ratio

3. Draw Access Cost Graph



(c)Expected access cost

2. Calculate based on u and r

Hybrid Cost: $C_{HY}(u,r) = C_{HR}(u,r) \cdot IO_R + C_{HW}(u,r) \cdot IO_W$

Hybrid Read Cost: $C_{HR}(u,r) = H_R(u,r) \cdot C_{PR} + (1 - H_R(u,r)) \cdot (C_{DR} + C_{PW}(u))$

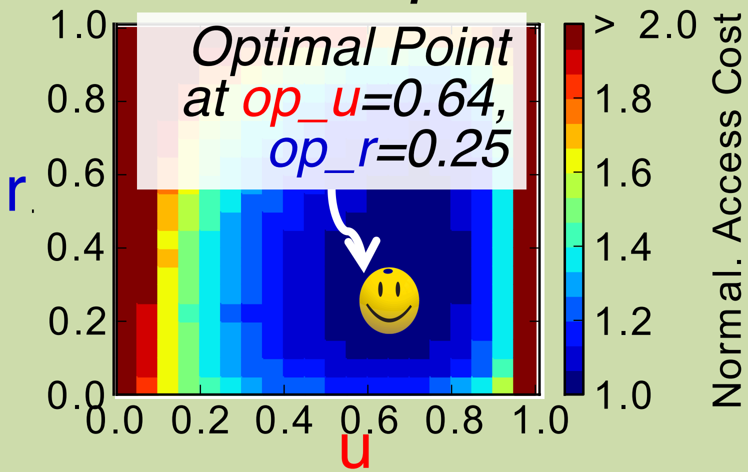
Hybrid Write Cost: $C_{HW}(u,r) = H_W(u,r) \cdot C_{WH} + (1 - H_W(u,r)) \cdot (C_{PR} + C_{DW} + C_{PW}(u))$

Optimal Partitioning Algorithm with Hybrid Cost Model

- ***Periodically Execute Optimal Partitioning Algorithm***

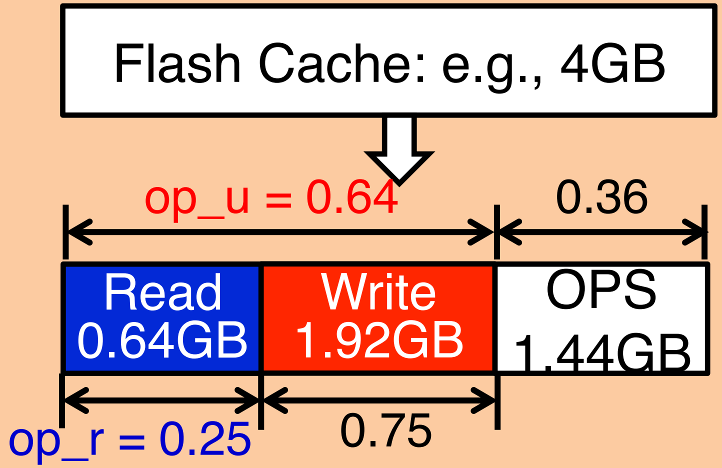
```
for u ← step; u < 1.0; u ← u + step do
    for r ← 0.0; r ≤ 1.0; r ← r + step do
        cur_cost ← C_HY(u, r)
        if cur_cost < op_cost then
            op_cost ← cur_cost
            op_u ← u, op_r ← r
        end if
    end for
end for
ADJUST_CACHE_SIZE(op_u, op_r)
```

- ***Find u and r resulting in Optimal I/O Cost***



op_u op_r

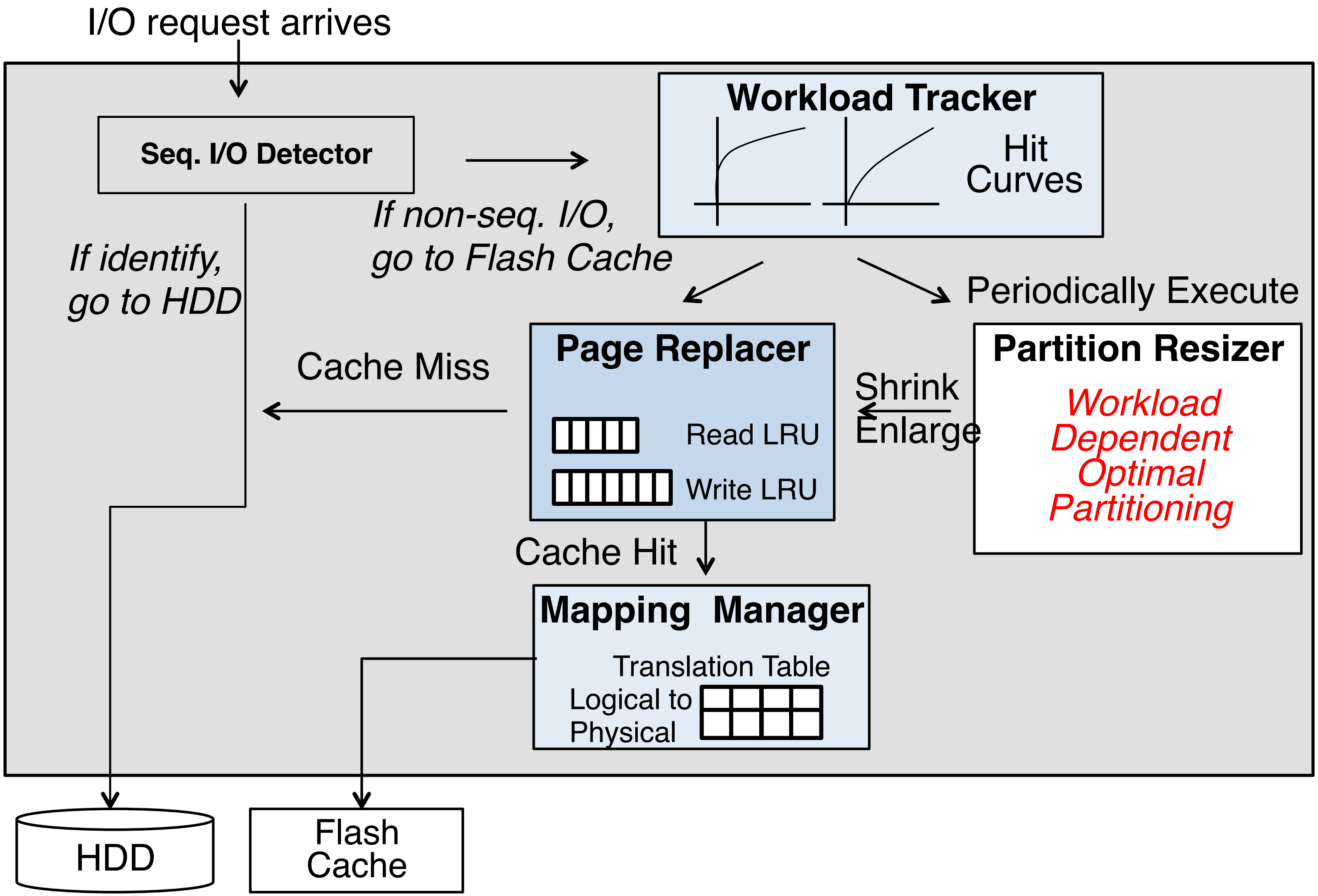
- ***Adjust Flash Cache partition***



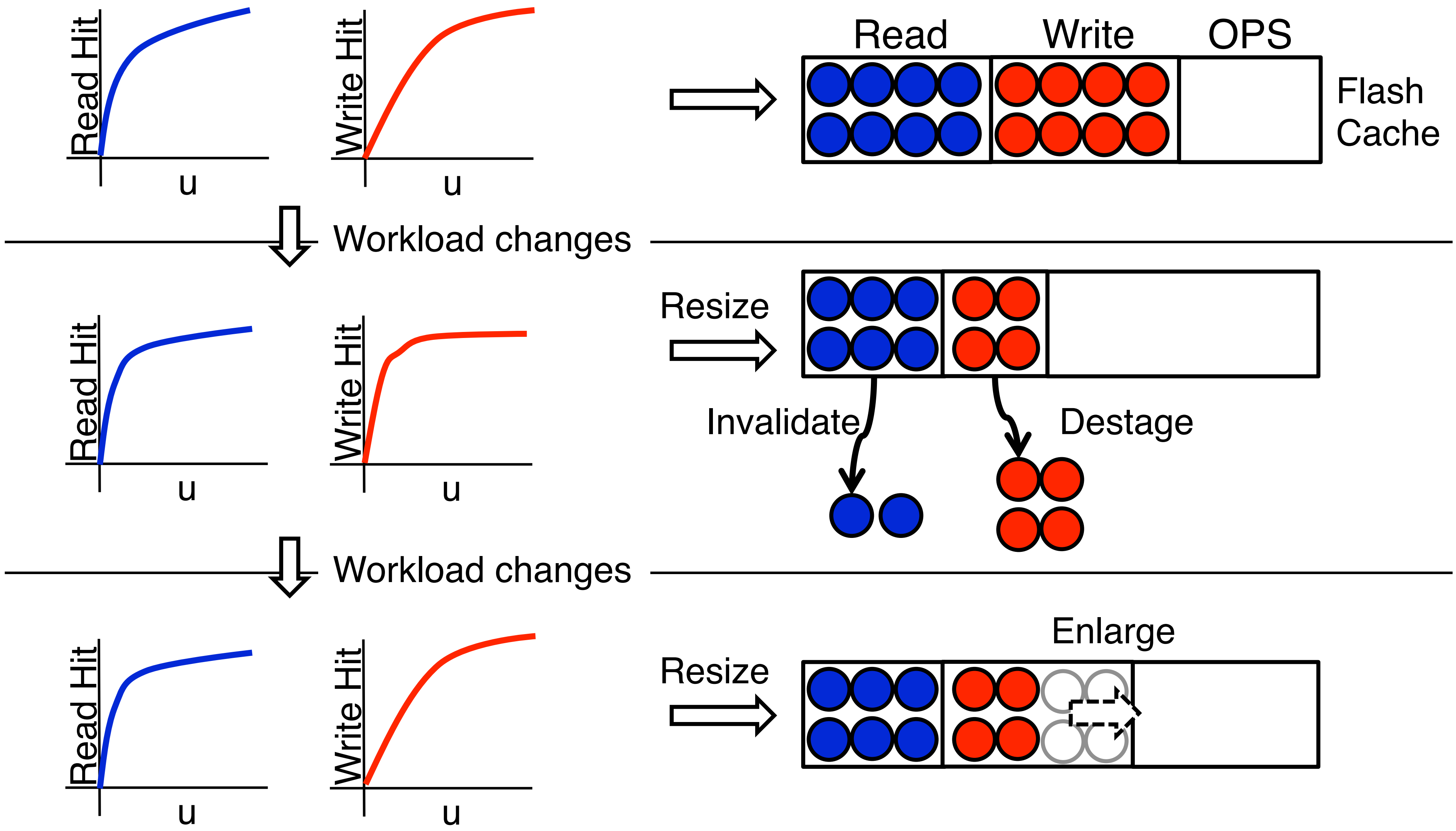
Outline

- Introduction
- Hybrid Cost Model
- Implementation
- Evaluation
- Conclusion

Optimal Partitioning Flash Cache Layer (OP-FCL)



Adapt to Workload Pattern



Outline

- Introduction
- Hybrid Cost Model
- Implementation
- Evaluation
- Conclusion

Evaluation Setup

- Hybrid Storage Simulator
  - CMU DiskSim 4.0 and MSR SSD extension
- Flash Cache Layers (FCLs)
  - Fixed Partitioning (FP-FCL) - Fixed size OPS
    - Typical SSD product
  - Read Write (RW-FCL) - Fixed size OPS
    - Distinguishes read and write
  - *Optimal Partitioning (OP-FCL)* – *Dynamically adjusted based on workload*
- Configurations
  - Config. 1: 4GB flash cache + 10K RPM HDD
  - Config. 2: 16GB flash cache + three 10K RPM HDDs

Workload Traces

Financial [UMass] with Config. 1

- Random write dominant
- OLTP application running at a financial institutions

Search Engine [UMass] with Config. 1

- Random read dominant
- Web search engine

Exchange [SNIA] with Config. 2

- Random read/write mixed
- Microsoft employee e-mail server

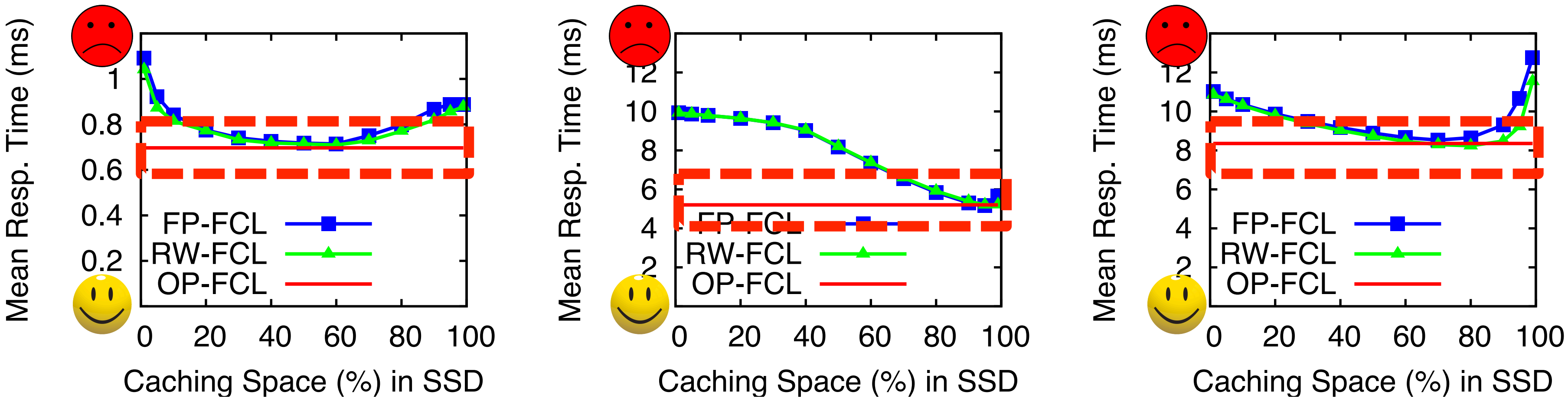
Home [FIU] with Config. 1

- Development, testing, and plotting in NFS Server

MSN [SNIA] with Config. 2

- MSN storage back-end file store

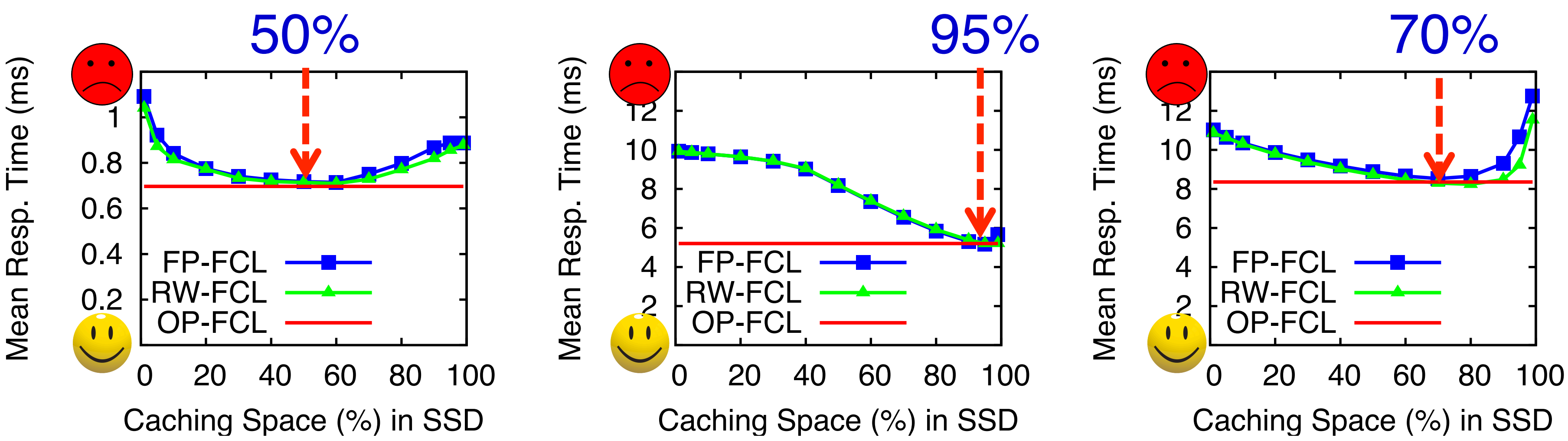
Response Time Results



(a) Financial (b) Search Engine (c) Exchange

- *OP-FCL shows near-optimal performance*
- *Optimal performance depends on workload characteristics*

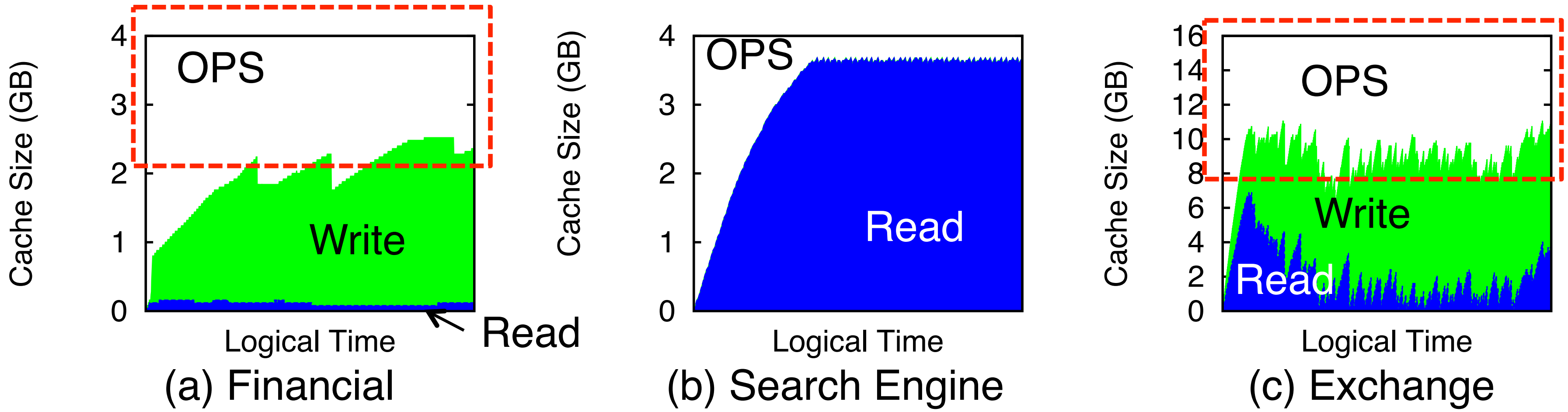
Response Time Results



(a) Financial (b) Search Engine (c) Exchange

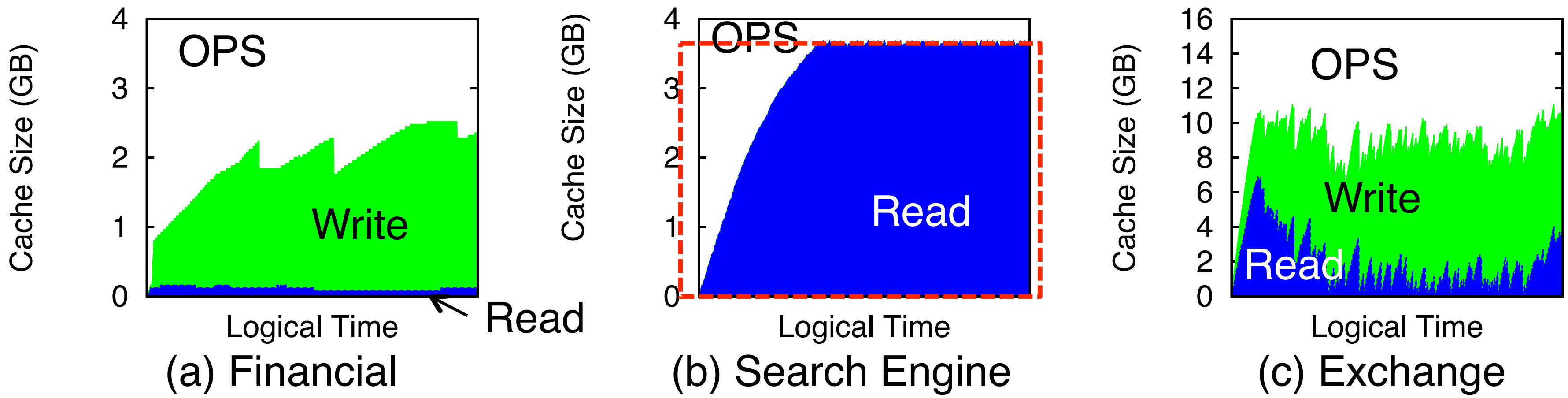
- *OP-FCL shows near-optimal performance*
- *Optimal performance depends on workload characteristics*

Dynamic Adjustment



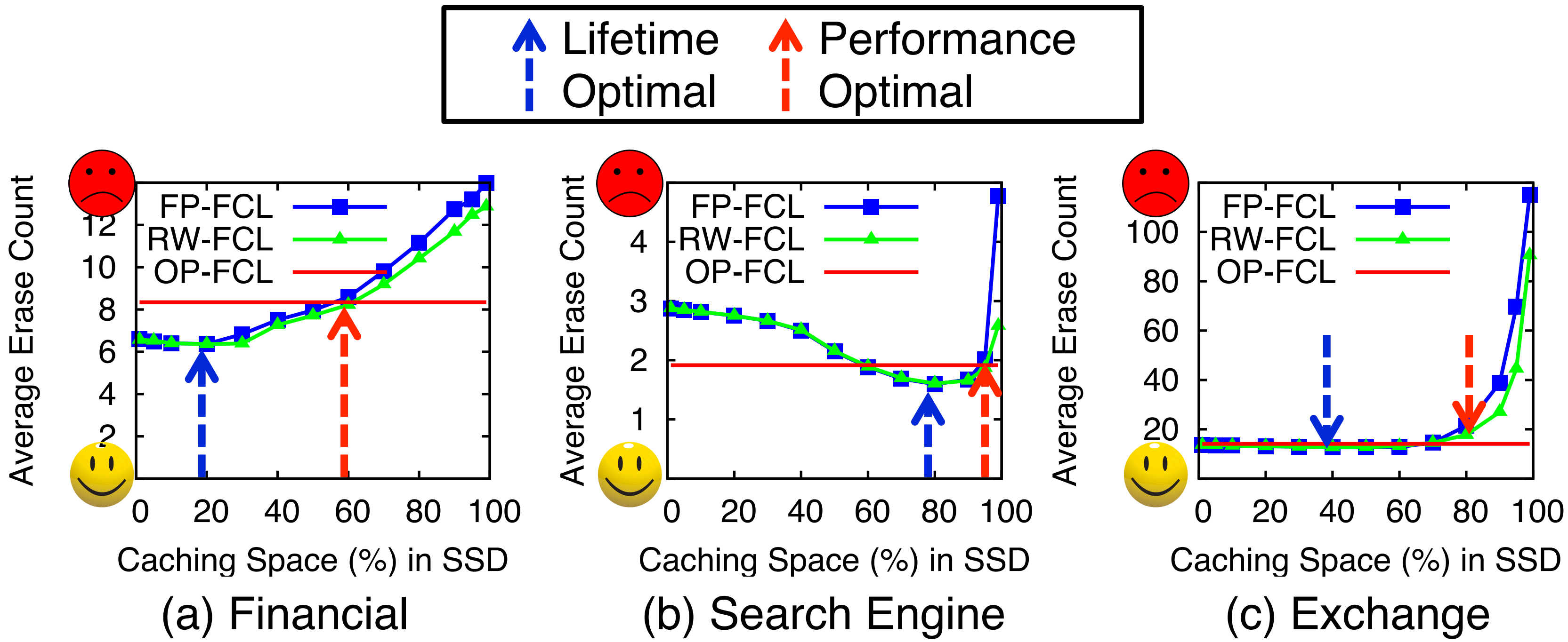
- OP-FCL dynamically adjusts cache spaces according to workloads
- Financial and Exchange
  - **Considerable OPS is used to lower garbage collection cost**
- Search Engine
  - *Most caching space* is used to maintain read data

Dynamic Adjustment



- OP-FCL dynamically adjusts cache spaces according to workloads
- Financial and Exchange
  - *Considerable OPS* is used to lower garbage collection cost
- Search Engine
  - **Most caching space is used to maintain read data**

Effect on Lifetime of Flash Cache



- *Lifetime of flash cache is an important issue*
- *Optimal point of lifetime differs from that of performance*
- Our focus is to improve the performance of flash cache
- Optimizing lifetime of flash cache left as future work

Conclusion

- Trade-off exists
  - Caching benefit vs update cost
- We proposed OP-FCL for Hybrid Storage Systems
  - Use workload dependent cost model
  - Adjust read, write, and OPS sizes based on proposed cost model
  - Show near-optimal performance compared to others
- Future direction
  - Develop better destaging and replacement algorithm
  - Make SSD lifetime aware hybrid storage system

Thank You!

Caching less for better performance: Balancing cache size and update cost of flash memory cache in hybrid storage systems

Yongseok Oh Jongmoo Choi Donghee Lee Sam H. Noh

University of Seoul
{ysoh,dhl_express}@uos.ac.kr

Dankook University
choijm@dankook.ac.kr

Hongik University
samhnoh@hongik.ac.kr