



TEXAS TECH UNIVERSITY™



OAK RIDGE
National Laboratory



A Low-cost Adaptive Data Separation Method for the Flash Translation Layer of Solid State Drives

Wei Xie , Yong Chen and Philip C. Roth

Texas Tech University and Oak Ridge National Laboratory

Presenter: Wei Xie



Data-Intensive Computing Storage Demand



- I/O performance and power-consumption are critical problems of data-intensive applications
- Massive I/O demand in HPC scientific application: emerging exa-scale computing system (10^{18} ops/s)
- Power is a big constraint in data centers and HPC system
 - the 10MW consumption of present day HPC systems is certainly becoming a bottleneck



Flash-based Solid State Drives



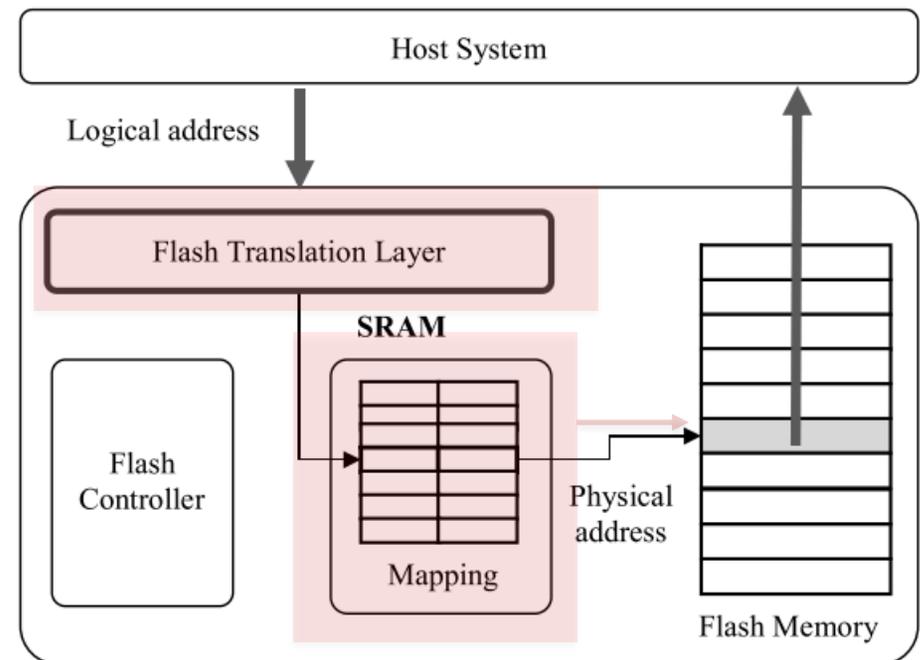
- Flash-based SSDs offer 10 -100x more performance than traditional hard disk drives
- Also 3-30x more power efficient
- Now replacing HDDs in many applications for mainly for its better performance
- Price falling down quickly
- Performance degradation problem



Solid State Drive and its Internals



- SSD Internals
 - Erase-before-write problem
 - Log-based write
 - Flash Translation Layer
 - Garbage collection



Qingsong, W., Bozhao, G., Pathak, S., Veeravalli, B., Lingfang, Z., & Okada, K. (2011). WAFTL: A workload adaptive flash translation layer with data partition. 2011 IEEE 27th Symposium on Mass Storage Systems and Technologies (MSST), 1–12. doi:10.1109/MSST.2011.5937217

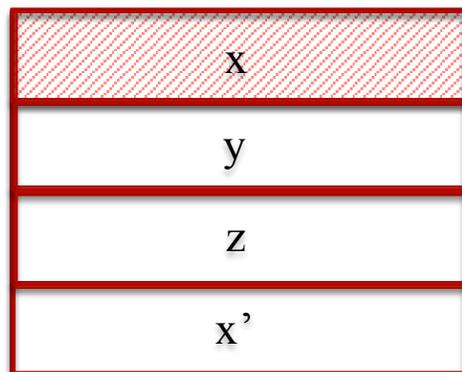


Garbage Collection

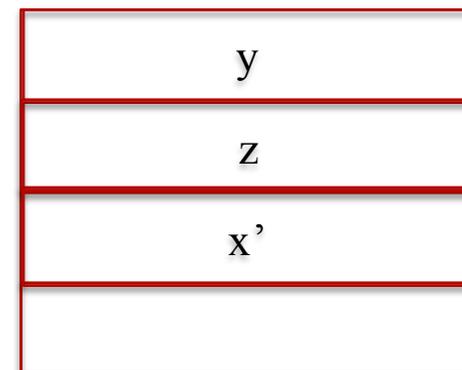


- Log-based write requires garbage collection
- Garbage collection involves overhead:
 - Copy valid data
 - Erase data block
- GC degrades SSDs' performance

Block 1000 (free block)



Block 2000 (free block)





Reduce Garbage Collection Overhead



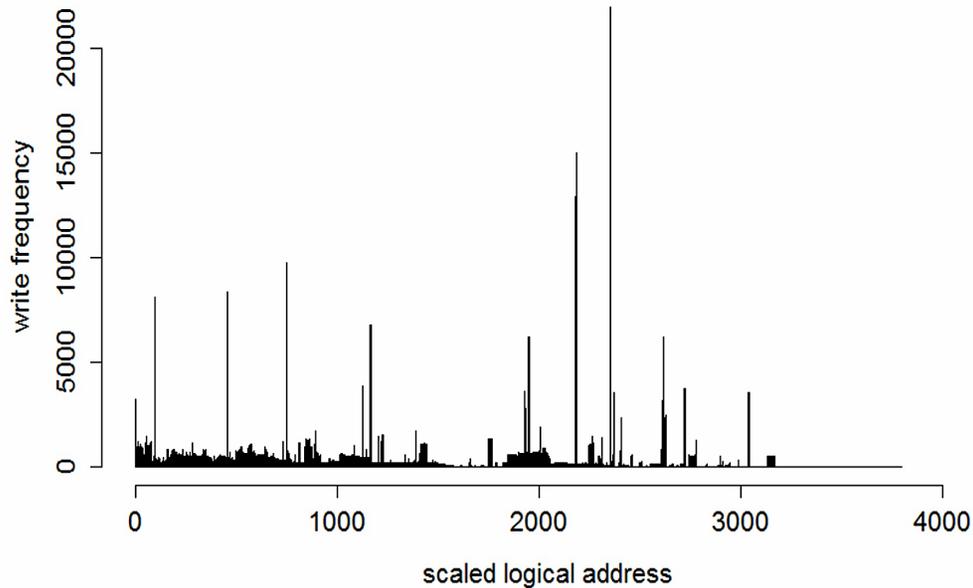
- To reduce GC overhead, reduce *valid data* in the selected *victim blocks*
 - Selection algorithm
 - Separate hot/cold data into different data blocks



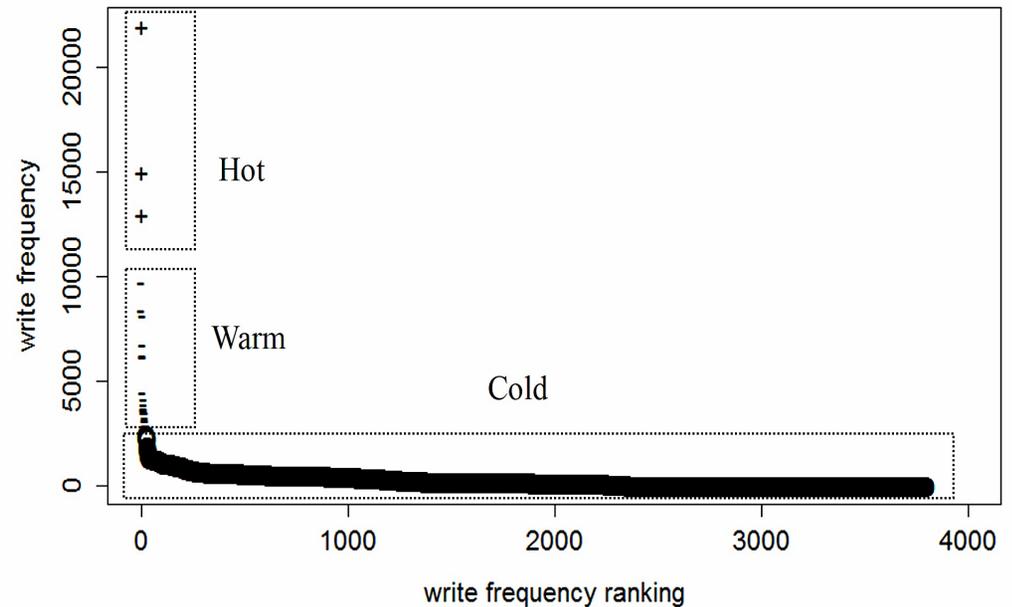
Motivation: I/O Workload Locality



Write frequency distribution on logical address



Write frequency ranking



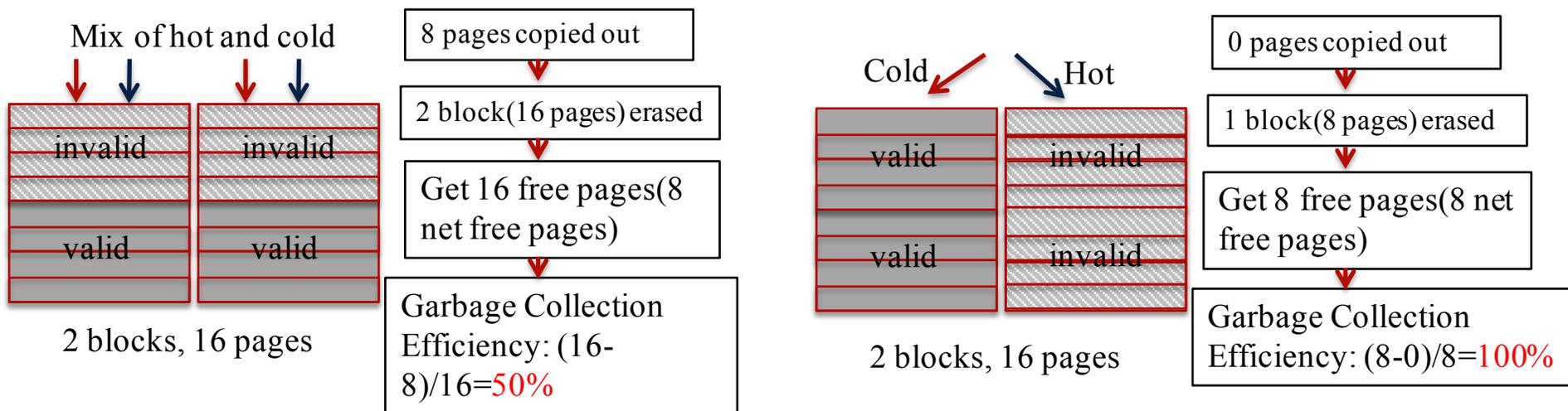
- Financial1 trace: bimodal distribution of hot and cold data



Motivation: Hot/Cold Separation Helps Garbage Collection



- GC overhead can be reduced when hot/cold data are separated





Motivation: Demand based page-mapping FTL



- Problem: DFTL does not consider hot/cold data separation, which is crucial for reducing Garbage Collection overhead
- Two questions:
 - How to detect hot/cold data
 - How to set the hot/cold criteria



Design: Hot/cold Separation



- How to detect hot/cold?
 - Inter-Reference-Recency: the time interval between last two writes
- How to set hot/cold criteria?
 - Fixed criteria
 - Criteria based on clustering algorithm: more accurate
- Issues:
 - *Memorizing and updating IRR information at page-level very costly*
 - *Calculating the hot/cold criteria costly*



Design: Selective-Cache *IRR* and *IRR* Updating



- *IRR* information is stored in flash and selectively cached
- *Age* is the time since last write, like a clock
- *Age* keeps tracking when a page is in cache
- A page evicted from cache: we use *Time_Evicted*
Time_Evicted: time that a data is lastly evicted from cache
- $IRR = (Current_Time - Time_Evicted) + Age$



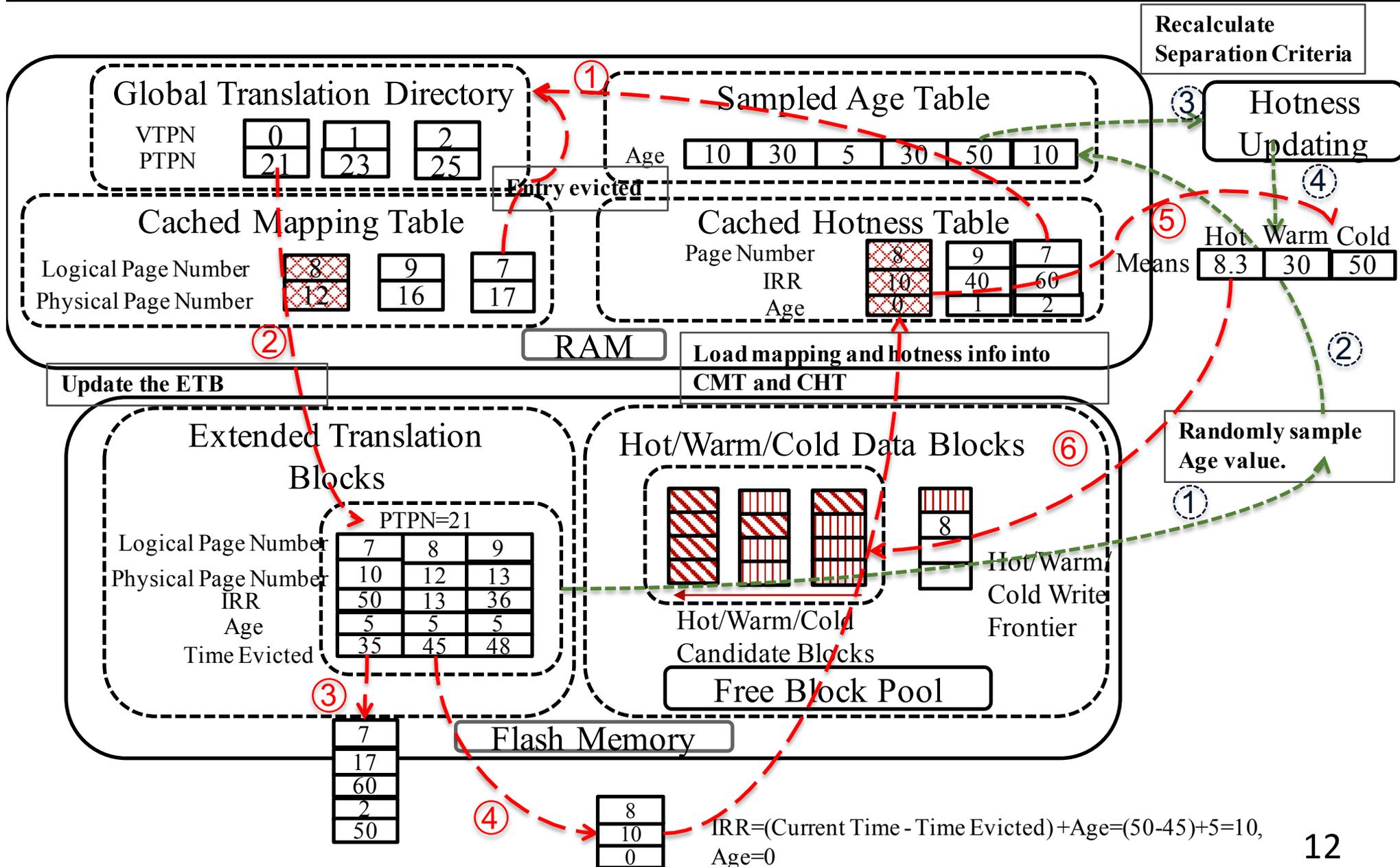
Design: Random Sampling and Clustering



- How to set hot/cold criteria?
 - Clustering based on *IRR* values
 - Too many *IRR* values to consider
- Random sampling: randomly select a small subset (100) of *IRR* values
- Clustering: k-means clustering on the sampled *IRR* values to generate the centroids for hot/warm/cold categories

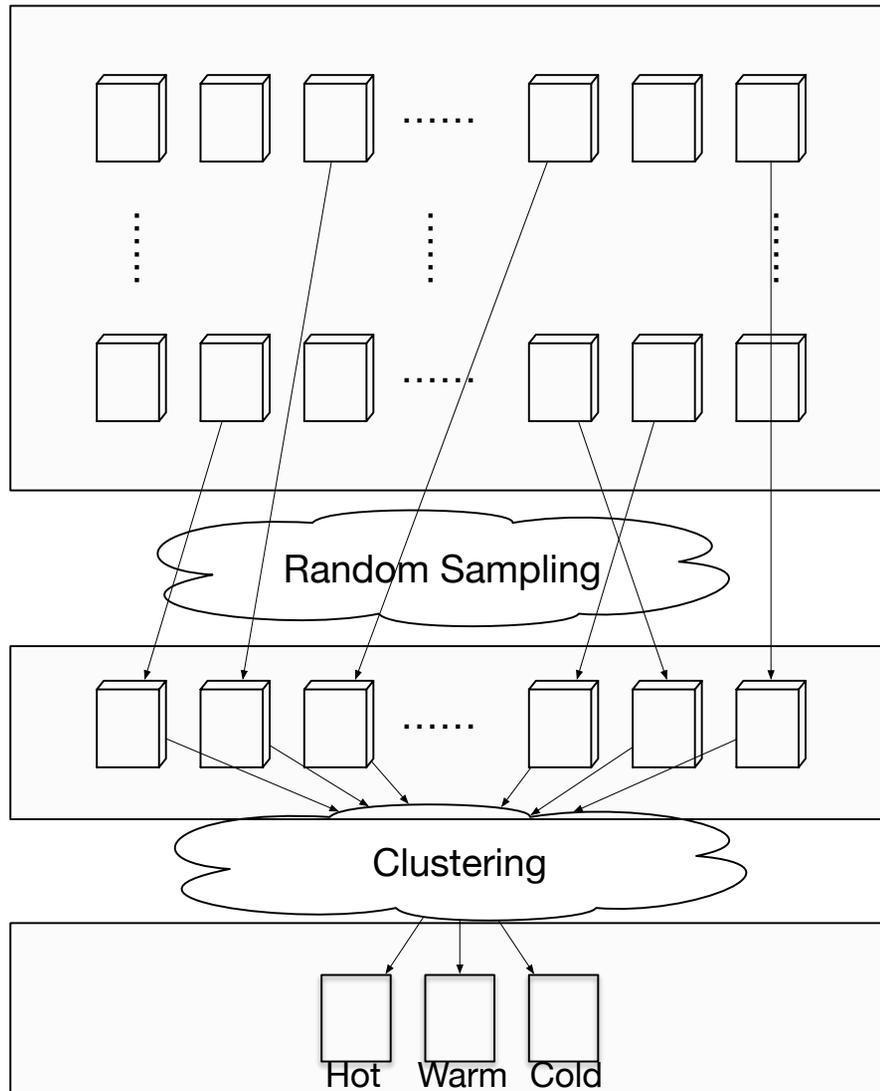


Design: Working Principle





Design: Random Sampling and Clustering



IRR value of a data page



Calculated separation criteria

A 128GB SSD contains 33.5 millions of pages

Randomly select a small subset of IRR values

3 numbers used a separation criteria



Implementation



- Implemented in FlashSim.
- Based on the DFTL algorithm implemented in FlashSim
- Random Sampling and Clustering is set to run every 5000 write requests
- Hot/warm/cold criteria are used to separate data on write



Workload Traces



I/O traces	Total Write Number (GB)	Total Write Number (Count)	Average Request Size (KB)
Financial1	9	2097152	4.5
Cambridge	46	1000000	48.8
TPCC1	4.6	484164	10.1

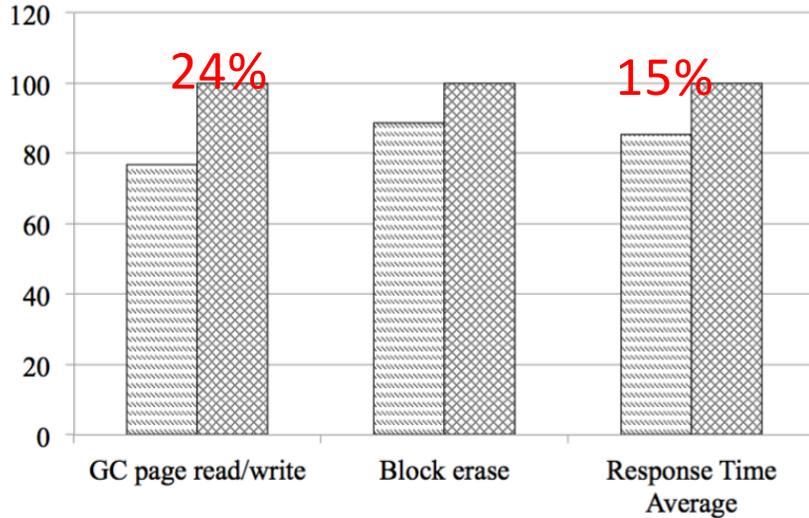
- Financial1: random, bimodal (hot and cold) distribution
- Cambridge: sequential, bimodal
- TPCC1: relative uniform access distribution



Evaluation Result

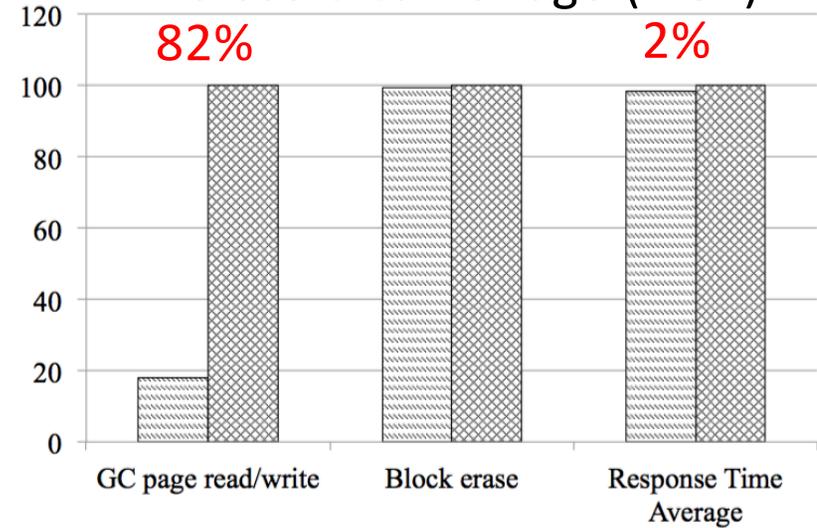


Financial1



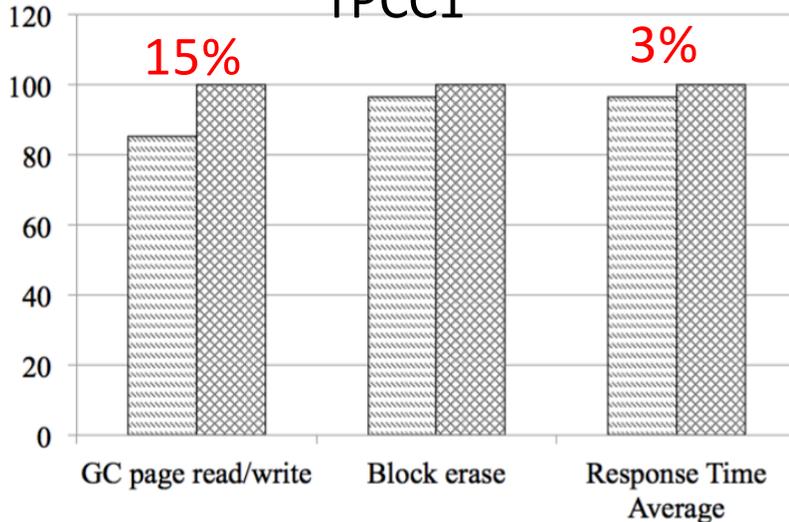
ASA-FTL DFTL

Microsoft Cambridge (MSR)



ASA-FTL DFTL

TPCC1

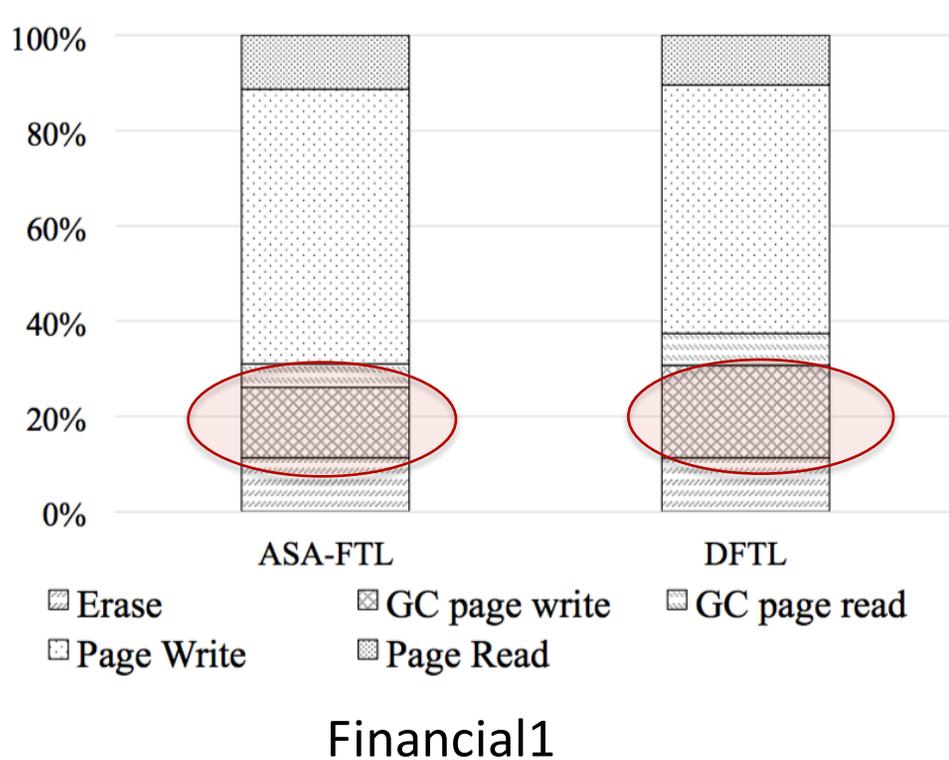


ASA-FTL DFTL

- Why 82% GC write reduction only translate to 2% less response time?



Evaluation Result (Proportion of Response Time)



- GC overhead only small portion in MSR, because of large request size
- Improvement in GC time does not translate to much better performance

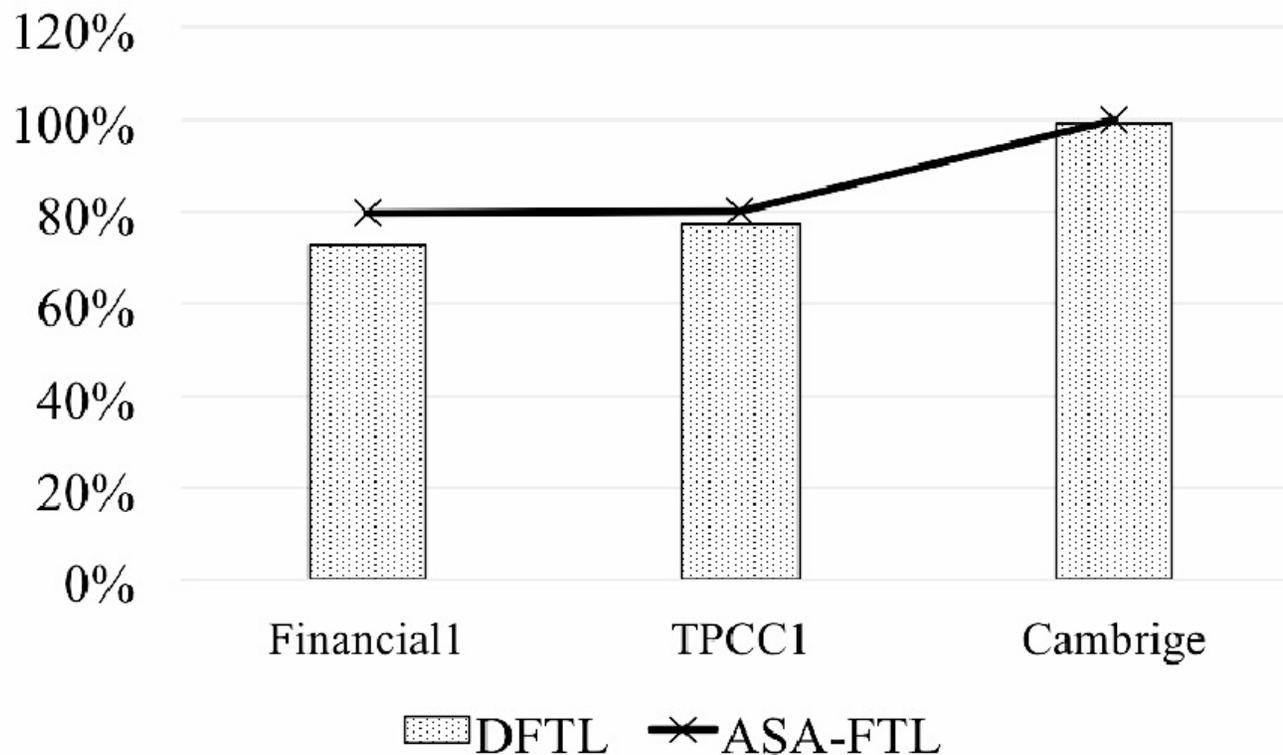


Evaluation of GC Efficiency



$$GC\ Efficiency = \frac{num\ pages\ created - num\ pages\ copied}{num\ pages\ created}$$

GC Efficiency Result: ASA-FTL vs. DFTL





Evaluation Result



- The improvement on GC overhead depends on the locality of the workload
- For the first 10,000 write requests:
 - Financial1 has around 1200 unique addresses accessed
 - MSR has around 280 unique addresses accessed
 - TPCC1 has around 4700 unique addresses accessed
- GC overhead largely depends on randomness of workload



Conclusions



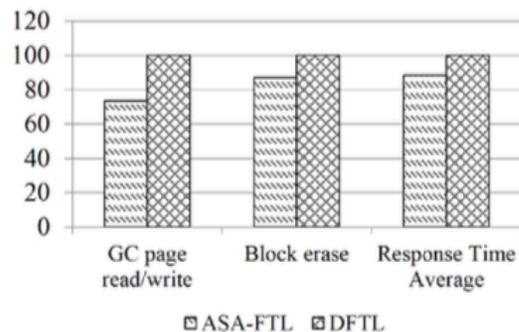
- Flash-based SSD is promising and being adopted in data-intensive applications
- The garbage collection overhead problem and hot/cold data access pattern motivate us the ASA-FTL scheme
- Selective caching and random sampling are used to achieve low-overhead data separation
- Implemented and evaluated using FlashSim
- Evaluation result shows performance improvement and its dependency on workload locality pattern



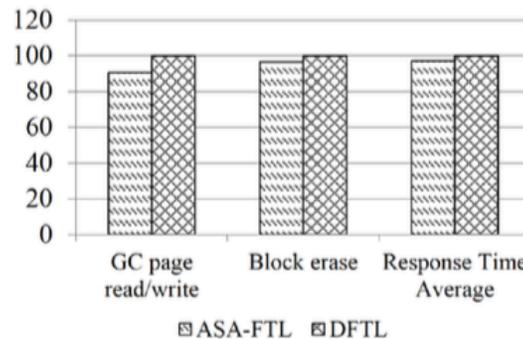
Future Work



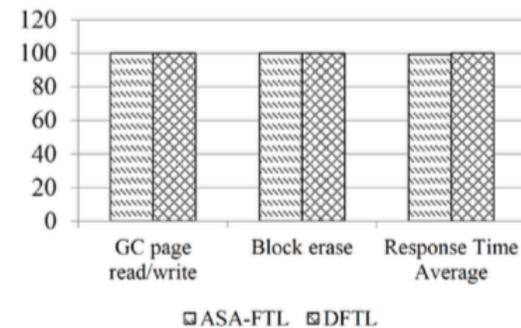
- More workload, including synthetic benchmark to evaluate the correlation between GC improvement and workload characteristics (IO size and locality)
- Extend the separation method to be applicable to other FTL schemes instead of just DFTL
- Build prototype SSD with the ASA-FTL to evaluate its performance in real devices



(a) Synth9/10



(b) Synth7/10



(c) Synth5/10



Acknowledgement



This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research.

This research is also supported by the National Science Foundation under grant CNS-1162488 and CNS-1338078.

Thanks! Questions?

DISCL@TTU: <http://discl.cs.ttu.edu>

FT@ORNL: <https://ft.ornl.gov>



U.S. DEPARTMENT OF
ENERGY