Impact of Data Locality on Garbage Collection in SSDs: A General Analytical Study

Yongkun Li, **Patrick P. C. Lee**, John C. S. Lui, Yinlong Xu

The Chinese University of Hong Kong

University of Science and Technology of China

SSD Storage

- Solid-state drives (SSDs) widely deployed
  - e.g., desktops, data centers
- Pros:
  - High throughput
  - Low power
  - High resistance
- Cons:
  - Limited lifespan
  - Garbage collection (GC) overhead

Motivation

- Characterizing GC performance is important for understanding SSD deployment

- We consider **mathematical modeling**:
  - Easy to parameterize
  - Faster to get results than empirical measurements

Challenges

- **Data locality**
  - Data access frequencies are non-uniform
  - Hot data and cold data co-exist
  - More general access patterns are possible (e.g., warm data [Muralidhar, OSDI'14])

- Wide range of GC implementations

Two Questions

- What is the impact of data locality on GC performance?

- How data locality can be leveraged to improve GC performance?

Our Contributions

A general analytical framework that characterizes locality-oblivious GC and locality-aware GC

- A non-uniform workload model
- A probabilistic model for a general family of locality-oblivious GC algorithms
- A model for locality-aware GC with data grouping
- Validation and trace-driven simulations

Related Work on GC

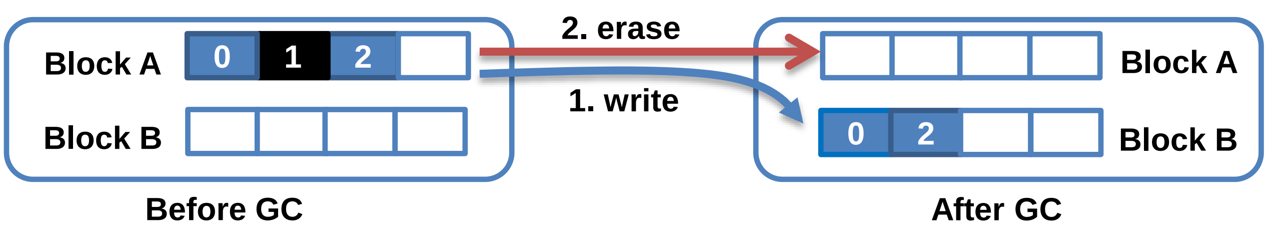
- Theoretical analysis on GC
  - Hu et al. (SYSTOR09), Bux et al (Performance10), Desnoyers (SYSTOR12): model greedy algorithm on GC
  - Li et al. (Sigmetrics13): model design tradeoff of GC between performance and endurance
  - Benny Van Houdt (Sigmetrics13, Performance13): model write amplification of various GC algorithms under uniform workload and hot/cold workload
  - Yang et al. (MSST14): analyzing the performance of various hotness-aware GC algorithms
- **Our work focuses on the impact of data locality on GC performance under general workload**

How SSDs Work?

- Organized into blocks
- Each block has a fixed number (e.g., 64 or 128) of fixed-size (e.g., 4-8KB) pages
- Three basic operations: read, write, erase
  - Read, write: per-page basis
  - Erase: per-block basis
- Out-of-place write for updates:
  - Write to a clean page and mark it as valid
  - Mark original page as invalid

How SSDs Work?

- **Garbage collection (GC)** reclaim clean pages
  - Choose a block to erase
  - Move valid pages to another clean block
  - Erase the block



- Limitations:
  - Blocks can only be erased a finite number of times
    - SLC: 100K, MLC: 10K, 3 bits MLC (several K to several hundred)
  - GC introduces additional writes (cleaning cost)
    - Degrades both performance and endurance

Workload Model

- Clustering
  - Only a small proportion of pages are accessed
  - Let f_a be proportion of logical pages that are active
- Skewness
  - Access frequency of each page varies significantly
  - n access types
  - Two vectors: $\boldsymbol{r} = (r_1, r_2, \dots, r_n), \boldsymbol{f} = (f_1, f_2, \dots, f_n)$
    - type-i pages account for a proportion f_i of active pages and are uniformly accessed by a proportion r_i of requests
- Both clustering and skewness are observed in real-world traces

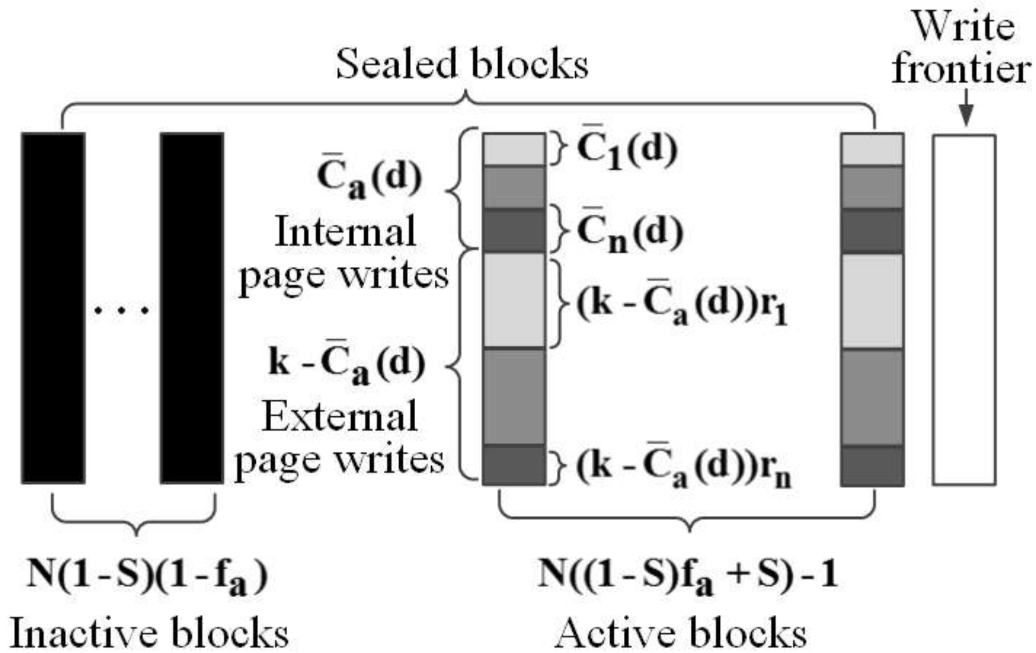
GC Algorithms

- Greedy Random Algorithm (GRA)
  - Defined by a window size parameter d
  - Two steps to select a block for GC
    - First select d blocks with the fewest valid pages (greedy)
    - Then uniformly select a block from the d blocks (random)
- Special cases
  - $d = 1$: GREEDY algorithm
  - $d = \mathrm{N}$: RANDOM algorithm

Locality-oblivious GC

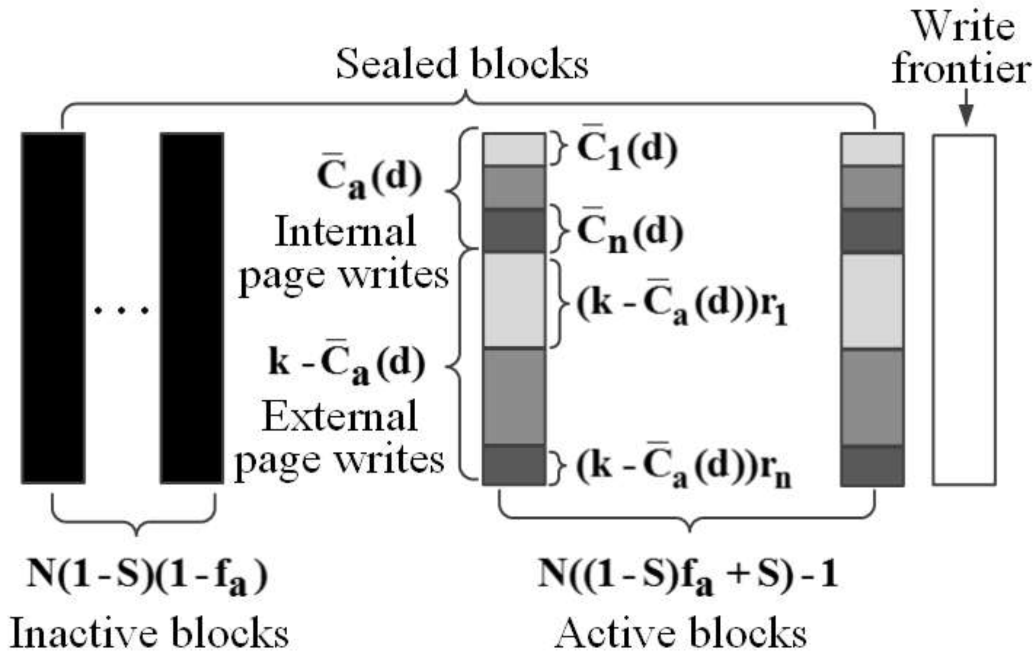
- Write and GC process with single write frontier
  - One block is allocated as the write frontier at any time
- Writes are sequentially directed to write frontier
  - Internal writes: due to GC
  - External writes: due to workload
- Write frontier is sealed until all clean pages in the block are used up
- Another clean block is allocated as write frontier
  - GC is triggered to reclaim a block

State of Blocks



- k: total number of pages in a block
- $\bar{C}_i(d)$: average number of type-i valid pages in the block chosen for GC
- $\bar{C}_a(d)$: Internal page writes (page writes due to GC)
  - Sum of $\bar{C}_i(d)$

State of Blocks



- Approximation: d candidate blocks are chosen from the d blocks sealed in the earliest time
  - Earlier sealed blocks have fewer valid pages on average

General Analysis Framework

- Average cleaning cost in each GC is

$$\bar{C}(d) = \begin{cases} \bar{C}_a(d) & \text{if } d \le N_a, \\ \frac{N_a}{d}\bar{C}_a(d) + (1 - \frac{N_a}{d})k & \text{otherwise.} \end{cases}$$

  - N_a is number of active blocks and $\bar{C}_a(d)$ can be computed via
  - $\begin{cases} \bar{C}_a(d) = \sum_{i=1}^{n} \bar{C}_i(d) \\ \bar{C}_i(d) = \frac{r_i(\mathrm{k} - \bar{C}_a(d))(1-\mathrm{P'})^{N_a - d}}{1 + P' \times d - (1-\mathrm{P'})^{N_a - d}} \end{cases}$ where $P' = \frac{r_i(\mathrm{k} - \bar{\mathrm{C}}_a(\mathrm{d}))}{(N_a + 1)(1 - S')kf_a}$

- $\bar{C}(d)$ is a function of d, f_a, r_i and f_i
  - GC cleaning cost is affected by GC algorithms and workload locality (both clustering and skewness)

Case Studies

- GRA with window size $d = o(N)$
  - Includes the case of GREEDY (d =1)

$$\bar{C}(d) = \sum_{i=1}^{n} (k - \bar{C}(d)) r_i / (e^{A_i} - 1), \text{ where } A_i = \frac{(k - \bar{C}(d)) r_i}{(1 - S') k f_i}.$$

- GRA with window size $d \geq N_a$
  - Includes the case of RANDOM ($d = N$)

$$\bar{C}(d) = (1 - NS / d)k$$

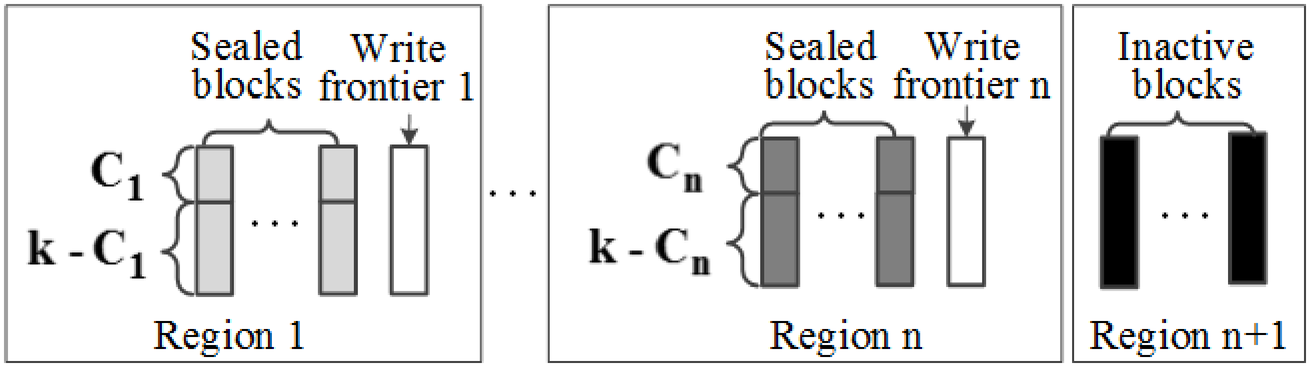
- GRA with window size $d = \alpha N_a$

$$\bar{C}(d) = \sum_{i=1}^{n} (k - \bar{C}(d)) r_i / [(1 + \alpha \mathrm{A}_i) e^{(1-\alpha) A_i} - 1], \text{ where } A_i = \frac{(k - \bar{C}(d)) r_i}{(1 - S') k f_i}.$$

Locality-aware GC

- Differentiating data reduces GC cleaning cost
- Consider locality-aware GC using data grouping
  - Differentiating different types of data pages
  - Storing them separately in separate regions
- Issues to address:
  - How data grouping influences the GC performance
  - How much is the influence for workloads with different degrees of locality

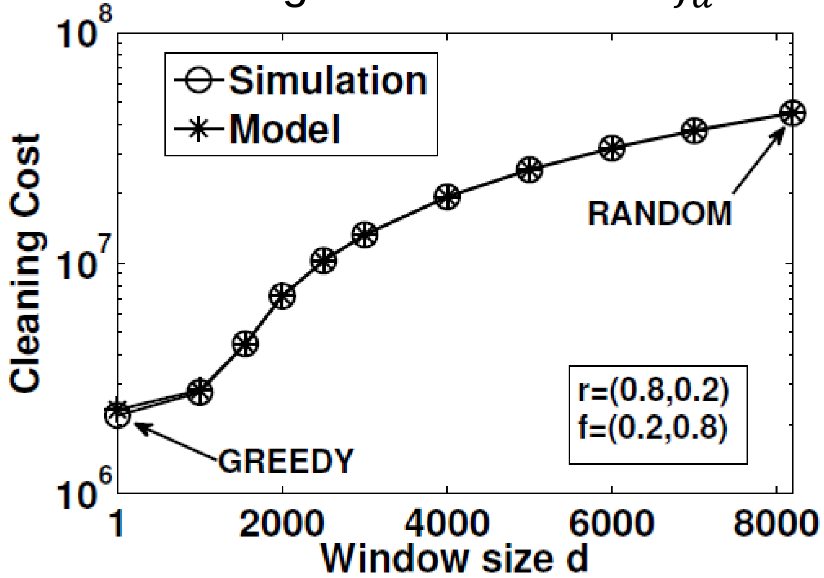
System Architecture



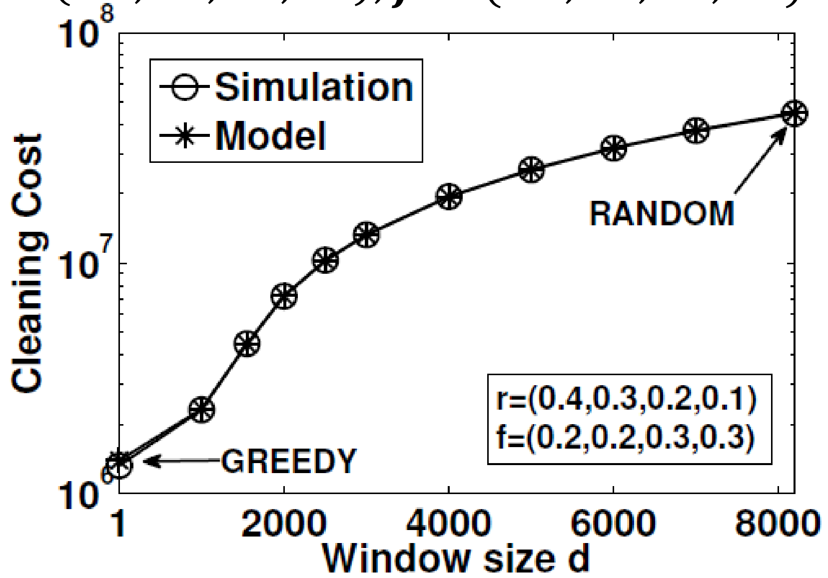
- The whole SSD is divided into $n + 1$ regions
- Each region is used to store one particular type of data
- The $n + 1$ regions can be viewed as $n + 1$ independent sub-systems
  - Each of the first n sub-systems is fed with a uniform workload
  - Previous analysis on locality-oblivious GC can be applied in each region

Model Validation

- DiskSim + SSD extension developed by Microsoft
- Workloads:
  - Skewed workload: $f_a = 0.1, n = 2, \boldsymbol{r} = (0.8,0.2),\ \boldsymbol{f} = (0.2,0.8)$
  - Fine-grained workload: $f_a = 0.1, \boldsymbol{r} = (0.4,0.3,0.2,0.1), \boldsymbol{f} = (0.2,0.2,0.3,0.3)$



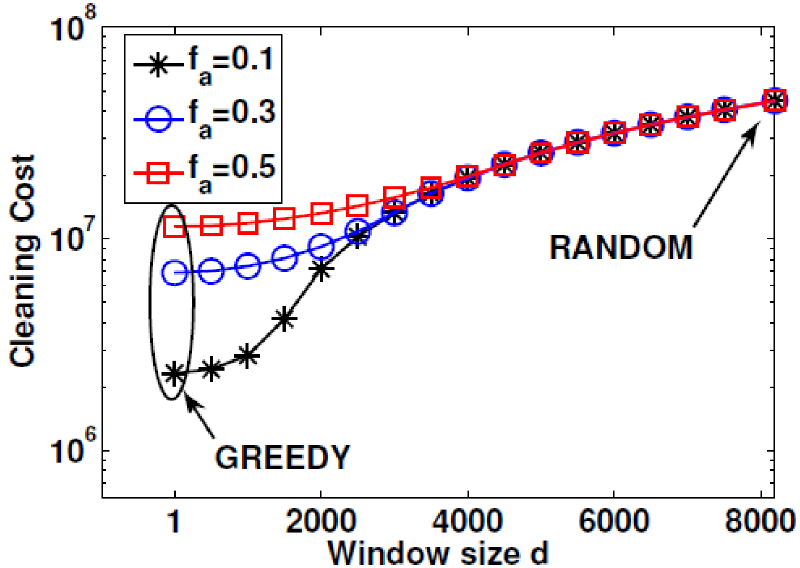
Skewed workload



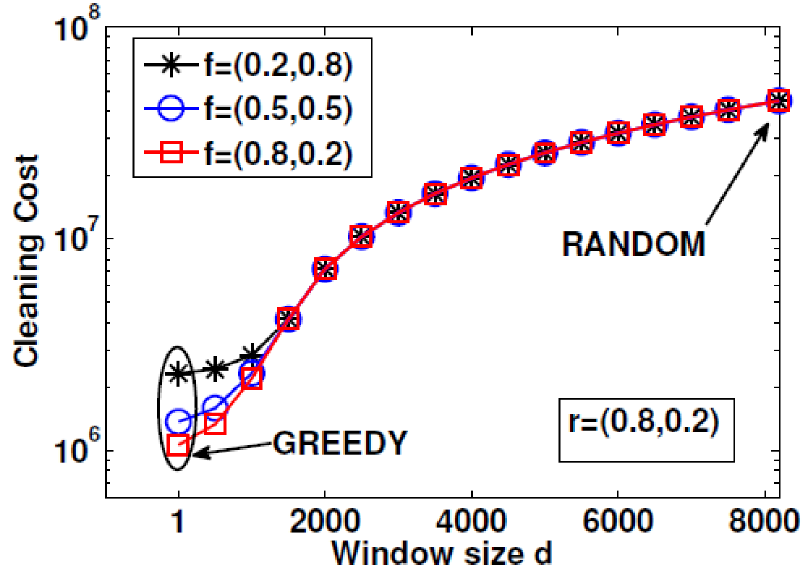
Fine-grained workload

Our model matches simulation results

Impact of Data Locality on Locality-oblivious GC



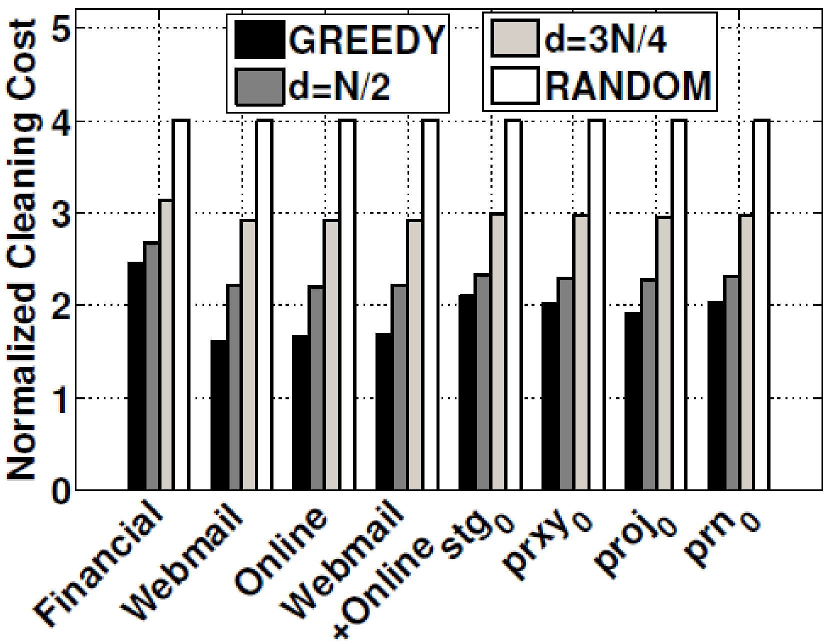
Impact of clustering



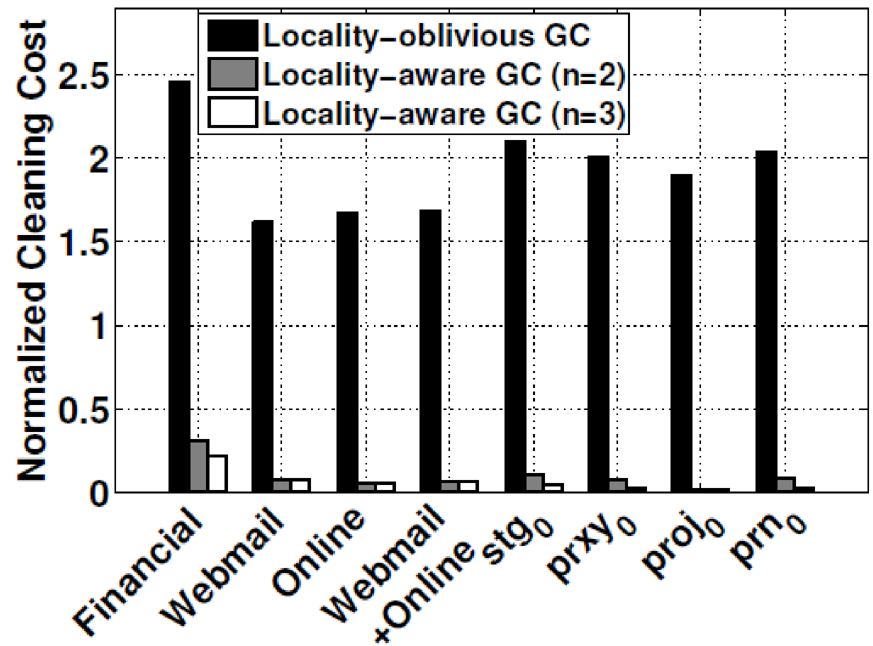
Impact of skewness

- Cleaning cost increases as either the active region size or skewness increases
- The increase is more pronounced for a smaller d
  - GREEDY algorithm shows the most increase
  - Data locality has no impact on RANDOM algorithm

Trace-driven Evaluation



Locality-oblivious GC



Locality-aware GC

- Locality-oblivious GC
  - GREEDY (RANDOM) gives the best (worst) performance
  - GREEDY has the most varying performance across workloads
- Locality-aware GC
  - Cleaning cost can be significantly reduced with data grouping
  - The further reduction is marginal when data is classified into more types

Summary

- Propose a general analytical model to study the impact of data locality on GC performance
  - Analyze various locality-oblivious GC under different workloads
  - Analyze the impact of locality-awareness with data grouping
  - Conduct DiskSim simulation and trace-driven evaluations
- Cleaning cost depends on clustering/skewness, and impact varies across algorithms
- Data grouping efficiently reduces the cleaning cost
  - Different spare block allocations show significant differences
- Future work
  - More validation beyond DiskSim simulations
  - GC implementation in SSD-aware file systems

Thank You!

- Contact:
  - Patrick P. C. Lee
    http://www.cse.cuhk.edu.hk/~pclee

Backup

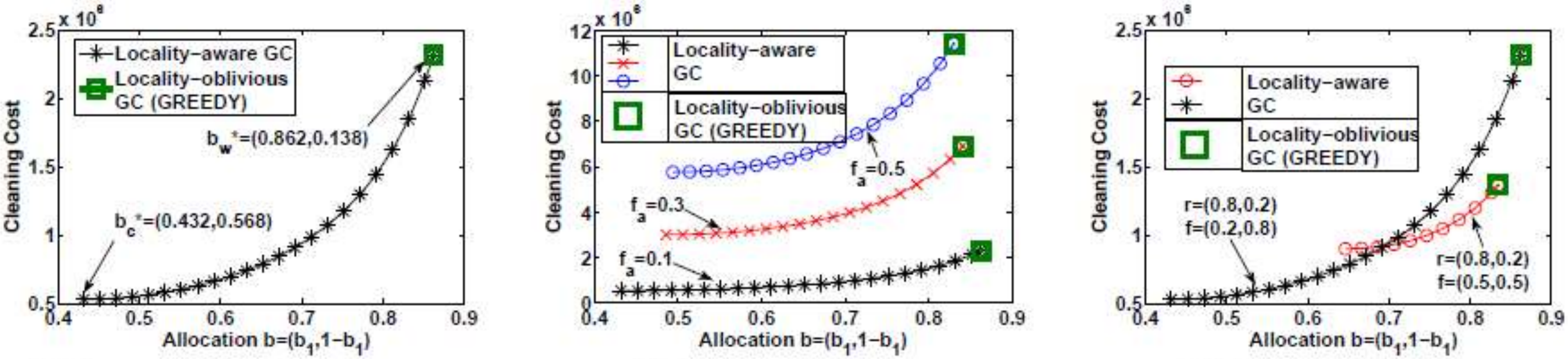
Analysis on locality-aware GC

- One design issue of locality-aware GC
  - How many spare blocks should be allocated to each region
  - Allocation $\boldsymbol{b} = (b_1, b_2, \ldots, b_n)$: proportion b_i of spare blocks are allocated to region i
- Average cleaning cost of locality-aware GC with GREEDY algorithm in region i is

$$C_i = -W_0\left(-\frac{1}{1-S_i} e^{-\frac{1}{1-S_i}}\right) / \left[\frac{1}{(1-S_i)k}\right], \text{ where } S_i = \frac{Sb_i}{(1-S) f_a f_i + Sb_i}.$$

  - Allocation of spare blocks affects the cleaning cost of locality-aware GC

Performance Gain with Locality Awareness



(a) Impact of locality awareness (b) Impact of clustering (c) Impact of skewness

- Data grouping effectively reduces GC cleaning cost
- Spare block allocation has significant impact on the performance of locality-aware GC
  - The impact decreases as the clustering increases
  - The impact increases as the skewness increases