RIPQ: Advanced Photo Caching on Flash for Facebook

COS 518: *Advanced Computer Systems*

Yu Zeng

3/11/2019

Problem Statement / Motivation

- Flash caches are used to reduce backbone traffic and backend IO for Facebook Photo.

- Advanced caching algorithms help
  - SLRU-3: 10% less backbone traffic
  - GDSF-3: 23% fewer backend IOs

- But Advanced caching algorithm cannot be implemented efficiently on Flash because many random small writes.

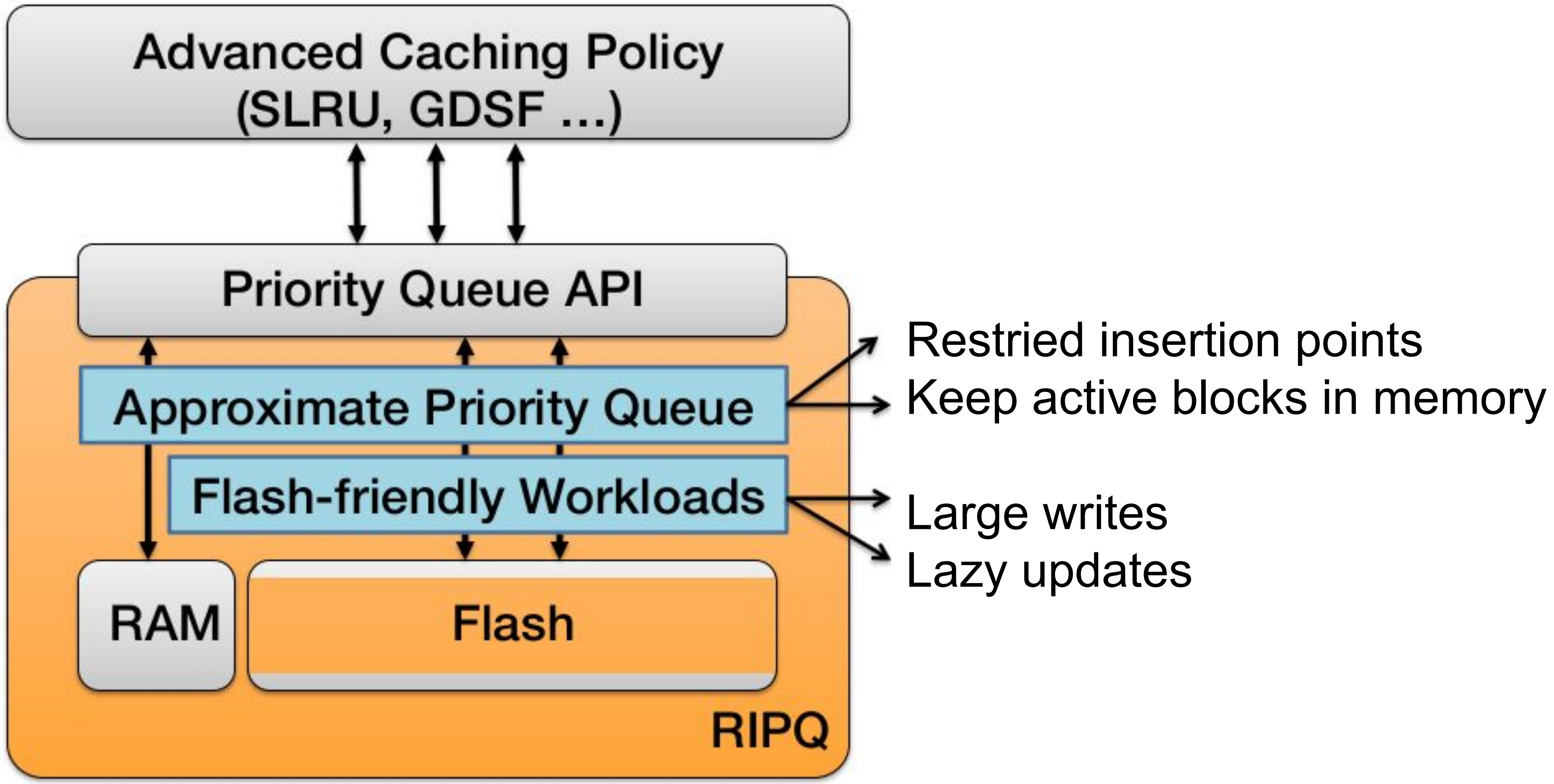
Previous Solutions

- Using simple caching algorithm
  - FIFO based

- reserve a significant portion of Flash for garbage collection
  - Decrease available cache capacity

- coarse-grained caching policies
  - Harms performance

Key Idea of RIPQ

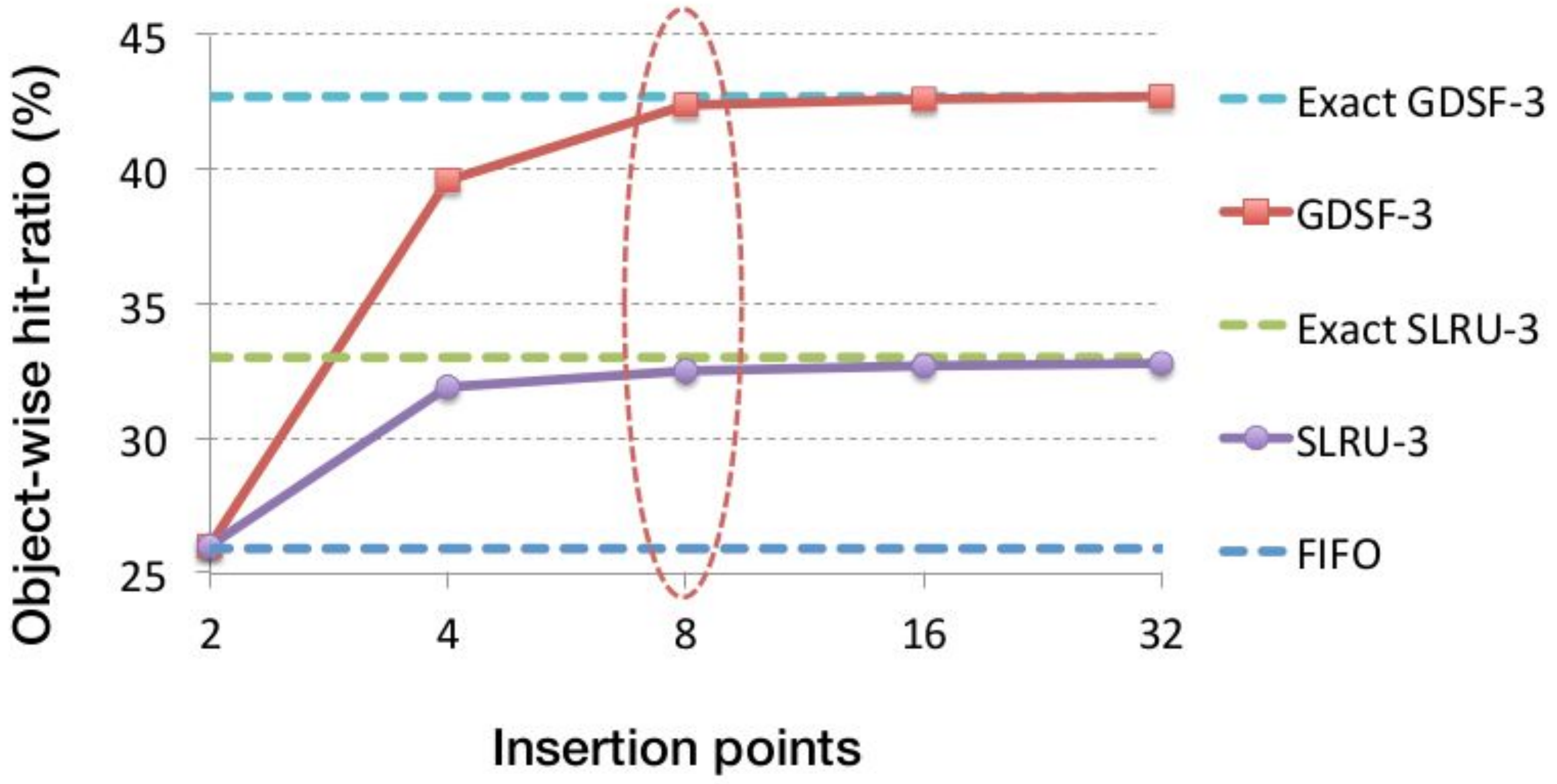


Key Challenges

- How to approximate a priority queue
  - Affect hit ratio
- Reduce the many small random writes introduced by advanced caching algorithm to improve throughput
  - Achieve balance between hit ratio and throughput

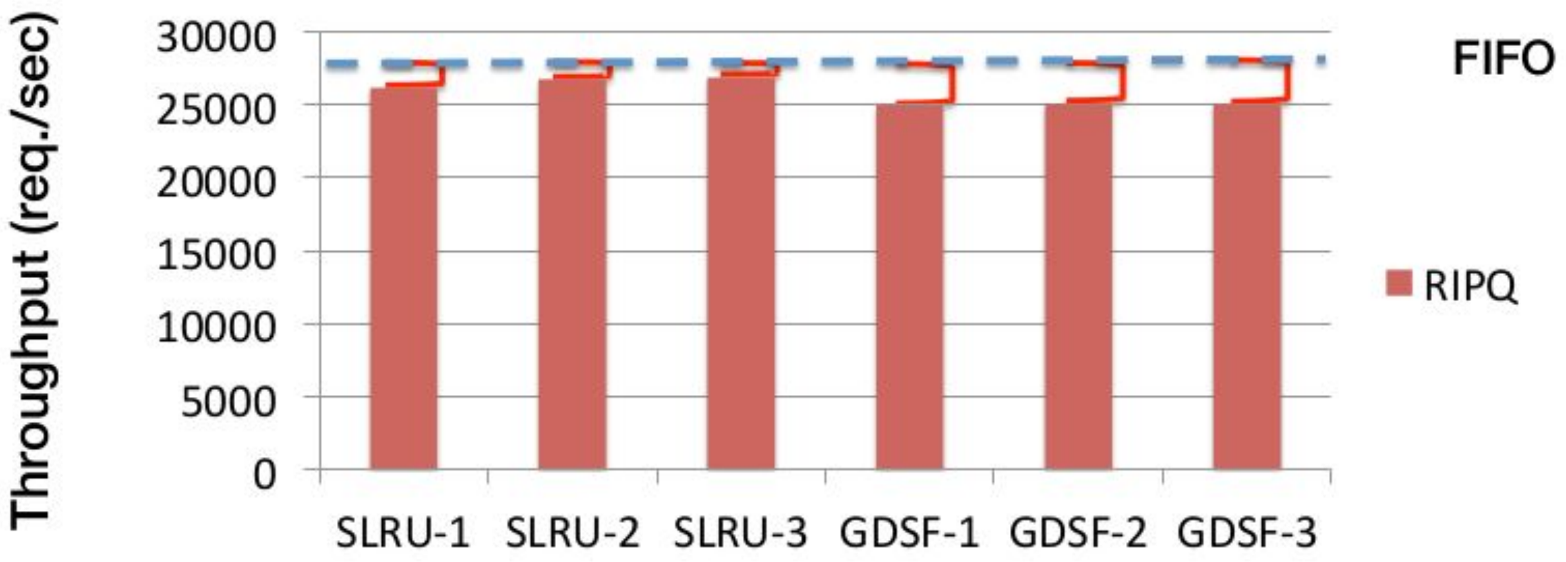
Key Results

- High hit ratio at low cost



Key Results

- Throughput near FIFO



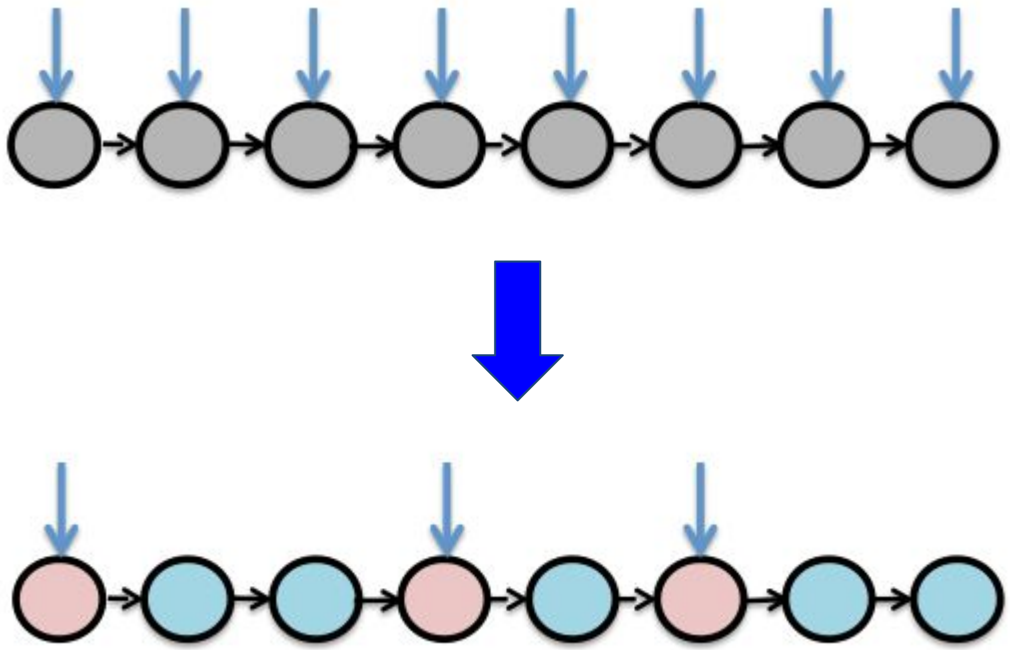
Impact

- First work that provides a flexible framework for efficiently implementing advanced caching algorithms on Flash.
- Used in Facebook photo serving stack, increasing hit ratio up to ~20% over previous system.

Technical Details

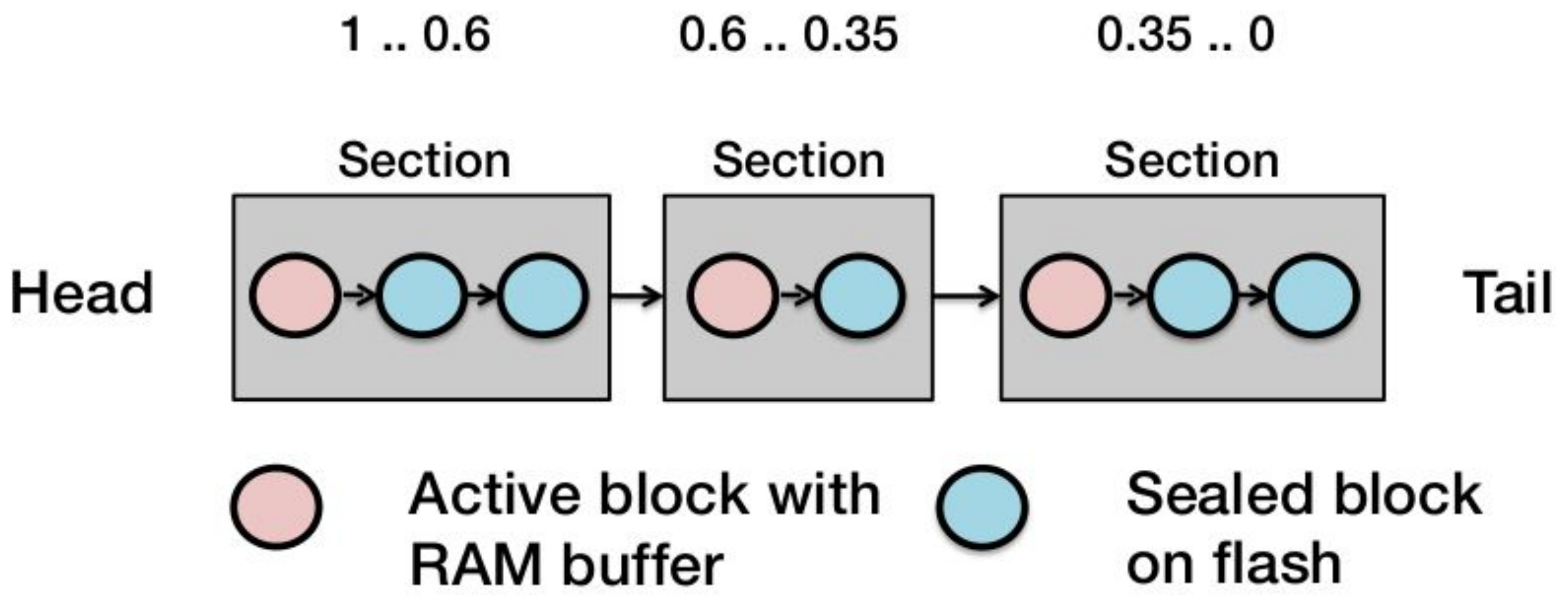
Key Idea: Restricted Insertion Points

- Do not want to write anywhere in the queue
    - Random small writes
- Restricted insertion points in the queue!



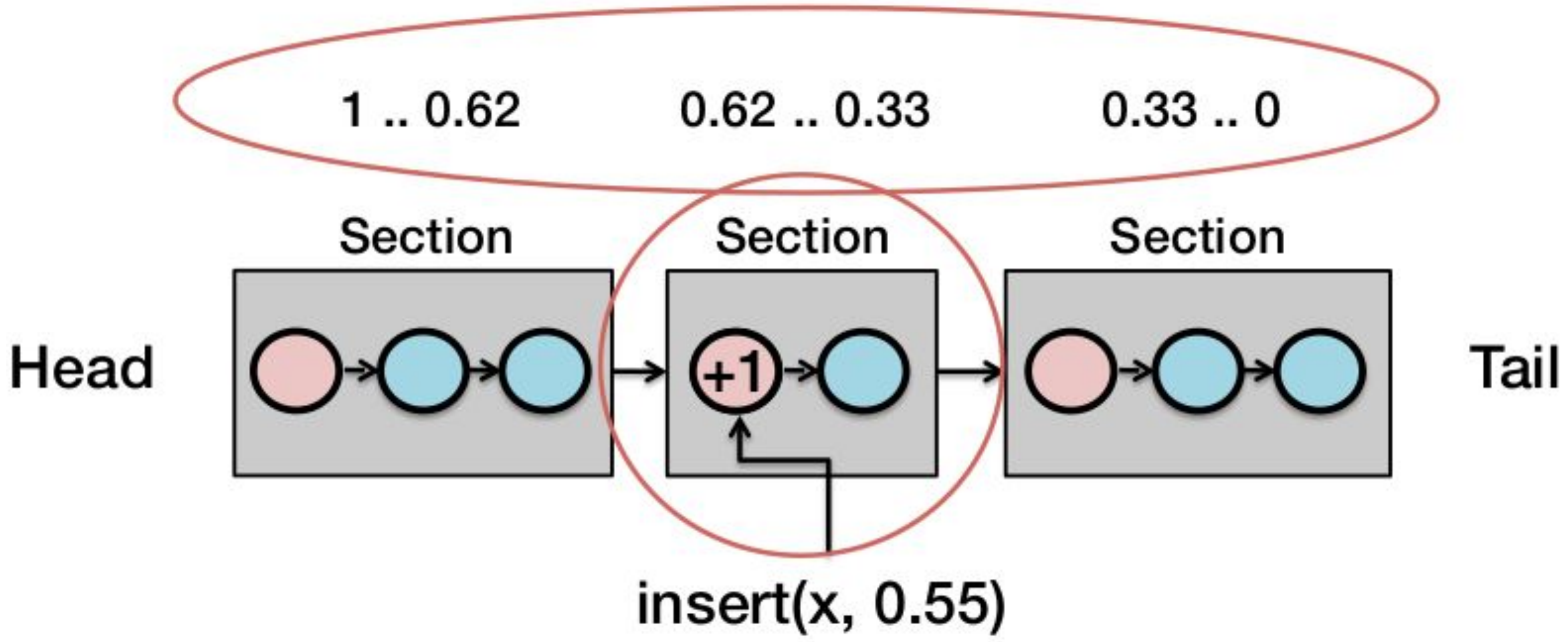
Key Idea: Restricted Insertion Points

- Divide the priority queue into sections, each one has a priority range
- Each section has only one active block



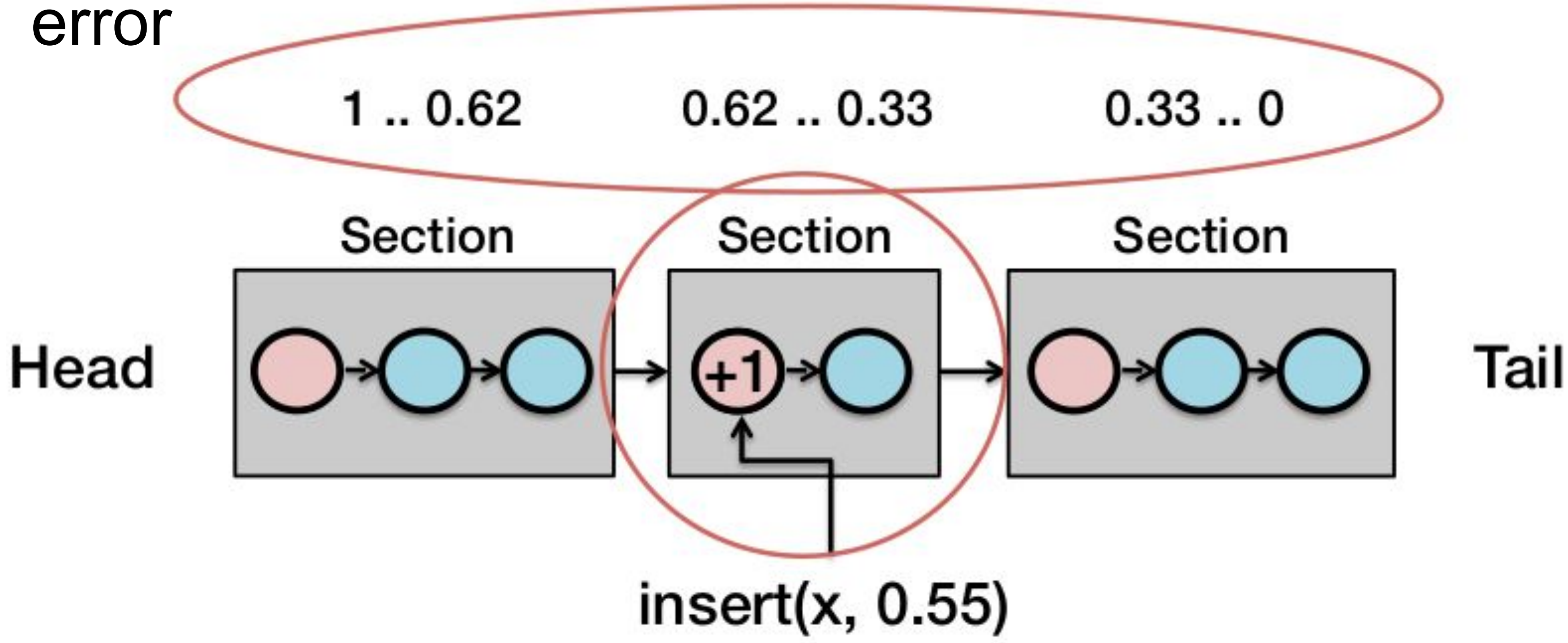
Key Idea: Restricted Insertion Points

- Insert new value into the active block of corresponding section
- Adjust priority ranges(stored in memory) of all sections.



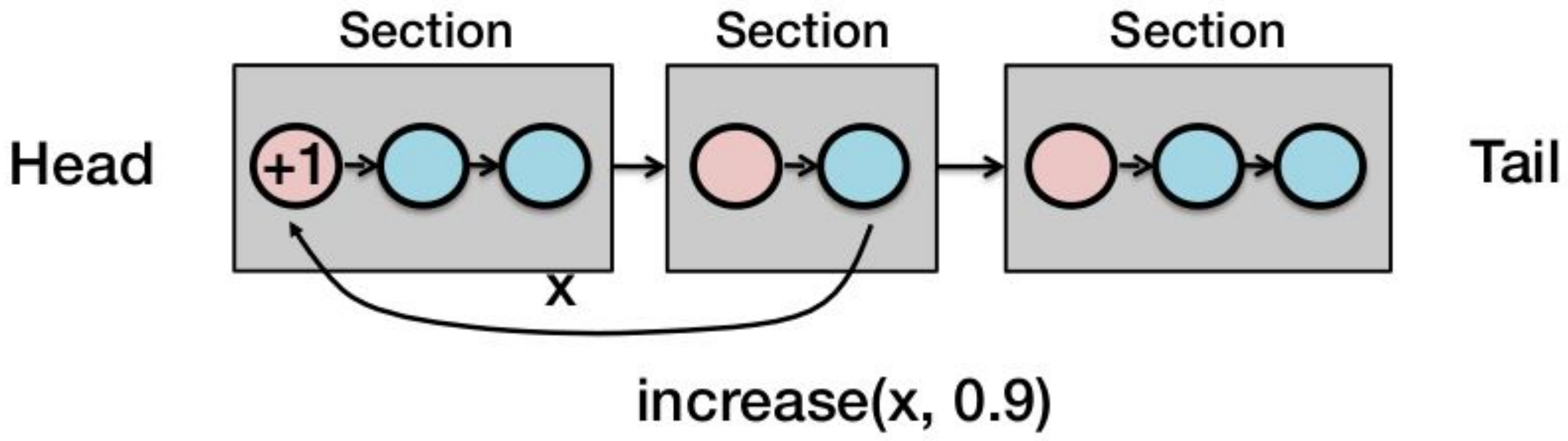
Key Idea: Restricted Insertion Points

- The orders within one section not guaranteed!
  - Result of approximation
- More sections, more MEM buffer, smaller approximation error



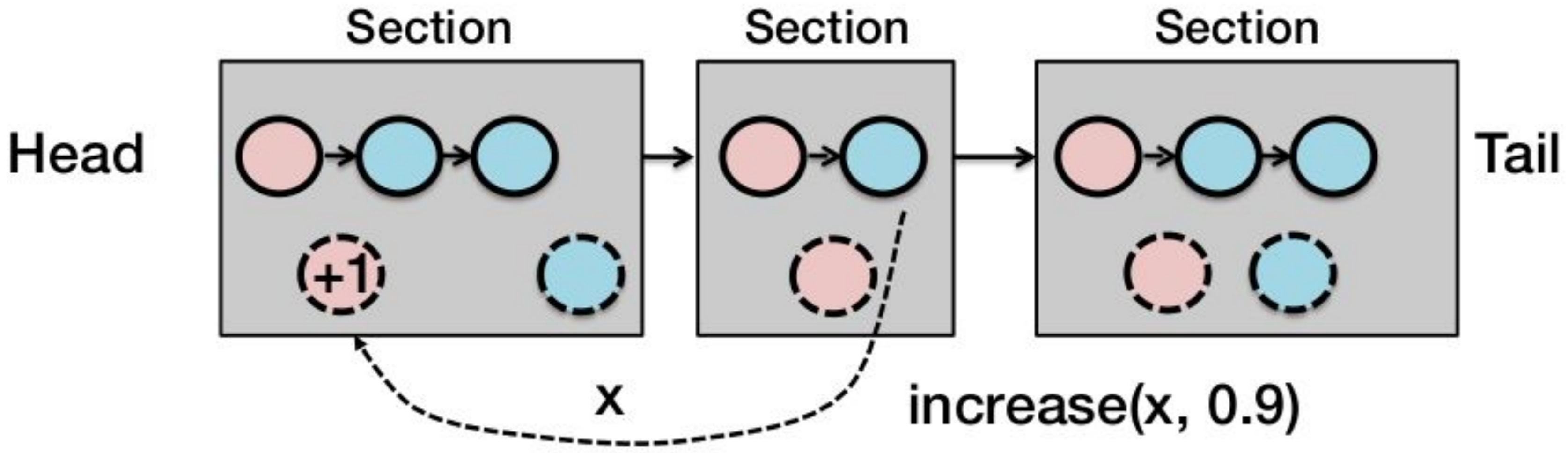
Key Idea: Lazy Update

- Naive approach for updating
  - Copy to the corresponding active block
  - Problem: Duplicated data on flash



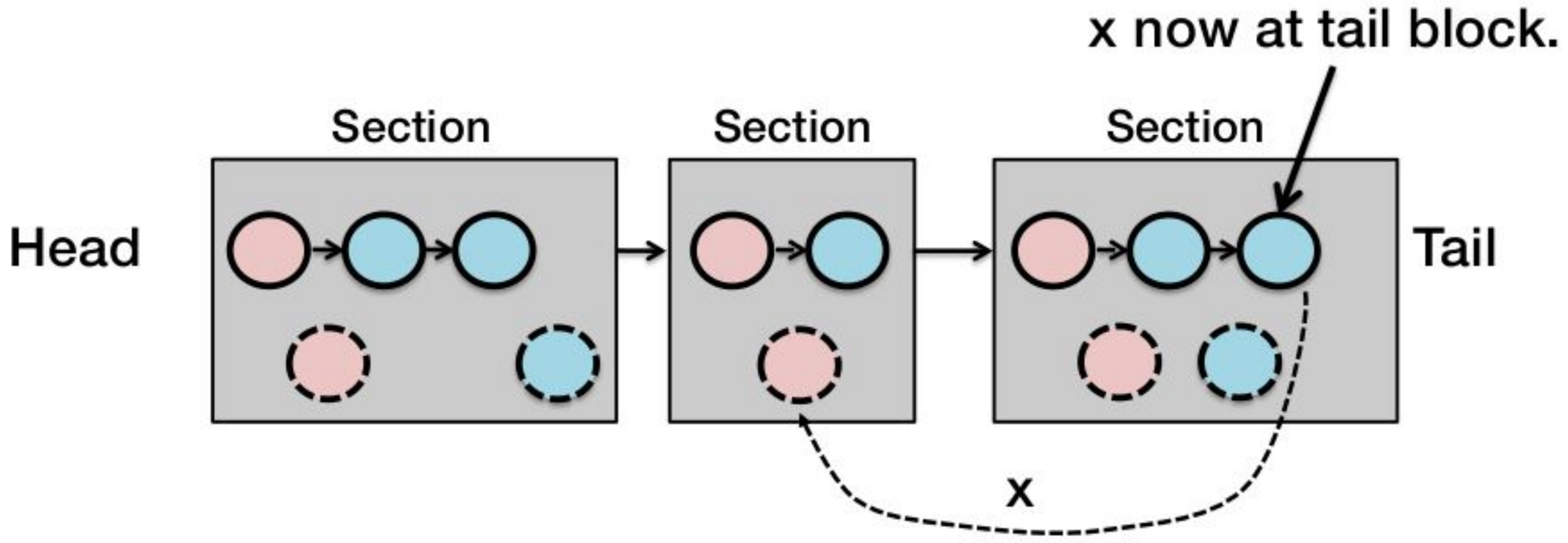
Key Idea: Lazy Update

- Use virtual block to track the updated location
  - Virtual block is in memory, so no data written to Flash



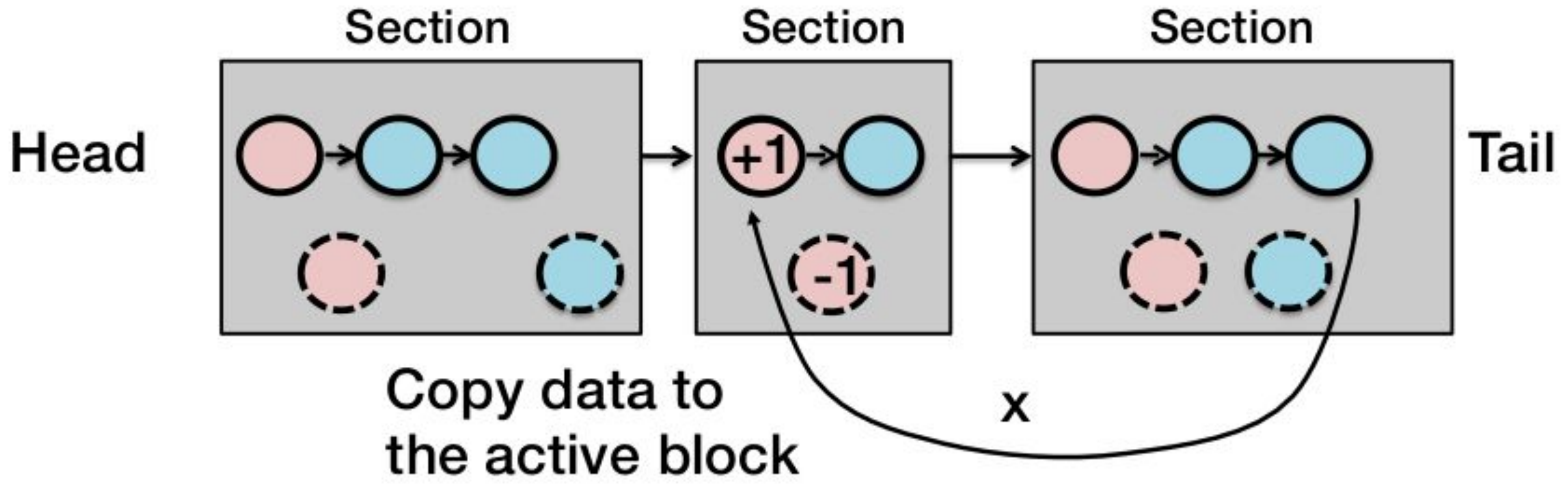
Key Idea: Lazy Update

- Copy data to the active block only when its block is evicted



Key Idea: Lazy Update

- Copy data to the active block only when its block is evicted
  - Delete the data from virtual block



Thank You