AutoStream: Automatic Stream Management for Multi-stream SSDs

Jingpei Yang, PhD, Rajinikanth Pandurangan, ***Changho Choi, PhD***, Vijay Balakrishnan

Memory Solutions Lab
Samsung Semiconductor

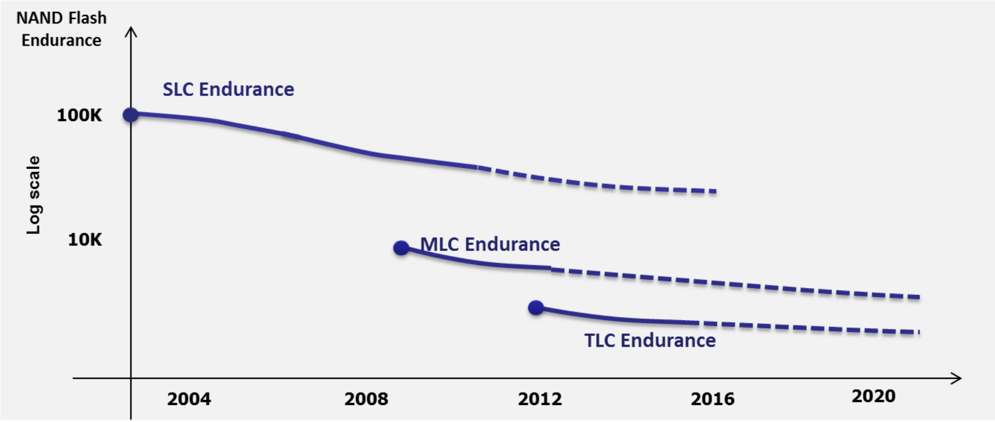
Agenda

- **SSD NAND flash characteristics**
- **Multi-stream**
- **Autostream: Automatic stream management**
  - Multi-Q
  - SFR
- **Performance enhancement**
- **Summary**

SSD NAND Flash Characteristics

- **Different IO units**
  - Read/Program: Page, Erase: Block (=multiple of pages)
- **Erase before program**
  - Out-of-place update
- **Unavoidable GC overhead**
  - The higher GC overhead, the larger Write Amplification*(= the lower endurance)
- **Limited number of Program/Erase cycles**

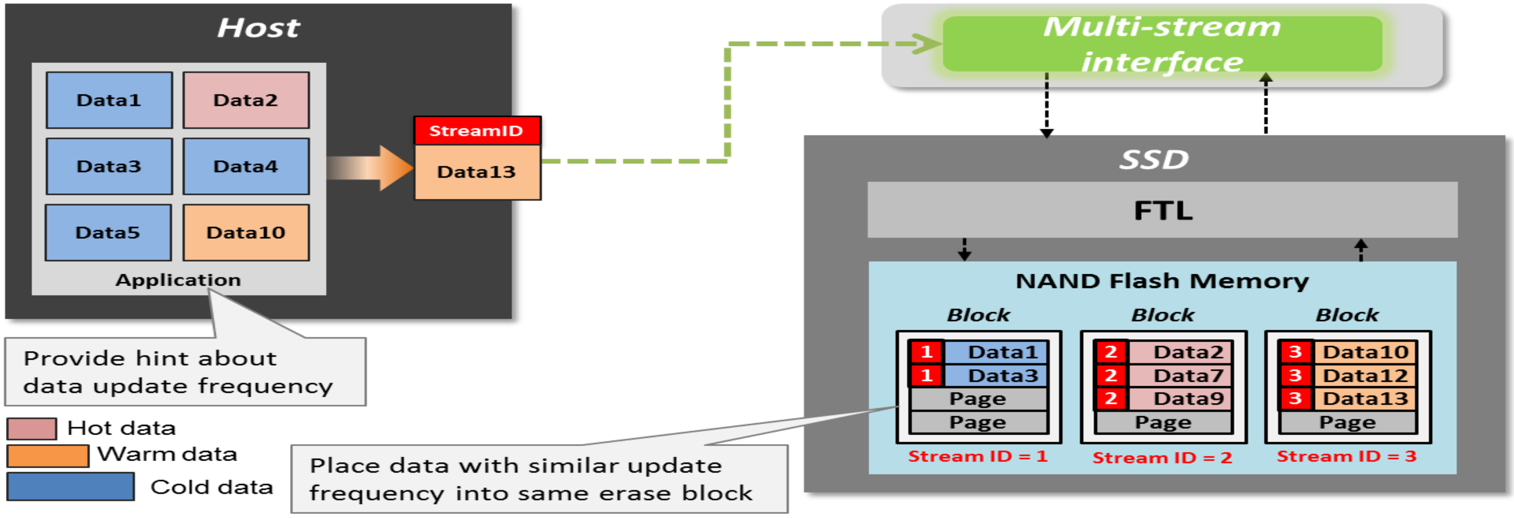
$$* WAF(Write\ Amplification\ Factor) = \frac{amount\ of\ data\ written\ to\ NAND\ Flash}{amount\ of\ data\ written\ by\ host}$$



To maximize SSD lifetime, need to minimize Write Amplification!

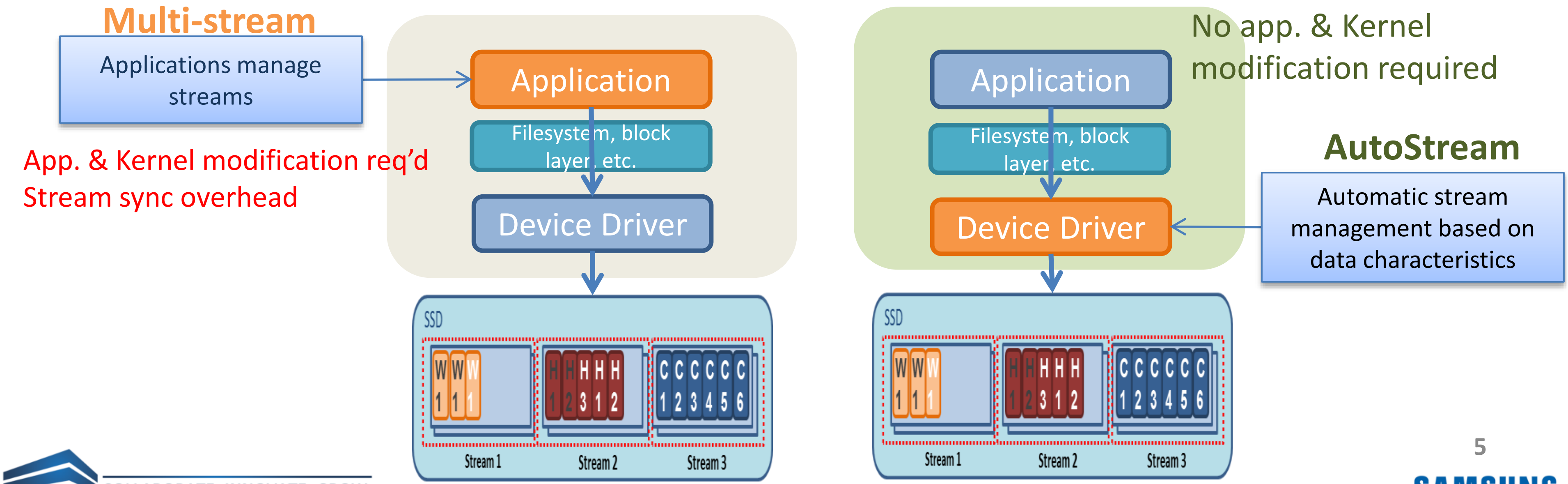
Multi-stream: Minimize Write Amplification

- **Store similar lifetime data into the same erase block and reduce WA (GC overhead)**
- **Provide better endurance and improved performance**
  - Host associates each write operation with a stream
  - All data associated with a stream is expected to be invalidated at the same time (e.g., updated, trimmed, unmapped, deallocated)
  - Align NAND block allocation based on application data characteristics(e.g., update frequency)

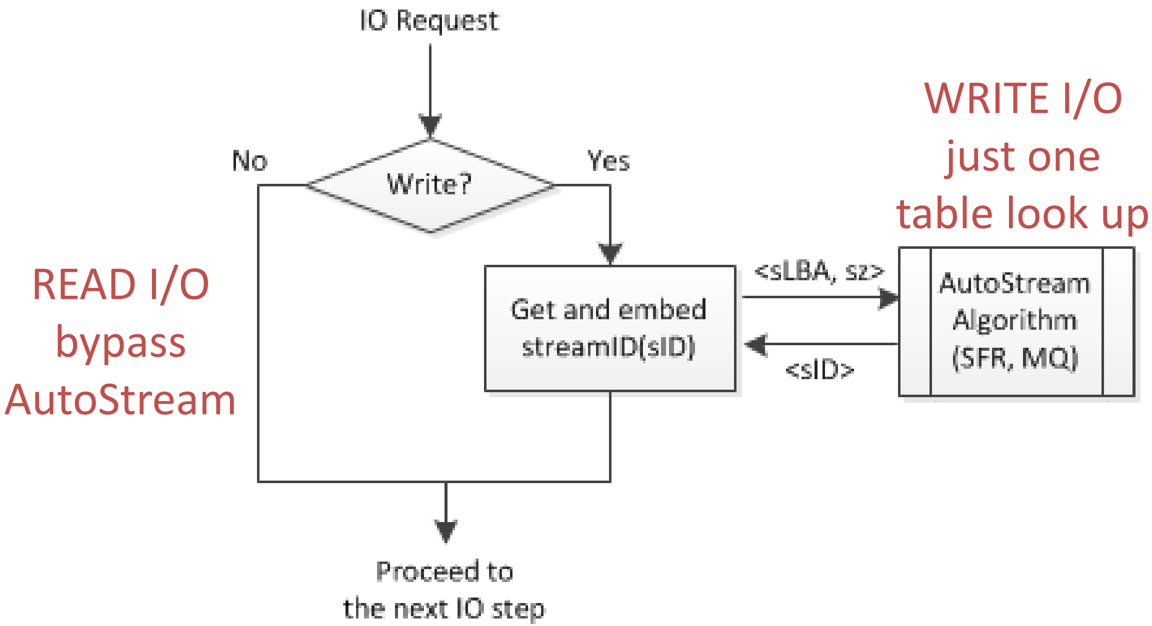


AutoStream: Automatic Stream Management

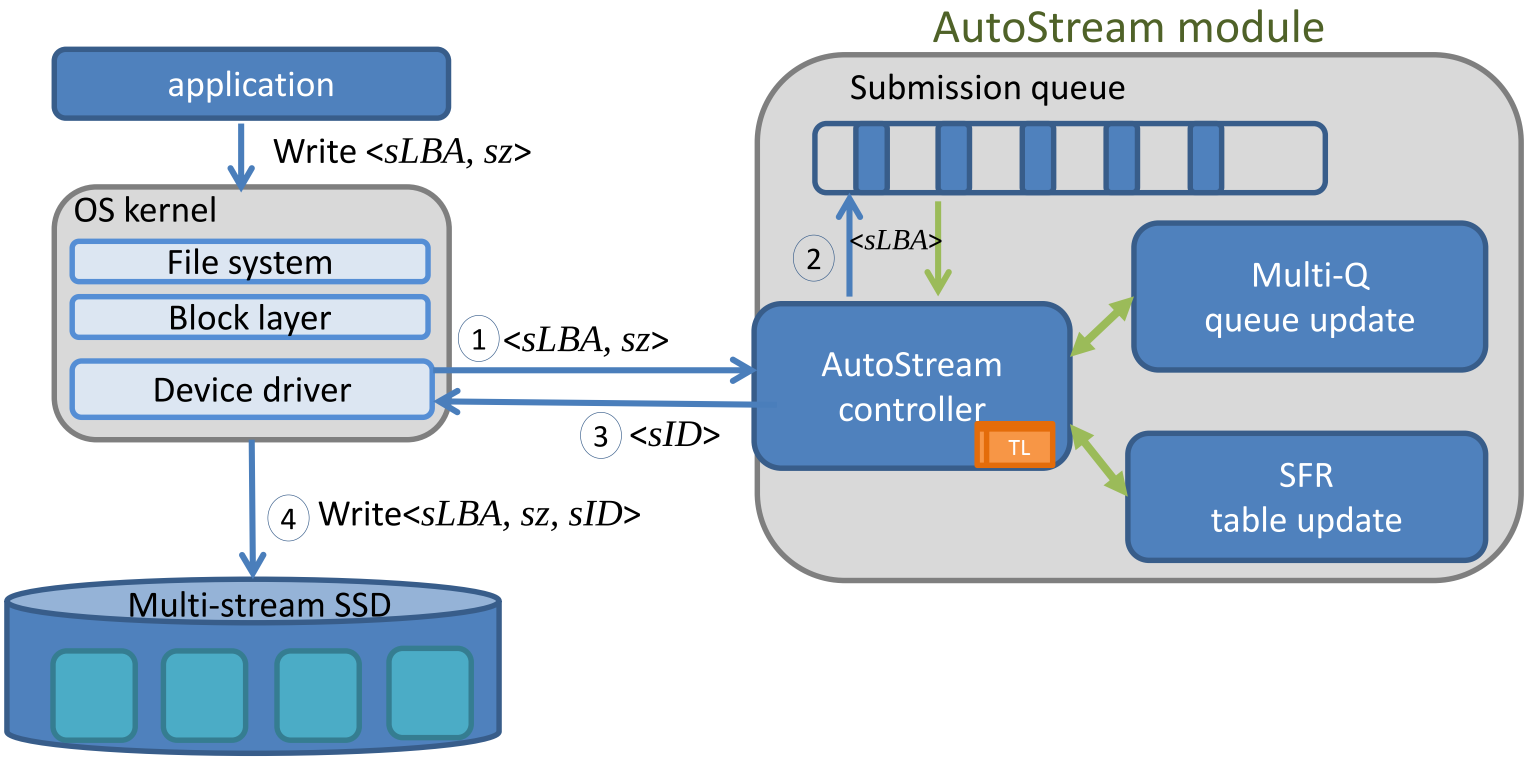
- **Multi-stream shows good benefit but requires application and system modification**
  - More challenges in multi-application, multi-tenant environments (e.g., VM or Docker)
- **AutoStream**
  - Make stream detection independent of applications (e.g., in device driver)
  - Cluster data into streams according to data update frequency, recency and sequentiality
  - Minimize stream management overhead in application and systems



AutoStream IO Processing with Minimal Overhead



AutoStream Implementation



Multi-Q Algorithm Basics

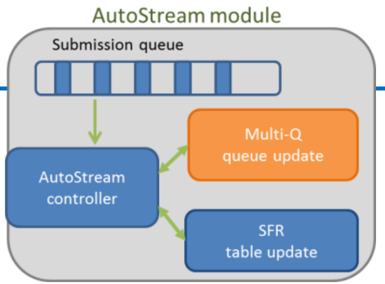
- **Divide a whole SSD space into the same size chunks**
  - 480GB SSD, 1MB chunk size -> 480,000 chunks
- **Track statistics for each chunk**
  - access time, access count, expiry time, etc.
  - Expiry time
    - *hottest chunk's lifetime := current time – last access time*
    - *Other chunk's expiry time:= current time + hottest chunk's lifetime*

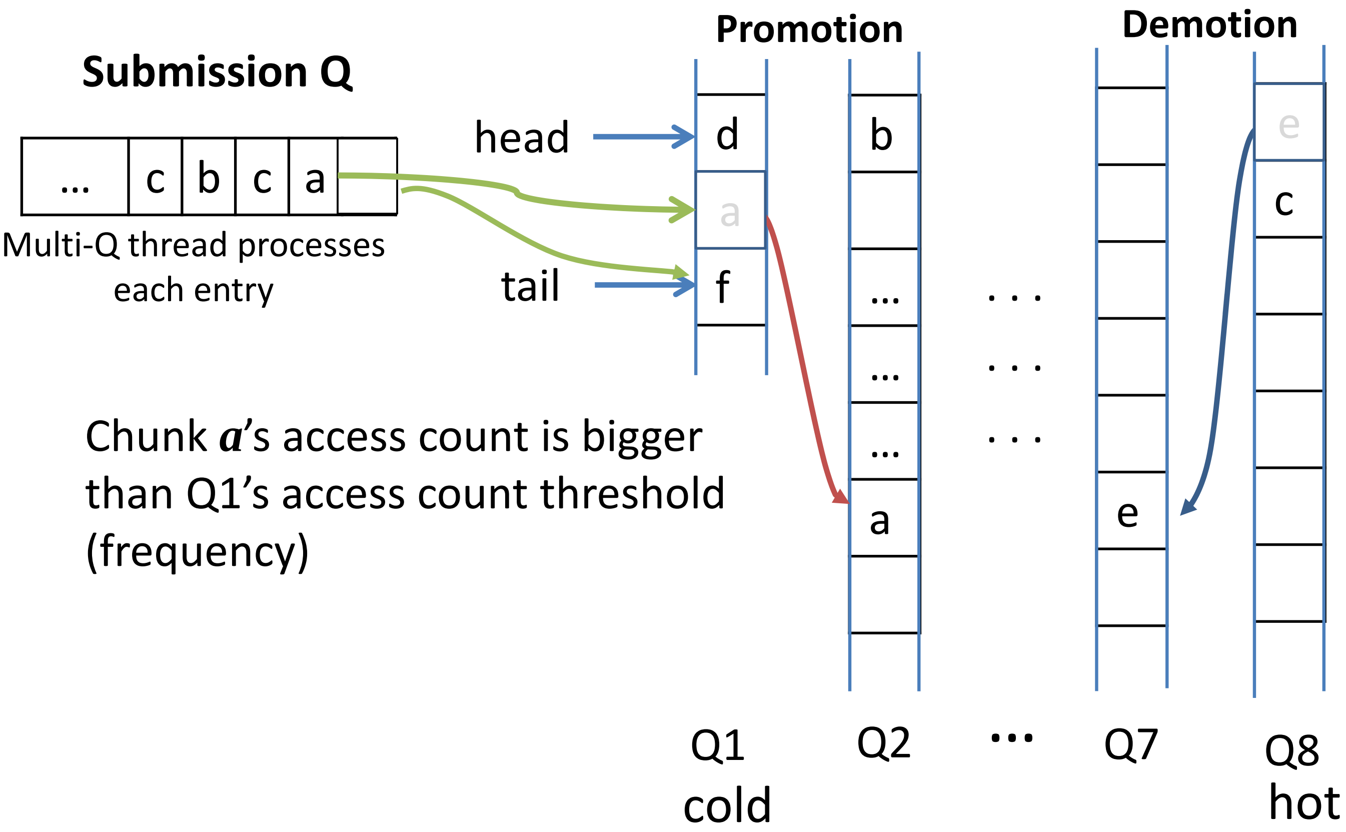
chunk id	……	c	c	x	y	z	u	w	c	d
access time	……	4	5	6	7	8	9	10	11	12
access count	……	1	2	1	1	1	1	1	3	1

Access time 11: *Hottest chunk =* ***c***
Chunk c's lifetime = 11 – 5 = 6

Access time 12:
chunk ***d*** *expiry time = 18 (12+6)*

Multi-Q Update (Promotion & Demotion)



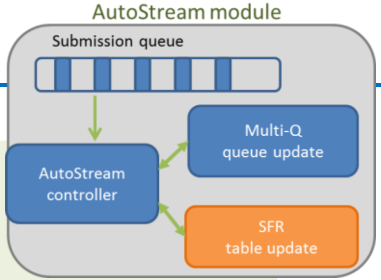


Chunk *a*'s access count is bigger than Q1's access count threshold (frequency)

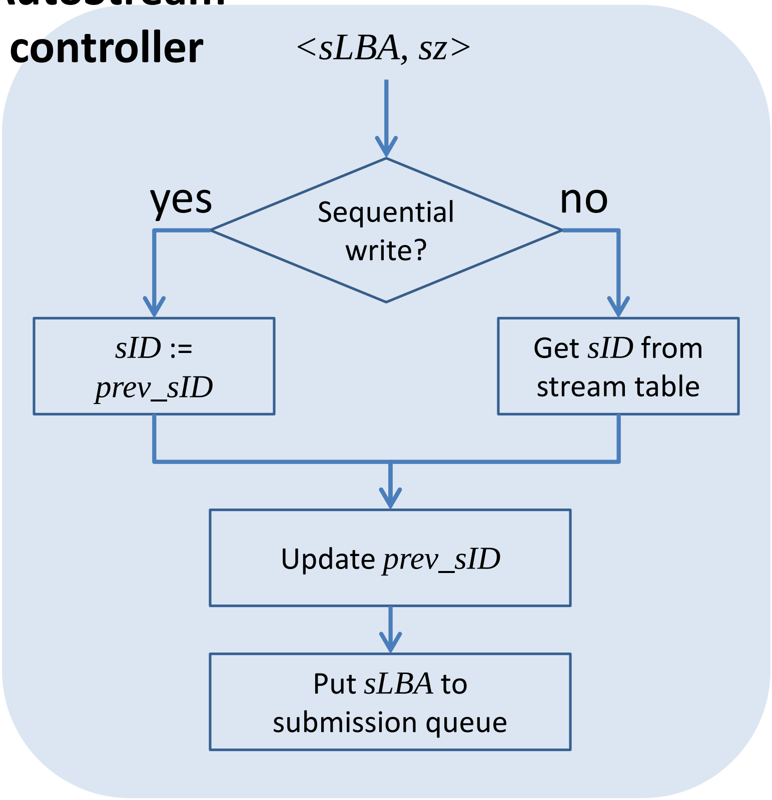
Chunk *e*'s expiry time has passed (recency*)

* Recency considers the last updated time

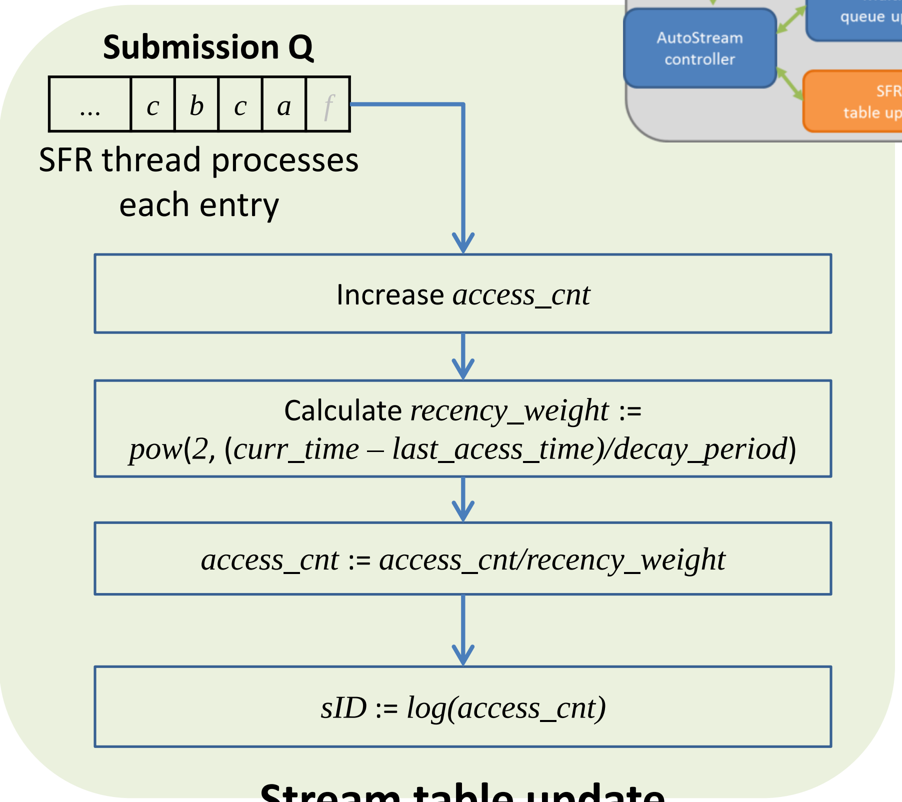
SFR - SequentialityFrequencyRecency Algorithm



AutoStream controller



Sequentiality

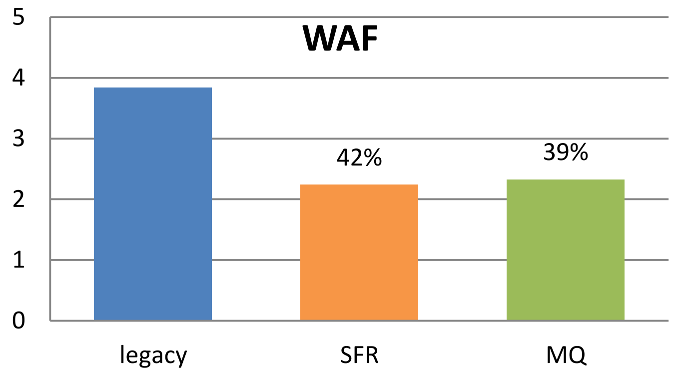


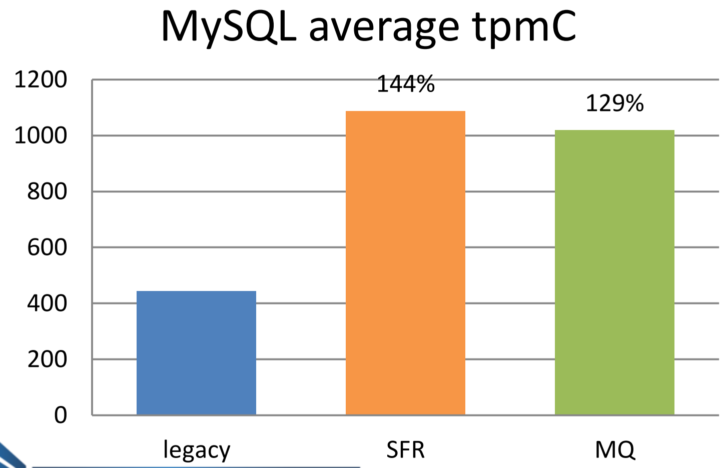
Stream table update (Frequency, Recency)

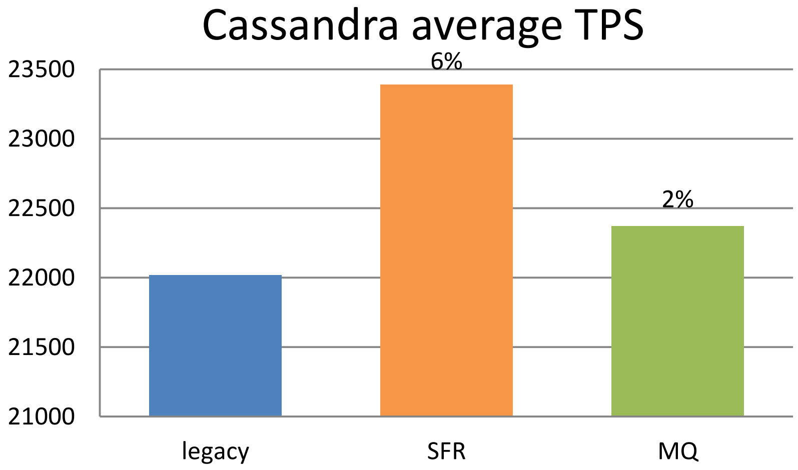
Docker Environment Performance Measurement

- Running 2 MySQL & 2 Cassandra instances simultaneously

	Database Size	Workload
MySQL TPC-C	800 warehouse	TPC-C: 30 connection
Cassandra -Stress	1KB record, 100 million entries	r/w: 50/50







Summary

- **AutoStream**
  - With no application and system modification, improve SSD lifetime and performance
- **AutoStream with minimal overhead**
  - Works well under different workloads for diverse applications on various system environments
  - Up to *60%* WAF reduction
  - Up to *237%* performance improvement
- **Future work**
  - Optimize resource utilization and performance to fit into devices

Thank You!