Open Source High Performance DRAM/SSD Hybrid Caching Engine

https://github.com/facebook/CacheLib
https://cachelib.org/docs/

Jaesoo Lee
Software Engineer, Meta

Meta

Agenda

01 Introduction

02 CacheLib Architecture

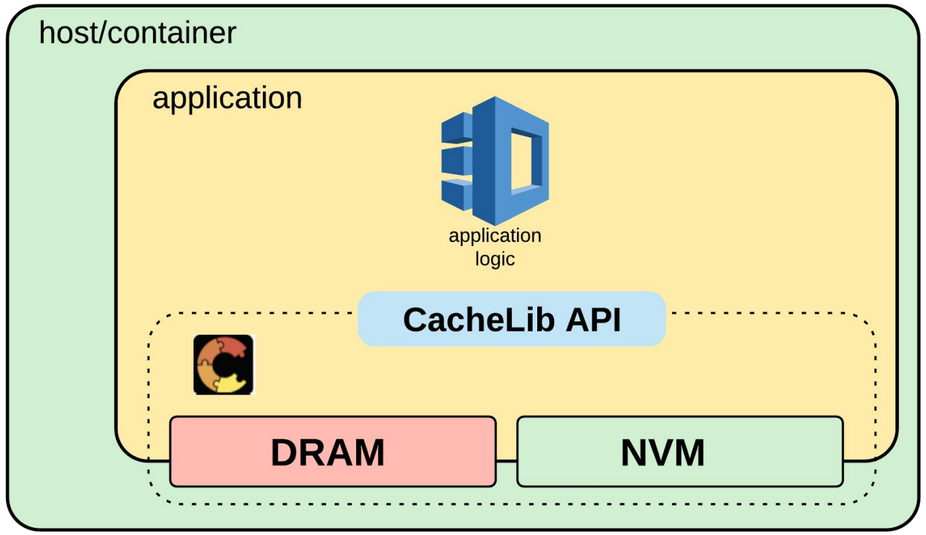
- a DRAM Cache
- b NVM Cache

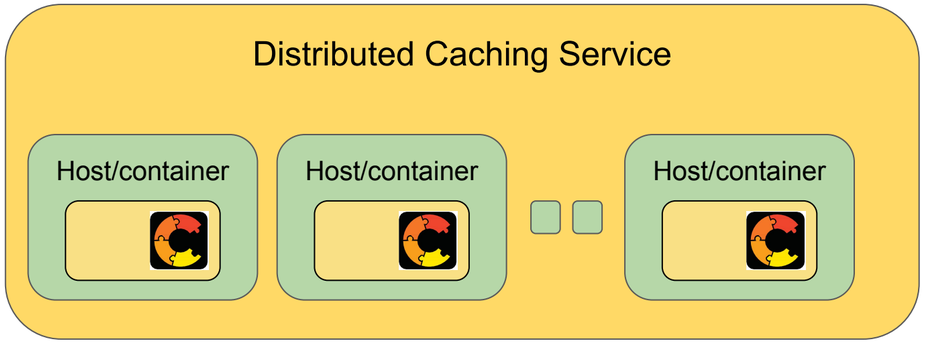
03 CacheBench and Workloads

04 Recent Open Source Innovations

CacheLib: DRAM/SSD Software Caching Engine

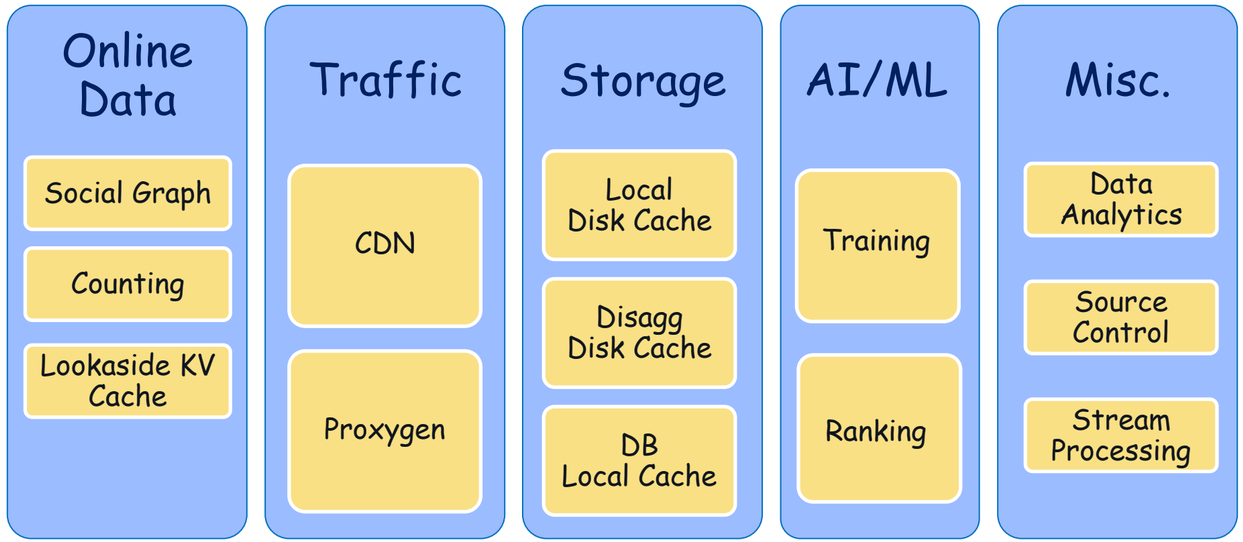
- Library implementing S/W caching engine
  - Thread-safe API for insert/find/remove
  - DRAM/SSD-hybrid multi-level (L1/L2)
  - High performance and highly configurable
  - Written in C++
- Two forms of usage
  - In-process local cache
  - Core engine for distributed caching service
- Mainly for web or database caches





CacheLib: Usage at Meta

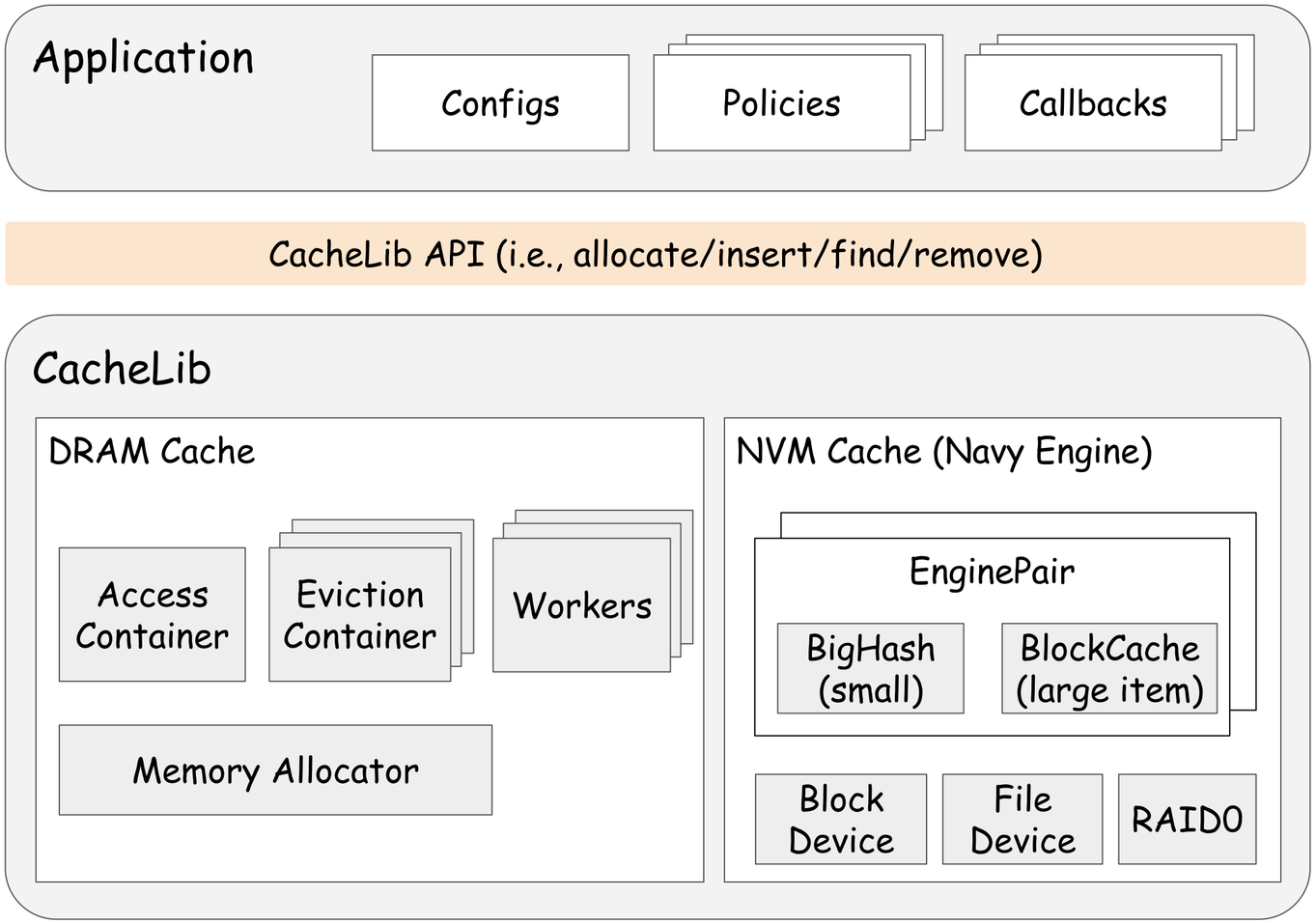
- CacheLib is widely adopted (~100s services) within Meta



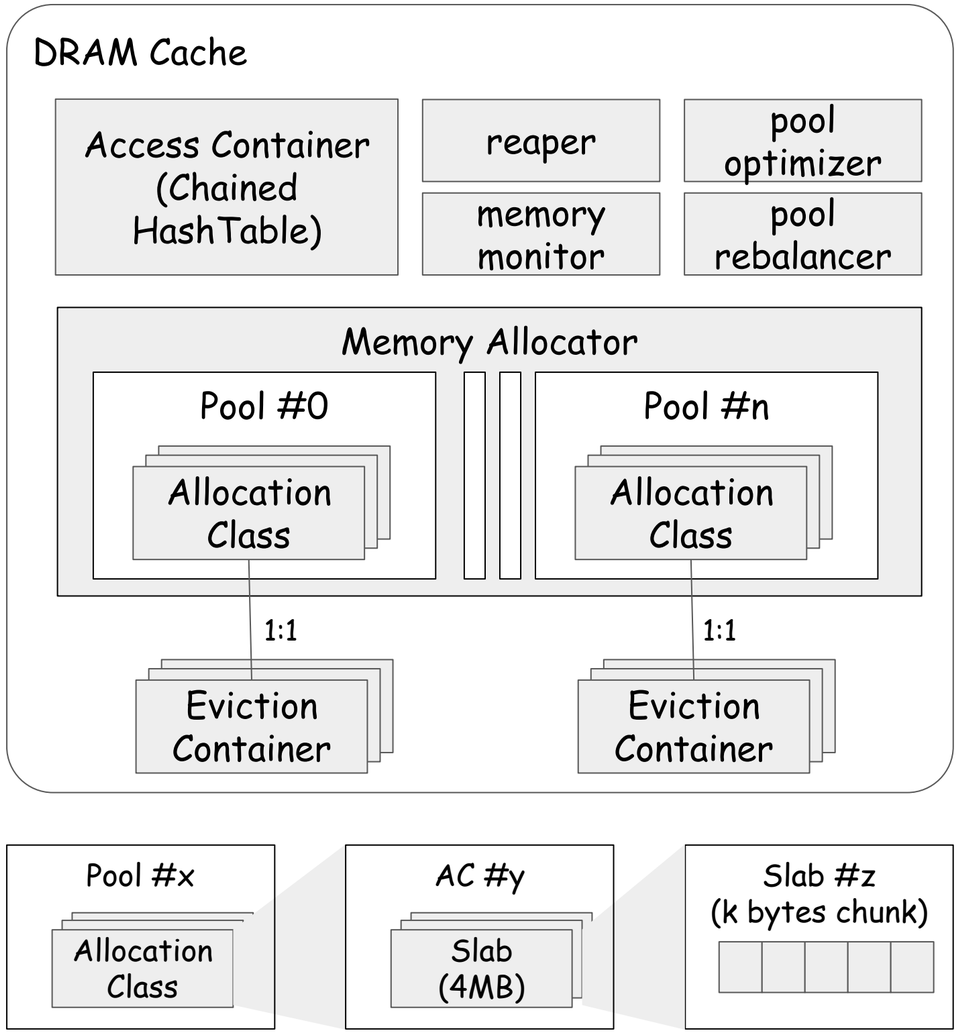
CacheLib : Open Source Project @github.com

- Open sourced in 2021
  - CacheLib: plug-and-play cache engine for your cache services
  - CacheBench: benchmarking tool with synthetic and real workloads on target platform
  - Workload traces: traces captured from large-scale cache services within Meta
- Adoption
  - Academic Research:
    - Kangaroo: Caching billions of tiny objects on flash
  - Companies using cachelib in their services
    - Pinterest, NebulaGraph
  - HW vendors optimizing for CacheLib or Benchmarking with CacheLib
    - Samsung, Western Digital, Kioxia, Intel, etc.

CacheLib Architecture



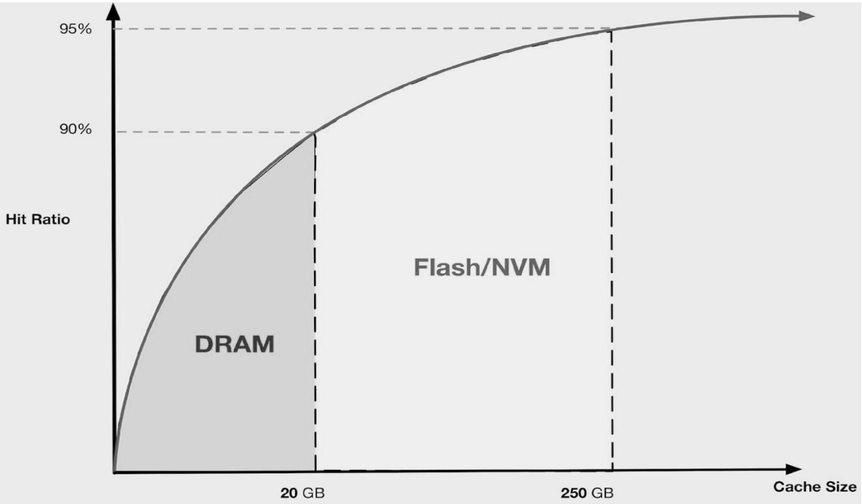
DRAM Cache

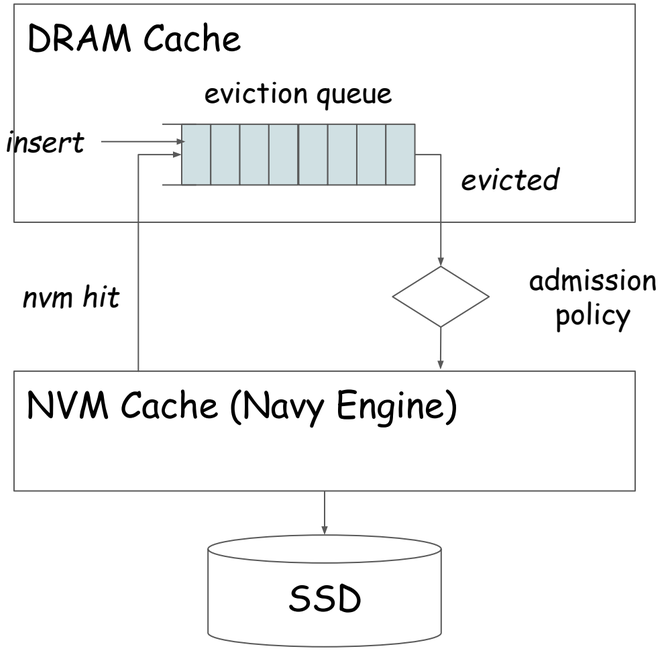


- Cache memory is organized into
  - `<Pool>.<Allocation Class>.<Slab>.<Chunk>`
    - Pool: isolate/protect memory usage
    - Allocation Classes of discrete sizes (minimize frag.)
- Eviction Container
  - List to maintain the eviction/replacement order per allocation class
    - e.g., LRU, LRU2Q, TinyLFU
- Workers
  - Memory monitor
    - Release slabs voluntarily to prevent OOM
  - Pool rebalancer
    - Move slabs between ACs for load balancing
  - Pool optimizer/resizer
    - Resize pool dynamically or on request
  - Reaper
    - Evict items proactively to enforce TTL (Time To Live)

DRAM/SSD Hybrid Layered Cache

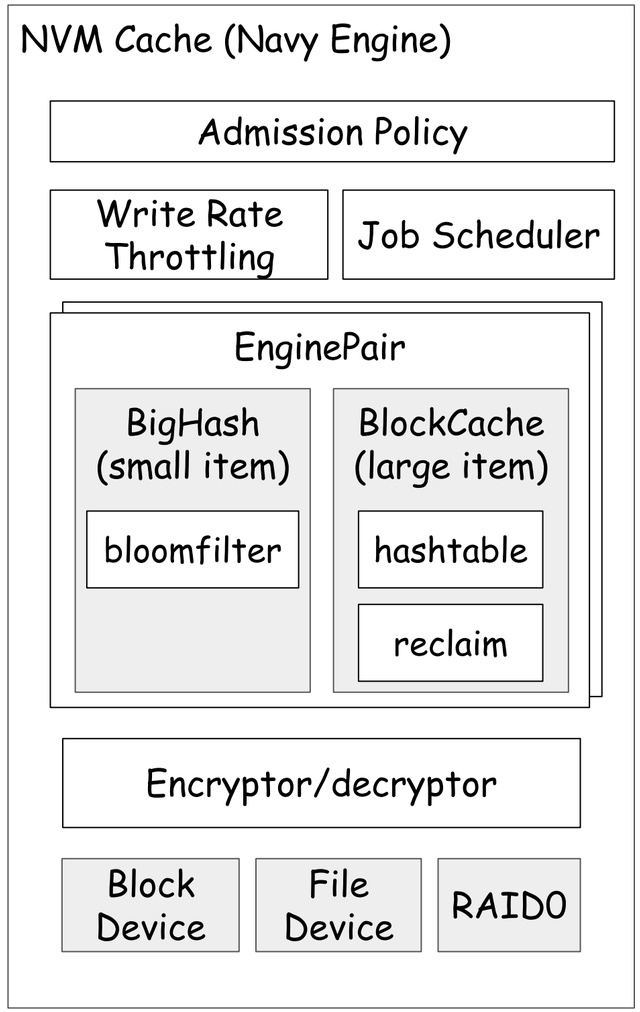
- SSD is a cheaper option to achieve higher hit ratio
  - Tradeoffs with latency increase





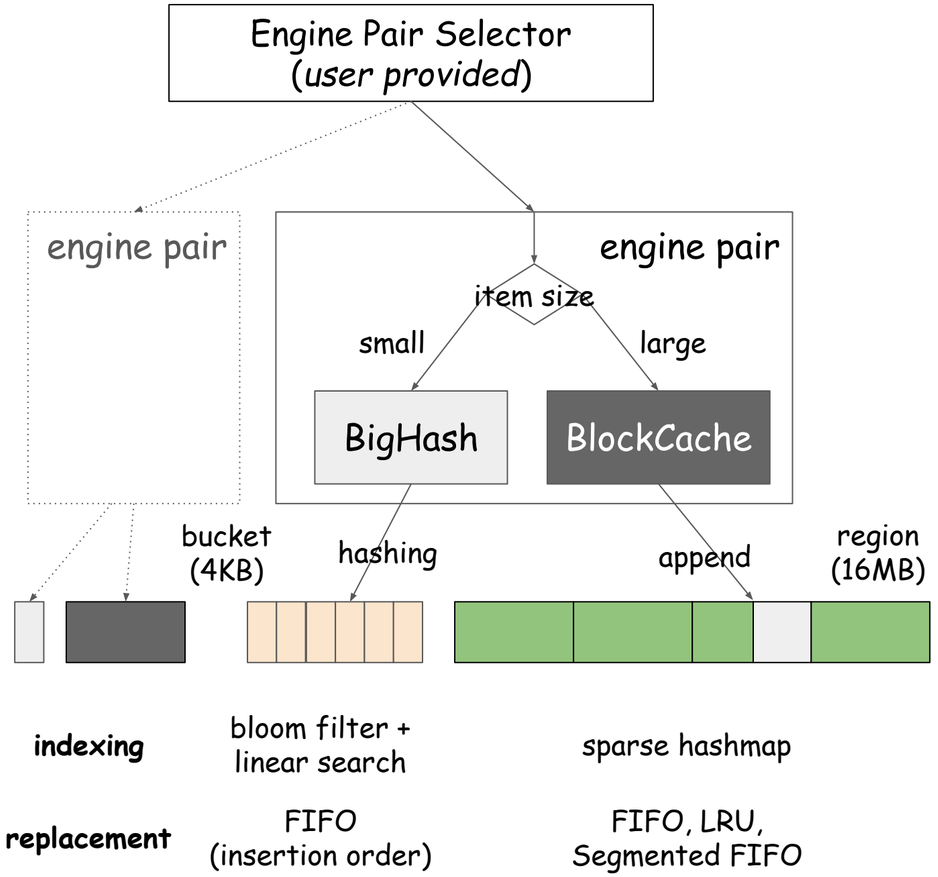
- *find()* is asynchronous for NVM access
  - Notified via callback (*folly::semifuture*)

NVM Cache



- NVM admission policy
  - Determine whether to admit in NVM cache
  - E.g., reject those soon to be expired, reject first seen, within recency threshold, ML-based
- Write rate throttling (a.k.a., Navy admission policy)
  - Random reject for limiting the write rate for SSD endurance guarantee
- Engine pair and engine pair selector
  - Logical partitioning of SSD for isolating space usage
  - BigHash is for small item while BlockCache is for larger item
- Target cache storage
  - Block device (O_DIRECT), regular file, RAID0
  - Optional block encryptor/decryptor

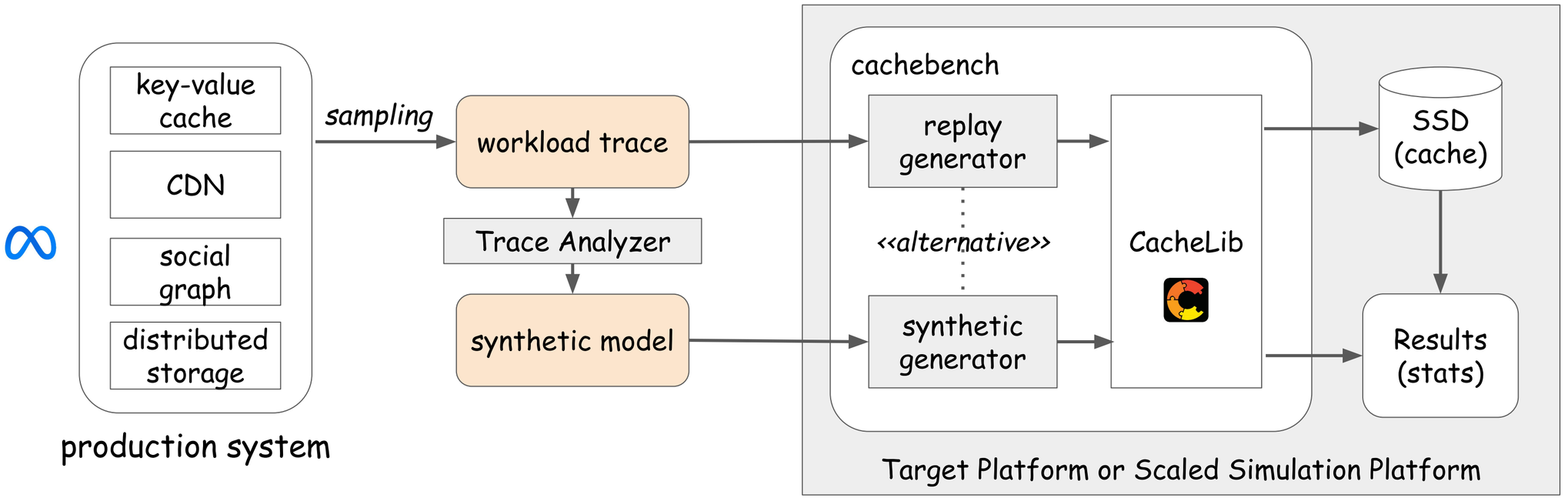
NVM Cache: BlockCache and BigHash



- BlockCache
  - Fully associative cache like full-page mapping FTL
    - Append to active clean region (16MB)
  - Reclaim for free region
    - Reinsert or evict based on reinsertion policy
  - Sparse hashmap for indexing
    - Relatively high mapping overhead (>20B per items)
- BigHash for small objects
  - Similar to n-way set associative cache
    - Set is of fixed size called bucket (4KB)
    - FIFO replacement for the collision within bucket
  - Bloom filter (4B) + linear search within bucket
    - Low memory overhead - good for smaller objects
  - High write amplification
    - 4K read-modify + random writes

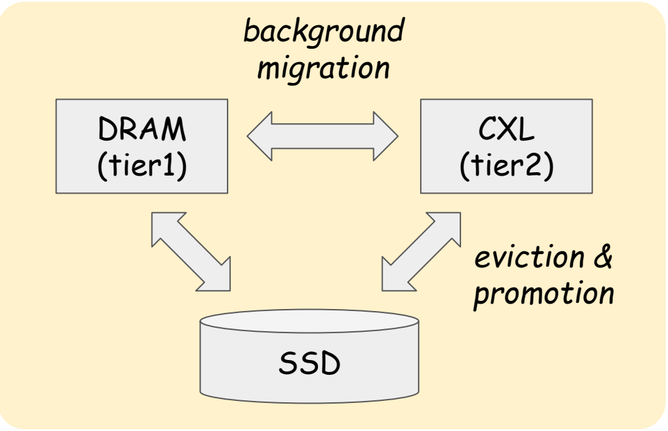
CacheBench: High Confidence Benchmarking Platform

- Cache simulator to evaluate cache performance on the target hardware
  - Support for both real trace workloads and synthetic workloads
- Example usages
  - New hardware evaluation for caching workloads
  - Explore/research on eviction/admission policies

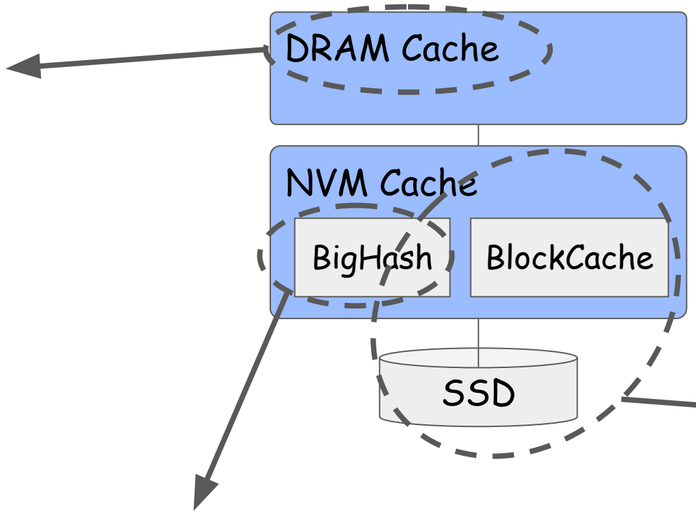


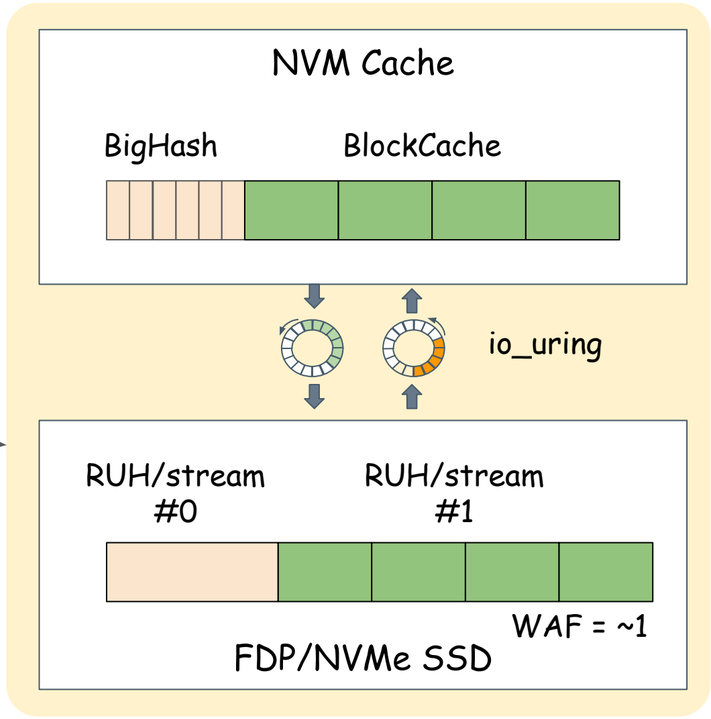
Recent Innovations from Open Source

Tiered Memory (intel, 2023): Power/Cost
- Heterogeneous byte-addressable memory support

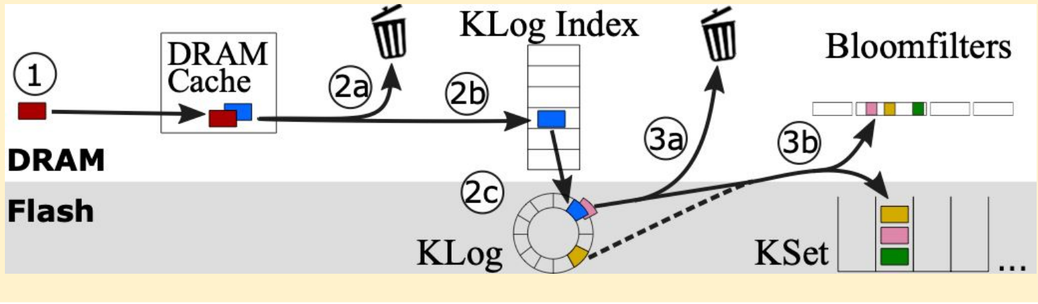


CacheLib/FDP-SSD (Samsung, 2023): WAF/Cost
- FDP-aware vertical optimization
- WAF ~1 at ~100% utilization





Kangaroo (CMU, 2021): Performance/WAF
- Enhanced BigHash engine



Useful Links

- Come talk to us at Cache Meetup (Early September)
- More details at cachelib.org
  - User guide
  - Architecture guide
  - CacheBench
- Papers related to CacheLib
  - https://www.usenix.org/system/files/osdi20-berg.pdf
  - https://www.pdl.cmu.edu/PDL-FTP/NVM/McAllister-SOSP21.pdf


Meta