Optimize Redis with NextGen NVM

Shu Kevin/Si, Peifeng/Li, Zhiming
Intel Corporation
Agenda

- Redis introduction
- NVM introduction
- Scenario 1: Use NVM to increase Redis capacity
- Scenario 2: Use NVM to improve the performance of Redis persistency
- Completed features support of Redis on NVM
  - LRU & Defrag
  - Linux Copy-on-Write on NVM
- Summary
Redis Introduction

Redis is an open-source in-memory K-V database that offers high performance, replication, and a unique data model with optional durability.

Fig 1. Redis persistence - AOF

Fig 2. Redis data structures
Agenda

- Redis introduction
- NVM introduction
- Scenario 1: Use NVM to increase Redis capacity
- Scenario 2: Use NVM to improve performance of Redis persistency
- Completed features support of Redis on NVM
  - LRU & Defrag
  - Linux Copy-on-Write on NVM
- Summary
NVM Highlight

- Large capacity
- Lower $ / GB
- Close to DRAM throughput and latency
- Persistency - may or may not use
Agenda

- Redis introduction
- NVM introduction
- Scenario 1: Use NVM to increase Redis capacity
- Scenario 2: Use NVM to improve the performance of Redis persistency
- Completed features support of Redis on NVM
  - LRU & Defrag
  - Linux Copy-on-Write on NVM
- Summary
Use NVM to increase Redis capacity

- Design Option #1 - Store all heap data in NVM
  - Replace jemalloc by libmemkind
  - Minimum Redis code change needed
Use NVM to increase Redis capacity

Design option #1 - Store all heap data in NVM

- Performance optimization opportunity:
  - Big and sequential data access pattern has better performance than small and random data access pattern in NVM
  - Meanwhile in Redis, a lot of management data structures are small and usually be accessed randomly
Use NVM to increase Redis capacity

Design option #2 - Store most of heap data in NVM
- Only store large value in NVM (>64 byte by default)
  - Keep Redis management data structures in DRAM
  - Optimize the data placement strategy for each data type

Performance:
- Close to Redis performance run on DRAM
- URL: https://github.com/pmem/pmem-redis
Use NVM to increase Redis capacity

- Implementation details:
- Encoding to STRING:
  - **RAW**: Store value (>threshold) in DCPMM and store SDS pointer in DRAM.
  - **INT**: Store in the DRAM
  - **EMBSTR**: Store in the DRAM
Use NVM to increase Redis capacity

- Implementation details:
  - Encoding to LIST: QUICKLIST
    - After Redis-3.2.7, quicklist is used to implement the list data type. Each quicklist node contains a ziplist structure.
    - ziplist is to save memory usage for small items like int/embedded string
    - Store value (>threshold) in DCPMM and store the pointer in ziplist
Use NVM to increase Redis capacity

- Implementation details:
- Encoding to HASH:
  - **ZIPLIST**: Store value (>threshold) in DCPMM and store the pointer in ziplist
  - **HASHTABLE**: Store value (>threshold) in DCPMM and store the pointer in hashtable
Use NVM to increase Redis capacity

- Implementation details:
  - Encoding to SET
    - **INTSET**: It is unnecessary to move intset into DCPMM due to its small size
    - **HASHTABLE**: Store value (>threshold) in DCPMM and store the pointer in hashtable
Use NVM to increase Redis capacity

- Implementation details:
  - Encoding to ZSET
    - **ZIPLIST**: Value (>threshold) stored in DCPMM and the pointer stored in ziplist
    - **SKIPLIST**: Value (>threshold) stored in DCPMM and the pointer stored in skiplist
Agenda

- Redis introduction
- NVM introduction
- Scenario 1: Use NVM to increase Redis capacity
- Scenario 2: Use NVM to improve the performance of Redis persistency
- Completed features support of Redis on NVM
  - LRU & Defrag
  - Linux Copy-on-Write on NVM
- Summary
Use NVM to improve the performance of Redis persistency

Design option #1 – Persist everything in NVM

- Use libpmemobj to store data and its mgmt. structures in NVM

Source:
- URL: [https://github.com/pmem/redis/tree/3.2-nvml](https://github.com/pmem/redis/tree/3.2-nvml)
Use NVM to improve the performance of Redis persistency

Design option #2 – Pointer based AOF
- Store key in DDR and AOF (same to Open Source Redis)
- Store value in NVM (for persistency) and only store its pointer in AOF (for recover)
- Leverage AOF to guarantee data integrity

Performance:
- Much better than Open Source Redis AOF (sync=always)
- URL: https://github.com/pmem/pmem-redis
Agenda

- Redis introduction
- NVM introduction
- Scenario 1: Use NVM to increase Redis capacity
- Scenario 2: Use NVM to improve the performance of Redis persistency
- Completed features support of Redis on NVM
  - LRU & Defrag
  - Linux Copy-on-Write on NVM
- Summary
Completed features support of Redis on NVM

Besides data movement optimization, also covers below features for NVM adoption:

- Data retirement support: LRU on NVM
  - Evict the objects only when both DDR and NVM are full
  - Support to evict objects in NVM
- Defragmentation on NVM
  - In Redis 4.0, leverage jemalloc to support defrag the space in NVM
Completed features support of Redis on NVM

- Pitfall – Linux Copy-on-Write is not supported on NVM

- Problem Statement
  - Redis leverages CoW to do RBD snapshot, replication, etc.
  - The NVM space doesn’t support CoW, because it is based on a memory mapped file. Hence the parent and child process share the same NVM address space, which may cause data corrupt during COW.

- Solution
  - During CoW, duplicate objects in parent process of Redis
  - Introduce two hash tables to help the parent process to decide if it needs to duplicate the object or not
Summary

Optimized Redis with Next Gen NVM can achieve:

- Full compatible API and Functionality
  - Compatible with open source Redis 4.0+
- Higher capacity per instance with lower TCO
  - Way to use NVM as a volatile device to provide larger capacity for in-memory database
- Higher Perf on persistency
  - Novel design to use NVM for high performance persistency on Redis
Reference

- Persistent Memory Development Kit
  - Libmemkind [https://github.com/memkind/memkind](https://github.com/memkind/memkind)

- Open Source Redis download [https://redis.io/](https://redis.io/)

- Optimized PMEM Redis Repo [https://github.com/pmem/pmem-redis](https://github.com/pmem/pmem-redis)