CacheKV: Redesigning High-Performance Key-Value Stores with Persistent CPU Caches

Yijie Zhong*, Zhirong Shen*, Zixiang Yu*, Jiwu Shu*†,
*Xiamen University, †Tsinghua University
yijiezhong@stu.xmu.edu.cn, shenzr@xmu.edu.cn, yuzixiang23@foxmail.com, shujw@tsinghua.edu.cn

I. Summary of Implementations

We implement our proposed CacheKV atop LevelDB [1] with C++. We make use of Intel CAT [2] to allocate space in the persistent CPU caches. We elaborate on the dependent softwares, representative KV stores, and testing tool as follows.

Dependent softwares: the implementation of CacheKV relies on a suit of softwares, including `Intel CAT`, `ndctl`, `ipmctl`, `Intel PMWatch`, and `PMDK`, which can be reached via the following URLs.

```
$ wget https://github.com/intel/intel-cmt-cat
$ wget https://github.com/pmem/ndctl
$ wget https://github.com/intel/ipmctl
$ wget https://github.com/intel/intel-pmwatch
$ wget https://github.com/pmem/pmdk
```

Representative KV stores: we use two representative KV stores for comparison, namely NoveLSM [3] and SLM-DB [4], which can be downloaded via the following URLs.

```
$ wget https://github.com/sudarsunkannan/lsm_nvm
$ wget https://github.com/WangEP/SLM-DB
```

Testing tools: we employ `db_bench` and `YCSB-C` in our evaluation, where `db_bench` is released with LevelDB [1] and YCSB-C [5] is a C++ version of YCSB. They can be reached via the following URLs.

```
$ wget https://github.com/google/leveldb
$ wget https://github.com/basicthinker/YCSB-C
```

II. Evaluation Details

Hardware configurations: We conduct extensive experiments on a single machine equipped with two 2.10 GHz Intel Golden 5318Y CPUs (with 24 cores in total), 128 GB of DRAM memory and four Optane PMem DIMMs of 200 series (128 GB per DIMM and 512 GB in total). The Optane PMem DIMMs are configured in interleaved `App Direct Mode`, which are connected to one processor. Figure 1 shows the major hardware configurations of our testbed and Figure 2 shows detailed information about the Optane PMem used in our evaluation.

We perform extensive testbed experiments, including (i) experiments to understand the properties of CacheKV; and (ii) experiments to understand the sensitivity of CacheKV.

Experiments with db_bench: We measure the access performance of CacheKV, NoveLSM, and SLM-DB using `db_bench`. The following script runs `db_bench` to evaluate NoveLSM:

```
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   46 bits physical, 57 bits virtual
CPU(s):                          48
On-line CPU(s) list:             0-47
Thread(s) per core:              1
Core(s) per socket:              24
Socket(s):                       2
NUMA node(s):                    2
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           106
Model name:                      Intel(R) Xeon(R) Gold 5318Y CPU @ 2.10GHz
Stepping:                        6
CPU MHz:                         800.779
CPU max MHz:                     3400.0000
CPU min MHz:                     800.0000
BogoMIPS:                        4200.00
Virtualization:                  VT-x
L1d cache:                       2.3 MiB
L1i cache:                       1.5 MiB
L2 cache:                        60 MiB
L3 cache:                        72 MiB
```

Figure 1: Configurations of our testbed.

```
 DimmID | Capacity    | LockState        | HealthState | FWVersion
============================================================================
 0x0010 | 126.742 GiB | Disabled, Frozen | Healthy     | 02.02.00.1516
 0x0210 | 126.742 GiB | Disabled, Frozen | Healthy     | 02.02.00.1516
 0x0110 | 126.742 GiB | Disabled, Frozen | Healthy     | 02.02.00.1516
 0x0310 | 126.742 GiB | Disabled, Frozen | Healthy     | 02.02.00.1516
```

Figure 2: The configurations of the Intel Optane PMem DIMMs of 200 series used in the evaluation.

```
#!/bin/bash
$./db_bench --benchmarks=fillrandom,readrandom
--threads=1 --num=1000000 --value_size=64
--num_levels=2 --nvm_buffer_size=16
```

As SLM-DB has realized a `db_bench` tool for evaluation, we run the following script to evaluate SLM-DB with `db_bench`:

```
#!/bin/bash
$ ./db_bench --benchmarks=fillrandom,readrandom
--threads=1 --num=1000000 --value_size=64
--nvm_dir=/mnt/pmem0dir-node0/dbbench
```

We amend `db_bench` to add the configurations required in CacheKV. The following script is used to evaluate CacheKV.

```
#!/bin/bash
$./db_bench --threads=1 --num=1000000
--benchmarks=fillrandom  --value_size=64
--num_read_threads=1 --dlock_way=4 --dlock_size=12582912
--subImm_thread=1 --skiplistSync_threshold=65536
--compactImm_threshold=10 --subImm_partition=0
```

Experiments with YCSB: We also employ YCSB-C as an example to clarify how we assess the performance of NoveLSM and CacheKV. The experiments with other YCSB workloads are similar. As SLM-DB is tested by its own `db_bench` tool that provides an API for YCSB test, the workload file of SLM-DB can be generated by using the following script:

```
#!/bin/bash
$./ycsb run basic -P workloada > trace_a.csv
$./db_bench --csv=1 --trace_dir=../trace
--benchmarks=workloada --value_size=64
--nvm_dir=/mnt/pmem0dir-node0/pool
--nvm_size=20480 --threads=1
--db=/mnt/pmem0dir-node0/dbbench
--write_buffer_size=4294967296
```

For NoveLSM and CacheKV, we can generate the workload file. The following script evaluates NoveLSM using the workload `YCSB-A`. The scripts to evaluate CacheKV with other YCSB workloads are similar by modifying the following script.

```
#!/bin/bash
$ ./ycsbc -db novelsm -path /mnt/pmem0dir-node0/
-threads 1 -P workloads/workloada.spec
```

REFERENCES

[1] "LevelDB," https://github.com/google/leveldb, 2021.

[2] "Introduction to Cache Allocation Technology in the Intel® Xeon® Processor E5 v4 Family," https://www.intel.com/content/www/us/en/developer/articles/technical/introduction-to-cache-allocation-technology.html?wapkw=intel%20cat, 2016.

[3] S. Kannan, N. Bhat, A. Gavrilovska, A. Arpaci-Dusseau, and R. Arpaci-Dusseau, "Redesigning LSMs for Non-Volatile Memory with NoveLSM," in *Proc. of USENIX ATC*, 2018.

[4] O. Kaiyrakhmet, S. Lee, B. Nam, S. H. Noh, and Y.-r. Choi, "SLM-DB: Single-Level Key-Value Store with Persistent Memory," in *Proc. of USENIX FAST*, 2019.

[5] B. F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, and R. Sears, "Benchmarking Cloud Serving Systems with YCSB," in *Proc. of ACM SOCC*, 2010.