Understanding Index Structures for Tiered Memory

RS3 LAB, EPFL

ROOPAM TANEJA

GOAL

- Benchmarking and understanding behaviour of index structures using PiBench (also used in OptiQL) for tiered memory (with a remote CXL-enabled node) under different tiering schemes : TPP+AutoNUMA and DAMON (HMSDK).
  - TPP and NUMAB-2 : NUMAB-2 (or Tiering-0.8) allows hot page promotion to fast NUMA node while TPP complements it with LRU-based demotion.
  - DAMON and HMSDK : DAMON is a lightweight and scalable memory monitoring subsystem. It also includes DAMOS, allowing users to conveniently perform system operations based on DAMON monitoring results. HMSDK was developed to particularly support CXL devices, one of the use cases being memory tiering based on DAMON.
- Aim to understand, how fast a small hot working set of an otherwise large index benchmark spanning both nodes, be promoted to the faster tier, and what improvement it brings in throughput.
- Ideally, we expect the hot working set to be quickly promoted and the no of promotions to stagnate.
- Glimpse of benchmarks configs :
  - 1B keys
  - Distribution : Selfsimilar (with skew), Uniform (no skew)
  - Mix of read-heavy and write-heavy workloads
  - Memory ratio achieved via Memeater

TIMELINE

1. Understanding applications (PiBench, GUPS) and running basic experiments to understand their behaviour.
2. Understanding tiering systems initially in a VM setup.
    1. Emulating separate memory tiers in QEMU for AutoNUMA promotions.
    2. Attempting CXL emulation in QEMU.
    3. Setting up HMSDK, then moving on to latest DAMON version.
    4. Exploring new DAMON features and looking for appropriate counters.
3. Benchmarking the applications on a real system. ← **Currently Here**

FINDINGS AND LEARNINGS

- Learnt about NUMA balancing and TPP and the associated counters relevant to us : 'pgpromote', 'pgdemote', 'numa_pages_migrated' etc.
- Learnt about the 'sysfs' and 'procfs' interfaces and how relevant stats can be retrieved through them.
- Got more comfortable with QEMU. Worked on a VM setup with DRAM NUMA nodes to run basic experiments.
- **ISSUE**: DRAM NUMA nodes are kept in same memory tier, making it impossible to observe effect of AutoNUMA page promotion/demotion.
  - Option 1: Emulate CXL memory in QEMU to achieve tiering.
  - Option 2: Emulate memory tiering via DRAM NUMA nodes only.

TWO OPTIONS

- Option 1: CXL emulation support in QEMU:
  - QEMU CXL support has been consistently improving. Spent some time going through documentation ([1], [2], [3]) and **was able to boot VM** with a **CXL-enabled CPU-less NUMA node**.
  - However, it doesn't support KVM yet. And any memory-intensive program touching the remote node runs **terribly slow.**
  - Also looking at this QEMU issue [4], perhaps as things currently stand, CXL with QEMU is suitable only for driver development and not benchmarking purposes. Have tried documenting all this in detail.
- Option 2: Memory tiering with DRAM NUMA nodes
  - Thanks to this kernel patch [5], specifying HMAT attributes in QEMU script [6] allows kernel to see different tiers for a CPU-less remote DRAM NUMA node (hence no local patch needed).
  - Hence able to emulate a **CPU-less node with custom latency and bandwidth** values (taken from [7]) in different tiers. Thus, able to **emulate effect of AutoNUMA promotions** and demotions successfully. [11]

HMSDK AND DAMON

- For memory tiering use case, HMSDK extended existing DAMOS framework by adding two new actions : migrate_hot and migrate cold. However, since they got upstreamed, it now only provides a Python script to generate YAML config for starting DAMON.
- However, since HMSDK's last update, DAMON itself has received updates [8] and now supports:
  - Applying separate DAMOS actions for different NUMA nodes from command line (making YAML generation redundant).
  - Introduced quota goals defined by used or free memory %age on a node, for self-tuned hotness/coldness thresholds for improved memory tiering.
  - Support auto-tuning of monitoring intervals as well as quotas based on defined goals. This enables to extract better performance from DAMON without requiring manual tuning.
  - Automatic updates to DAMON stats exposed via sysfs, allowing faster retrieval at time of access [9].

FINDINGS AND LEARNINGS

- Learnt about design of DAMON through its docs and slides, realised using latest DAMON instead of HMSDK would be better.
- Looked for counters comparable to ‘pgpromote’ of AutoNUMA, that allow to track no. of applied promotions/demotions at page granularity (for apples-to-apples comparison).
- Consulted with **SeongJae Park** (DAMON maintainer) for understanding available counters and better using new DAMON features [10].
- Received valuable suggestions: Comparing results across collaborative configs like combining DAMON with LRU-demotion or TPP with DAMON-promotion.
- Configured new DAMON version and extended existing scripts to monitor DAMON action statistics. Currently working on srv4 machine with kernel 6.17-rc7.

THANK YOU

REFERENCES

[1] https://www.qemu.org/docs/master/system/devices/cxl.html

[2]https://www.fujitsu.com/jp/documents/products/software/os/linux/catalog/Exploring_CXL_Memory_Configuration_and_Emulation.pdf

[3] Tu Tran et al. 2024. OMB-CXL: A Micro-Benchmark Suite for Evaluating MPI Communication Utilizing Compute Express Link Memory Devices. (PEARC '24). https://doi.org/10.1145/3626203.3670533

[4] https://gitlab.com/qemu-project/qemu/-/issues/3075

[5] https://lkml.org/lkml/2024/3/27/240

[6] https://www.qemu.org/docs/master/system/qemu-manpage.html

[7] Musa Unal et al. “Tolerate It if You Cannot Reduce It: Handling Latency in Tiered Memory”. (HotOS XX 2025)

[8] https://lore.kernel.org/all/20250420194030.75838-1-sj@kernel.org/

[9] https://lore.kernel.org/damon/20250717055448.56976-1-sj@kernel.org/

[10] https://github.com/damonitor/damo/issues/34

[11] https://github.com/RoopamTaneja/epfl-research-intern-roopam/blob/main/docs_created/qemu-cxl-hmsdk.pdf