Early Experience of Utilizing Persistent Memory for Database Bulk Loading

Tsuyoshi Ozawa *UTokyo*
Ryoji Kawamichi *UTokyo*
Yuto Hayamizu *UTokyo*
Kazuo Goda *UTokyo*
Masaru Kitsuregawa *UTokyo & NII*

1 Introduction

Persistent memory is a non-volatile and byte-addressable memory device, being much faster than NAND flash memory and offering larger memory capacity at a lower cost than DRAM. Since Intel's Optane launched, early papers have been reporting its performance properties from a variety of aspects [4–7]. This article reports our latest experience of utilizing persistent memory for bulk loading as a typical scenario of database record processing. Bulk loading is a process of inserting a bunch of record contents (often given in text files) into database stored in secondary storage. This is usually composed of multiple steps; a former step transfers its intermediate result to its next step by using memory space (i.e., DRAM) or secondary storage space (e.g., flash memory and magnetic disks.) Persistent memory is potentially a new promising device for this intermediate memory space. Experimentally understanding the performance of this use case helps future software design.

2 Experiment and analysis

A typical process of database bulk loading is (1) iteratively to read a block from given text files into sort buffer, interpret and sort records in the block, and stores the block into intermediate space; (2) to merge all the blocks from and store the records into a data structure (e.g., leaf nodes of B+ tree) in database; and (3) to arrange the data structure (e.g., internal nodes of B+ tree.) We compared four configuration scenarios: sDiD (using DRAM for the sort buffer and the intermediate space,) sPiD (using persistent memory for the sort buffer and DRAM for the intermediate space,) sDiP (using DRAM for the sort buffer and persistent memory for the intermediate space,) and sPiP (using persistent memory for both.)

The experiment was performed on a four-socket machine with four Intel Xeon 6252 processors (each having 24 cores at 2.1 GHz) with 768 GB DRAM, 2048 GB persistent memory (Intel Optane 2,666 MHz,) and four flash SSDs, running Ubuntu Linux 20.04. We utilized our home-grown multi-threaded database engine [2] and TPC-H (scale factor: 100) [1] datasets. The dataset was initially stored as text files in btrfs [3] space striped on the flash SSDs and finally inserted into the database organized in another block space striped on the space SSDs.

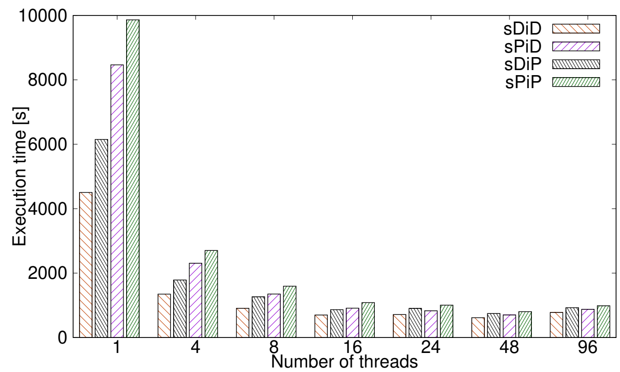

Figure 1: Execution time spent for bulk loading with DRAM and persistent memory.

Figure 1 summarizes the total execution time spent for bulk loading with different configurations. We observed 2.2 times longer execution time on persistent memory than DRAM when the bulk loading was single-threaded. This observation seems different from other micro-benchmarking reports [6], which reported moderate performance degradation. Our current analysis is that the processing of variable-length records tends to exhibit this phenomenon in particular, but further study is necessary. By contrast, when the bulk loading was more intensively multi-threaded, the performance gap became smaller, reaching 1.3 times at 96 threads.

Persistent memory is potentially promising, but still emerging. We would like to further investigate the reality of its performance properties for a variety of data-intensive use cases.

References

[1] Transaction Processing Performance Council. TPC-H benchmark specification. http://www.tpc.org/tpch/, Last accessed on Jan 21, 2021.

[2] Kazuo Goda, Yuto Hayamizu, Hiroyuki Yamada, and Masaru Kitsuregawa. Out-of-order Execution of Database Queries. *Proc. VLDB Endow.*, 13(12):3489–3501, 2020.

[3] Ohad Rodeh, Josef Bacik, and Chris Mason. BTRFS: The Linux B-Tree Filesystem. *ACM TOS*, 9(3):9:1–9:32, 2013.

[4] Xingda Wei, Xiating Xie, Rong Chen, Haibo Chen, and Binyu Zang. Characterizing and Optimizing Remote Persistent Memory with RDMA and NVM. In *Proc. USENIX ATC 2021*, pages 523–536, 2021.

[5] Michèle Weiland, Holger Brunst, Tiago Quintino, Nick Johnson, Olivier Iffrig, Simon D. Smart, Christian Herold, Antonino Bonanni, Adrian Jackson, and Mark Parsons. An early evaluation of Intel's Optane DC persistent memory module and its impact on high-performance scientific applications. In *Proc. SC 2019*, pages 76:1–76:19, 2019.

[6] Jian Yang, Juno Kim, Morteza Hoseinzadeh, Joseph Izraelevitz, and Steven Swanson. An Empirical Guide to the Behavior and Use of Scalable Persistent Memory. In *Proc. USENIX FAST 2020*, pages 169–182, 2020.

[7] Jinfeng Yang, Bingzhe Li, and David J. Lilja. Exploring Performance Characteristics of the Optane 3D Xpoint Storage Technology. *ACM TOMPECS*, 5(1):4:1–4:28, 2020.