Accelerating the Data Deduplication Performance with GPU in Hybrid Storage Systems

Prince Hamandawana[†], Awais Khan, Changgyu Lee, Sungyong Park, Youngjae Kim
Sogang University, Seoul, Republic of Korea

I. Introduction

The explosive increase in data production has made the problem of data storage space worse. Data compression has been actively adopted in backup storage as a way to improve storage space. However, the compression not only failed to completely remove the replica across the cluster, but also had difficulties applying it immediately to in-line mode due to high compression overhead. As a result, the community began applying deduplication techniques that divide the data into smaller chunks, calculate the fingerprint of the chunks, identify the fingerprint in the lookup table, and then insert new data or maintain references to existing data. This paper explores the inline deduplication of the SSD layer in a hybrid scale-out tiered storage system for cost and performance efficiency that employs a high-speed SSD as the cache layer for low-speed HDDs. We configure Ceph [1] as tiered storage and build inline deduplication on the SSD layer. Deduplication has a significant impact on storage performance and cost in a tiered architecture. First, we investigate the performance degradation of Ceph [1], after enabling data deduplication across the cluster. We observed an overhead of 27% due to fingerprint computation during an investigation that seriously degraded the overall cluster performance. To reduce fingerprinting overhead in inline deduplication, we propose a technique that uses a GPU accelerator to optimize fingerprint computation to improve cluster performance. By using our Approach we obtain a 50% dedup performance improvement.

II. GPU-Accelerated Data Deduplication

We have structured the Ceph cluster in two tiers by configuring the SSD layer for caching writes and the underlying HDD storage for data storage. In particular, we developed an inline data deduplication framework for the SSD tier. Using the CRUSH algorithm in Ceph, all object writes are mapped to the OSDs of the SSD layer, chunked using a fixed-size chunking scheme, fingerprinted using the SHA-1 algorithm, and deduplication checks performed before I/O disk submission. We implemented an object redirection scheme that maps different objects (already stored copies in storage) with the same hash value, but with different names. An object can refer to other objects if they have same hash value, to achieve maximum space saving. We adopted a database partitioning approach for efficient dedup metadata management, to reduce query contention at individual nodes. We achieved this by maintaining partitioned database shards on each object storage server (OSS) on the SSD tier. Consequently, bulk metadata requests are spread to different nodes at the SSD tier for dedup query whilst maintaining negligible amount of metadata space on the SSDs. But we found that inline deduplication caused I/O performance degradation due to fingerprint computation. As nowadays, state of the art compute nodes are equipped with high-performance GPUs. We get the motivation to leverage their massive parallelism of computation to accelerate fingerprinting. GPU usage, however raises challenges on how best we can manage GPU memory in bursty incoming requests as allocation is mostly predetermined at compile time without full knowledge of memory request patterns at run-time. These static predefined GPU memory allocations are heavily programmer dependent and they can cause a huge buffer queue increase because it has to wait for a GPU memory release. In addition, GPU utilization may be lowered when using the GPU for fingerprinting because the data is transferred to the GPU, the fingerprint is calculated, the calculated fingerprint is sent back to the CPU memory, and the GPU buffer is cleared before another transfer begins. It can also trigger unnecessary swapping when GPU memory is full. By using GPU memory-aware allocation which keeps track of and allocates according to available GPU memory, we can achieve higher GPU memory utilization. It will also prevent blind GPU memory allocations, which leads to unnecessary swapping operations. Moreover, overlapping data transfers and GPU executions, we can further optimize the fingerprinting overhead to achieve higher dedup efficiency.

[†]Mr. Prince is currently a graduate student in Ajou University, Suwon, Republic of Korea.

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No.2015-0-00590)

III. Evaluation

We conducted a preliminary evaluation of our approach in Ceph Jewel v10.2.5. Our testbed setup consists of 4 OSS servers equipped with Intel Xeon CPU ES-2640 v3 processor, 32GB memory and 12GB NVIDIA Tesla K80 GPU. We provisioned 2xSSDs for cache and 4xHDDs in storage tier. A separate server was used as the Ceph client. We configured rados block device (RBD) and used fio benchmark tool to generate random writes to the RBD of total 1 GiB size with a dedup ratio of 100%. Figure 1 depicts the dedup I/O processing time overhead from client request to disk submission. We observed fingerprinting constituted 77% of total dedup overheads (chunking+fingerprinting+dedup query) in the baseline approach. The second bar shows that GPU fingerprinting achieves 65% reduction in fingerprint computation overhead and consequently, 50% dedup overhead reduction. We plan to further optimize our GPU approach to reduce the fingerprint overhead near to negligible through implementing a GPU memory-aware transfer mechanism in conjunction with overlapping CPU-GPU data transfers with GPU executions.

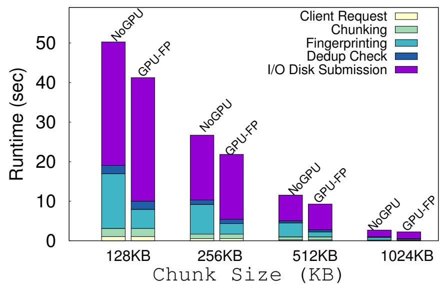

Fig. 1: Deduplication Performance with CPU vs GPU.

References

[1] S. A. Weil, S. A. Brandt, E. L. Miller, D. D. E. Long, and C. Maltzahn, "Ceph: A Scalable, High-performance Distributed File System," in *Proceedings of the 7th Symposium on Operating Systems Design and Implementation*, ser. OSDI, 2006.