KSII The 17th International Conference on Internet (ICONI) 2025.
Copyright ⓒ 2025 KSII

Accelerating I/O Performance for AI Frameworks on DAOS

Sooyoung Lim, Taehyun Yoo, Jaegi Son, and Dongmin Kim*
Medical IT Convergence Research Center, Korea Electronics Technology Institute (KETI)
Seongnam-si, Gyeonggi-do, Republic of Korea
[e-mail: slim@keti.re.kr, yootommy0113@gmail.com, jgson@keti.re.kr, dmkim@keti.re.kr]
*Corresponding author: Dongmin Kim

Abstract

Recent advances in next-generation hardware and storage technologies have driven the adoption of the Distributed Asynchronous Object Storage (DAOS) for AI and high-performance computing (HPC) workloads. However, integrating DAOS with AI frameworks poses performance limitations, due to excessive data movement between compute and storage layers. To address this issue, we explore a design direction that minimizes data transfer overheads by enabling computation closer to storage, thereby enhancing overall I/O efficiency.

Keywords: Distributed Asynchronous Object Storage (DAOS), File systems, High-performance computing (HPC), Near-data processing, Storage systems

1. Introduction

The growing demand for large-scale AI and high-performance computing (HPC) workloads has accelerated the development of storage systems optimized for next-generation hardware [1]. Among these, Distributed Asynchronous Object Storage (DAOS) has emerged as a promising architecture, leveraging persistent memory (PMem) and high-speed interconnects such as RDMA and CXL [2].

However, our preliminary experiments using DAOS with AI frameworks reveal notable inefficiencies in data access paths. Specifically, repeated data transfers between the CPU and DAOS storage engines trigger excessive I/O overheads in preprocessing workloads. These effect accumulate into a significant bottleneck in end-to-end training pipelines. To mitigate this, we investigate architectural directions that minimize inter-layer data movement and enable more efficient processing near storage. Our ongoing work explores lightweight mechanisms that leverage the DAOS's extensible design to achieve improved I/O efficiency without modifying its core architecture.

2. Background & Motivation

Modern AI and HPC workloads require intensive data preprocessing, where data movement betwe-en storage and CPU memory often dominates execution time. In DAOS, POSIX compatibility through DFUSE incurs user-kernel context switches and memory copies, causing significant overhead in I/O-intensive workloads.

Table. 1 presents the comparison of DFUSE and a POSIX filesystem, using image and video preprocessing pipelines in PyTorch and

(Support note) This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (No. RS-2025-02215256), Development of High-Speed Parallel File System Technology for AI Semiconductor Cloud

Tensorflow. For lightweight image dataset such as cifar-10 [3], the performance gap was minimal, largely due to caching effects and small I/O sizes.

File	Image		Video	
System	PT	TF	PT	TF
DFUSE	6.35	4.12	26.08	49.59
POSIX	6.51	2.39	12.33	15.05

Table. 1. Total execution time (seconds) of image and video preprocessing on DFUSE and POSIX (PT as PyTorch, TF as TensorFlow, respectively).

However, Tensorflow's frequent file open-close operations and large continuous I/O for video datasets (e.g., UCF101 [4]) exposed severe overheads, leading to up to 3.3x longer preprocessing times in DFUSE. These results highlight that DFUSE's POSIX layer, while ensuring compatibility, becomes a major source of inefficiency for data-intensive AI frameworks.

3. Discussion

To mitigate I/O bottlenecks observed in DAOS, Fig. 1 displays the redesigned I/O stack that integrates two complementary strategies: zero-copy I/O and in-storage computing (ISC).

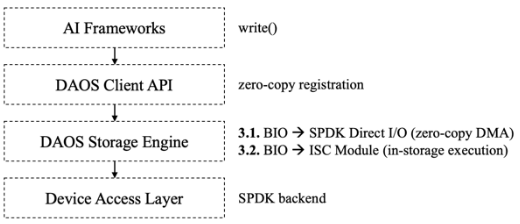

Fig. 1. The overview of redesigned DAOS I/O stack

3.1 Zero-Copy I/O

To reduce data movement overhead between user space and storage, we introduce a zero-copy I/O mechanism in the DAOS client API. When AI frameworks issue a *write()* request, the data buffer is directly registered with the DAOS Client API through a zero-copy registration process. Once registered, data is transferred via SPDK Direct I/O, which performs DMA-based data movement directly between the user buffer and the storage. This mechanism eliminates redundant *memcpy* operations between the user and kernel space that occur in DFUSE-based communication.

3.2 In-Storage Computing (ISC)

Beyond host-side optimization, we extend the BIO (Block I/O) layer of the DAOS Storage Engine to support ISC. Instead of repeatedly transferring large datasets from storage to the host for preprocessing, lightweight operations are executed directly within the storage node. The ISC module is invoked through the *bio_isc_submit()* interface, which allows the DAOS Storage Engine to dispatch computation tasks to the SPDK-managed device context. This design leverages the DAOS's modular service architecture, enabling near-data execution without modifying existing APIs. As a result, computation is brought closer to data, reducing I/O latency and improving overall system efficiency under data-intensive AI frameworks.

4. Conclusion

This work identified system overheads in the DFUSE layer as the dominant source of I/O inefficiency when running AI workloads on DAOS. Future work will involve implementing a prototype of zero-copy I/O and ISC mechanisms to validate their potential for accelerating data-intensive AI pipelines on object storage systems.

References

[1] J. Soumagne et al., "Accelerating HDF5 I/O for Exascale Using DAOS," *IEEE Transactions on Parallel and Distributed Systems*, vol.33, no.4, pp.903-914, 2022.

[2] M. Hennecke, M. Matsuda, and M. Nakao, "Evaluating DAOS Storage on ARM64 Clients," in *Proc. of the HPC Asia 2023 Workshops (HPCAsia)*, 2023.

[3] A. Krizhevsky and G. Hinton, "Convolutional deep belief networks on cifar-10," *Unpublished manuscript*, vol.40, no.7, pp.1-9, 2010.

[4] K. Soomro, A. R. Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild," *arXiv:1212.0402*, 2012.