Korea Electronics Technology Institute

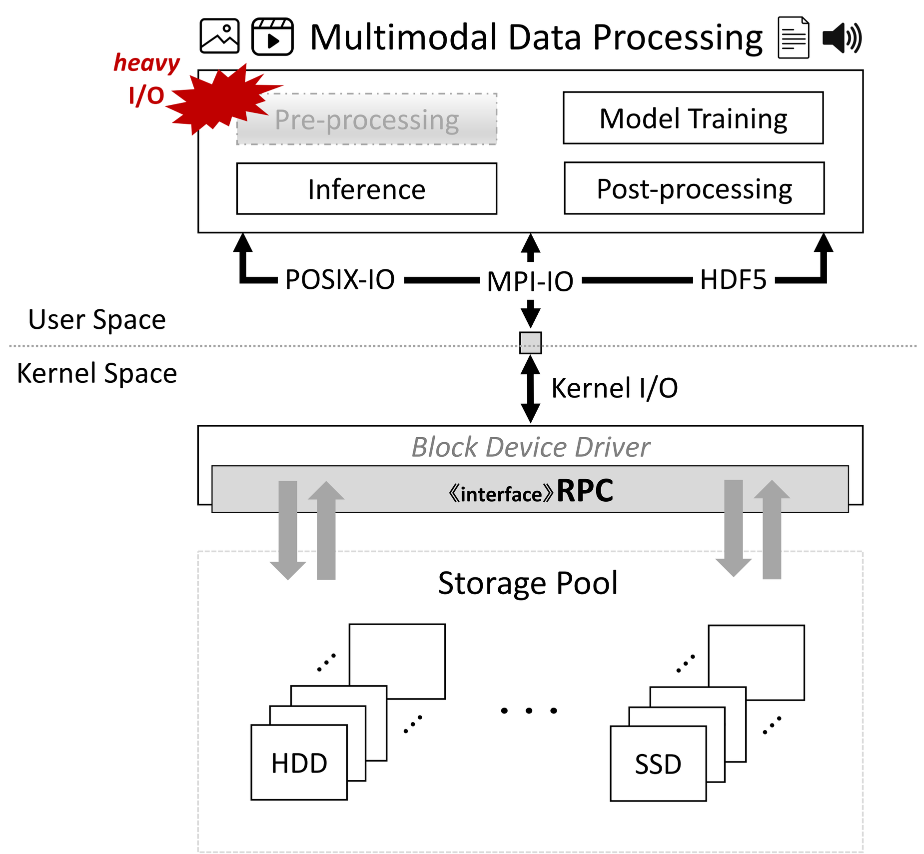
Accelerating I/O Performance for AI Frameworks on DAOS

Aurora-FS

Sooyoung Lim, Taehyun Yoo, Jaegi Son, and Dongmin Kim
slim@keti.re.kr, yootommy0113@gmail.com, jgson@keti.re.kr, dmkim@keti.re.kr

Korea Electronics Technology Institute (KETI)
Medical IT Convergence Research Center

Overview



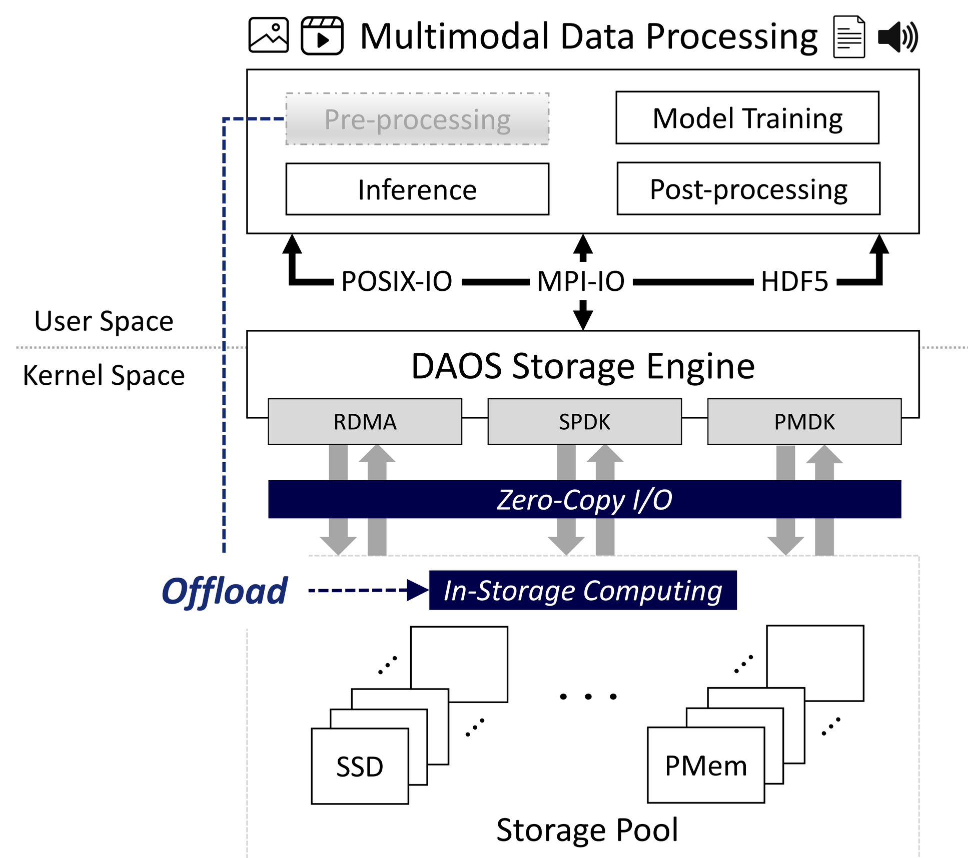
<Legacy Distributed Storage System>

As-is

- ❑ **Heavy I/O overhead**
  - Modern AI and HPC workloads require intensive data preprocessing.
- ❑ **Excessive redundant copies and context switches**
  - The default DAOS I/O path traverses between user space and kernel space via block drivers and RPC.

To-be

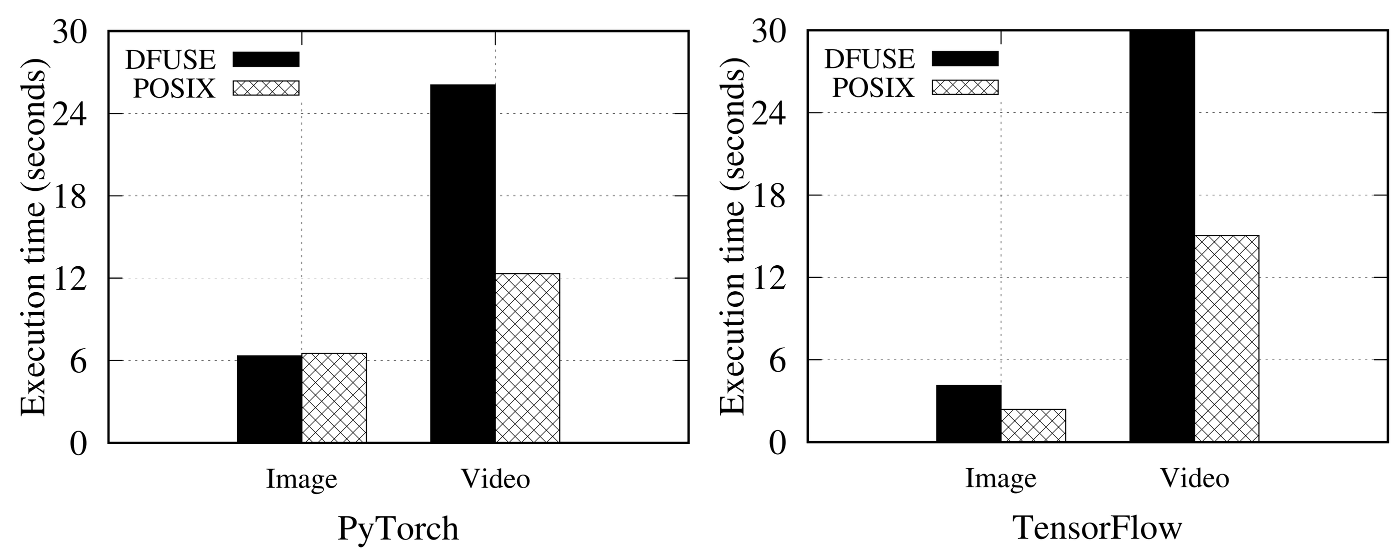
- ❑ **Two solutions for accelerating I/O performance**
  - To reduce unnecessary data movement: ***Zero-copy I/O***
  - To move computation closer to data: ***In-Storage Computing***



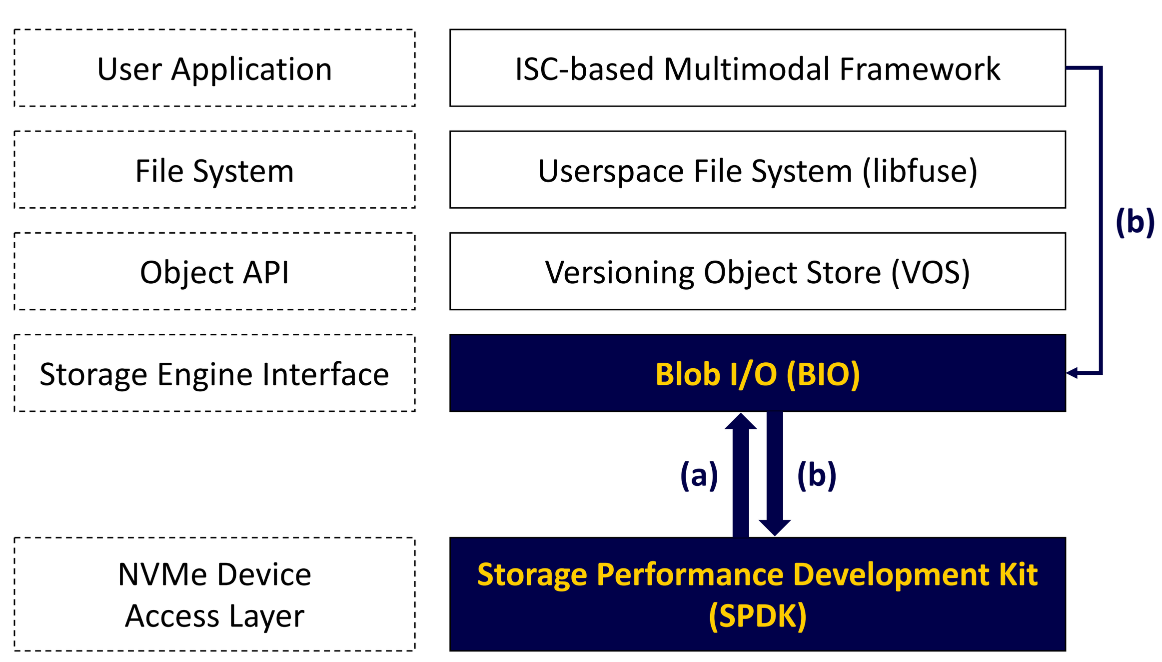
<Proposed Distributed Storage System>

Motivation

- ❑ **Setup** – Evaluated PyTorch and TensorFlow preprocessing pipelines on DFUSE (DAOS FUSE) vs. POSIX filesystem (xfs) using image (CIFAR-10) and video (UCF101) datasets
- ❑ **Result** – Up to ***3.3x slower*** preprocessing on DFUSE for video pipelines
- ❑ **Cause** – DFUSE introduces frequent user-kernel transitions, redundant copies, and heavy POSIX operations
- ❑ **Insight** – Bottleneck stems from ***data movement*** between compute and storage!



Proposed Design



< Redesigned DAOS I/O Stack>

Redesigned DAOS I/O Path

(a) Zero-Copy I/O

- **BIO → SPDK**: Submits DMA-ready user buffers directly to SPDK's NVMe channels
- **BIO ← SPDK**: Returns completion events and signals the DMA transfer to devices

(b) In-Storage Computing

- **User → BIO**: Sends ISC requests to BIO by specifying the target object extents
- **BIO → SPDK**: Issues targeted block-access requests to SPDK

(a) Zero-Copy I/O

Limitation

- The software I/O path (user ↔ DFUSE ↔ kernel ↔ device) dominates latency.

Design Proposal

- Direct data transfer without kernel

Design Elements

- ❑ **DAOS storage engines**
  - **BIO**: Acts as the zero-copy entry point that receives DMA-ready buffers from the DAOS client
- ❑ **SPDK modules**
  - **Blobstore**: Manages physically contiguous I/O regions that can be directly mapped from user buffers
  - **Bdev**: Provides DMA-capable I/O channels and NVMe queue pair management
  - **SPDK memory module**: Handles hugepage-based physical address to make user buffers DMA-ready

(b) In-Storage Computing

Limitation

- Preprocessing workloads repeatedly transfer large volumes of I/O from compute to storage.

Design Proposal

- Offloading preprocessing pipelines to the storage layer

Design Elements

- ❑ **DAOS storage engines**
  - **VOS**: Manages versioned object metadata and transactional consistency-guaranteed epochs
  - **BIO**: Hosts ISC execution and enables SPDK-driven I/O
- ❑ **SPDK modules**
  - **Blobstore**: Executes ISC with block-granular access to NVMe -backed objects
  - **Bdev**: Provides I/O channel control with block-level extent
- ❑ **HW emulation**
  - **QEMU**: Provides a *controllable* NVMe device environment