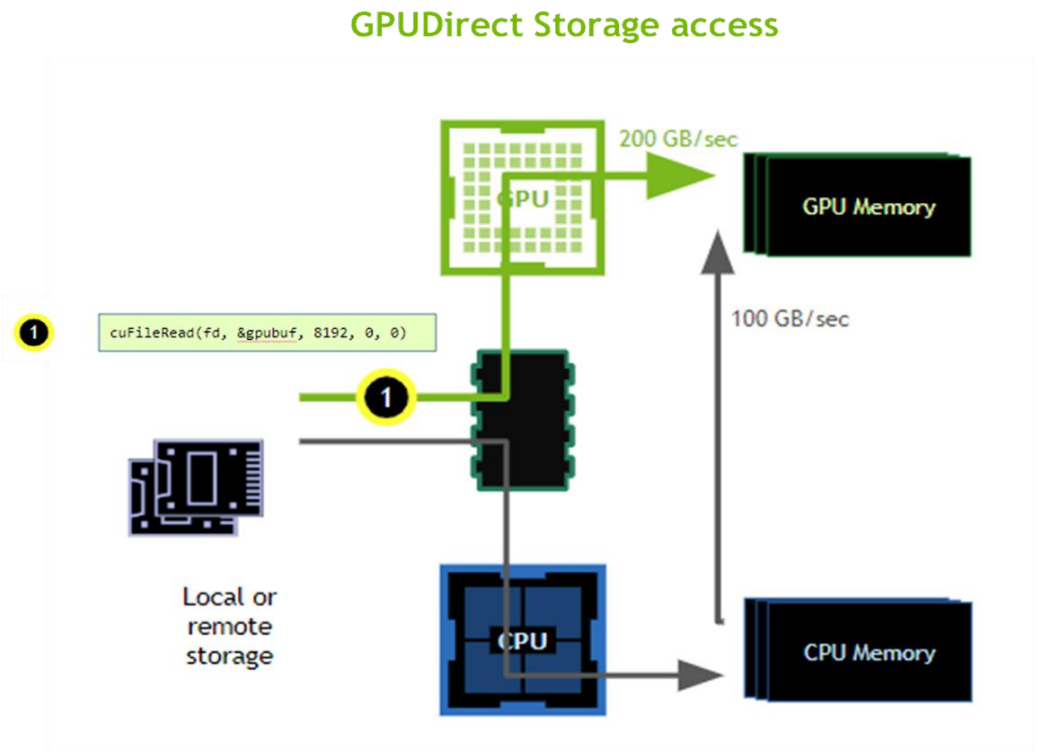
GPUDirect™ Storage: Accelerated File APIs for CUDA Applications

Presenter: Kiran Kumar Modukuri, NVIDIA

NVIDIA® Magnum IO GPUDirect® Storage

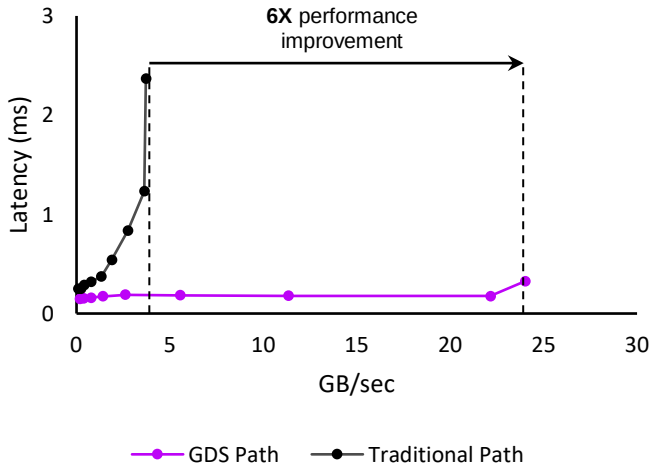


- NVIDIA Magnum IO GPUDirect Storage (GDS) is designed to accelerate data transfers between GPU memory and remote or local storage by reducing CPU overheads
- GDS enables a direct data path for direct memory access (DMA) transfers between GPU memory and storage, which avoids a bounce buffer through the CPU page cache.
- The GDS direct path increases system bandwidth, decreases the CPU utilization load and IO latency by avoiding cudaMemcpy calls
- The GDS feature is exposed using cuFile APIs, integrated into NVIDIA CUDA Toolkit and supported by multiple NVIDIA frameworks including RAPIDS and DALI.
- Unified APIs for host memory and GPU memory

GDS vs Traditional IO Path - Busy System

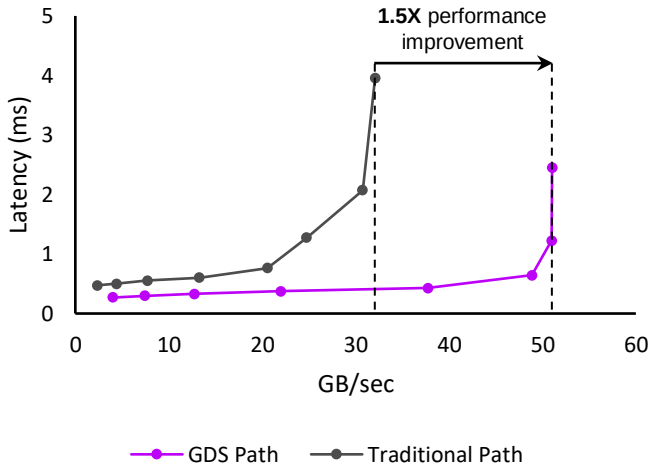
NVIDIA DGX A100, 8x Micron 9400 PRO NVMe, CPU + Memory at 80%

GDSIO[1] 4KB 2,048 Threads



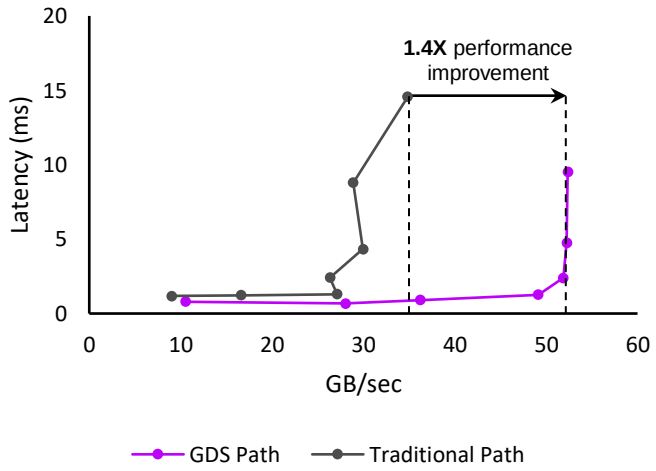
✓ On a busy system[2], GDS increased 4KB transfer size performance by up to **6X** and improved response time by **3.8X** compared to the traditional IO path

GDSIO[1] 128KB 1,024 Threads



✓ GDS improved 128KB transfer size performance by up to **1.5X** and improved 128KB transfer size response time by **3X**

GDSIO[1] 1MB 512 Threads



✓ GDS improved 1024KB transfer size performance by up to **1.4X** and improved responsive time by **1.5X**

1) NVIDIA GDSIO Tool

NVIDIA GDSIO tool is a synthetic IO benchmarking tool that uses cufile APIs for IO and it can be found in the /usr/local/cuda/gds/tools directory. In this test we use GDSIO transfer types 0 (Storage -> GPU) and 2 (Storage -> CPU -> GPU) to perform randread IO operations for different block sizes (4k, 128k, 1M) with a file size of 5GB

2) Google Stressful Application Test

Stressapptest generates randomized traffic to memory from processor and I/O, with the intent of creating a realistic high load. The "Busy" system testes were run with 80% load.

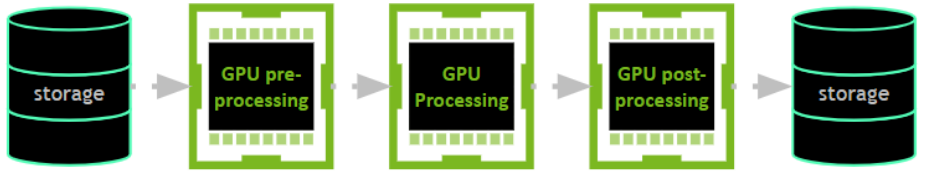
NVIDIA GPUDirect Storage – Use Cases

Proven success for many use cases and application profiles

Flash Memory Summit

IO performance boost

Pre-Stack Time Migration	1.19x
Reverse Time Migration	1.3x-6.3x
HPC Visualization	8x
DeepCAM Inference	4.6x
NVTabular	1.4x
Genotyping	3x-6x
Seismic Simulation	2.5x
RAPIDS/cuCIM	11x
HDF5	5x
KvikIO/Zarr	2.9x
Cosmoflow DL Training	1.2x

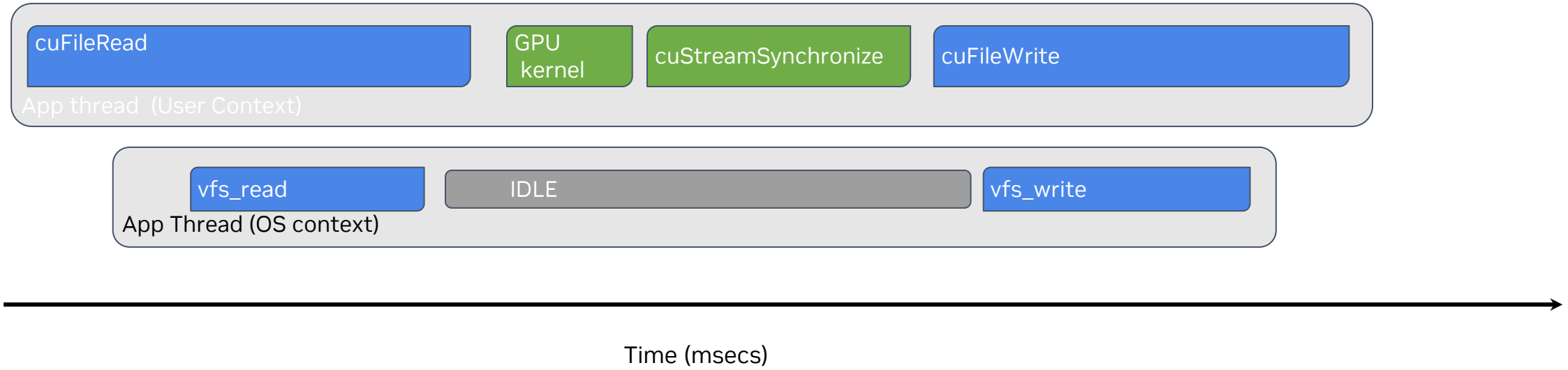


BENEFITS	REQUIREMENTS
Up to 3x lower CPU utilization	O_DIRECT only
File IO in CUDA	Use CTK
1.2x-8x bandwidth boost	System Topology (PCIe Switch)
Faster IO to GPU memory	GPUDirect / Vidmem Only, NVME or RDMA based FS

NVIDIA

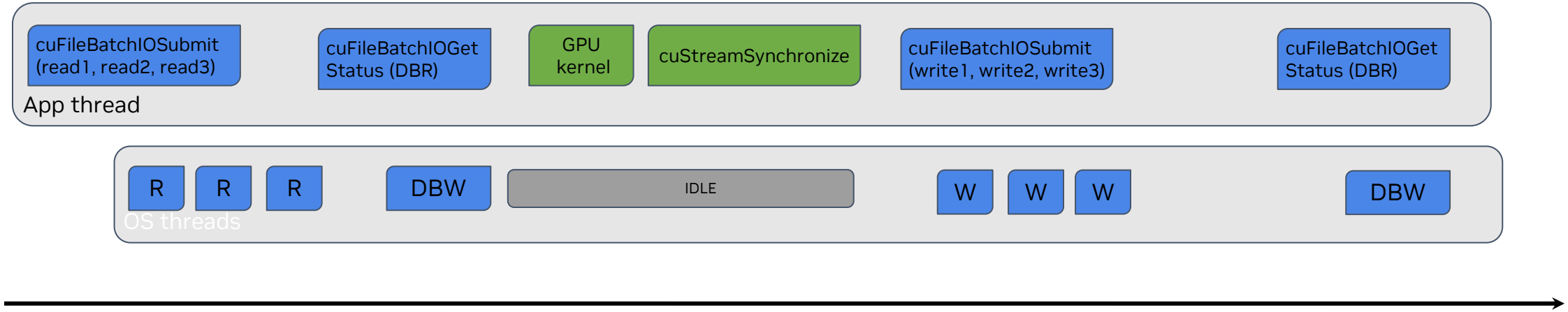
cuFile Synchronous IO Flow

Synchronous submission and completion



cuFile Batch IO Flow

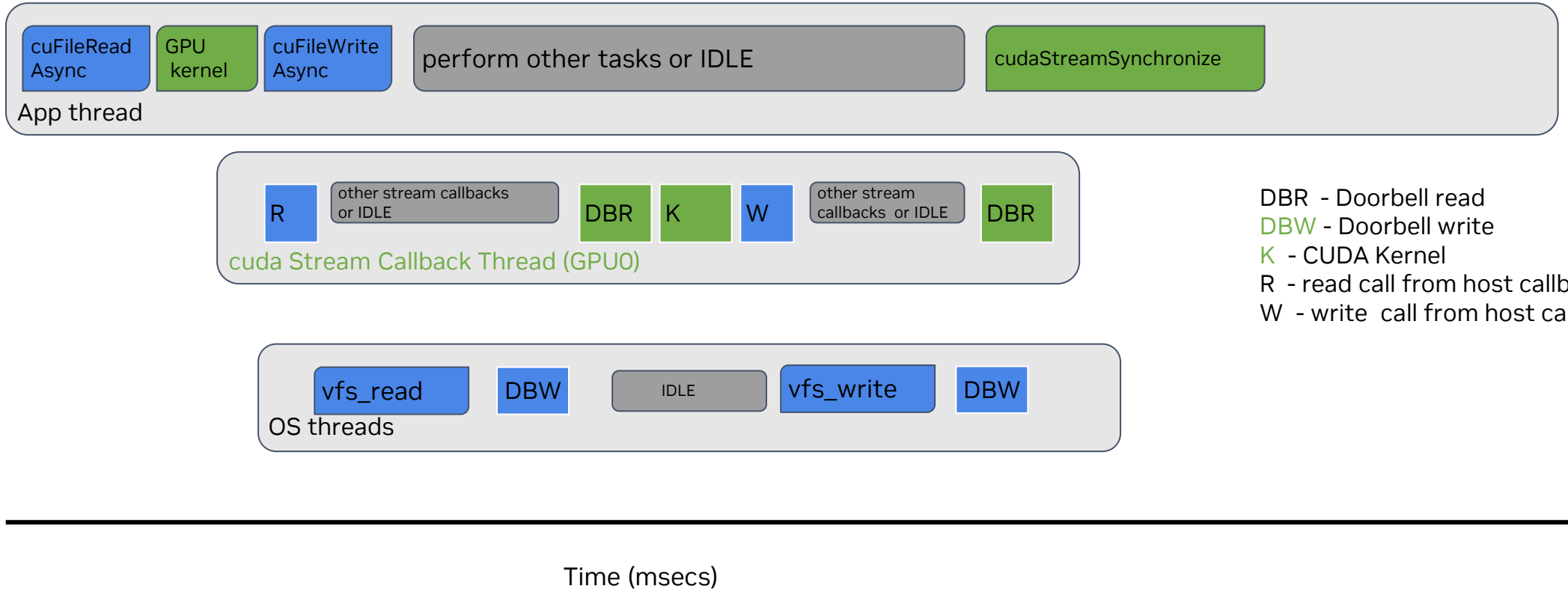
Synchronous submission and Asynchronous completion



DBR - Doorbell read
DBW - Doorbell write
R - VFS read
W - VFS write

cuFile Stream IO Flow

Asynchronous submission and Asynchronous completion

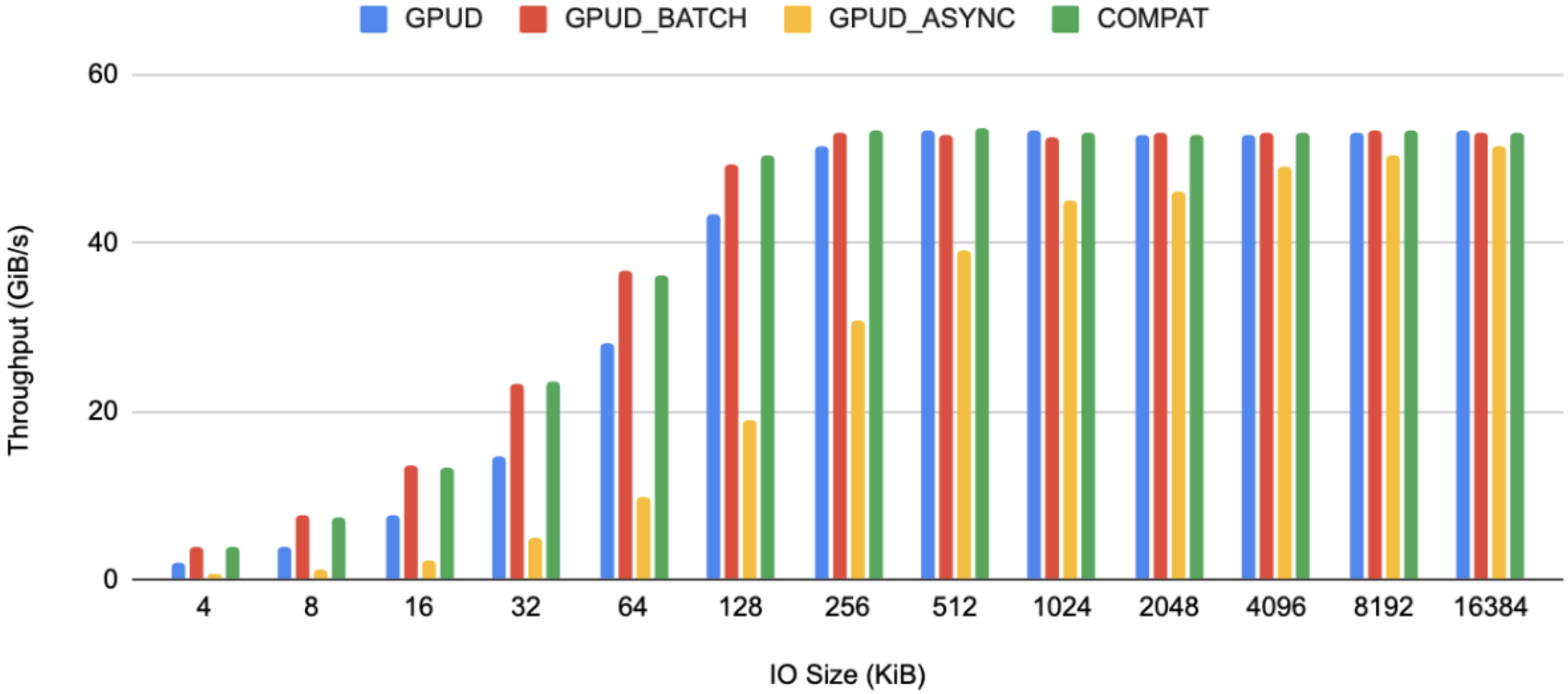


DBR - Doorbell read
DBW - Doorbell write
K - CUDA Kernel
R - read call from host callback
W - write call from host callback

GDS Performance

Read throughput comparison with different modes of cuFile IO

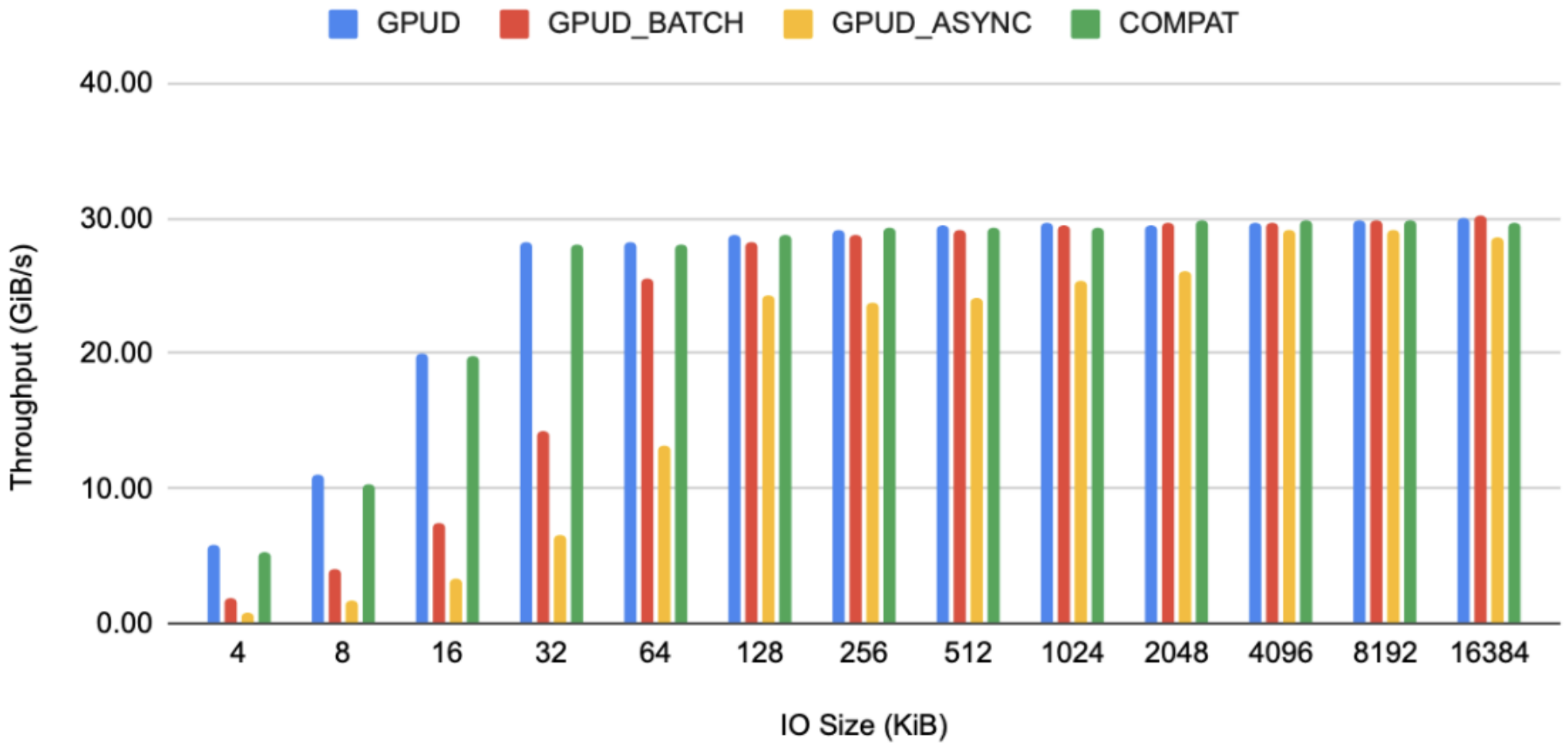
DGX A100 with 8x Local NVMes, Batch (8TxB16) vs. Threaded (128T), GDS 1.7.x



GDS Performance

Write throughput comparison with different modes of cuFile IO

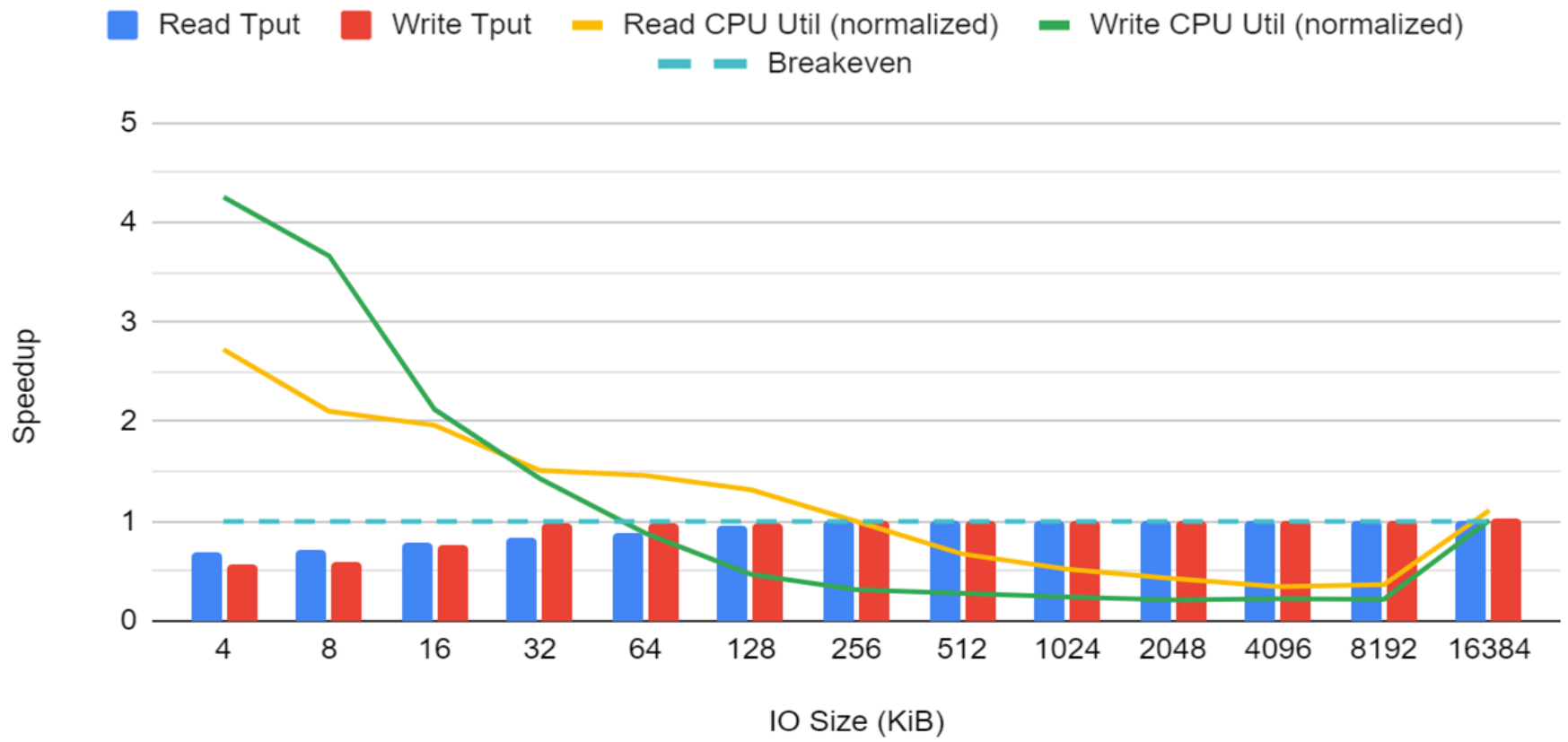
DGX A100 with 8x Local NVMes, Batch (8TxB16) vs. Threaded (128T), GDS 1.7.x



cuFile Batch APIs vs cuFile Sync APIs

CPU utilization between Batch and Sync APIs

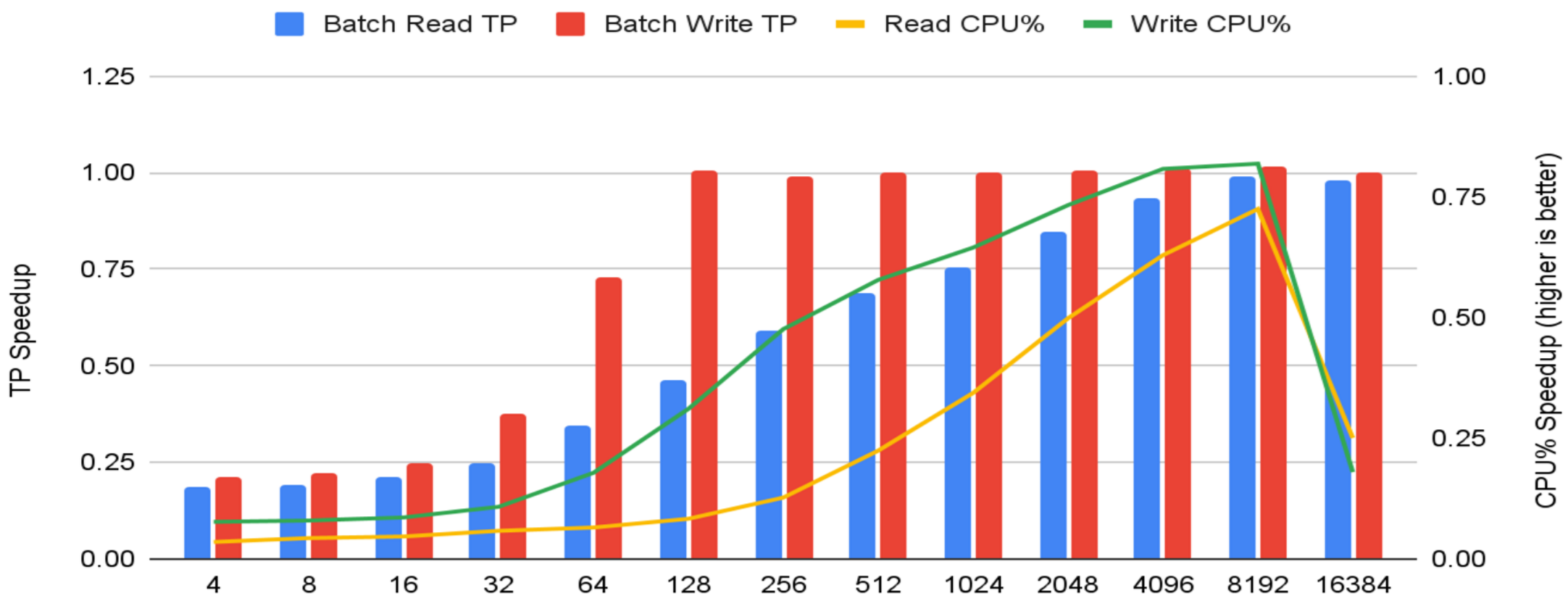
DGX A100 with 8x Local NVMes, Batch (8TxB16) vs. Threaded (128T), GDS 1.7.x



cuFile APIs with CUDA Streams

Throughput and CPU utilization between Stream and Sync APIs

DGX A100 with 8x Local NVMes, Batch (8TxB16) vs. Threaded (128T), GDS 1.7.x



cuFile API Use Cases

Mode	IO Behavior	Use case	Pros/Cons
cuFileRead cuFileWrite **cuFile Thread Pool enabled**	Synchronous	Single threaded application using standard file system calls for single large file and large buffers(>16MB)	**Pros** simple to use **Cons** Does not help for multiple files or multiple buffers
cuFileRead cuFileWrite	Synchronous	• Multithreaded application using standard file system calls for multiple files, buffers • Application has thread pools for IO pipeline	**Pros** • simple to use • lower submission latency • work good for medium size IO request 64K and above. **Cons** • scalability limited by number of CPU threads used. • higher CPU cost for smaller IO sizes(4K-64K)
cuFileBatchIOSetup cuFileBatchIOSubmit cuFileBatchGetStatus	Synchronous submission Asynchronous completion	• Single threaded application using standard filesystem calls needs to perform IO for multiple non-contiguous file offsets, sizes and GPU buffers. • Each IO request is small < 64KB • Has ability to track completion of IOs asynchronously or wait in same thread.	**Pros** • lower average completion latency • lower CPU cost because of batch submission **Cons** • more complex to code, submit followed by polling for completion of the batch • higher submission latency. can be reduced by partial submission
cuFileReadAsync cuFileWriteAsync	Asynchronous submission Asynchronous completion	Single threaded application using standard filesystem calls needs to perform IO for multiple non-contiguous file offsets, sizes and GPU buffers. IO sizes , buffer data is dependent on prior async CUDA work.	**Pros** works with CUDA semantics. lower submission latency **Cons** higher execution latency for IO size (<1MB) needs multiple streams to submit IO in parallel

GPUDirect Storage Partner Ecosystem

The Partner ecosystem is expanding in response to customer demands

Flash Memory Summit

ALL GA PARTNERS

NVIDIA GPUDirect Storage integrated solution in production.

ddn · DELL EMC · Hewlett Packard Enterprise · HITACHI Inspire the Next · IBM · KIOXIA · LIQID · Micron · NetApp · SAMSUNG · ScaleFlux · SUPERMICRO · VAST · WEKA

FILE SYSTEM PARTNERS

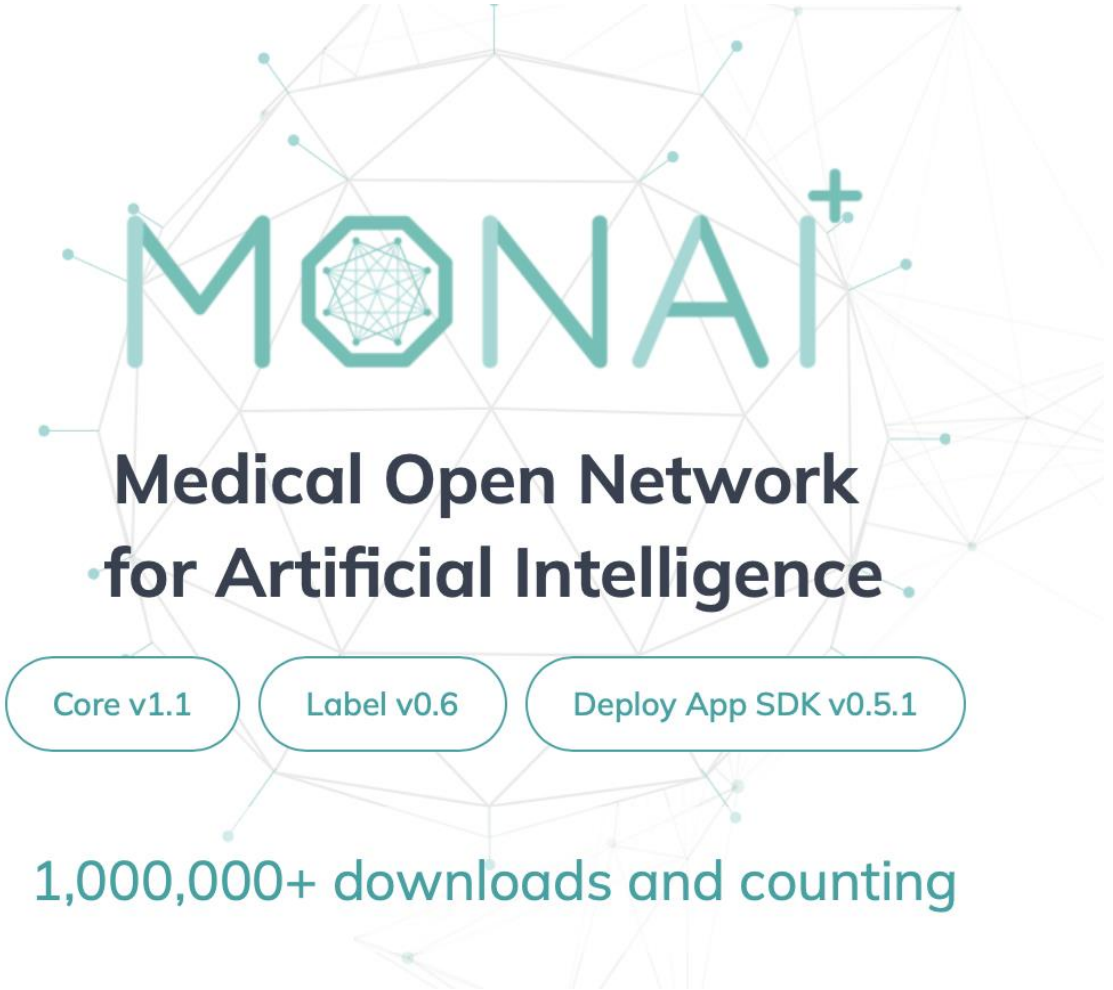
Partner company	Partner Product	GDS Version
WekaIO	WekaFS 3.13	1.0
DellEMC	PowerScale 9.2	1.0
Hitachi Vantara	HCSF	1.0
IBM	Spectrum Scale 5.1.2	1.1 and higher
DDN	EXAScaler 6.0*	1.1 and higher
VAST	Universal Storage 4.1	1.1 and higher
NetApp	ONTAP 9.10.1	1.0 and higher
NetApp ThinkParQ System Fabrics Works	BeeGFS Tech Preview	1.1 and higher
IBM	Spectrum Scale 5.1.5	1.5 and higher
HPE	HPE Ezmeral 5.5	1.5 and higher
HPE Cray ClusterStor	Neo 4.2 and newer	1.0 and higher

*Open source Lustre 2.15 supports GPUDirect Storage

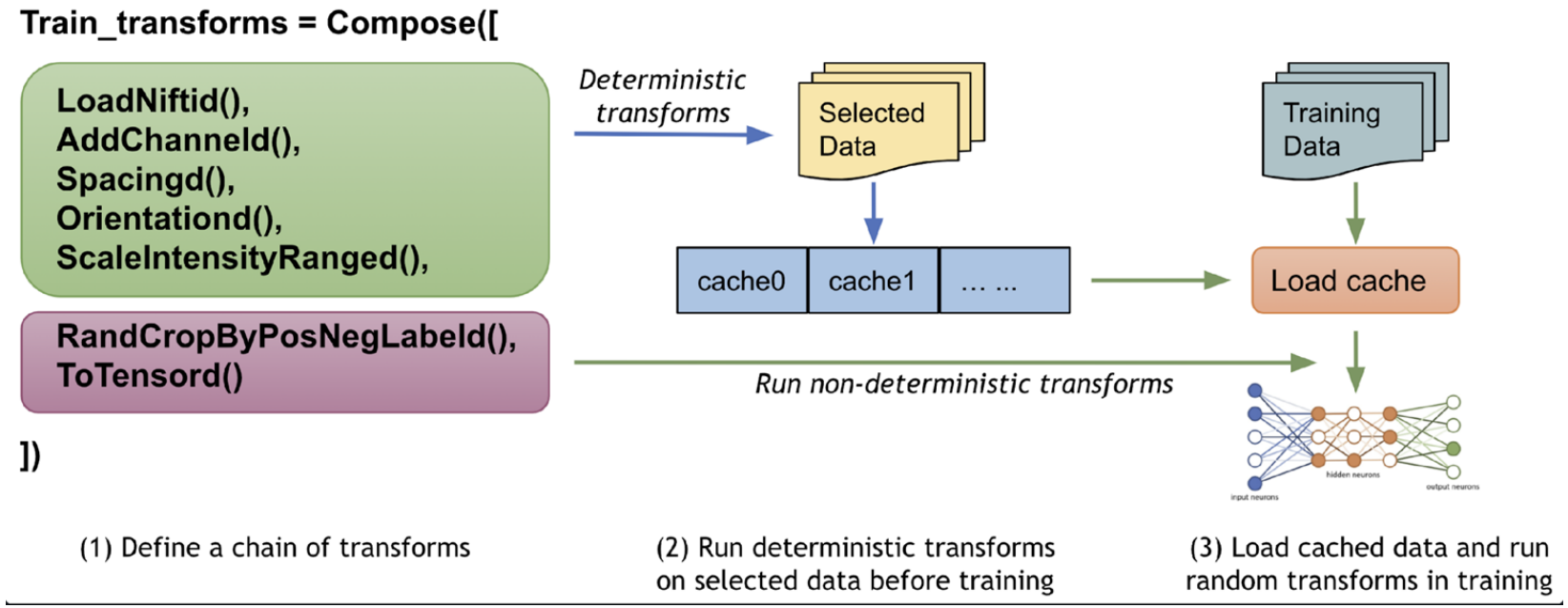
NVIDIA

GDS with MONAI

- MONAI is an open-source built on top of PyTorch for accelerating research and clinical collaboration in Medical Imaging
- Brain tumor 3D segmentation training with MONAI using BraTS dataset
- Used Persistent Dataset and GDSDataset
  - Stores pre-computed values to efficiently manage larger than memory dictionary format data,
  - operates on transforms for specific fields.
  - Results from the non-random transform components are computed and cached
  - when first used, results are stored in a *cache_dir* for rapid retrieval on subsequent uses.
  - GDS integration done with **KvikIO** Python library for cuFile and no changes to core MONAI library.
- Single PCIe Gen3 NVMe and Tesla V100S-PCIE-32GB

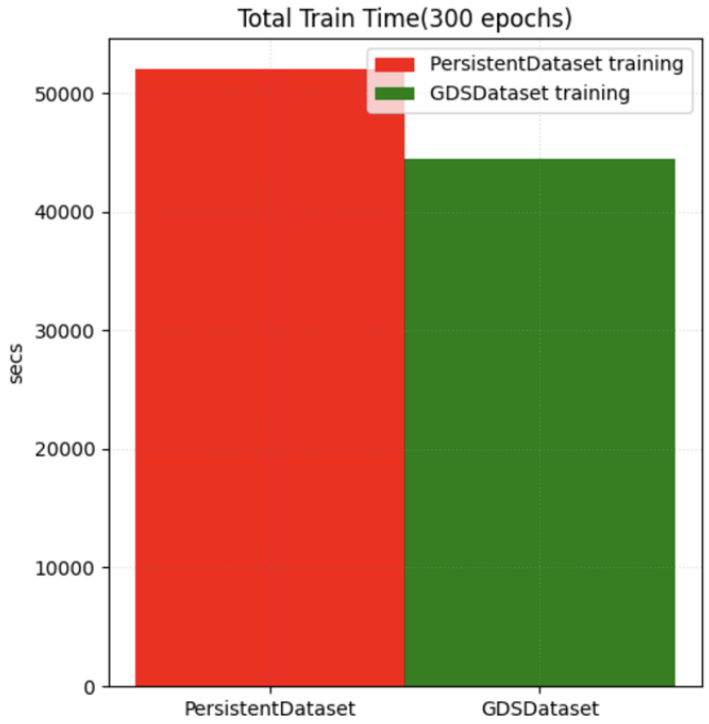


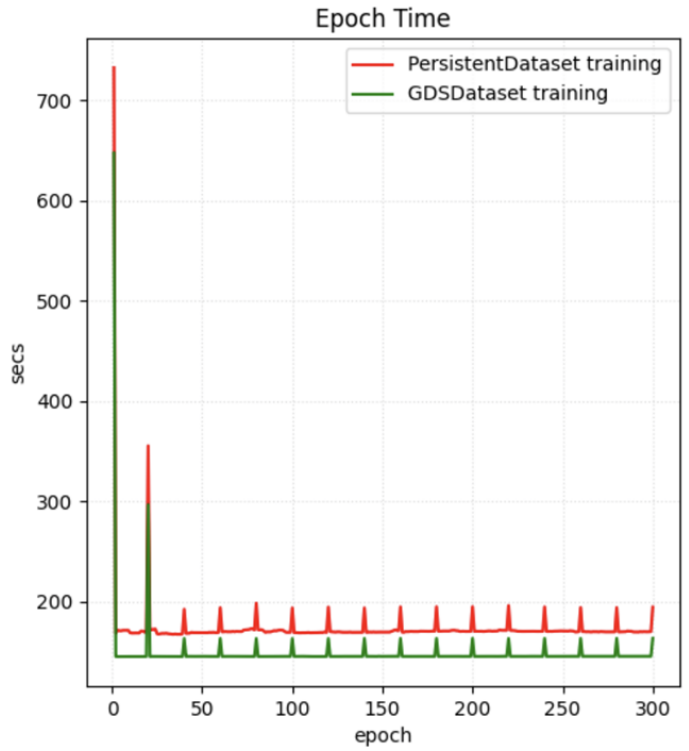
MONAI Dataset Pipeline with Persistent Caching



GDS vs Persistent Caching Dataset

- Dataset is read from compressed dataset in epoch 0, transformed and persistently cached for deterministic transforms in NVMe disk.
- Data is read from the NVMe cache dataset for all epochs (1-300).
- With GDS and cuFile APIs, the model achieved **1.1x** speedup in total time compared to standard persistent disk caching mechanism.

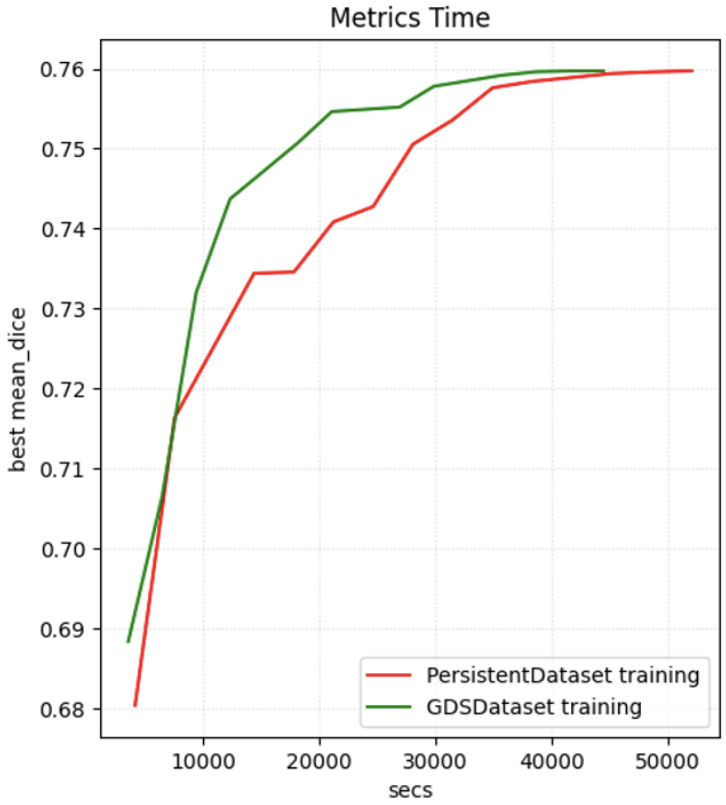


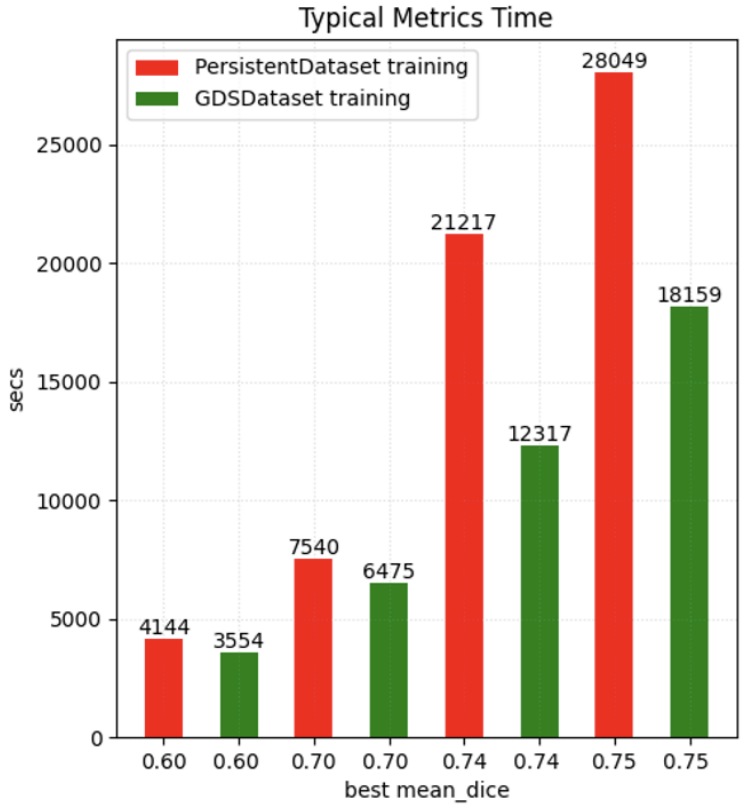


NVIDIA

GDS vs Persistent Caching Dataset

- With GDS, the model achieves 0.75 mean_dice in 18159 secs compared to 28049 secs. This is a **1.54x** speedup.





Call to Action

Flash Memory Summit

Check it out

- Documentation - https://docs.nvidia.com/gpudirect-storage/index.html

Try it

- Get from CUDA - https://developer.nvidia.com/cuda-downloads
- MagnumIO repo - https://github.com/NVIDIA/MagnumIO/tree/main/gds
- NVIDIA Frameworks - DALI, cuCIM, SPARK-RAPIDS, RAPIDS-cuDF, MONAI

Bring more use cases

NVIDIA