Oak Ridge National Laboratory

HVAC: Eliminating I/O bottleneck for Large-Scale Deep Learning Applications

Awais Khan,
Arnab K. Paul,
Chris Zimmer,
Sarp Oral,
Sajal Dash,
Scott Atchley,
Feiyi Wang

ORNL is managed by UT-Battelle LLC
for the US Department of Energy

U.S. DEPARTMENT OF ENERGY

Introduction

- Deep Learning (DL) is an emerging technology gaining dominance
  - to solve critical problems and predicting trends in
    - ***Computer vision, Speech recognition, Natural language processing, Scientific and Climate sciences***
- Efficient training of Deep Neural Network (DNN)
  - Requires ***large volumes of input datasets*** and ***high-speed compute accelerators***
- Therefore, DL applications are becoming an increasingly important workload on supercomputers
  - ***Summit and Frontier (Upcoming No. 1 Supercomputer)***

Introduction

- Large-scale HPC parallel file systems provide massive capacity
  - To store huge volumes ***(TBs ~ PBs) of DL datasets***
- Each compute nodes on Summit supercomputer
  - Offer exceptional computing capabilities to fulfill the DL application needs, e.g., ***Six NVIDIA Tesla V100 GPUS per Node***
- Despite, to efficiently ***Run and Scale DL Applications*** to leverage state-of-the-art ***HPC supercomputers*** remains a challenge
  - Running scientific DL application such as ***DeepCAM at scale on 1,024 compute nodes of Summit is limited due to slow I/O***

Motivation

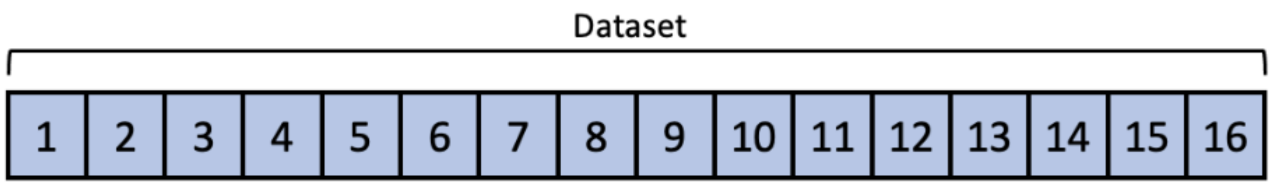
- ***I/O optimization for DL Applications*** is non-trivial challenge to solve on ***Large-scale Supercomputers***
  - ***Dataset characteristics, DL Access patterns, and I/O properties***

- ***Dataset characteristics of DL applications***
  - ***ImageNet-1K:*** 1.28 Million files in 1000 categories
  - ***ImageNet-21K:*** 11 Million images (average size: 163KB)
  - ***OpenImages:*** 9 Million images

Motivation

- ***Access patterns of DL Applications***
  - ***Stochastics Gradient Descent (SGD)*** randomly shuffles dataset after each epoch

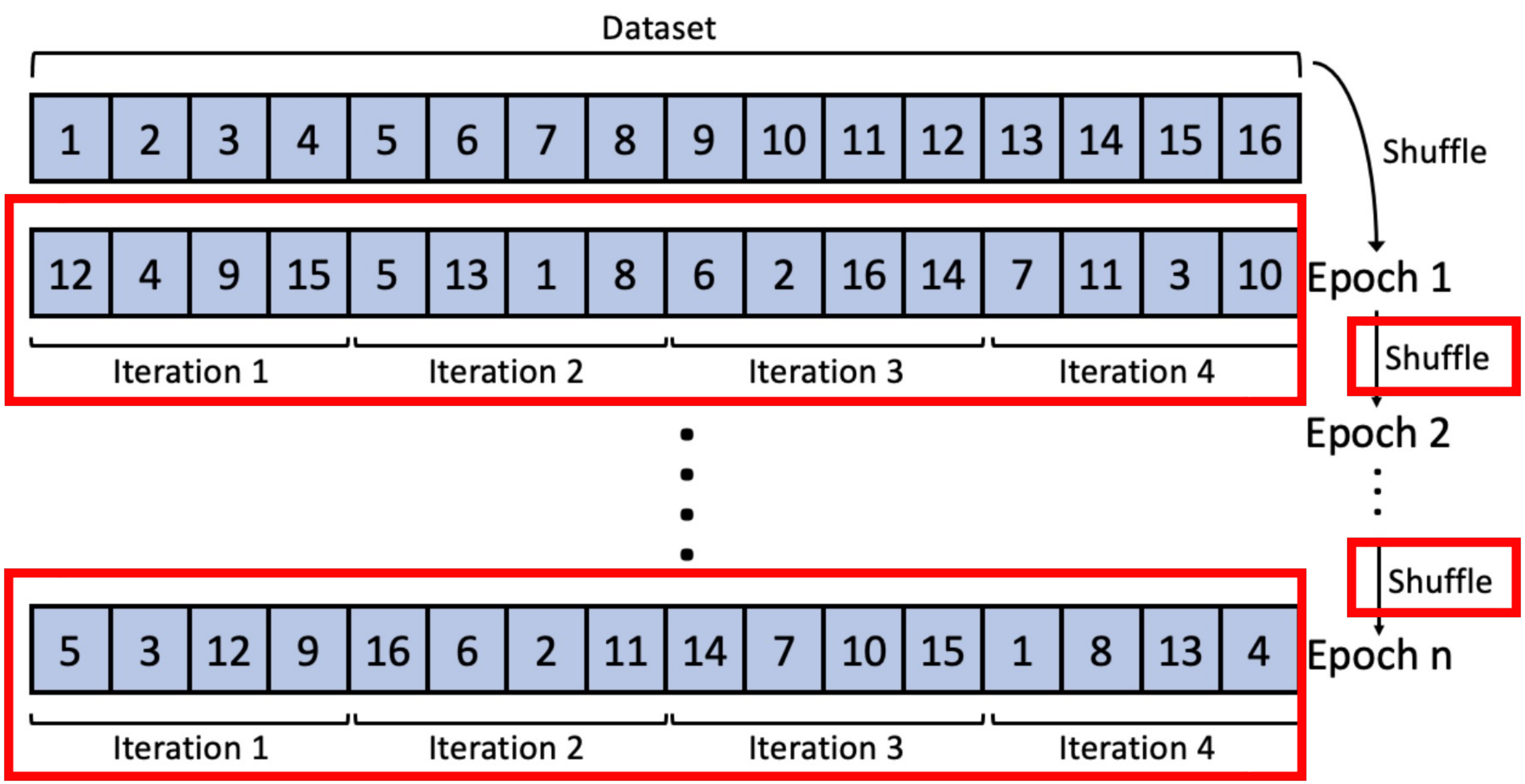
An epoch is defined as the unit of time a DL application takes to touch the whole dataset at least once…



Dataset: 16 Files
Batch Size: 4
Iterations in an Epoch: 4
No. of Epochs: n ***(depends on achieving desired % of training accuracy from the DL model)***

Motivation

- ***Access patterns of DL Applications***
  - ***Stochastics Gradient Descent (SGD)*** randomly shuffles dataset after each epoch

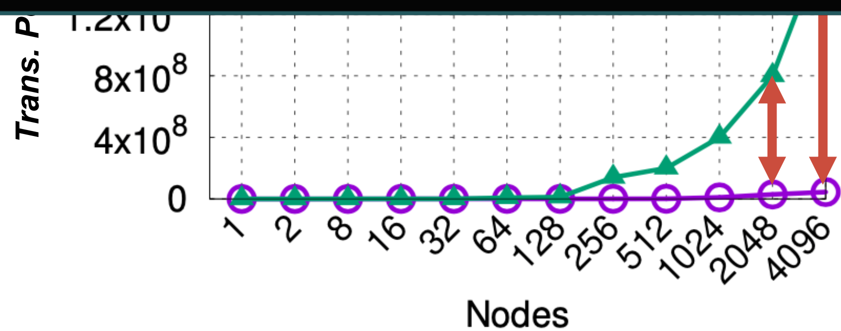


Motivation

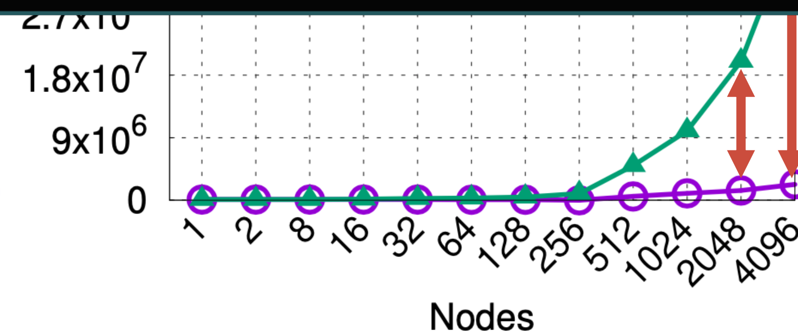
- ***HPC I/O subsystem*** is not built to deal with the manner DL frameworks ***Read Data at Scale***
  - Easily saturated with a large number of concurrent and random accesses on small files

Small Files: PFS’s metadata performance is an impediment to large-scale DL training jobs

Large Files: Shifts to bandwidth constraints of PFS, i.e., Read Performance 22.5TB/s vs 2.5 TB/s



32KB <Open-Read-Close>



8MB <Open-Read-Close>

Motivation

- ***File Access Characteristics of DL Applications***
  - Training DNN is ***Read-intensive***
  - ***Read-only Access*** to the complete dataset in ***One Epoch***

Opportunity to exploit node-local or near-node local storage on compute nodes and solve I/O Scalability limitations by layering a Caching System

- ***Key I/O Properties***
  - High degree of ***Shareability*** in ***DL I/O***
  - ***Shareability*** *of I/O makes DL jobs* ***Cache-Friendly***
  - ***Cache Adversarial*** if complete dataset do not fit in cache
  - I/O is substitutable

Challenges and Limitations

- ***Limited to small-scale testbeds or simulation-based scalability***
  - Really do not scale well on HPC supercomputers, e.g., over 1000 nodes
- ***Lack Portability and require modifications***
  - Existing studies mostly require changes to application, input pipelines or underlying file systems
- ***Metadata bottlenecks for large no. of small files***
  - Millions of requests touching metadata server for both small and large files

Challenges and Limitations

- ***Lack of Generality***
  - Tailored to meet specific application/dataset requirements, often hardware dependent

- ***No support for POSIX interface***
  - Lack support for POSIX interface and are not suitable choice for scientific DL applications running on supercomputer

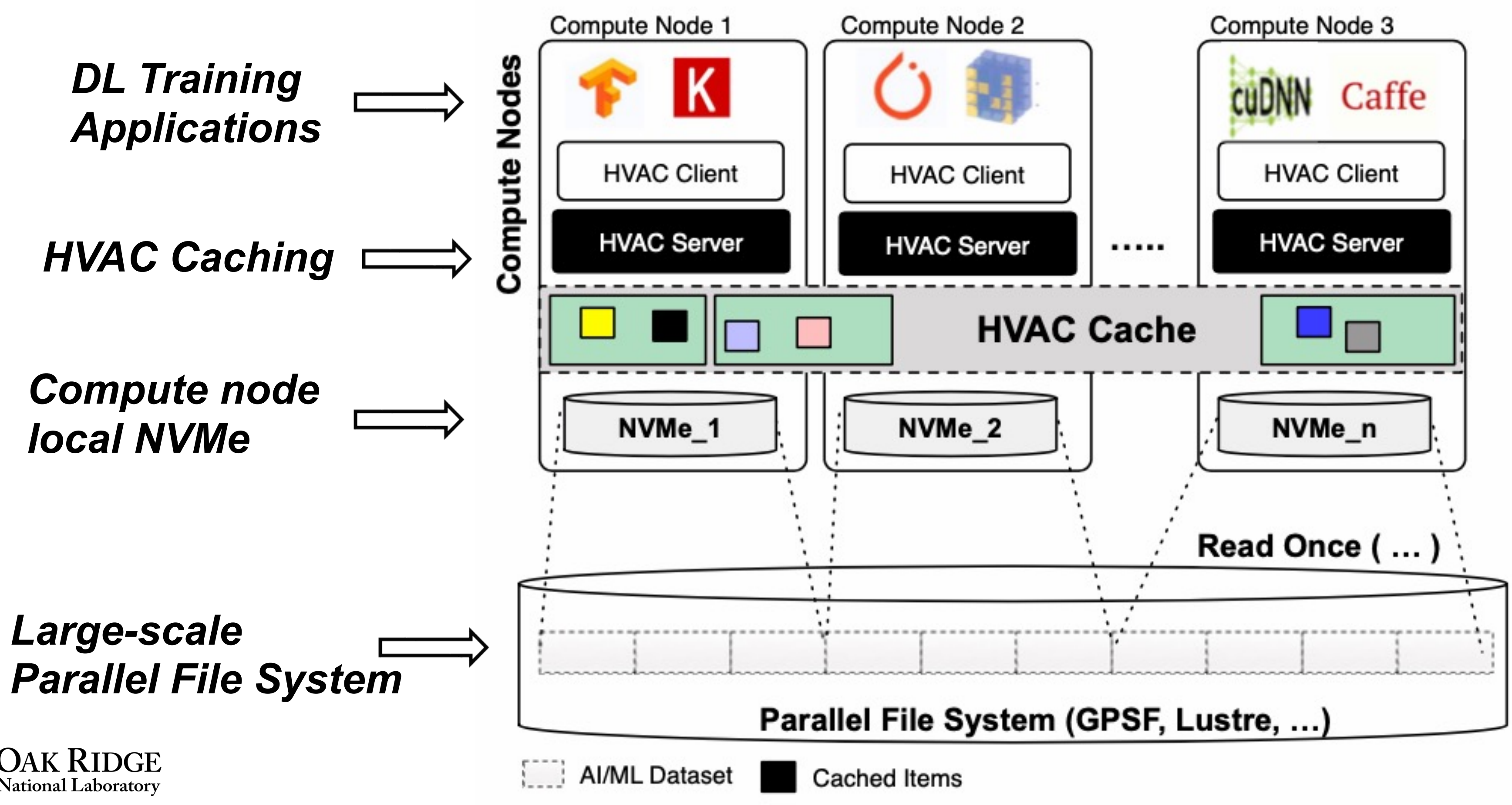
High-Velocity AI Cache

- ***High-Velocity AI Cache (HVAC)***, a transparent read-only caching layer, for large-scale supercomputers

To Scale on thousands of compute nodes on leadership-class supercomputer such as Summit and Frontier without modifying DL applications and additional metadata bottlenecks and storage overhead

HVAC

- ***Architectural overview on Summit***
  - A Client-Server Library intended to accelerate I/O accesses for DL applications that utilize read-only data with a high re-read rate.



HVAC

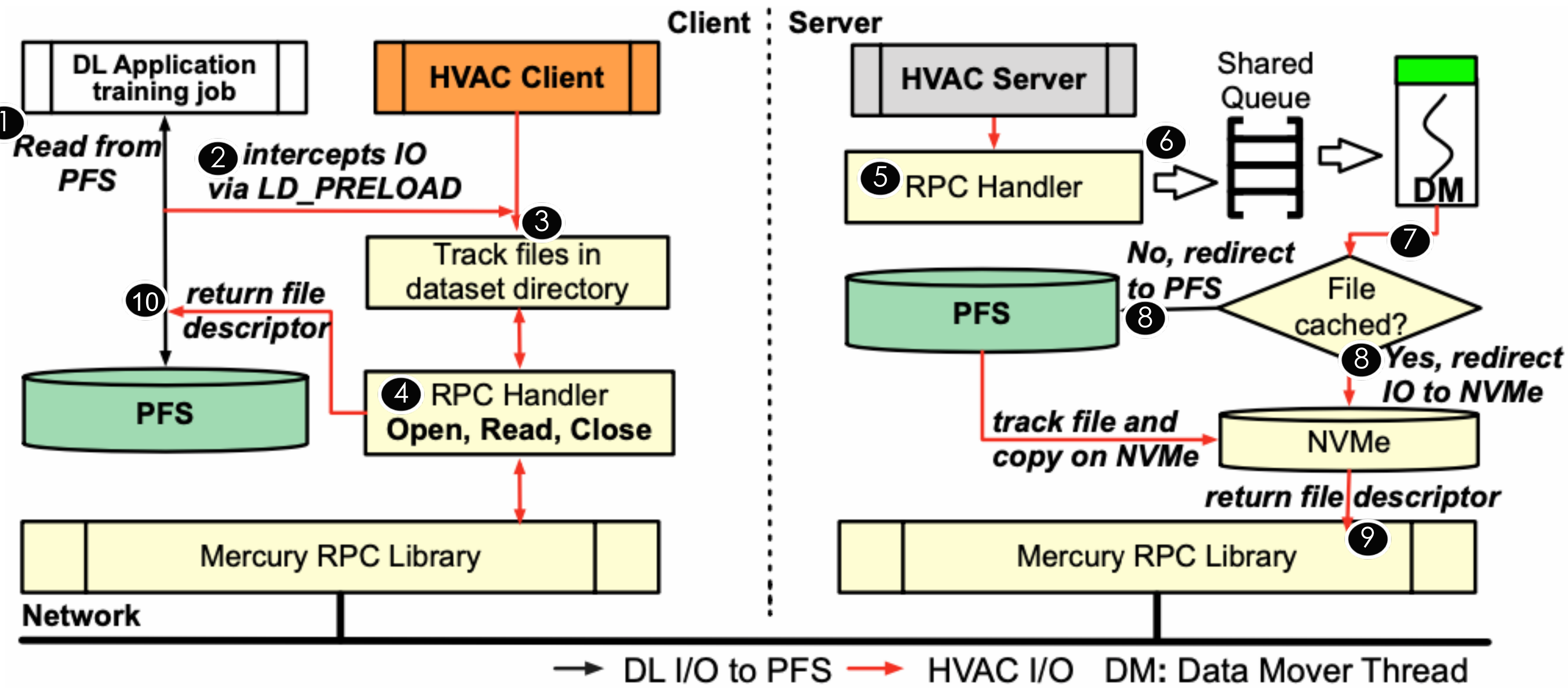
- ***Generic for diverse deployments models***
  - Agnostic and can be adopted on Node-local NVMe SSDs, near-node local or Rack-local storages
- ***Portable and POSIX support***
  - Intercepts the <open-read-close> file I/Os via LD_PRELOAD
    - No changes to application required
- ***No Metadata Bottleneck***
  - Employs distributed hashing to calculate location of cached contents across the nodes
- ***No repeated re-reads from GPFS required***

HVAC

- ***HVAC Server***
  - builds an aggregate cache layer atop of node-local fast storage
    - Purpose is to process forwarded file system operations from HVAC clients and to retrieve data from PFS or cache
    - Can be co-located with HVAC client and servers
- ***HVAC Client***
  - Consists of an interception interface that catches relevant file system calls to PFS and redirects to the respective HVAC server
- ***HVAC communication Framework***
  - uses High-Performance Mercury library
    - Remote communications and bulk data transfers over the InfiniBand network

Proposed Solution

- ***I/O service flow in HVAC***



OAK RIDGE National Laboratory

Evaluation

- ***Summit Testbed***
  - *Compute node specifications*

Attribute	Description
Supercomputer	Summit
CPU	2 x IBM POWER9 22Cores 3.07GHz
GPU	6 x NVIDIA Tesla Volta (V100)
Memory Capacity	512 GB DDR4
Node-local Storage	1.6 TB Samsung NVMe SSD with XFS
Network Interconnect Family	Dual-rail Mellanox EDR Infiniband

- *Datasets*
  - ***ImageNet21K*** – 11M images (1.1TB, avg. 163KB size)
  - ***CosmoUniverse*** – 600K TRF files (1.3 TB, avg. 16MB size)

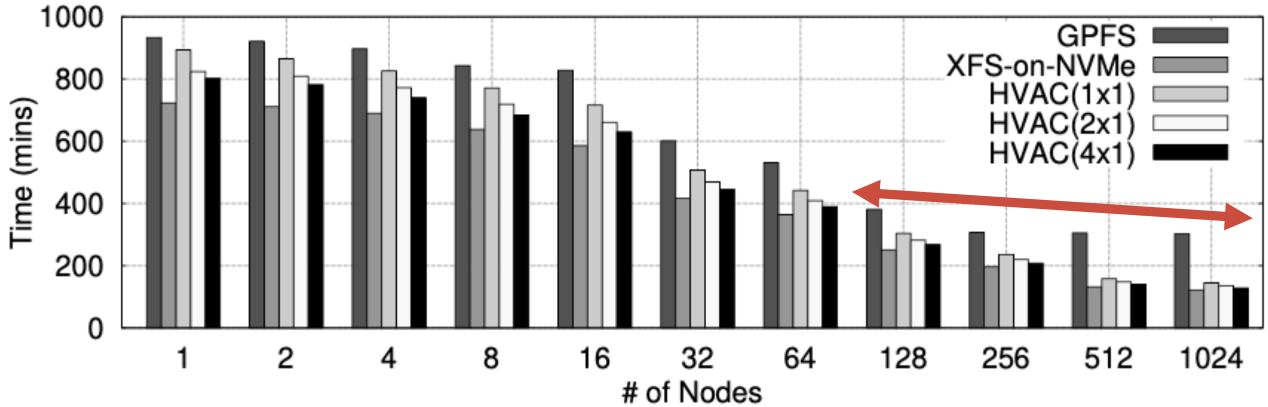
- Distributed Training
  - Pytorch with Horovod
- Applications
  - ResNet50, CosmoFlow, DeepCAM (Gordon bell prize in 2019)

Evaluation

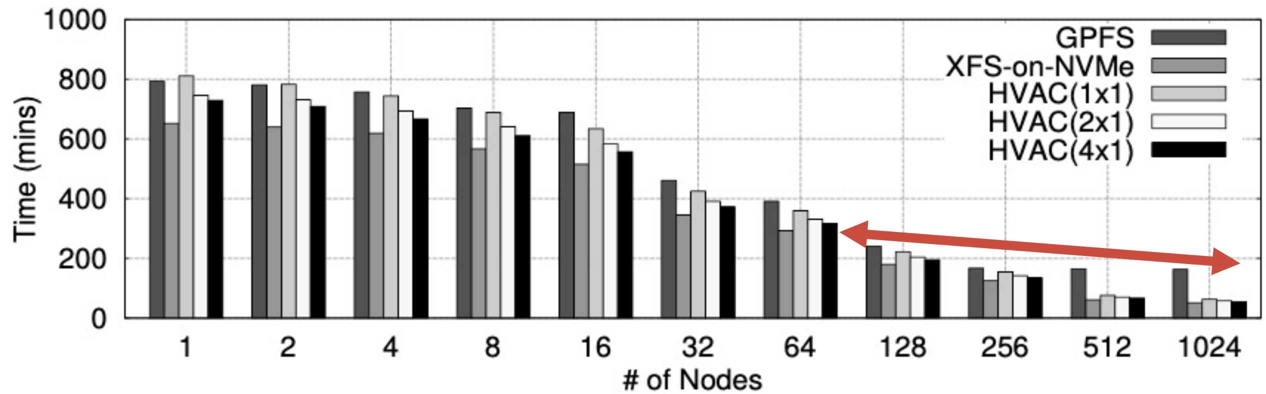
- ***Compared against***
  - ***GPFS:*** Large-scale IBM Spectrum Scale shared PFS hosting the complete dataset. Each epoch reads from GPFS
  - ***XFS-on-NVMe:*** The complete dataset is staged on compute node-local NVMe formatted with XFS file system prior to application run (Upper I/O bound)
  - ***HVAC (i x1): i instance***(s) running on each compute node. The dataset is read from GPFS only in the first epoch
  - **Multiple HVAC servers** on a single compute node to show its flexibility, portability and ease of deployment

Evaluation

- ***Effect of scaling no. of compute nodes on training time***



(a) ResNet50 [BS=80, Eps=10, nProcs/node=2]

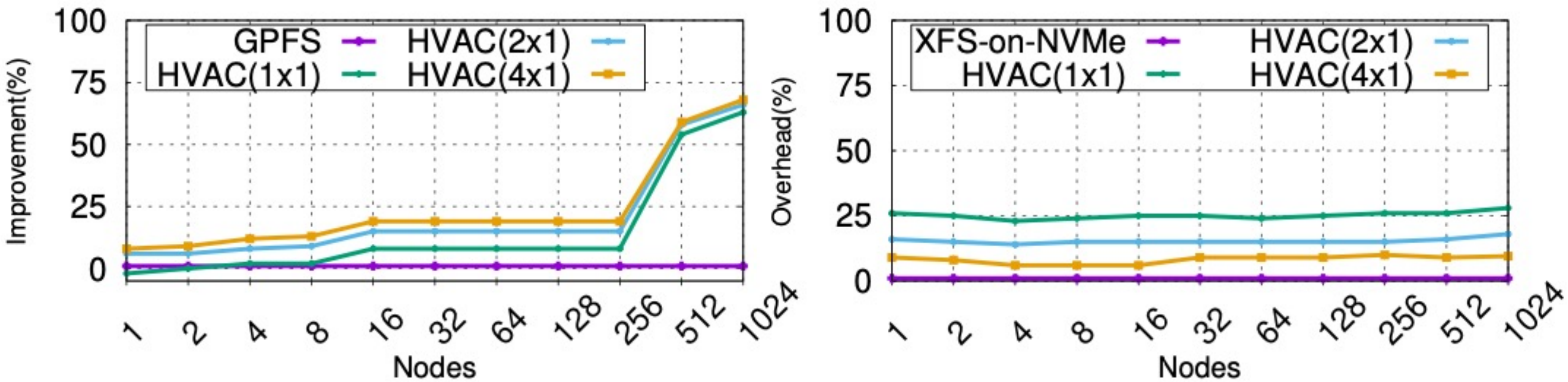


(a) CosmoFlow [BS=80, Eps=10, nProcs/node=2]

Oak Ridge National Laboratory

Evaluation

- ***Performance gain and overhead with scaling no. of compute nodes***



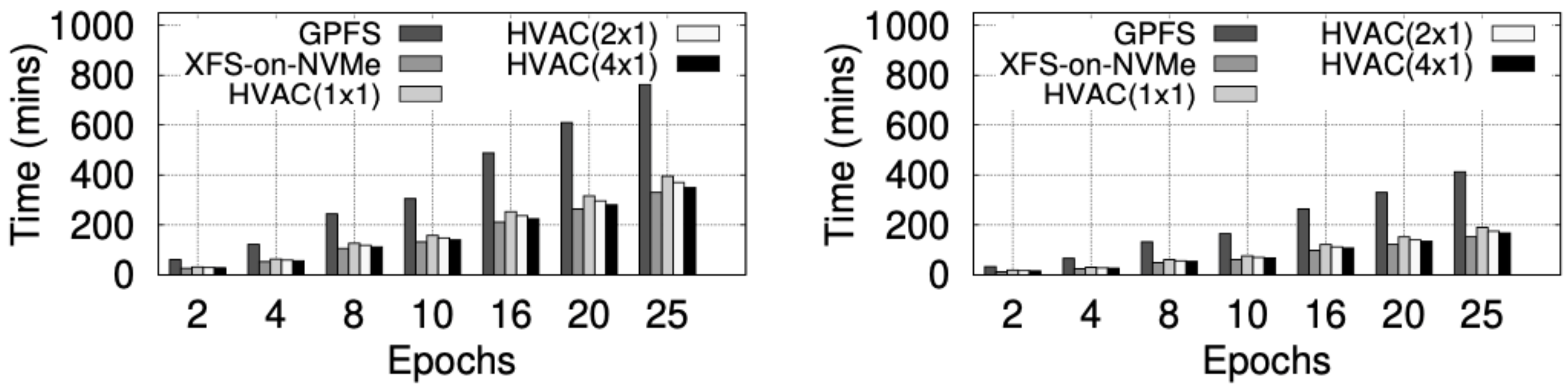
(a) Normalized with GPFS. (b) Normalized with NVMe-SSD.

- ***Average performance improvement is over 50% for all HVAC variants atop GPFS***

- ***HVAC (1x1) shows higher overhead around 25% compared to other HVAC variants, i.e., HVAC(2x1) 14% and HVAC(4x1) with 9%***

Evaluation

- ***Effect of scaling no. of epochs on training time***



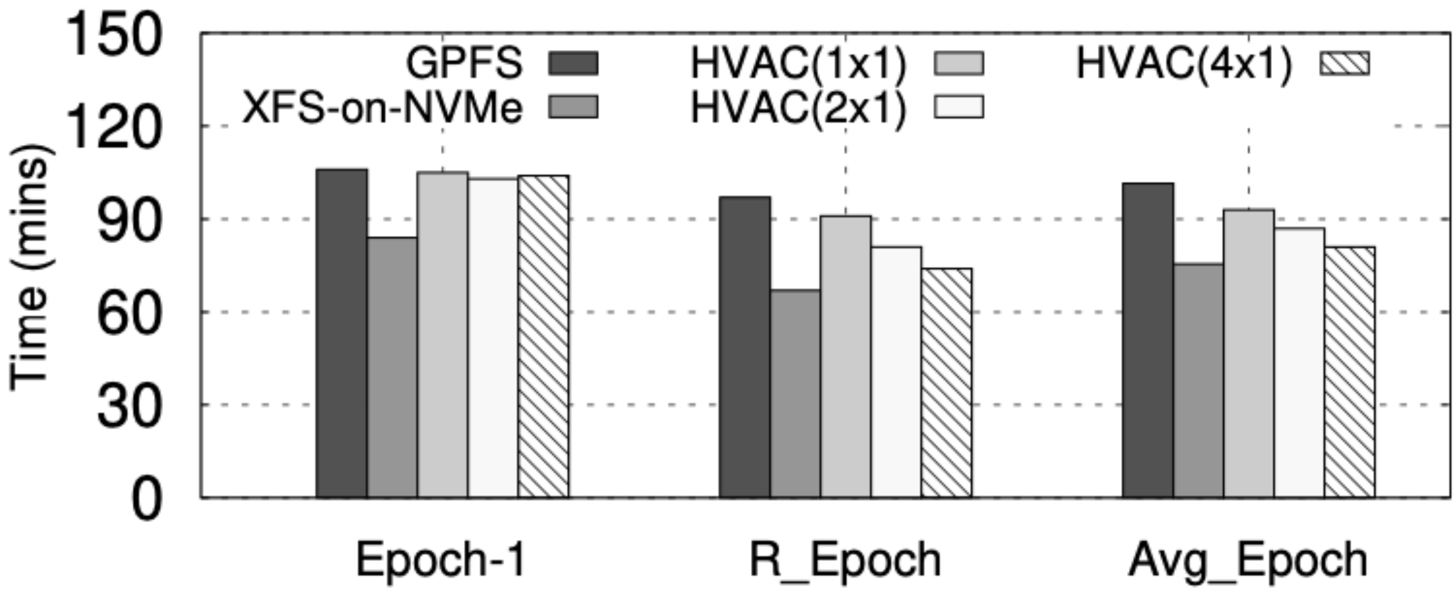
(a) ResNet50 [BS=80, nNodes=512] (b) CosmoFlow [BS=4, nNodes=512]

  - ***Linear scaling showing strong scaling property of HVAC***

OAK RIDGE National Laboratory

Evaluation

- ***Per Epoch Analysis***



- ***Every HVAC servers reads the file from GPFS and then caches it***
- ***(Best and Average Training time), has reduced to a factor 3x per epoch by HVAC(4x1) com- pared to GPFS***

OAK RIDGE National Laboratory

Summary and Current Status

- HVAC is a scalable caching system for HPC systems such as Summit and Frontier
  - Exploits compute node-local storage and builds an aggregate cache layer atop to accelerate the DL application training

- Current Status
  - Deployed and evaluated on 1,024 Summit nodes (with over 6000 NVIDIA V100 GPUS)
  - Development to support Slingshot network on Frontier

Questions?

Awais Khan

Contact: khana@ornl.gov