Intelligent and Progressive Scientific Data Management

Dr. Qing Liu

Associate Professor
Electrical and Computer Engineering
New Jersey Institute of Technology

CTO
X-Byte Research

Joint Faculty
Computer Science and Mathematics Division
Oak Ridge National Laboratory



Advanced Modeling & Simulation (AMS) Seminar Series
NASA Ames Research Center, March 13th, 2025

NJIT New Jersey Institute of Technology

X-BYTE RESEARCH

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE U.S. DEPARTMENT OF ENERGY

Quickly About Myself

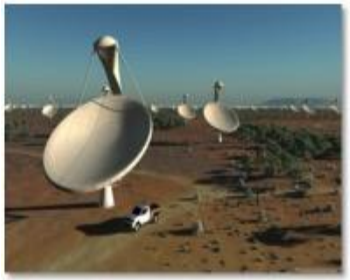
- Associate Professor at New Jersey Institute of Technology
- CTO at X-Byte Research, a small startup founded in 2022 for HPC-related research and development
- Various roles at Oak Ridge National Laboratory since 2009
  - Postdoc at Oak Ridge Leadership Computing Facility
  - Staff Scientist II, and III
  - Now as Joint Faculty
- Research interests
  - HPC, strong interest in both systems and applications

End-to-End Workflow in Big Science

Experiments



Observations



Validation

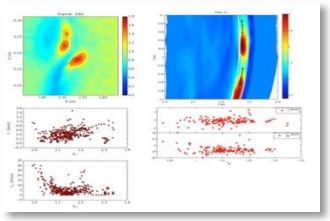
Prediction



High Performance Computing

Adjustment for new runs (e.g., parameters)

Knowledge



Data Analysis & Visualization



Data Storage



- Solid-state drive
- Non-volatile memory (PCM, STT-RAM)
- Die-stacked memory

Preprocessing



- Data inspection
- Transformation
- Annotation and indexing

Data Storage

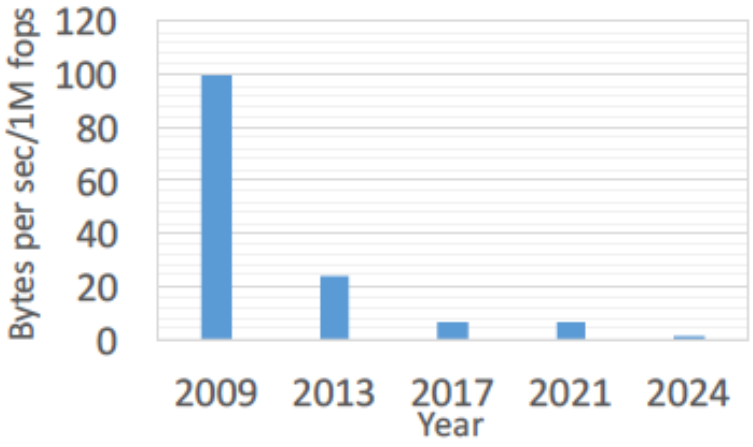


Working with Applications

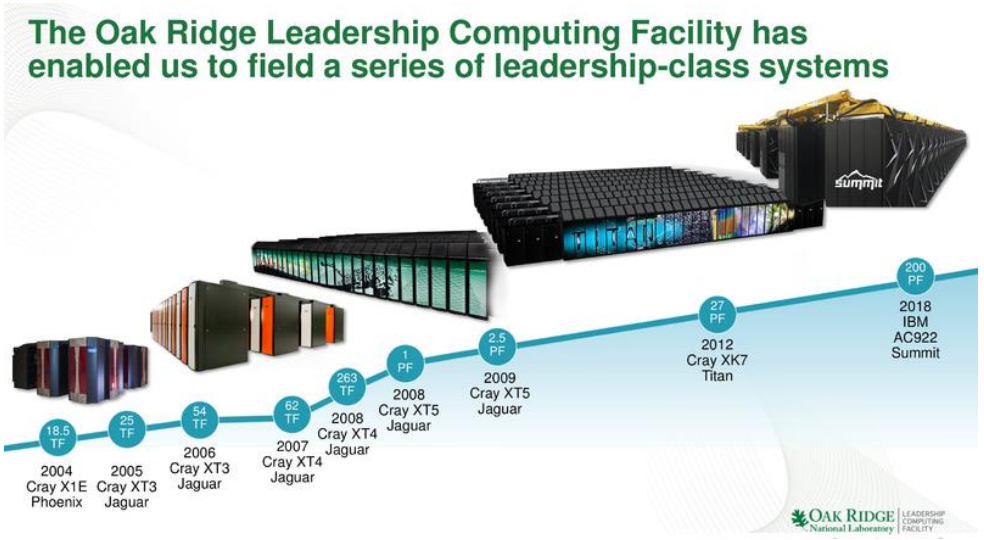
- We have worked with many HPC applications
  - Fusion, high-energy physics (accelerator), geoscience (earthquake), cancer research, nuclear physics, material science, and etc.
  - Commercial applications: Ramgen Power Systems/Numeca (turbine engine design), FireFoam

Key Data Challenges in HPC

- The gap between I/O and compute is increasing
  - E.g., Titan (27-petaflops) -> Summit (200-petaflops)
  - Peak file system throughput remains approximately same



Storage-to-compute trend for large HPC systems



Slides from Oak Ridge Leadership Computing Facility

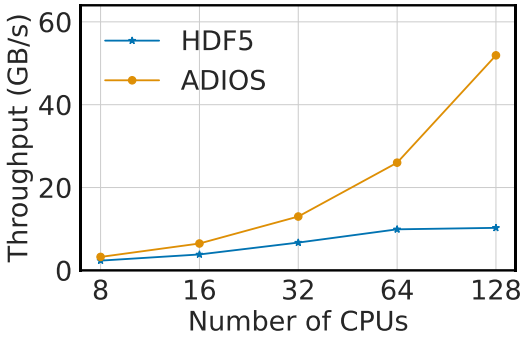
Scalable Data Management Framework

- ADIOS (adaptable I/O system)
  - Declarative APIs and open framework to develop customized I/O methods
  - Fast I/O methods for large storage systems
  - In-situ processing and data streaming capabilities
  - Rich software ecosystem built around ADIOS, e.g., visualization tools, data reduction/compression/transformation, schemas for particle code, etc.
  - https://github.com/ornladios/ADIOS2

```
adios2::IO io = adios.DeclareIO("Output");
io.SetEngine("S3");

auto S3ZSTDOp    = adios.DefineOperator("S3CompressorZSTD", adios2::ops::S3ZSTD);
auto var_temperature = io.DefineVariable<float>("temperature", shape, start, count);
var_temperature.AddOperation(S3ZSTDOp, {{adios2::ops::s3zstd::key::level, l}});

auto S3MGARDOp    = adios.DefineOperator("S3DecomposerMGARD", adios2::ops::S3MGARD);
auto var_pressure = io.DefineVariable<double>("pressure", shape, start, count);
var_pressure.AddOperation(S3MGARDOp);
        .....
```



Write throughput

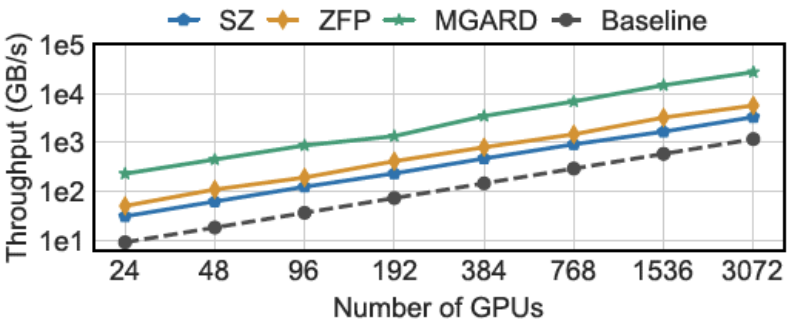
(CESM on Summit)

Liu, Q., Logan, et. al., Hello ADIOS: the challenges and lessons of developing leadership class I/O frameworks. Concurrency Computat.: Pract. Exper., 26: 1453-1473. https://doi.org/10.1002/cpe.3125

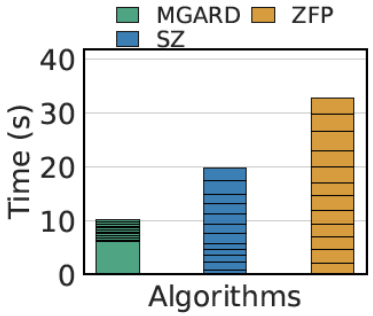
William F. Godoy, et. al., ADIOS 2: The Adaptable Input Output System. A framework for high-performance data management, SoftwareX, Volume 12, 2020, https://doi.org/10.1016/j.softx.2020.100561.

Data Compression and Decomposition

- MGARD (MultiGrid Adaptive Reduction of Data)
  - https://github.com/CODARcode/MGARD
  - Data decomposition, compression, and progressive retrieval
  - Lossy reduction with guaranteed error for derived quantities
    - Compress X, but guarantee the error of f(X)
  - Implementations portable across CPU and GPUs (Nvidia, AMD)



Scalability (Summit)



Progressive retrieval (Summit)

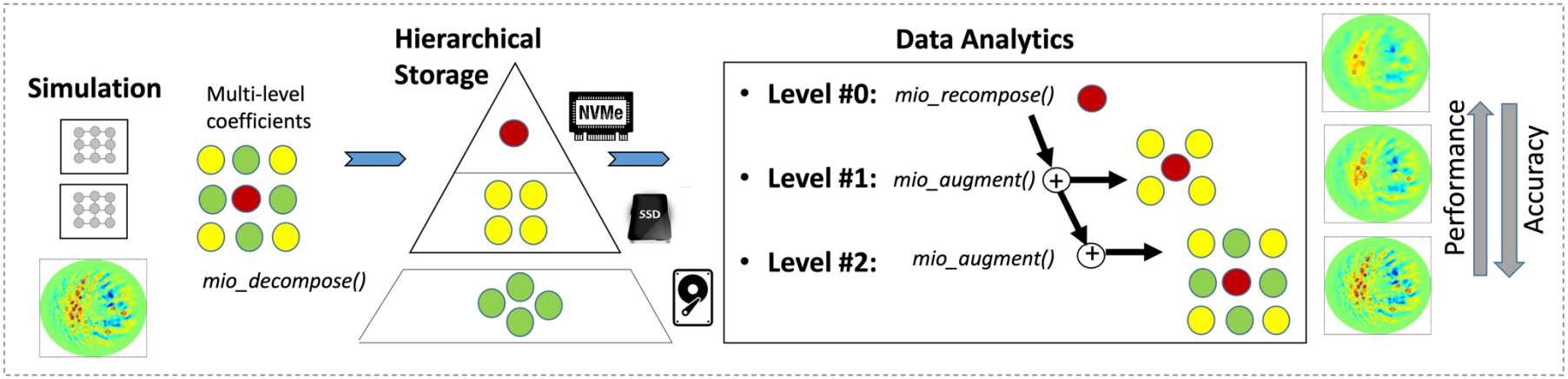
Jinzheng Wang et al. Improving Progressive Retrieval for HPC Scientific Data using Deep Neural Network. IEEE International Conference on Data Engineering (ICDE), 2023

Xin Liang et al. Error-controlled, progressive, and adaptable retrieval of scientific data with multilevel decomposition. the International Conference for High Performance Computing, Networking, Storage and Analysis 2021, Nov, 2021

Jieyang Chen et al. Accelerating Multigrid-based Hierarchical Scientific Data Refactoring on GPUs. 35th IEEE International Parallel & Distributed Processing Symposium, May 17–21, 2021.

Multi-level I/O System (MIO)

- Under development since last year to achieve multi-level capabilities over modern I/O systems
  - HPC storage: distribute data across storage tiers
  - Network: transmit data through multiple streams across the network
- Built on top of MGARD to achieve data decomposition
- Data placement, progressive retrieval, and intelligent migration.



Multi-level I/O System (MIO)

- Programming model
  - Mimic Linux C/C++ file operations
  - A new model of progressively interacting with data

Simulation

```
mio_file_handle * f = mio_open(”temperature.f32", "w");

vector<string> paths;
paths.push_back ("path1");
paths.push_back ("path2");

mio_write (f, size, data, paths);

mio_close(f);
```

System Daemon

```
mio_file_handle * f = mio_open(”temperature.f32", ”m");

vector<int> levels = {2}; // Level 2 is selected to be moved.
vector<string> paths = {“path3”}; // the migration dest
mio_move (f, size, data, paths);

mio_close(f);
```

Data Analytics

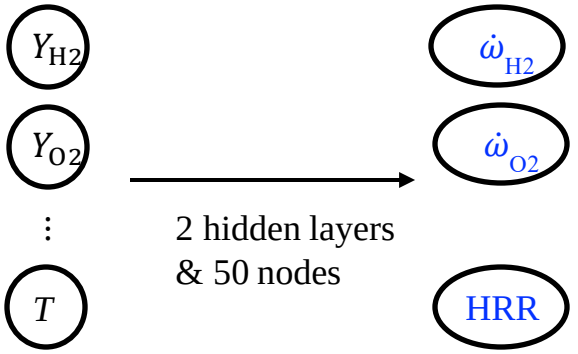
```
mio_file_handle * f = mio_open("path1", "r");

float err = 0.01;
mio_read (f, err, data);

while (!accuracy_satisfied(data))
{
   free (data);
   err = 0.1*err;
   mio_read (f, err, data);
}

mio_close(f);
```

Using Multi-level Decomposition for Combustion CFD



Turbulent combustion simulations

- **Combustion simulation is a high dimensional problem (dimension (rank): # of chemical species)**
  - **Methane/air mixture: $CH_4 + 2O_2 \rightarrow CO_2 + 2H_2O$**
  - **In reality, $CH_4 \rightarrow CH_3 \rightarrow CH_2O \rightarrow HCO \rightarrow CO \rightarrow CO_2$**

Governing equations

$$\frac{\partial \rho}{\partial t} = -\nabla_\beta \cdot (\rho \mathbf{u}_\beta) \quad \text{(continuity)}$$

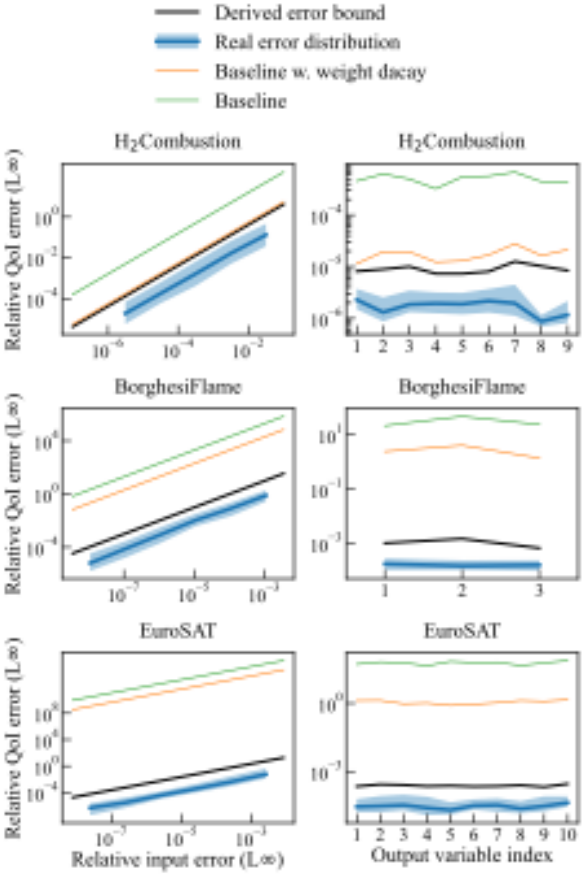
$$\frac{\partial (\rho \mathbf{u}_\alpha)}{\partial t} = -\nabla_\beta \cdot (\rho \mathbf{u}_\alpha \mathbf{u}_\beta) + \nabla_\beta \cdot \boldsymbol{\tau}_{\beta\alpha} - \nabla_\alpha p \quad \text{(momentum)}$$

$$\frac{\partial (\rho e_0)}{\partial t} = -\nabla_\beta \cdot [\mathbf{u}_\beta (\rho e_0 + p)] + \nabla_\beta \cdot (\boldsymbol{\tau}_{\beta\alpha} \cdot \mathbf{u}_\alpha) - \nabla_\beta \cdot \mathbf{q}_\beta \quad \text{(total energy)}$$

$$\frac{\partial (\rho Y_i)}{\partial t} = -\nabla_\beta \cdot (\rho Y_i \mathbf{u}_\beta) - \nabla_\beta \cdot (\rho Y_i V_{\beta i}) + W_i \dot{\omega}_i, \qquad i = 1, n_s \quad \text{(species transport)}$$

Formation (consumption) rate of species *i*

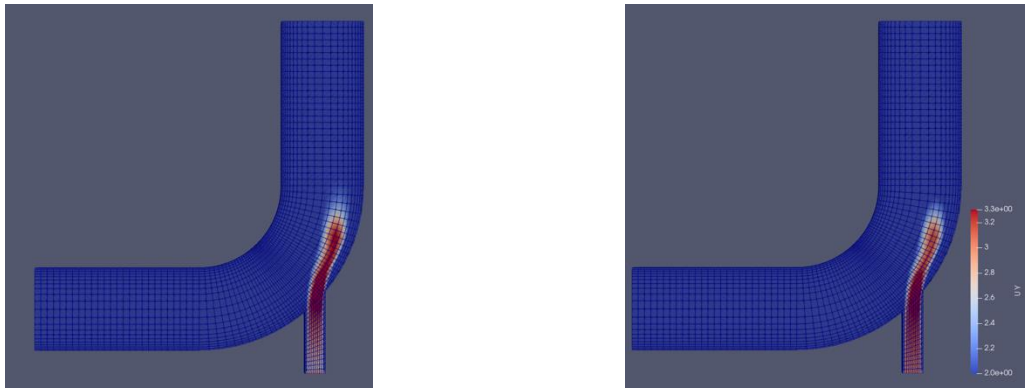
Solution: Use an ANN model for predicting $\dot{\omega}_i$!
Input: mass fraction (Y_i), output: $\dot{\omega}_i$
e.g., for air, Y_{O2} and Y_{N2} are 0.233, 0.767 (volume fractions are 0.21 and 0.79, respectively)



Slides from Dr. Kisung Jung from Sandia National Laboratory

Using Multi-level Decomposition for OpenFOAM

Velocity field (Y) of the elbow fluid simulation

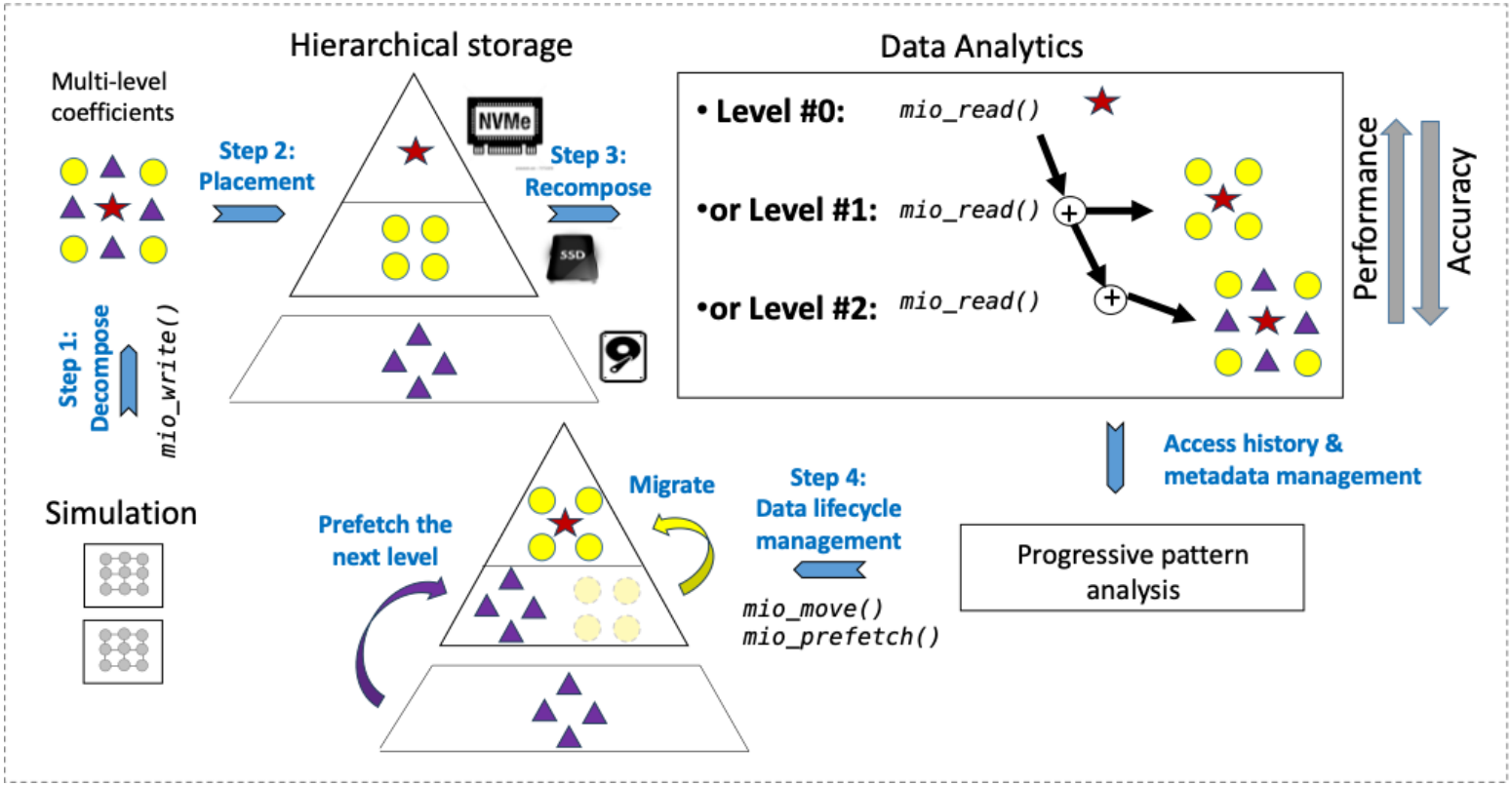


(a) Reduced **(b) Full**

- For the left sub-figure, we retrieved the first bitplane (out of 32 bitplanes) from the first level (out of 13 levels) of velocity Y,
- For the right sub-figure, we additionally retrieved the rest of the bitplanes from all levels.
- A substantial difference is observed in the bottom inlet area.

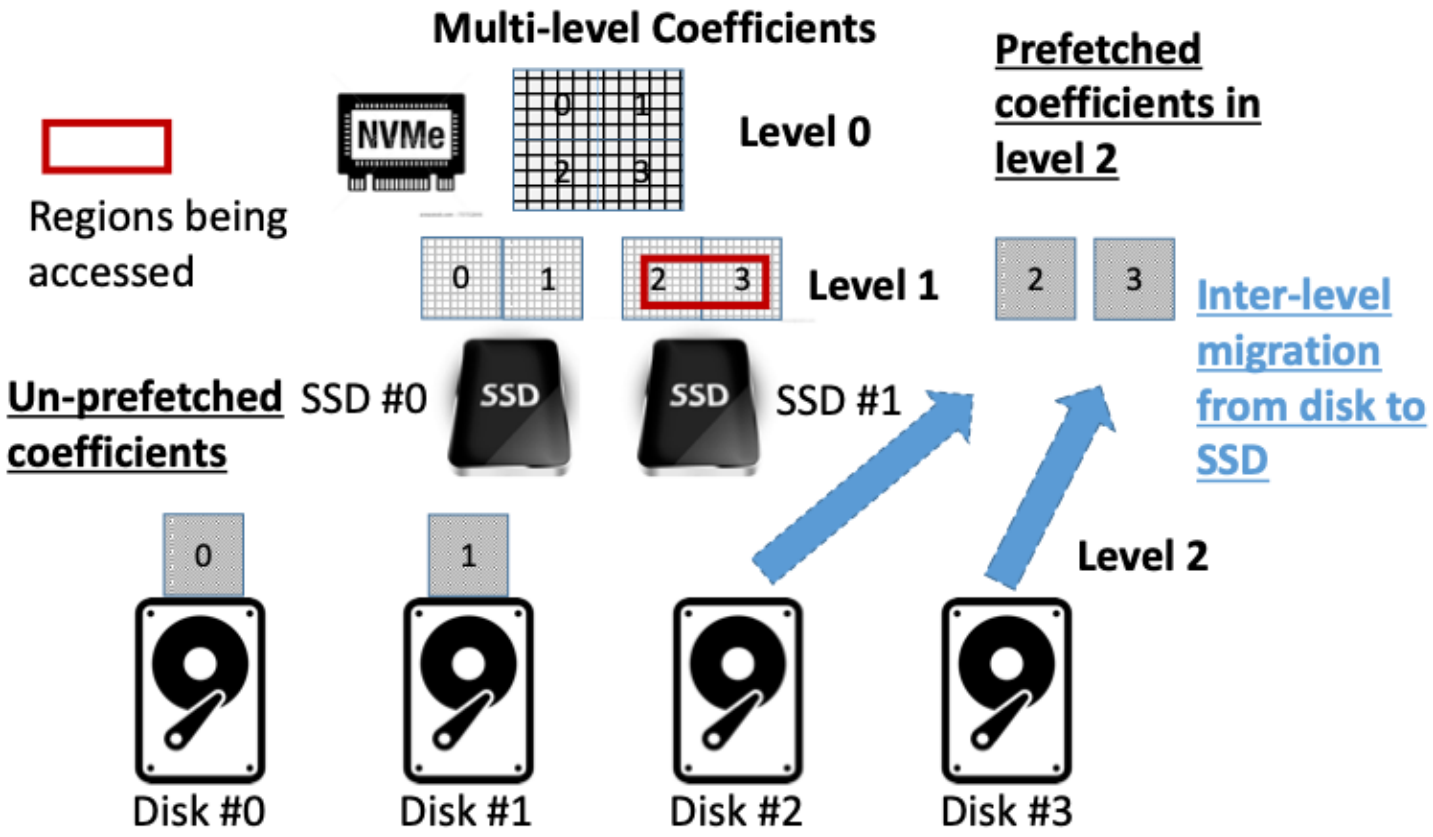
Multi-level Data Management

- Intelligent data prefetching and migration (under development).



Multi-level Data Management

- Multi-level prefetching



Reducing I/O Variation on HPC Storage

Application
Middleware
Storage

- Significant I/O variations on HPC systems
  - Due to the lack of explicit QoS at the application level
  - Due to high parallelism on HPC systems
- Prior works required all applications to change the I/O interface and interact with each other to alleviate the interference

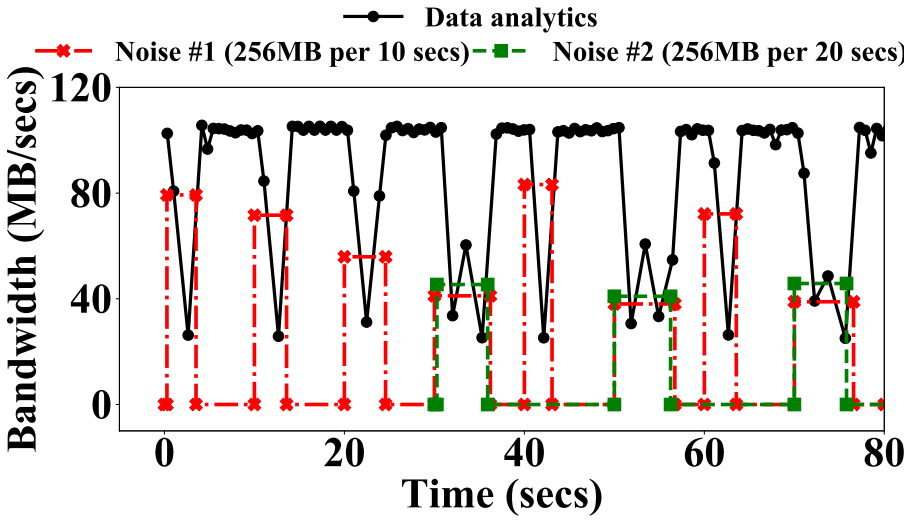
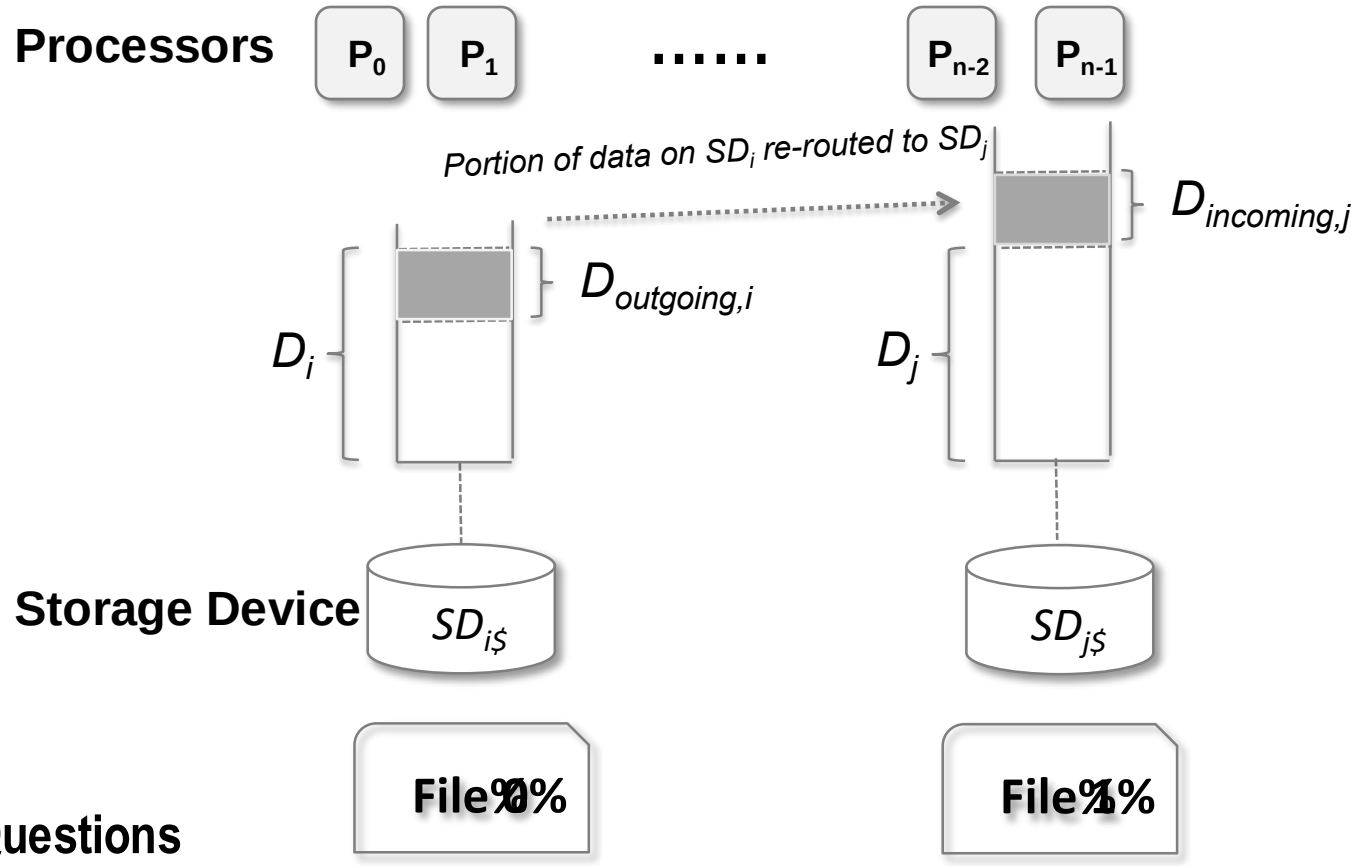


TABLE I: QoS in HPC file systems.

File system	HPC System	QoS mechanism
Ceph (13.2.6)	CERN cluster	None. A new mechanism is under development using *dmclock*.
Lustre (>2.6)	Theta, Cori	Token bucket filter. However, it can only throttle I/O performance, without providing performance guarantee. Can only limit RPC rate, instead bandwidth and IOPS.
Spectrum Scale (5.0.4)	Summit	QoS can only be controlled using *mmlsqos* for each storage pool, instead of for individual applications.
OrangeFS (2.9.7)	AWS	None.

Reducing I/O Variation on HPC Storage

- Re-route I/O requests away from congested storage devices



Application

Middleware

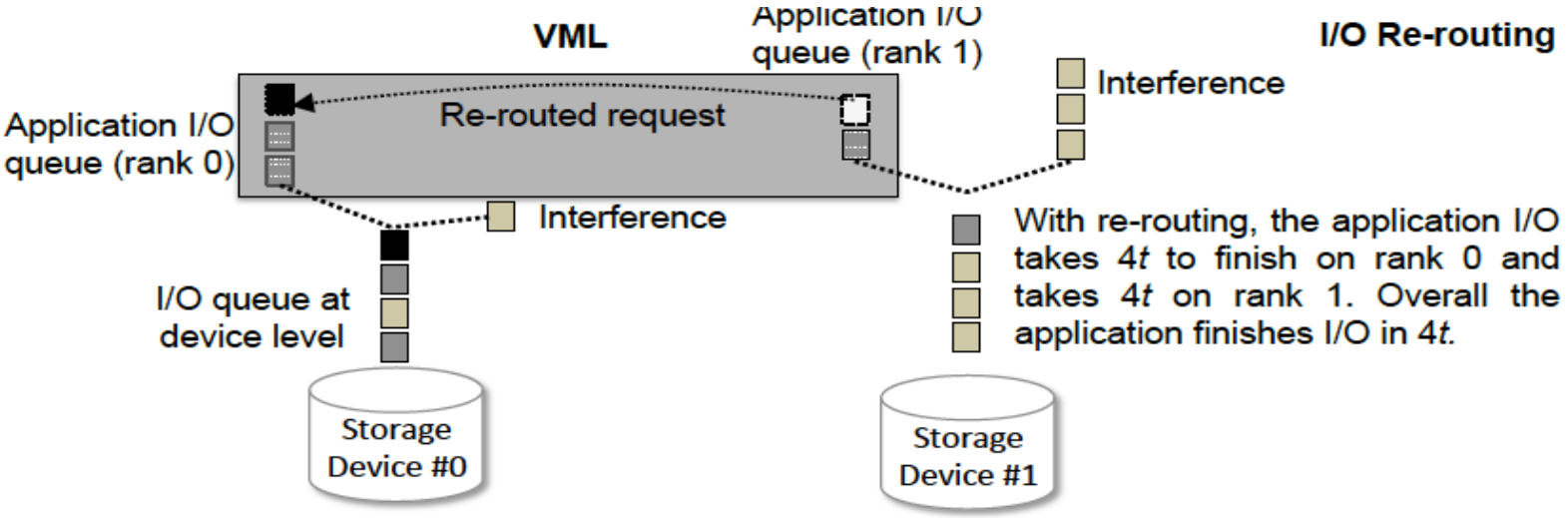
Storage

- Questions
  - How to identify relatively slow storage device? Modern storage systems are still simple block devices, and there are no performance semantics in APIs
  - How to re-direct a write processor to a new location?

D. Huang *et al*., "Can I/O Variability Be Reduced on QoS-Less HPC Storage Systems?," in *IEEE Transactions on Computers*, vol. 68, no. 5, pp. 631-645, 1 May 2019, doi: 10.1109/TC.2018.2881709.

Key Design Decisions

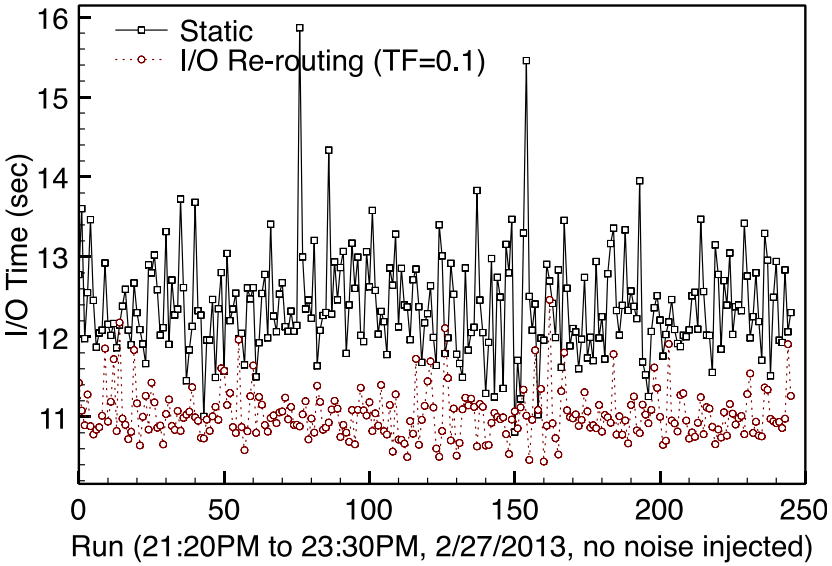
- No periodic measurements/sampling on storage device performance
  - Use ”my” application I/O queue as an indicator for noise intensity
  - Lazy rerouting, i.e., reroute occurs *only* when the per application I/O queue of a particular storage device becomes empty

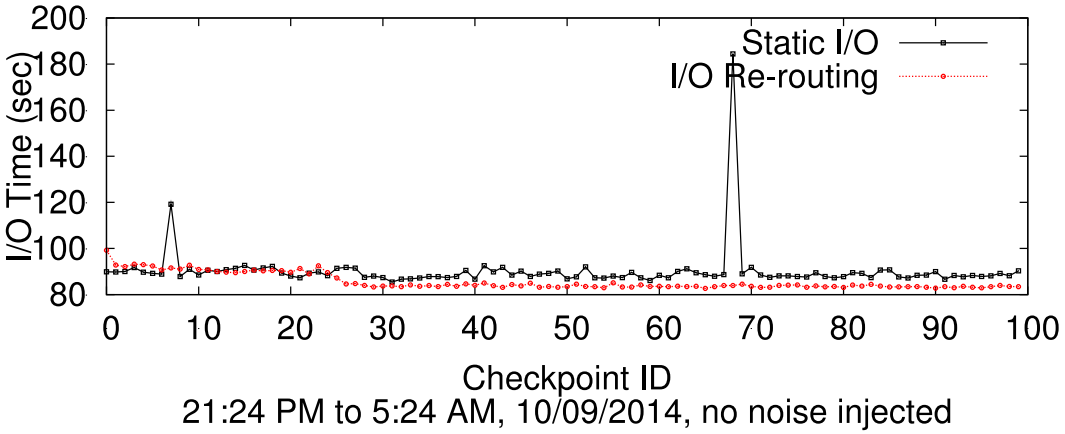


- Structure the scheduling system into hierarchies in order to be able to scale up

Write Performance

No artificial noise injected





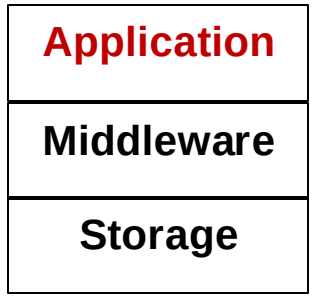
Synthetic benchmark

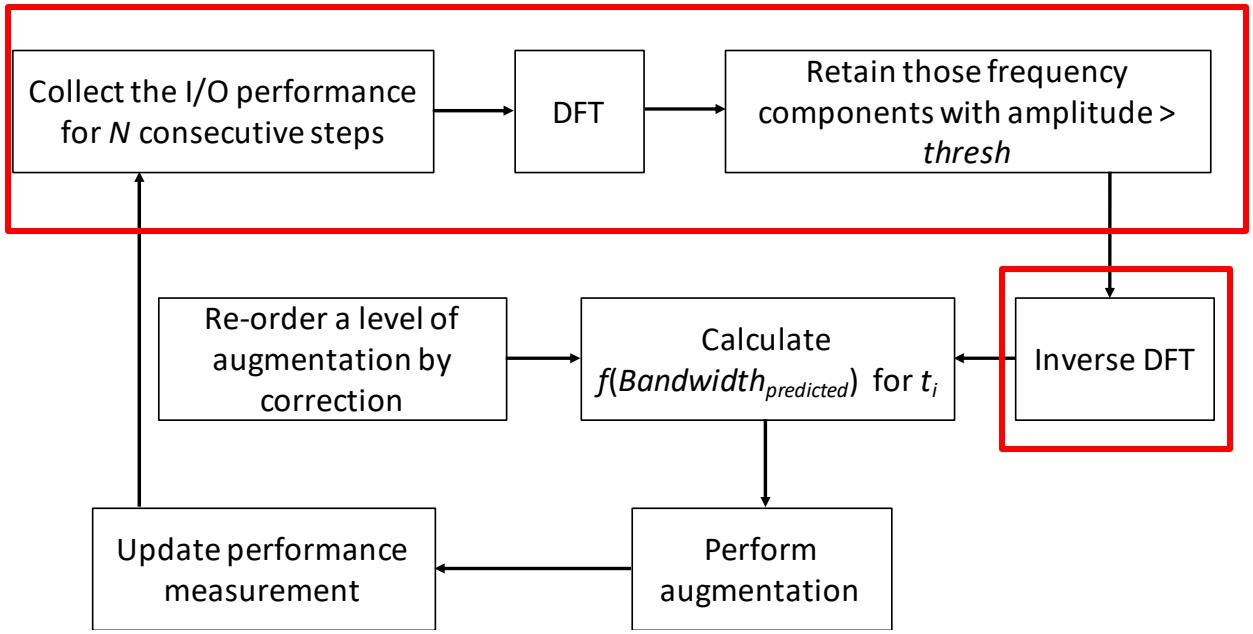
Fusion XGC

- **I/O rerouting improves I/O performance in terms of average I/O time and variability, despite the added communications**
- **How much is the improvement? It really depends on the noise intensity.**

What Can Applications Do?

- **Idea: why not control the interference at the application level?**
- **Key questions:**
  - **Q1:** Can we learn the pattern of interference?
    - ✓ Most parallel applications on HPC systems are deemed to follow certain patterns that can be described as I $(C^xW)^*F$.
    - ✓ Therefore, the interference is $\sum_i I_i$ (CixWi)∗Fi , where the subscript *i* denotes the *i*-th competing application.

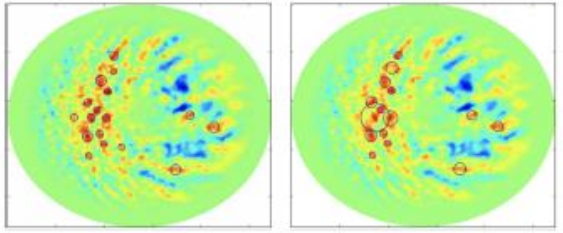


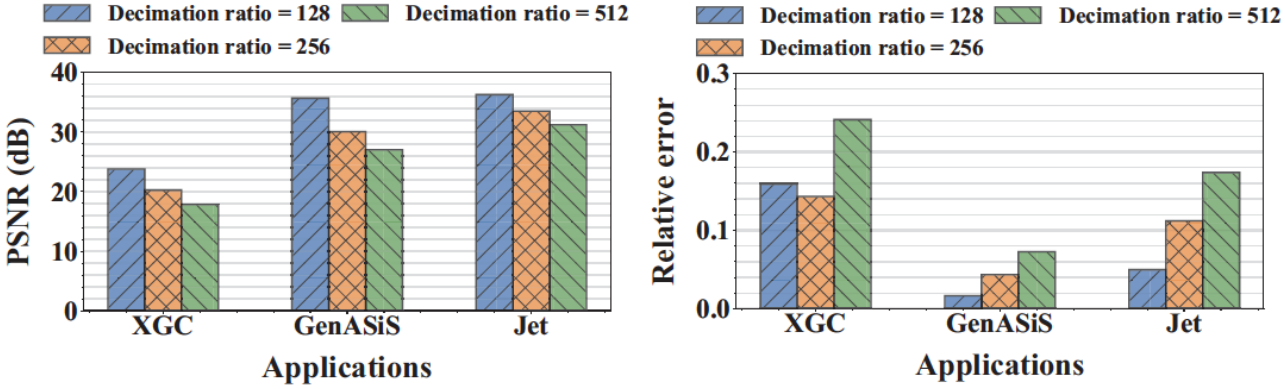


Z. Qiao, Q. Liu, N. Podhorszki, S. Klasky and J. Chen, "Taming I/O Variation on QoS-Less HPC Storage: What Can Applications Do?," *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, 2020

What Can Applications Do?

- **Q2**: Can data analytics tolerate lower accuracy?



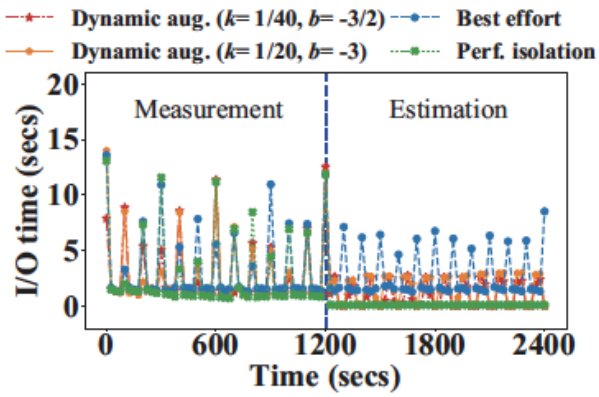


(a) PSNR. (b) Analysis error.

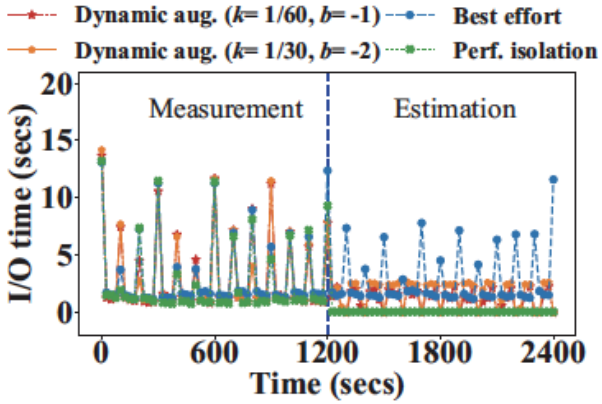
Accuracy of using the reduced representation

- ✓ **What if the error is not satisfactory?**
  - ❖ **Progressive augmentation**

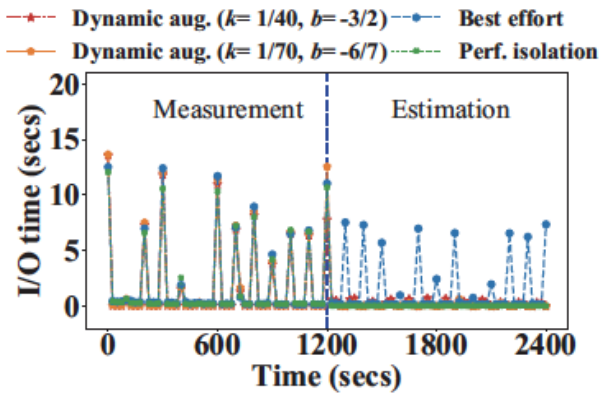
Performance Evaluation



(a) XGC.

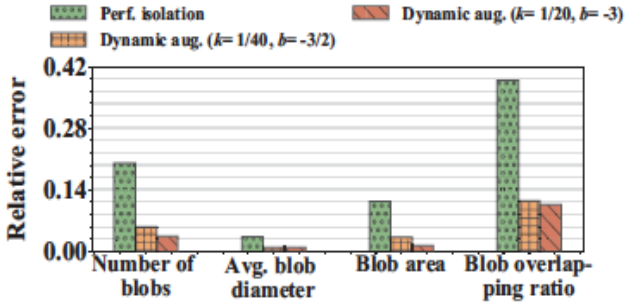


(b) GenASiS.

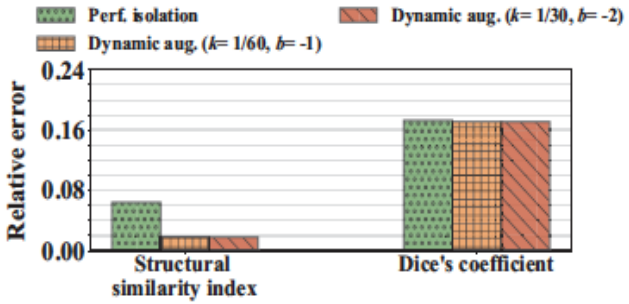


(c) Jet.

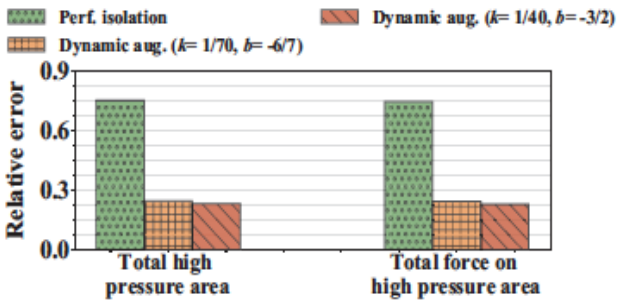
Estimation and dynamic augmentation



(a) XGC.



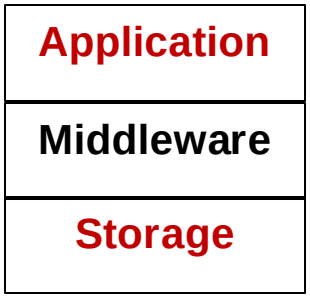
(b) GenASiS.



(c) Jet.

Assessing data quality

Cross-layer Error bounded Data Recomposition



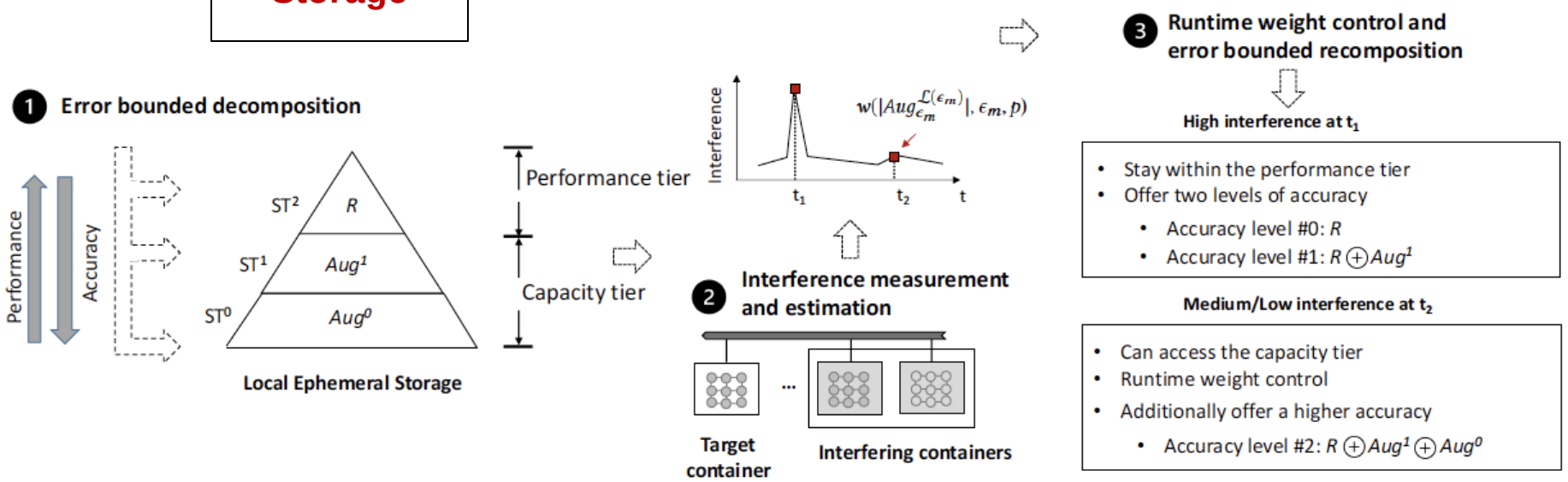
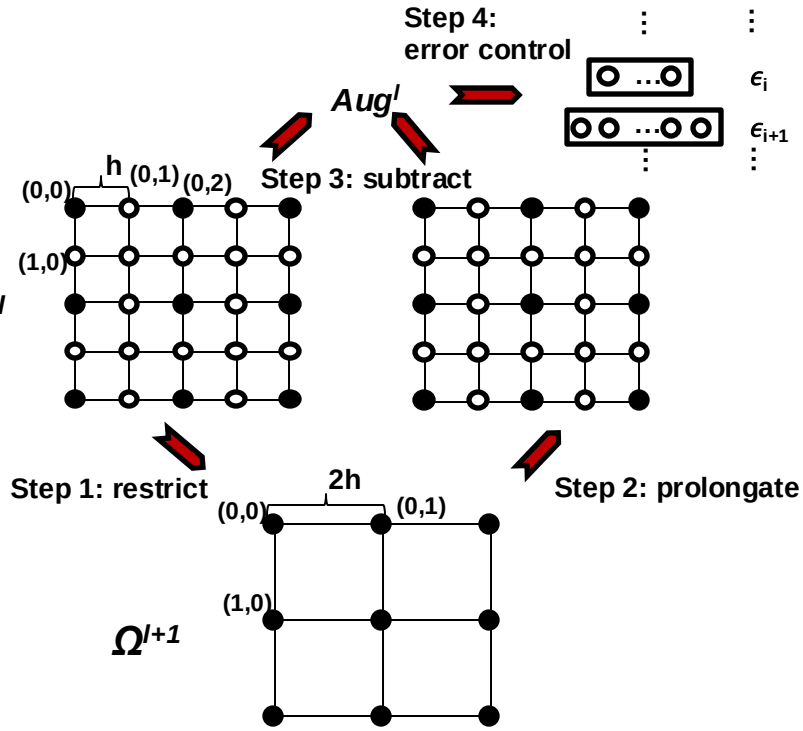


Fig. 3: An illustration of Tango. Herein we assume the interference is over the shared ST^0 and ST^1. The data analytics can be performed in three accuracy levels. If there is interference experienced, the data analytics can adapt and lower the accuracy, so that there is less or no data retrieved from the capacity tier. The operator $\oplus$ denotes the recomposition.

Z. Qiao *et al.*, "Tango: A Cross-layer Approach to Managing I/O Interference over Local Ephemeral Storage," *SC24: International Conference for High Performance Computing, Networking, Storage and Analysis*, Atlanta, GA, USA, 2024, pp. 1-15, doi: 10.1109/SC41406.2024.00020.

Error bounded Data Decomposition

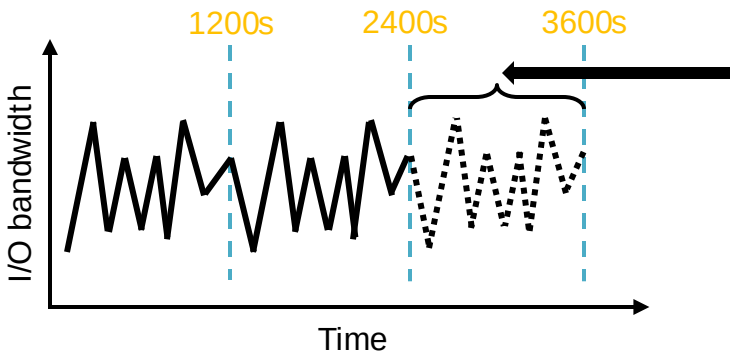
- **Step 1: restrict**
  - $\Omega^{l+1}_{(0,1)} = \Omega^{l}_{(0,2)}$
- **Step 2: prolongate**
  - $\text{prolongate}(\Omega^{l+1})_{(0,1)} = \frac{1}{2}(\Omega^{l+1}_{(0,0)} + \Omega^{l+1}_{(0,1)})$
- **Step 3: subtract**
  - $\text{Aug}^{l} = \Omega^{l} - \text{prolongate}(\Omega^{l+1})$
- **Step 4: error control**
  - Sort Aug^{l} by absolute value
  - Shuffle subsets with certain error bounds together

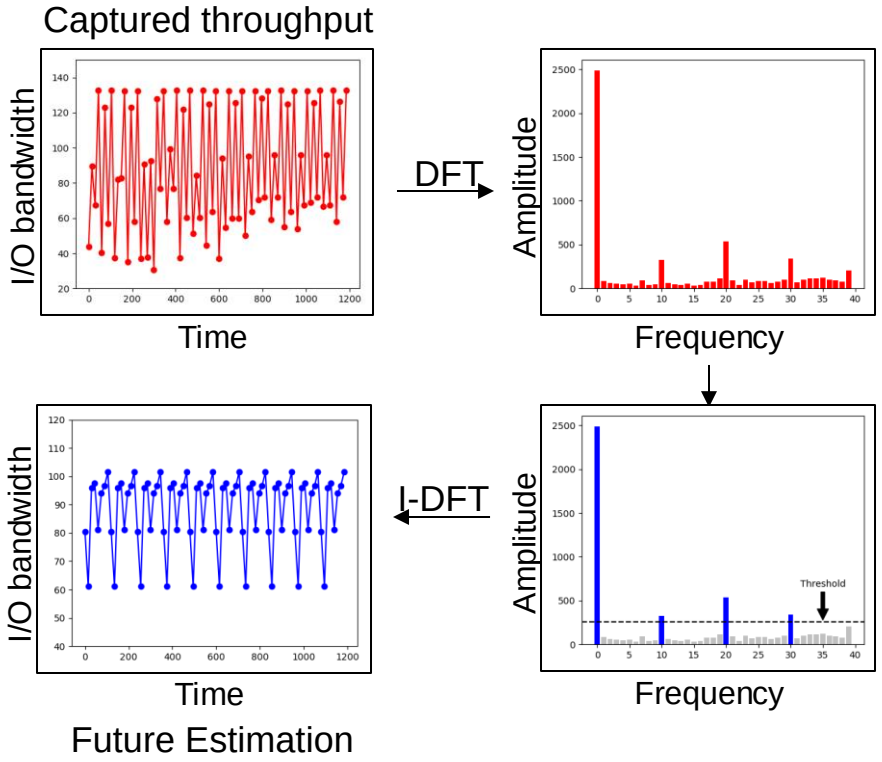


An example when decimation ratio = 2.

Interference Measurement & Estimation

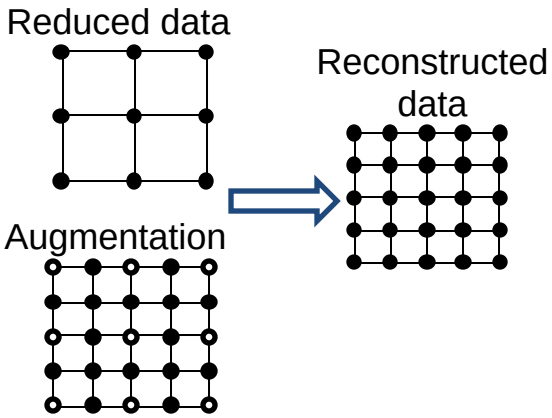
- Step 1: Collect I/O throughput
- Step 2: Filter periodic components
  - Discrete Fourier Transform (DFT)
  - Filter by amplitude
  - Inverse DFT

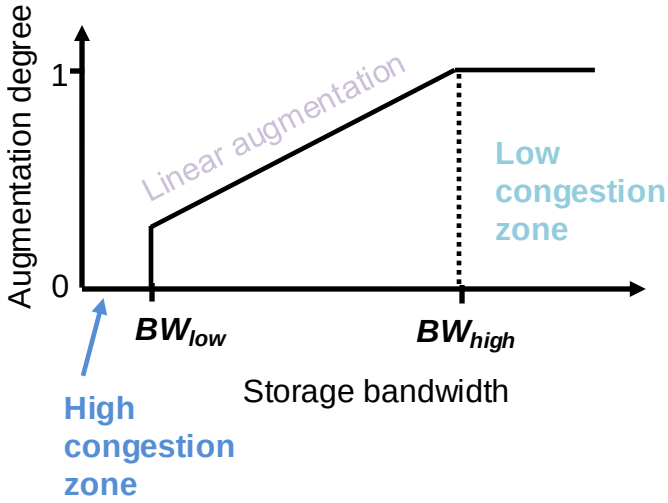




Cross-layer Interference Mitigation

- Application layer: Dynamic augmentation



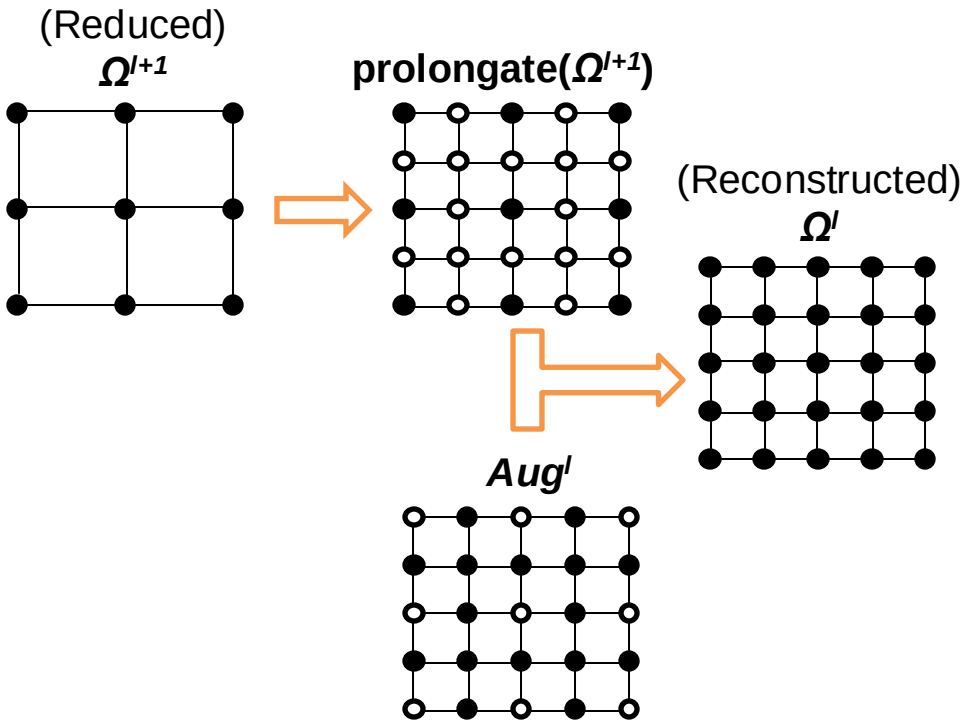


  - Dynamic augmentation retrieval
    - Threshold: BW_{low}, BW_{high}

Cross-layer Interference Mitigation

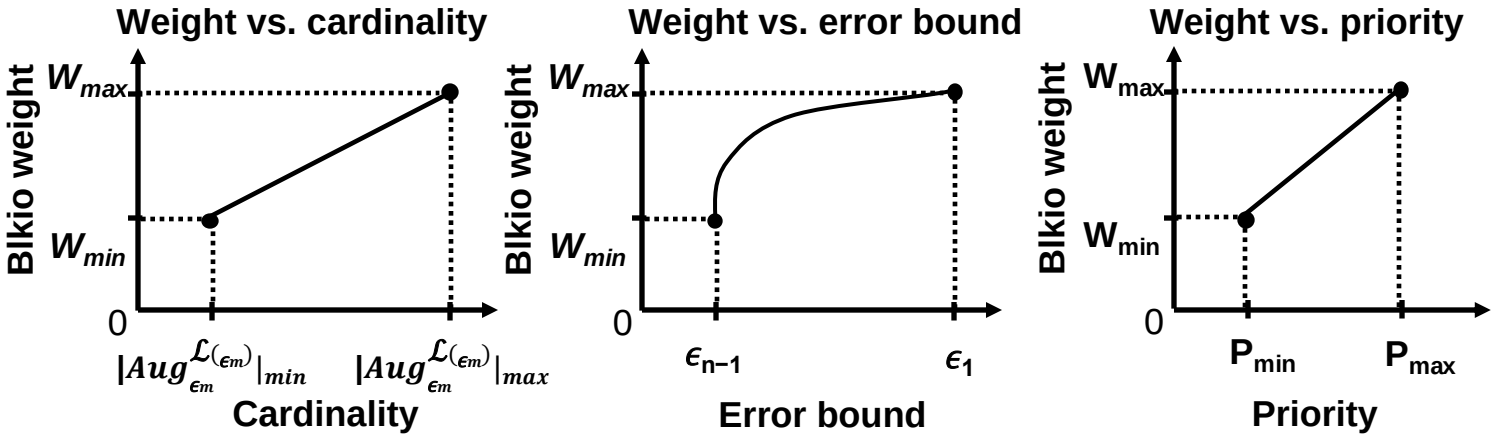
- Application layer: Dynamic augmentation
  - Data reconstruction

$$\Omega^{l} = \text{prolongate}(\Omega^{l+1}) + \text{Aug}^{l}$$



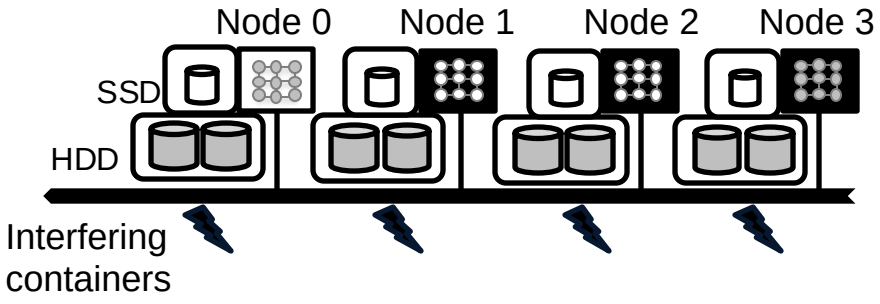
Cross-layer Interference Mitigation

- Storage layer: Runtime resource control via *blkio.weight* adjustment
  - Weight function parameters:
    - Data cardinality
    - Error bound
    - Priority



Testbed Configuration

- Chameleon cloud experimental facility
- Each physical compute node:
  - Processors: Intel Xeon 2.3GHz, 10 cores
  - SSD: Intel 400GB SATA
  - HDD: Seagate 2TB 7200RPM SAS
- Applications in Docker containers



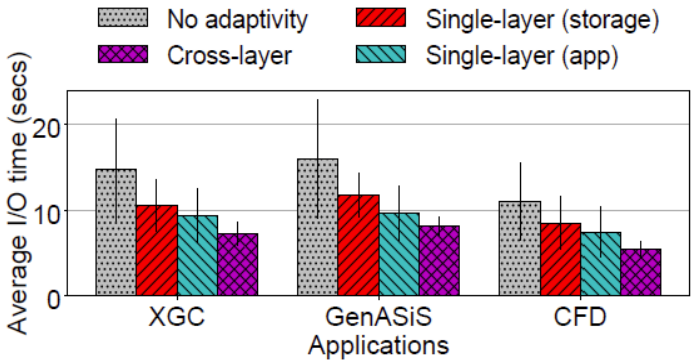
Intensity of interference

Interference	Period	Checkpoint size
Container #1	200s	768MB
Container #2	225s	512MB
Container #3	360s	512MB
Container #4	180s	1024MB
Container #5	150s	1024MB
Container #6	120s	1024MB

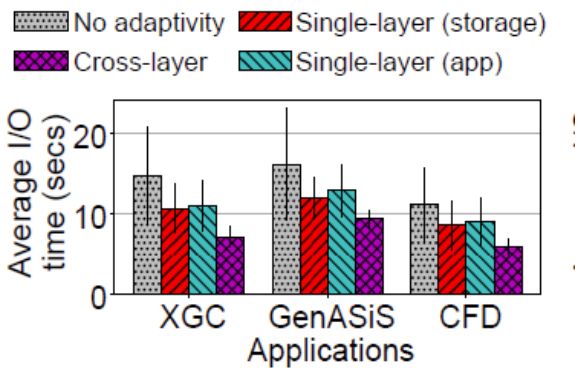
Cross-layer Approach

- **Cross-layer performance improvement**
  - **Without error control**

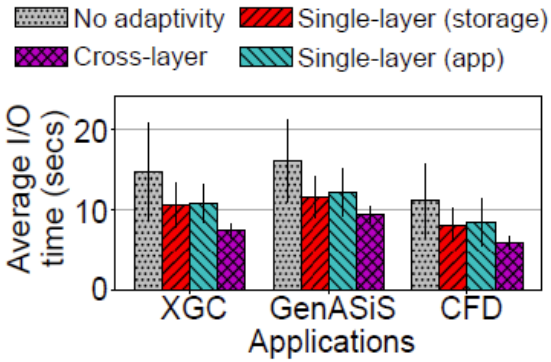
No error control



  - **With error control**



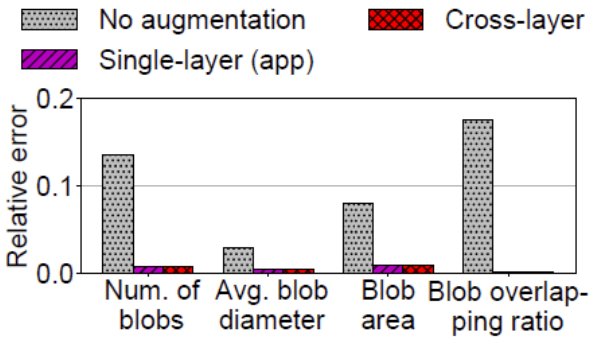
(a) ε = 0.01 (NRMSE)



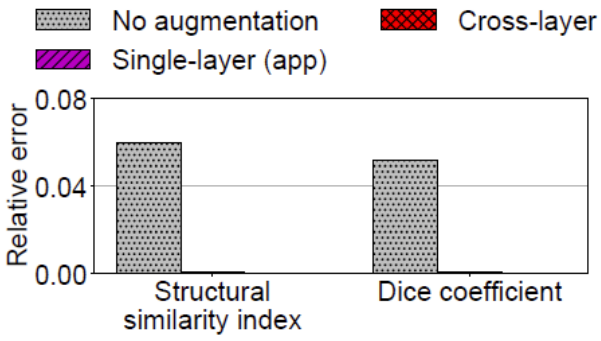
(b) ε = 30 (PSNR)

Error control with data decomposition

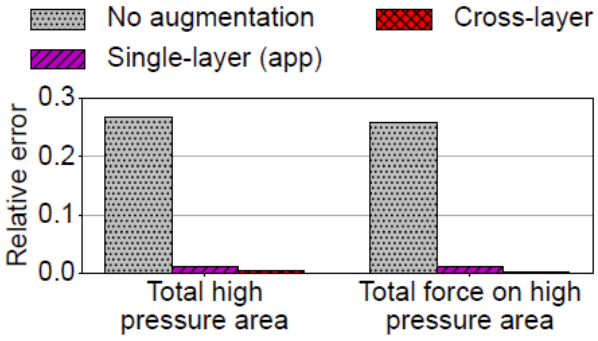
- Data quality through analysis results
  - Cross-layer approach maintains acceptable data quality



(a) XGC



(b) GenASiS

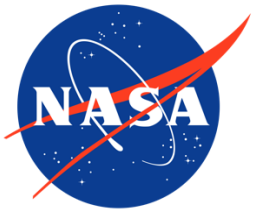


(c) CFD

Summary

- We developed various progressive data management schemes, involving applications, middleware, and storage, to managing data in an HPC environment.
- We intelligently retrieve and migrate data so that the I/O performance can be vastly improved.

Thanks!


NASA


DEPARTMENT OF ENERGY
UNITED STATES OF AMERICA


OAK
RIDGE
National Laboratory


NSF