Data Deduplication in a Hybrid Architecture for Improving Write Performance

Chao Chen, Jonathan Bastnagel, Yong Chen

Data-intensive Salable Computing Laboratory
Department of Computer Science
Texas Tech University
Lubbock, Texas

June 10th, 2013

Big Data Problem

- Many scientific simulations become highly data intensive
- Simulation resolution desires finer granularity both spacial and temporal
  - e.x. climate model, 250KM ⇒ 20KM; 6 hours ⇒ 30 minutes
- The output data volume reaches tens of terabytes in a single simulation, the entire system deals with petabytes of data
- The pressure on the I/O system capability substantially increases

PI	Project	On-Line Data	Off-Line Data
Lamb, Don	FLASH: Buoyancy-Driven Turbulent Nuclear Burning	75TB	300TB
Fischer, Paul	Reactor Core Hydrodynamics	2TB	5TB
Dean, David	Computational Nuclear Structure	4TB	40TB
Baker, David	Computational Protein Structure	1TB	2TB
Worley, Patrick H.	Performance Evaluation and Analysis	1TB	1TB
Wolverton, Christopher	Kinetics and Thermodynamics of Metal and Complex Hydride Nanoparticles	5TB	100TB
Washington, Warren	Climate Science	10TB	345TB
Tsigelny, Igor	Parkinson's Disease	2.5TB	50TB
Tang, William	Plasma Microturbulence	2TB	10TB
Sugar, Robert	Lattice QCD	1TB	44TB
Siegel, Andrew	Thermal Striping in Sodium Cooled Reactors	4TB	8TB
Roux, Benoit	Gating Mechanisms of Membrane Proteins	10TB	10TB

Figure 3: Data volume of current simulations

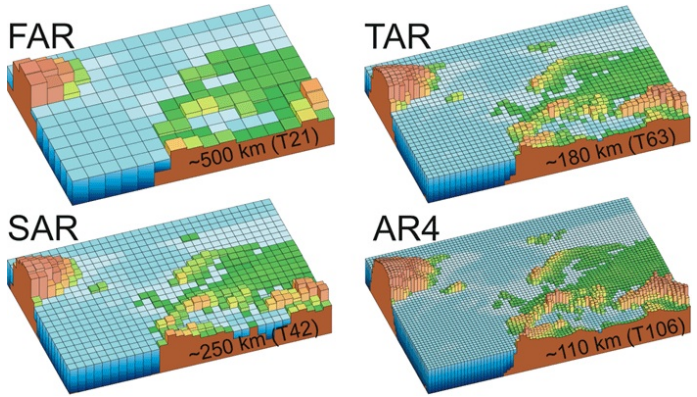


Figure 4: Climate Model Evolution: FAR (1990), SAR (1996), TAR (2001), AR4 (2007)

Motivation - Gap between Applications' Demand and I/O System Capability

- Gyrokinetic Toroidal Code (GTC) code
  - Outputs particle data that consists of two 2D arrays for electrons and ions, respectively
  - Two arrays distributed among all cores, particles can move across cores in a random manner as the simulation evolves
- A production run with the scale of 16,384 cores
  - Each core outputs roughly two million particles, 260GB in total
  - Desires $O(100MB/s)$ for efficient output
- The average I/O throughput of Jaguar (now Titan) is around 4.7MB/s per node
- Large and growing gap between the application's requirement and system capability

Motivation - Reducing Data Movement

- For large-scale simulations, reducing data movement over network is essential for performance improvement
- E.g. active storage demonstrated a potential for analysis applications (read-intensive)
  - Moving analysis kernel near to data
  - Transfer small size results over the I/O network
- What about scientific simulations?
  - Write a large volume of data
  - Data need to be stored for future processing and analysis
  - Storage capacity and bandwidth demands

Motivation - Data Deduplication

- Widely used in backup storage systems to reduce storage requirement
- Eliminates the duplicated copies of data using a signature representation
- Identical segment deduplication most widely used
  - Breaks data or stream into contiguous segments
  - Utilizes the Secure Hash Algorithm (SHA) to calculate a fingerprint for each segment
- Impressive performance in terms of both compression ratio and throughput, e.g. up to 20:1 compression ratio

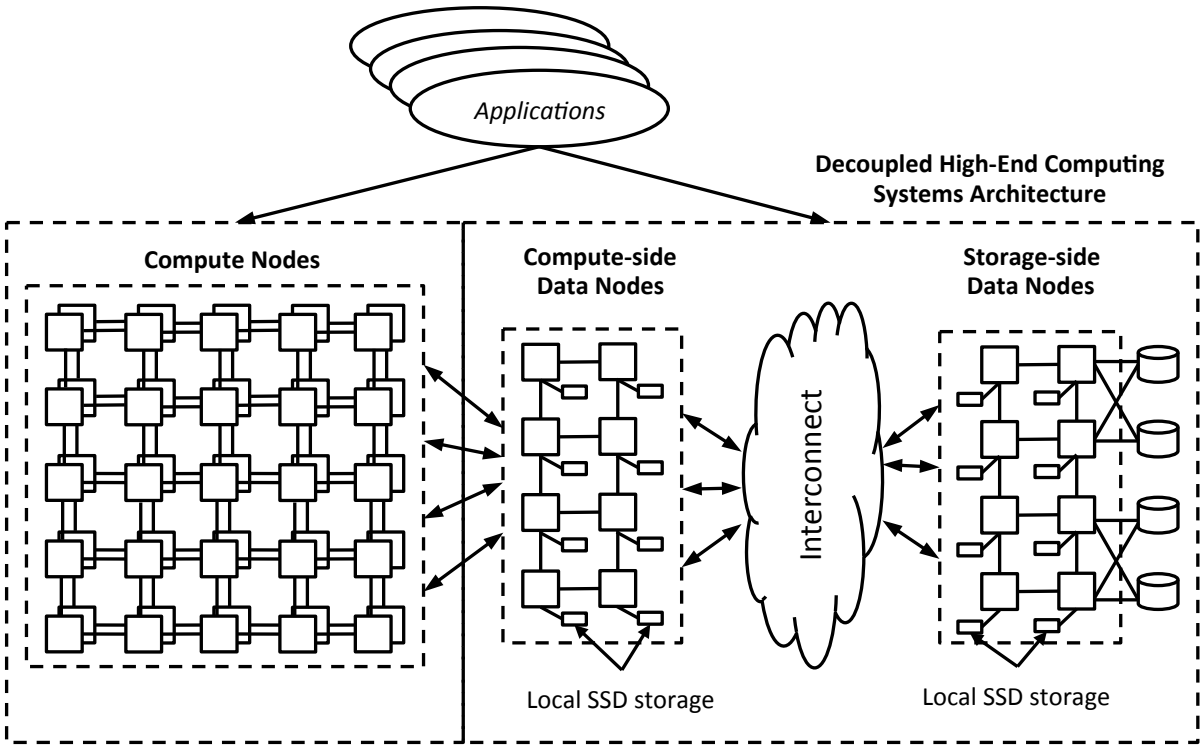
Data Deduplication in a Hybrid Architecture (DDiHA)

We explore utilizing data deduplication to reduce the data volume for write intensive applications on the data path:

- Using dedicated nodes (deduplication nodes) for deduplication operations
- Building a deduplication framework before data is transferred over the network
- Evaluating the performance with both theoretical analysis and prototyping

Decoupled Execution Paradigm

- DDiHA is designed based on a decoupled system architecture
- Explores how to use the compute-side nodes to improve the write performance



Decoupled HPC System Architecture

DDiHA Overview

- Dedicated deduplication nodes serve for compute nodes and connected through high-speed network
- Deployed with a global address space library to support a shared data deduplication approach
- Fingerprints are organized in a global shared table (global index) for a systematic view and improving deduplication ratios
- Writes going through streaming deduplication API (SD API, an enhanced MPI-IO API)
- Deduplicated data written to storage through a PFS API

DDiHA Architecture

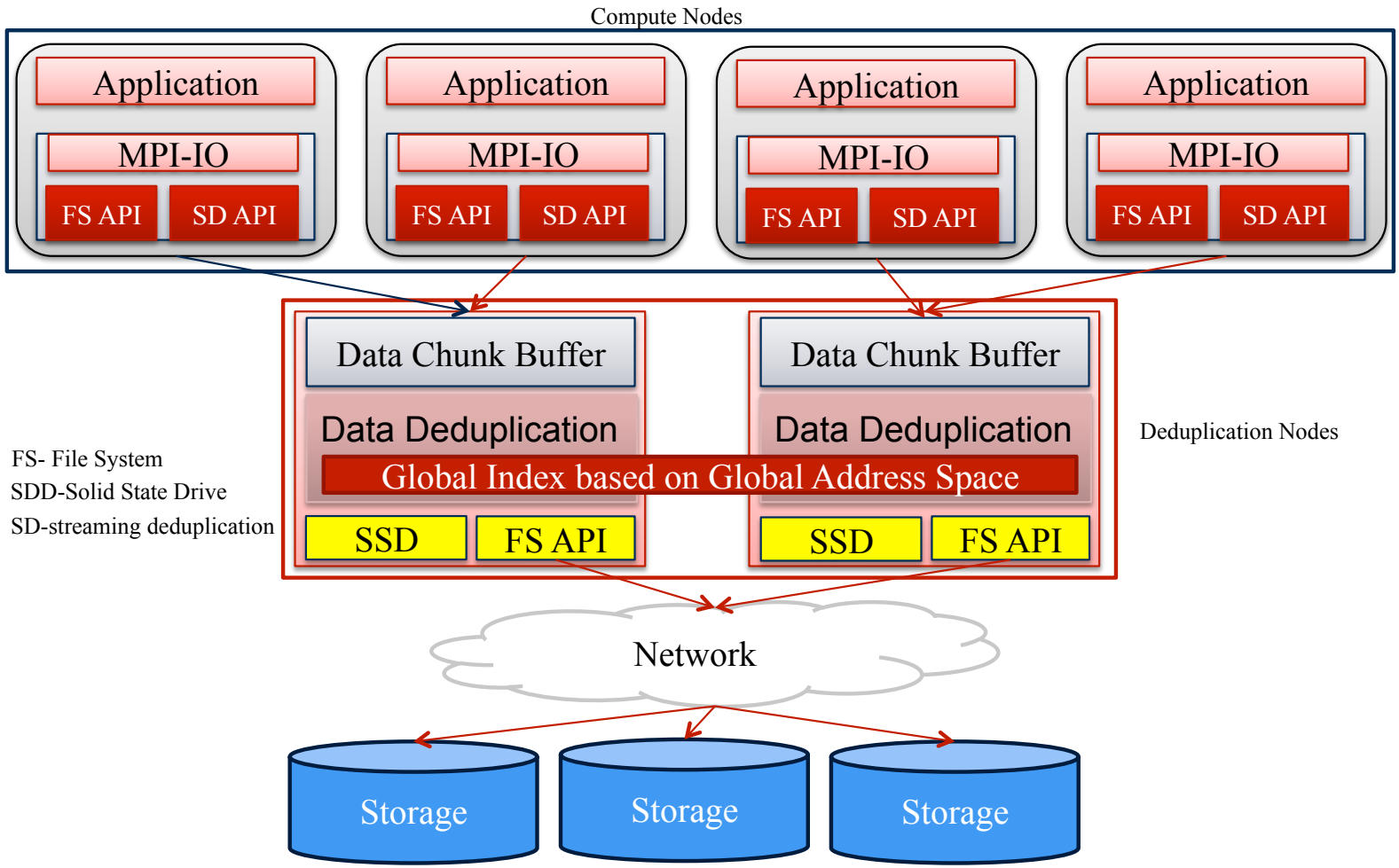


Figure 5: Architecture of DDiHA

DDiHA API

- Two APIs: *SD_Open* & *SD_Write*. (No read)
- MPI-IO open/write was revised
- Users need to add prefix "SD" to the file name when open a file
- Data will be directed to the deduplication nodes until file is closed

sample code

```
MPI_Comm com;
char *filename="SD:PVFS2:/path/data";
MPI_Info info;
MPI_File fh;
MPI_File_open(com, filename, ...)
...
```

DDiHA Runtime

- Uses the identical segment deduplication algorithm for deduplications
  - Divides data stream into fixed size segments, e.g. 8KB
  - Computes fingerprints with SHA1 algorithm for each segment
  - Compared against stored fingerprints or inserted as a new one
- Deduplication nodes maintain a global fingerprint table
- Fingerprint table can be implemented as a binary search tree that distributes across different deduplication nodes

Theoretical Modeling

Aim to answer following questions:

- What are the performance gain/loss?
- What are desired configurations for deduplication nodes?

Methodology:

- Assume deduplication nodes are the same with compute nodes
- Assume the task of an application can be divided into two parts: computation and I/O
- Compare the performance with traditional system given same resources

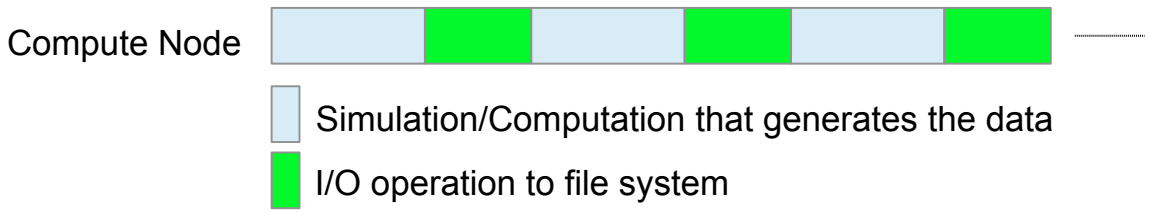


Figure 6: Execution Model of Scientific Applications

Theoretical Modeling

Execution Model with DDiHA:

- Deduplication cycle longer than computing cycle
- Deduplication cycle shorter than computing cycle
  - Expected, overlapping computation, deduplication, and I/O
  - Requirement of deduplication efficiency

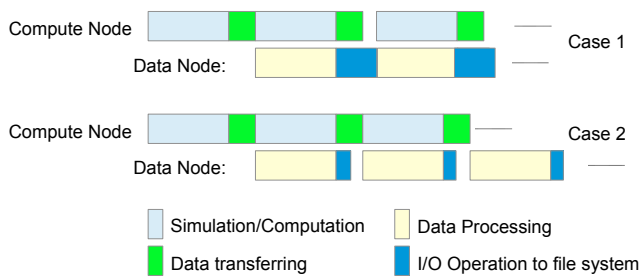


Figure 7: Execution Model with DDiHA

Notations:

W	Workload, data generated by simulation, measured with MB
W'	Output data volume of deduplication
C	Simulation/Computation capability each compute node for given operation, measured with MB/s.
C'	Deduplication efficiency of each deduplication node for given operation, measured with MB/s.
b	Average I/O bandwidth of each compute node in current architecture.
b_h	Bandwidth of high speed network between compute nodes and deduplication nodes.
r	Ratio of data nodes compared to compute nodes.
n	Number of nodes.
k	The number of compute-I/O phases of an application.
λ	b_h/b
φ	W/W'

Theoretical Modeling

Traditional Performance:

$$P = \frac{1}{T} \quad (1)$$

$$T = (\frac{W}{n \cdot C} + \frac{W}{n \cdot b}) \cdot k \quad (2)$$

DDiHA Performance:

$$P' = \frac{1}{T'} \quad (3)$$

$$T' = (\frac{W}{n \cdot (1-r) \cdot C} + \frac{W}{n \cdot (1-r) \cdot b_h}) \cdot k + \frac{W}{n \cdot r \cdot C'} + \frac{W'}{n \cdot r \cdot b} \quad (4)$$

Overlapping Requirement:

$$\frac{W}{n \cdot r \cdot C'} + \frac{W'}{n \cdot r \cdot b} \leq \frac{W}{n \cdot (1-r) \cdot C} + \frac{W}{n \cdot (1-r) \cdot b_h} \quad (5)$$

Performance gain:

$$\Delta = \frac{P'}{P} = \frac{(\frac{W}{n \cdot C} + \frac{W}{n \cdot b}) \cdot k}{(\frac{W}{n \cdot (1-r) \cdot C} + \frac{W}{n \cdot (1-r) \cdot b_h}) \cdot k} = \frac{\lambda(C+b)(1-r)}{\lambda b + C} \quad (6)$$

Theoretical Analysis

Baseline:

- Platform: Jaguar XT5
  - I/O bandwidth per node: 4.67MB/s
- Application: GTC
  - Compute capacity at Jaguar XT5: 1.08MB/s

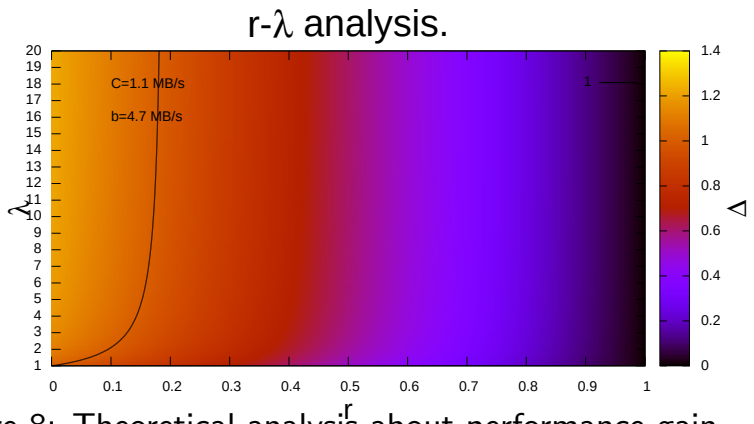


Figure 8: Theoretical analysis about performance gain. Assuming deduplication is efficienct enough for overlapping, DDiHA can achieve better performance by configuring appropriate deduplication nodes (around 10%)

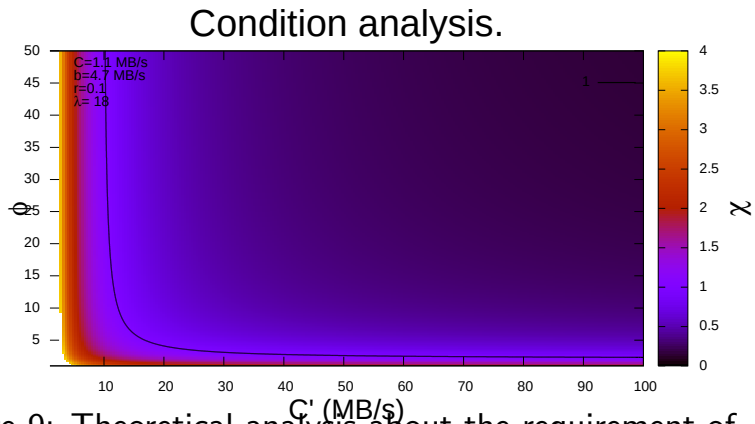


Figure 9: Theoretical analysis about the requirement of deduplication efficiency. If the deduplication can reduce the data size more than 10%, The deduplication efficiency requirement is no more than $20MB/s$, easy to be achieved

Platform and Setup

Platform:

Name	DISCFarm Cluster
Num. of nodes	16 nodes
CPU	dual quad-core 2.6GHz
Memory	4GB per node

Setup:

	Traditional	DDiHA
Computing cores	42	40
Deduplication cores	0	2
Total cores	42	42
Methodology	simulating GTC application behavior	
Data size	260MB – 8GB	

Deduplication Bandwidth

- 65MB/s processing capacity/bandwidth without any optimization
- Meet the basic requirement analyzed from theoretical modeling (20MB/s)

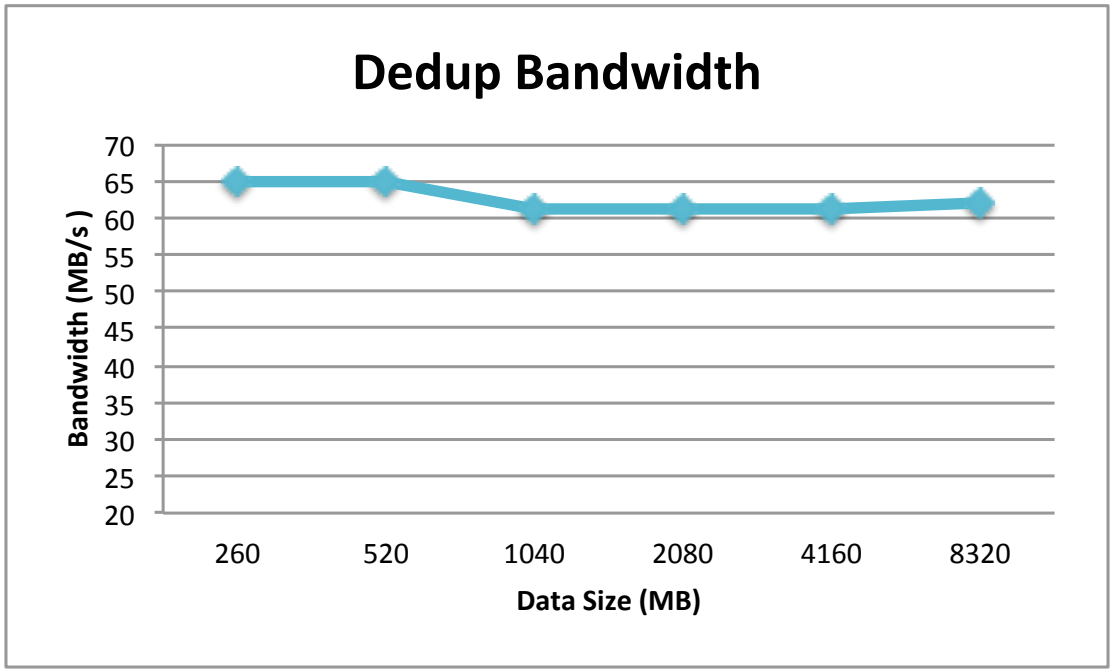


Figure 10: Efficiency of data deduplication.

Compression Ratio

- Analyzed and compared compression ratio
- Observed different compression ratios (from 0 (dataset4) to 16 (dataset3)) for different data sets.

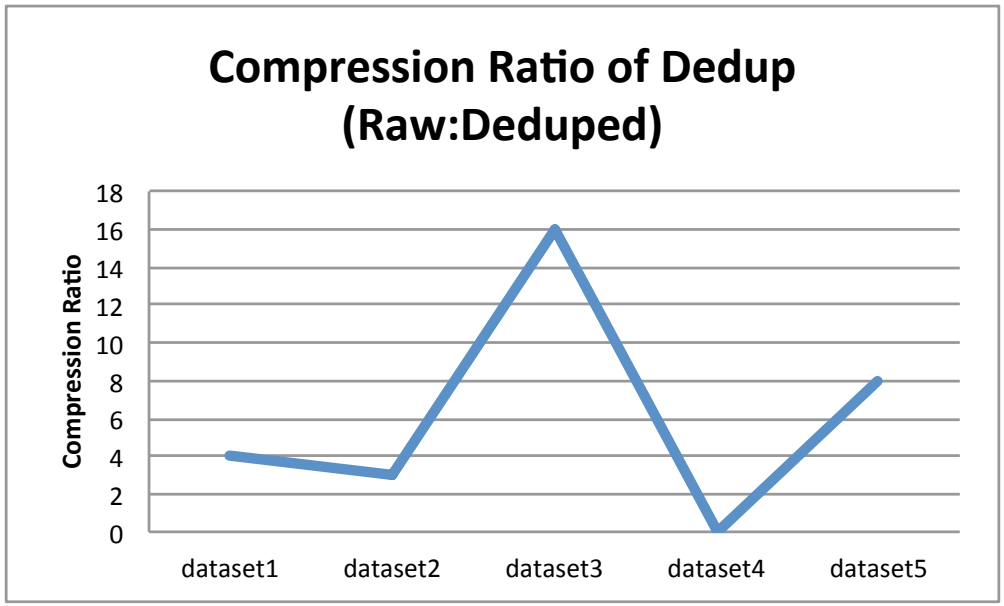


Figure 11: Compression ratio of data deduplication for different data sets.

I/O Improvement

- For the traditional paradigm, it measures the time of writing data to the storage
- For DDiHA, it measures the time of writing data to deduplication nodes (I/O from application's view)

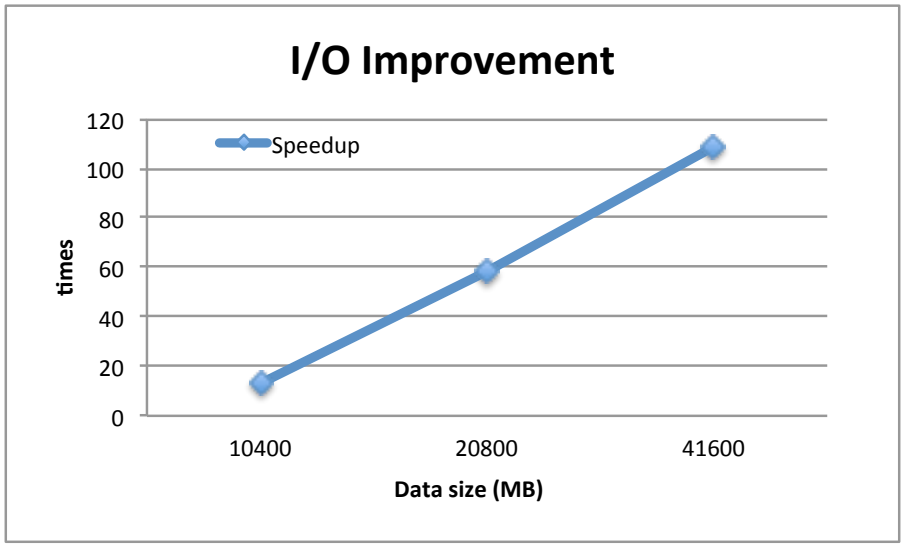


Figure 12: I/O improvement from application's view

Overall Performance Comparison

- Measures the total execution time of an application (a simulation of GTC in this work)
- Both computation and I/O were included, data was transfered to the storage
- DDiHA reduced more than 3 times than the traditional paradigm

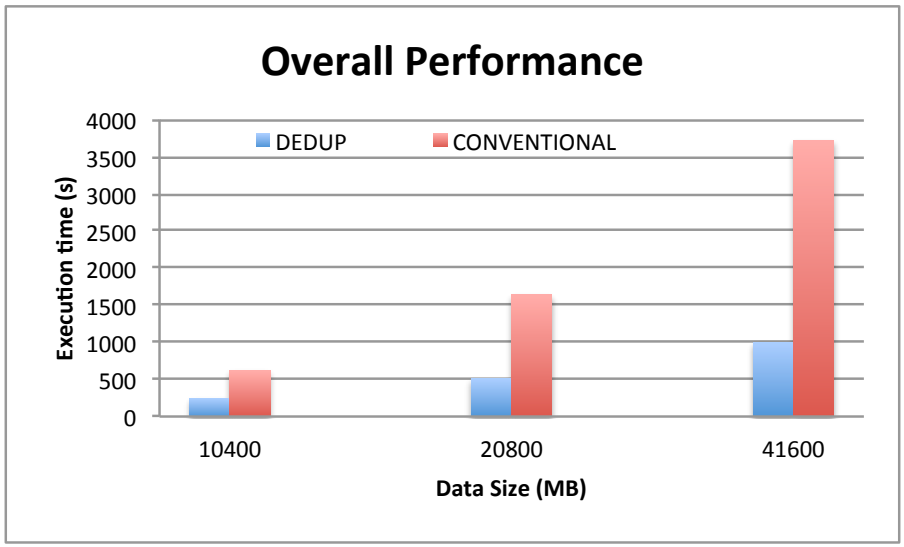


Figure 13: Overall performance comparison between DDiHA and traditional paradigm.

Conclusion and Future Work

- Big data computing brings new opportunities but also poses big challenges
- Trading part of computing resources for data reduction can be helpful and critical for system performance
- An initial investigation of data deduplication on the write path for write-intensive applications
  - Beneficial because of efficiency and compression ratio
- Theoretical analysis and prototyping were conducted to evaluate the potential of DDiHA
- Plan to further explore deduplication algorithms and how to leverage features of scientific datasets to achieve high compression ratio

Any Questions? Thank You!
Welcome to visit: http://discl.cs.ttu.edu/