High Level file system and parallel I/O optimization of DNS Code

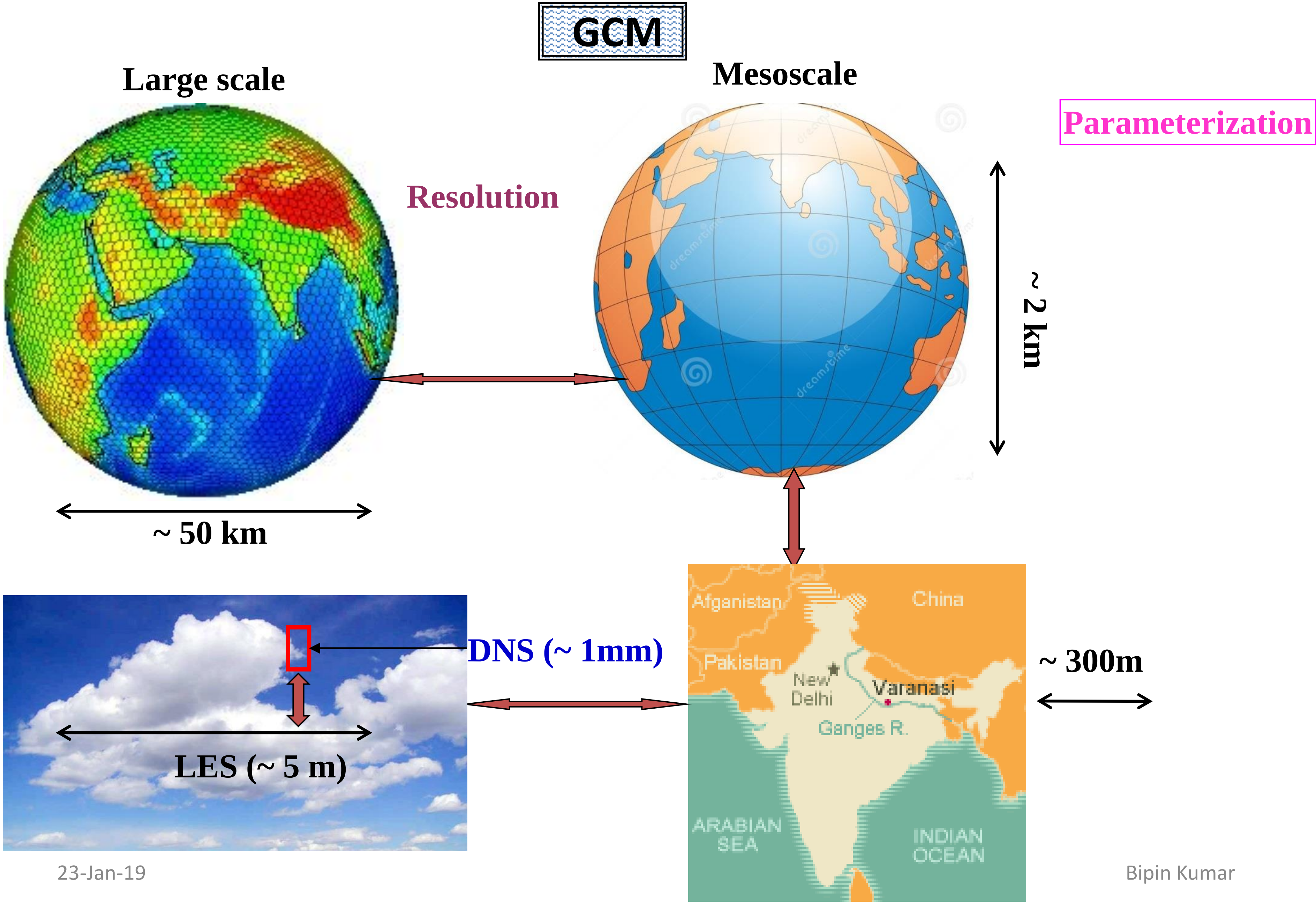
Bipin Kumar[1], Nachiket Manapragada[2] and Neethi Suresh[1]

[1]Indain Institute of Tropical Meteorology, Pune, India
[2]Cray Supercomputers (India) Pvt Ltd., Pune India

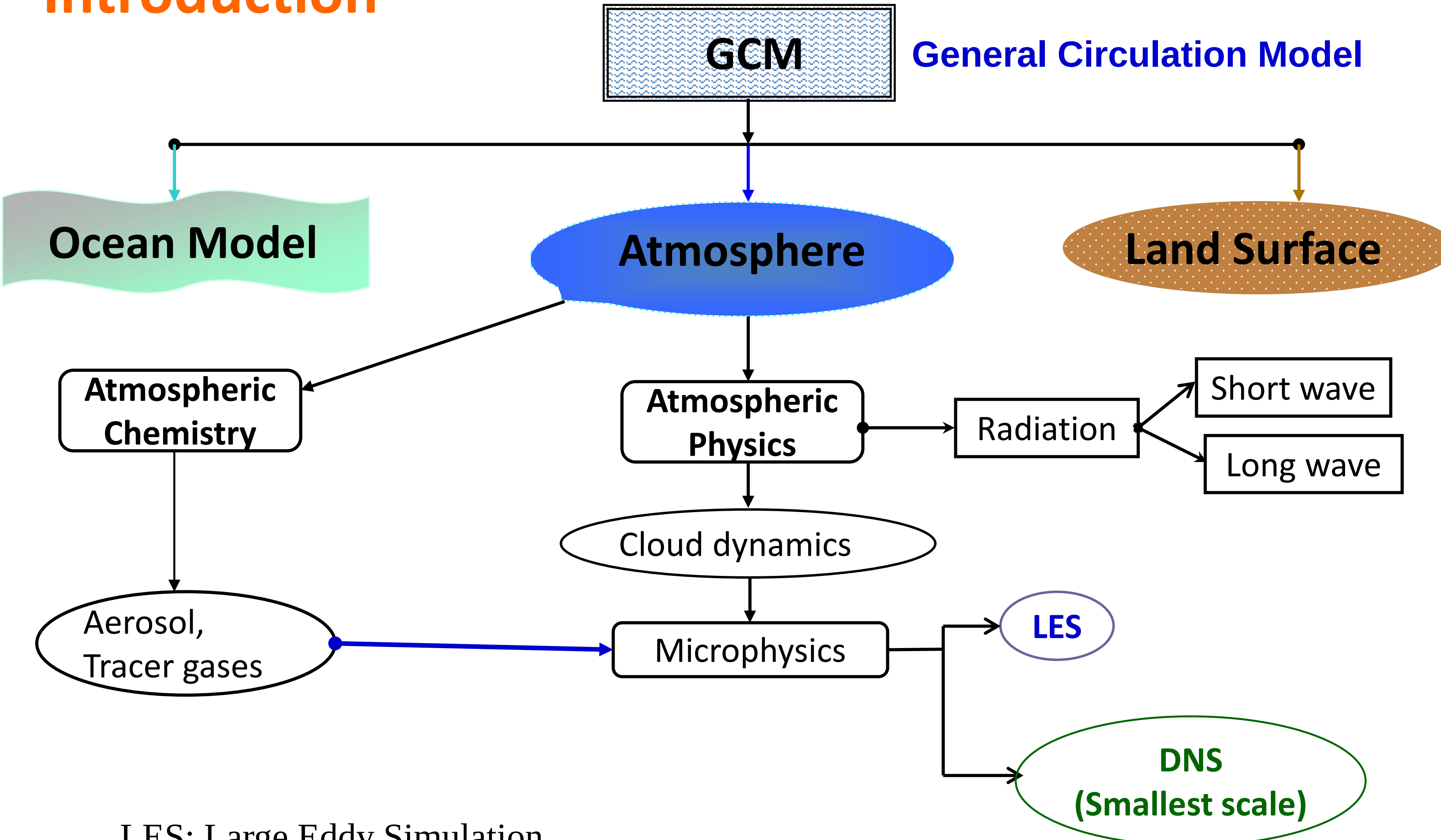
Contents

- Introduction
- Mathematical model
- Computation details
- Motivation for I/O optimization
- Optimization techniques
- Result and conclusions

Introduction: Simulation in Atmospheric science



Introduction



LES: Large Eddy Simulation

DNS: Direct Numerical Simulation

Mathematical Model

Eulerian (Fluid flow equations)

$$\nabla \cdot \mathbf{u} = 0$$

$$\partial_t \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho_0}\nabla p + \nu \nabla^2 \mathbf{u} + g\left[\frac{T - T_0}{T_0} + \epsilon(q_v - q_{v0}) - q_l\right]\vec{e}_z + f_{LS}$$

$$\partial_t q_v + (\mathbf{u} \cdot \nabla) q_v = D\nabla^2 q_v - C_d$$

$$\epsilon = \frac{R_v}{R_d} - 1$$

$$\partial_t T + (\vec{u} . \nabla)T = \kappa \nabla^2 \vec{u} + \frac{L}{c_p} C_d$$

κ : thermal conductivity

R_v : vapor gas constant

R_d : dry air gas constant

q_v : vapor mixing ratio

q_l : liquid water content

f_{LS}: turbulent forcing

(form large scale)

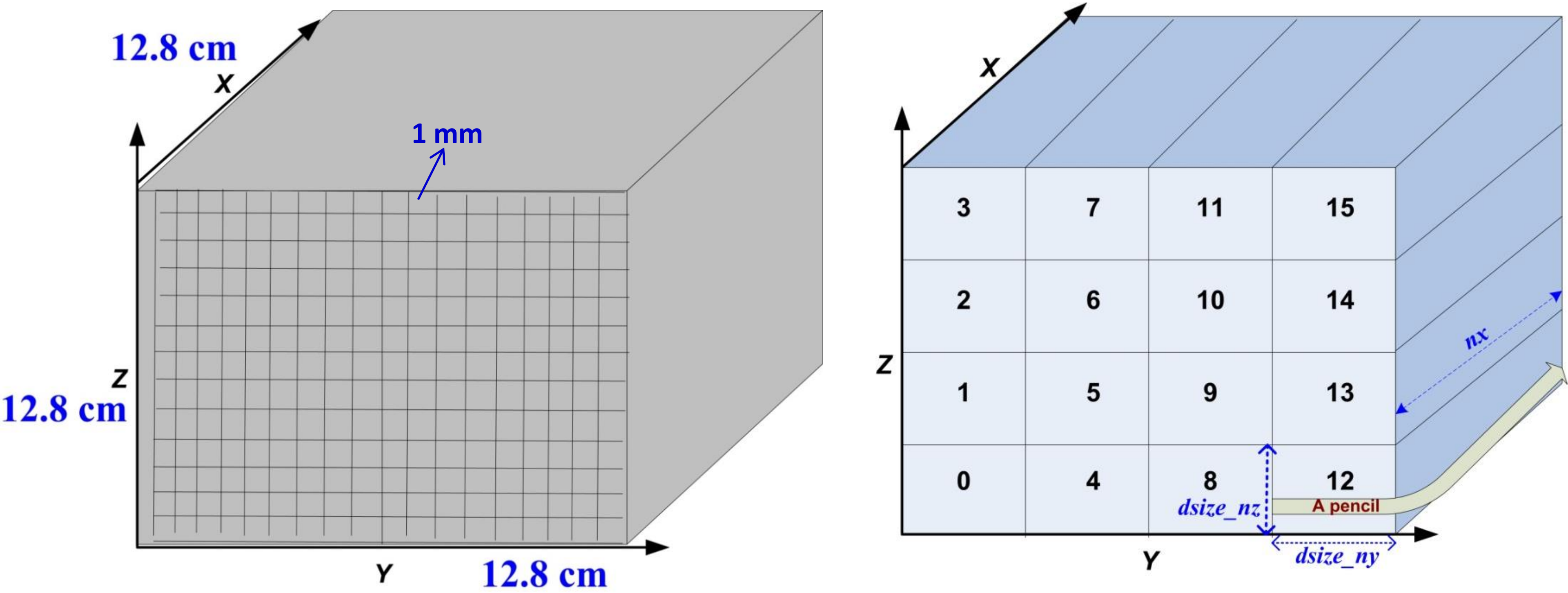
Periodic BC

Kumar et al., *JAS, 2014, JAMES, 2017, Götzfried et al., JFM 2017*

Computational Details

Domain decomposition on 2D processor topology

- Pseudo-spectral method used to convert Partial Differential Equations (PDE) to set of Ordinary Differential Equations (ODE).
- System ODE is solved by 2nd order Predictor-corrector method.



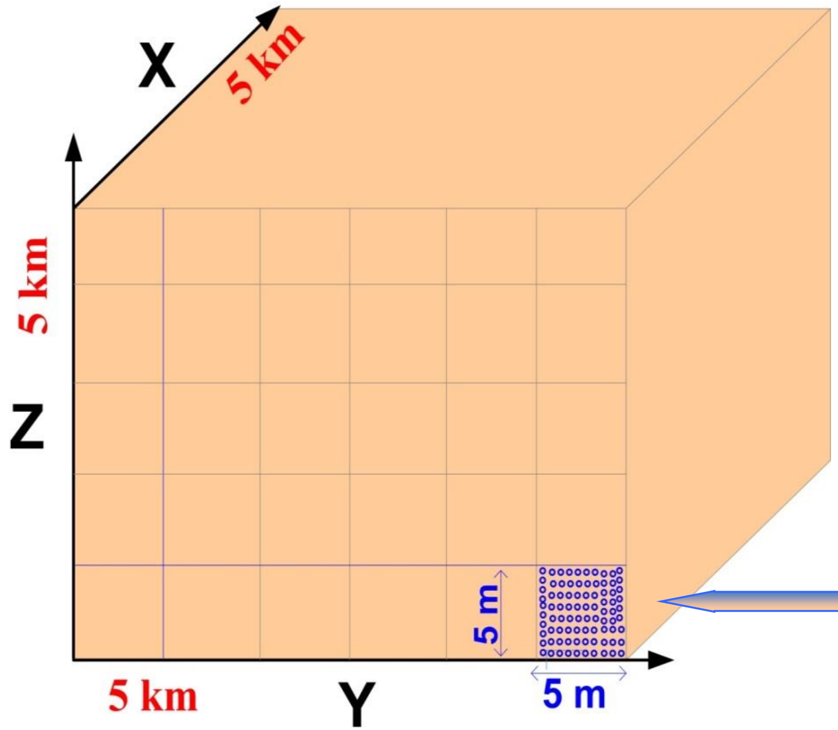
FORTRAN 90 + MPI + OpenMP

Motivation

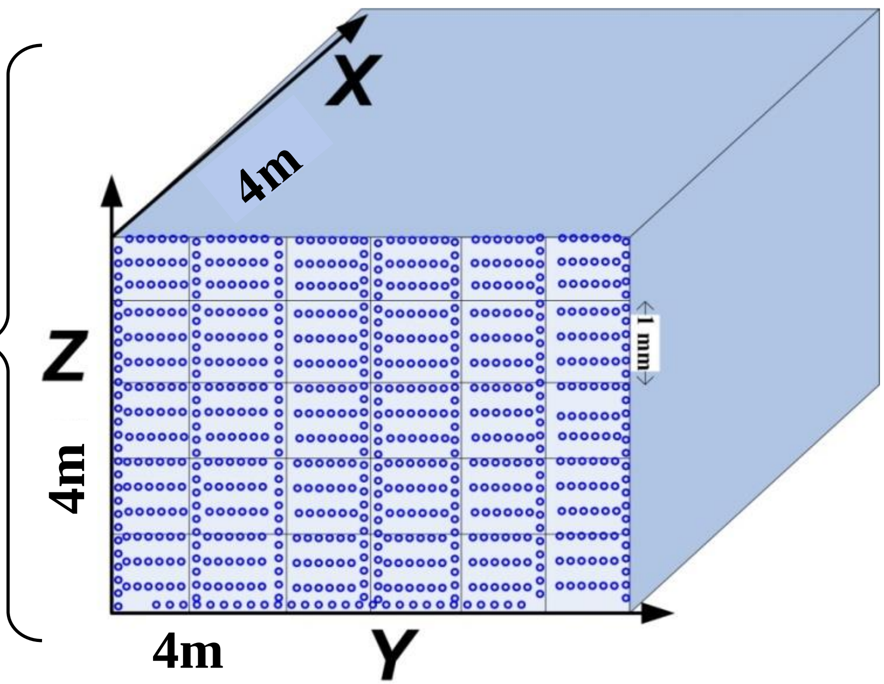
➢ Simulation in larger domain.

LES ← → DNS

Domain : $4m^3$
Resolution : 4096^3 grid cells
# of droplets : 3.46 billion
Time step : $2.0\,e^{-4}$



➢ Computationally more complex



Data file size : 2700 GB

Optimized parallel I/O is required.

Contributions:

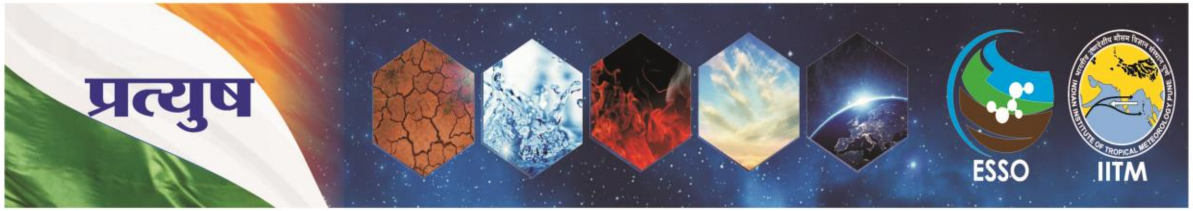
✓Better understanding of microphysics

✓Can provide seamless information to LES

✓Improvement of LES will be helpful for parameterization of large models.

IITM HPC, Pratyush

A 4.0 Petaflops supercomputer



Details:

Compute Node

Name	Details
Number of Nodes	3315
Number of cores	119340
Processor	Intel Xeon Broadwell E5-2695 v4 CPU (18 core, 2.1 GHz)
Memory Per Node	128 GiB Memory DDR4-2400 w/ Chipkill TM technology
Total Memory	414 TiB

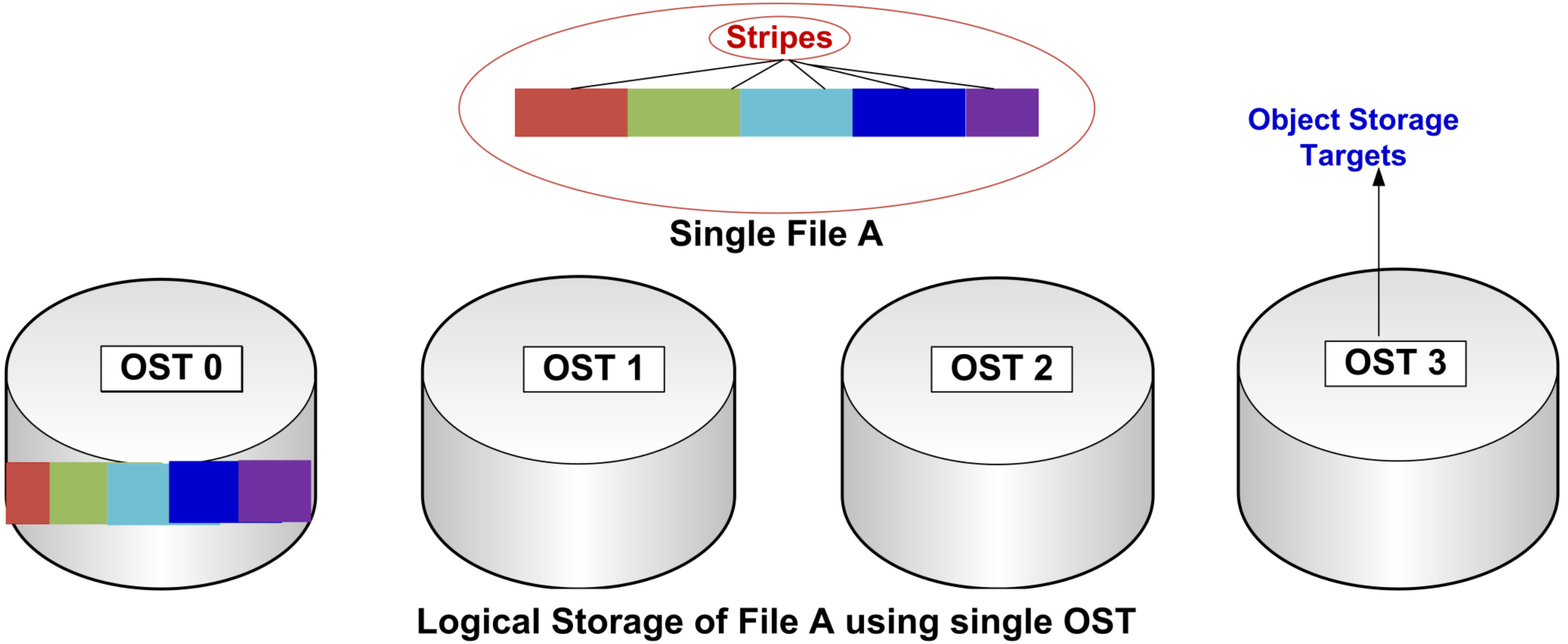
Accelerator Node

Name	Details
Number of Nodes	16
Accelerator	Intel KNL 7210, self-hosted mode, single socket per node
Memory Per Node	96 GiB DDR4-2133 w/ Chipkill TM technology
Total Peak Performance	42.56 TFLOPS

Lustre File system

Lustre = Linux + Cluster

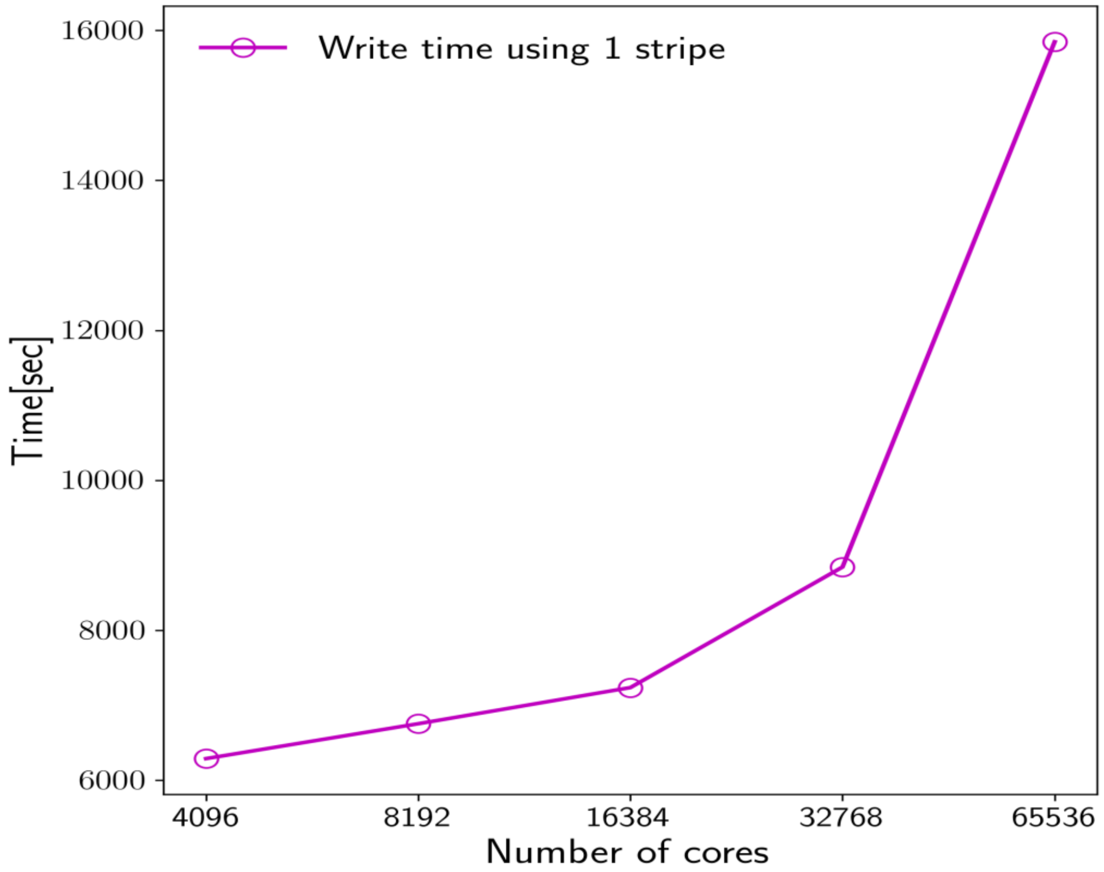
- Parallel distribute file system
- Used in large-scale cluster computing
- Highly scalable
- Can support several thousands nodes
- Multi-petabytes storage with 100s GB per sec I/O throughput.
- Set of many small file systems called OST.



Lustre File system

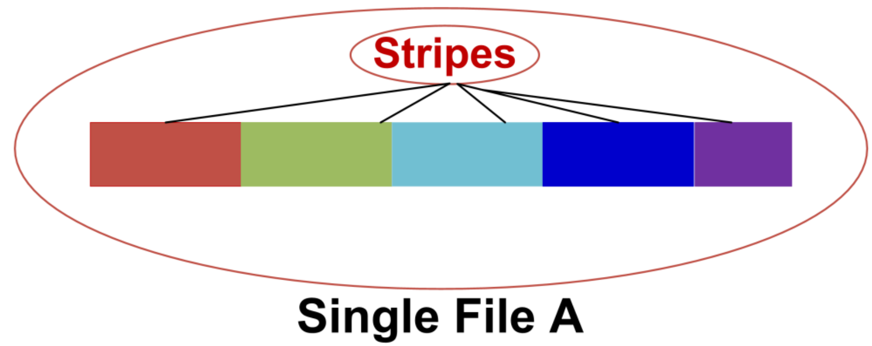
File processing time:

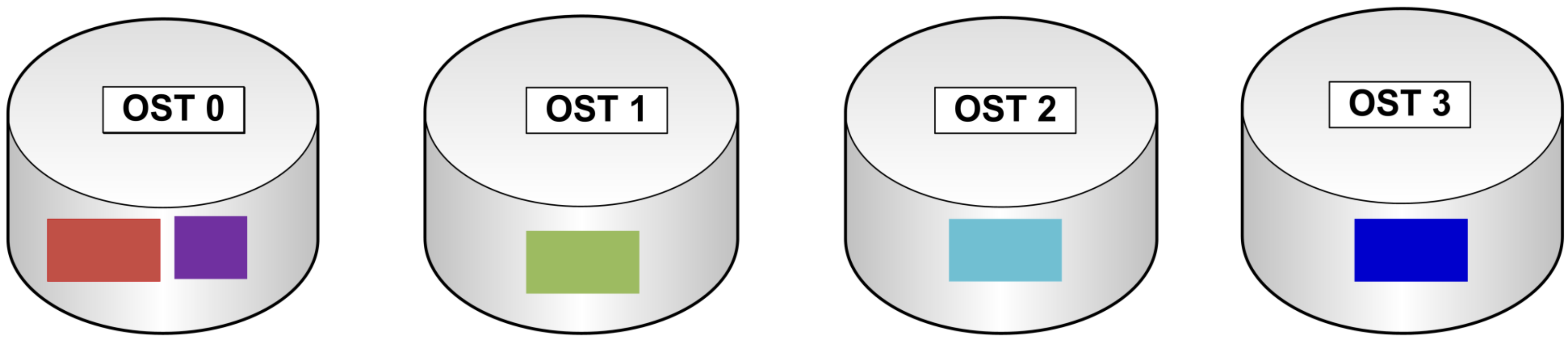
- **File size is 2.7 TB**
- **Reading/Writing time can go up to 4 hours.**
- **Increases total simulation time.**
- **Parallel I/O optimization is required.**



Lustre Optimization: striping

- File can be striped on Lustre
- Transparently divided in to chunks
- Read/write simultaneously using load balancing.





Logical Storage of File A using multiple OSTs

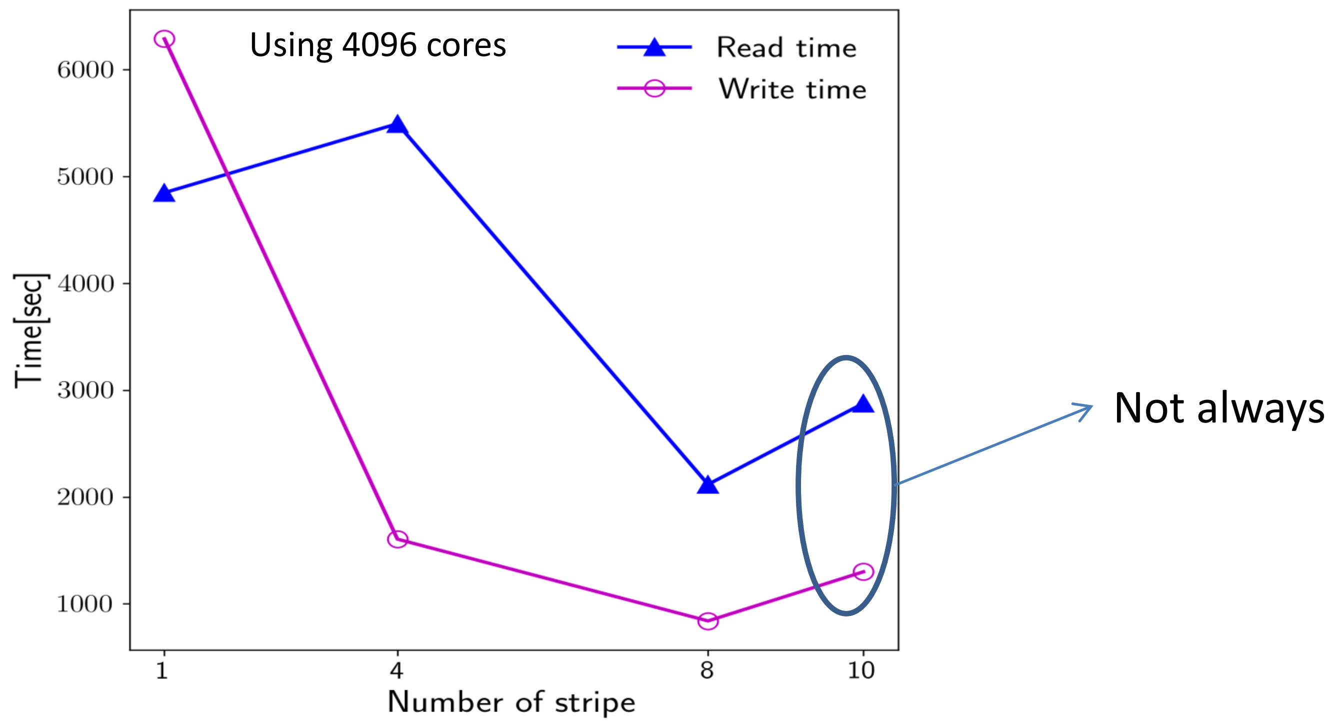
Advantages:

- Increase bandwidth utilization
- To store large file compare to single OST.

Lustre Optimization: striping

Dis-advantages:

- Increase overhead due to network operations & server contention.
- Load on OST by another application can cause bottleneck.
- **An optimal number of stipe is required.**

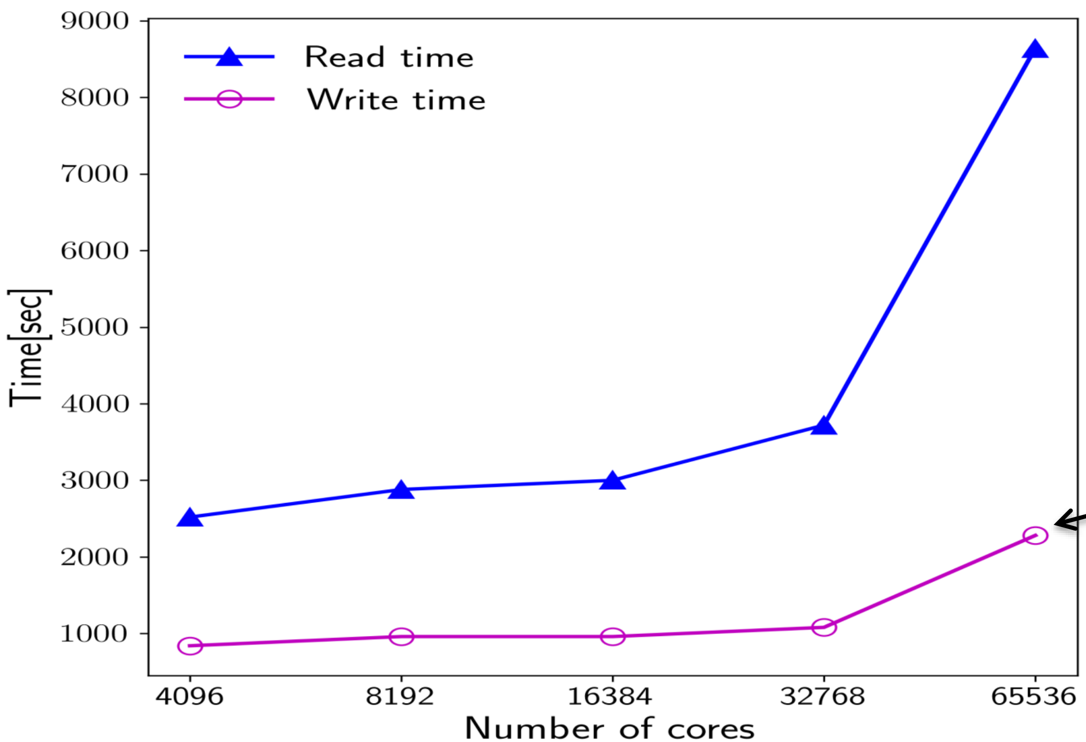


Experiment with up to 10 OSTs.

Result: Parallel I/O optimization

Striping

- Scaling of file processing using 8 OSTs
- Increasing trend in file processing time



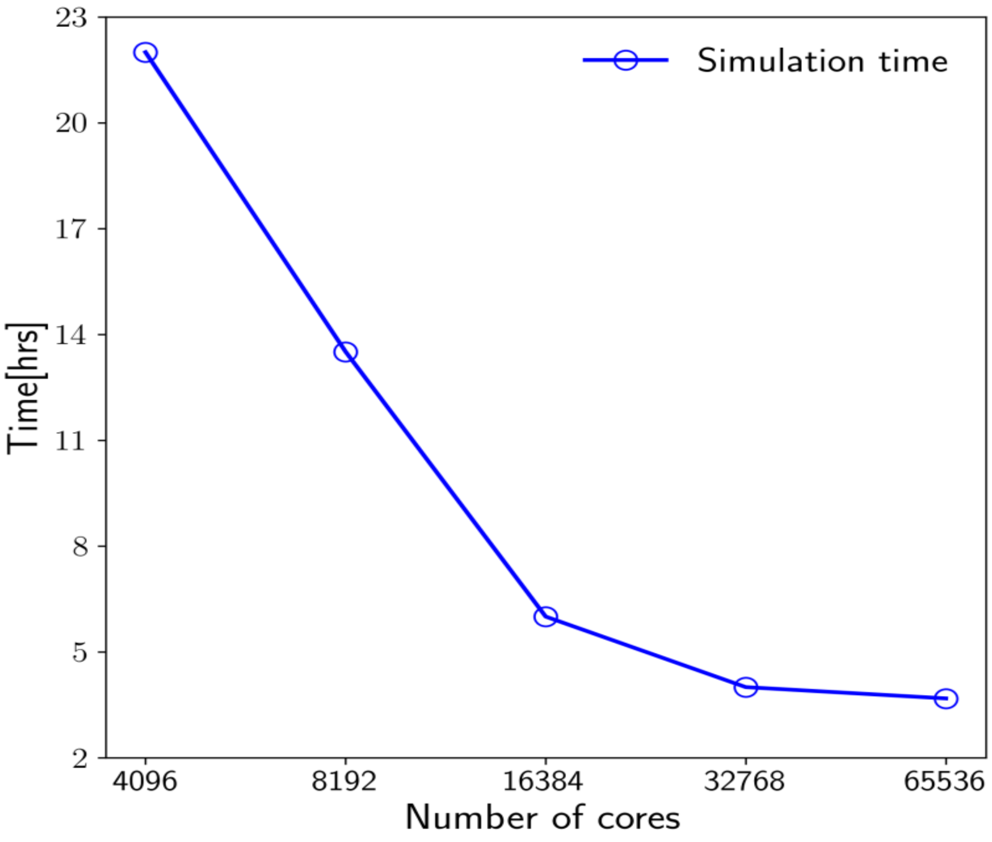
No. of cores	Reading time	Writing time
4K	2250	840
8K	2880	960
16K	3000	960
32K	3720	1080
65K	8640	2280

4 hours → 40 minutes

Result:

Scaling: with striping

- Linear speedup till 16384 cores with.



Result: Parallel I/O optimization

IOBUF

- Library; enables asynchronous caching and prefetching.
- No source code modification required.
- IOBUF optimization reduced 14% total time.

#4096 Cores	I/O Operations	Total time (sec)
No IOBUF	2 reads + 2 writes	23120
With IOBUF	2 reads + 2 writes	19908 | (14% reduction)

Vectorization

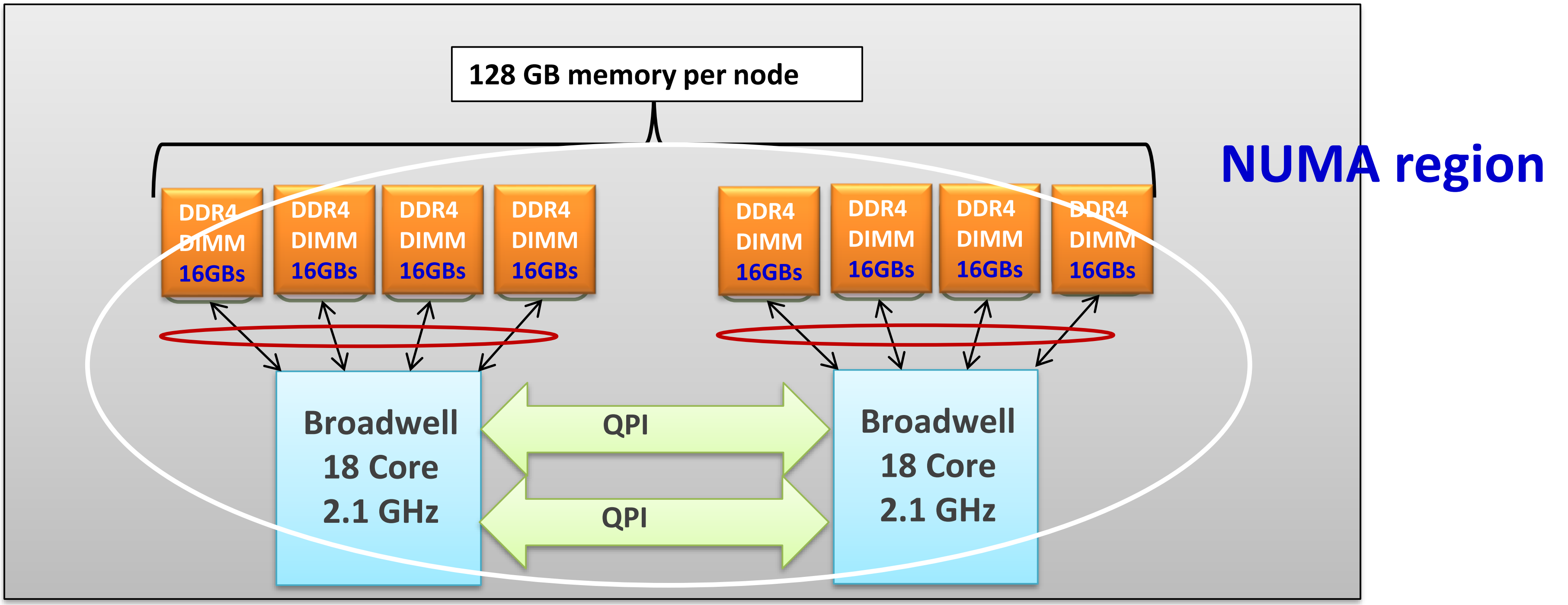
- Pratyush HPC has Broadwell processor which supports AVX2.
- Provided 5% speedup in total time.

#4096 Cores	I/O Operations	Total time (sec)
AVX	1 reads + 1 writes	9005
AVX2	1 reads + 1 writes	8555 | (5% reduction)

Result: NUMA optimization

Cray XC40 Compute Node

NUMA= Non uniform memory access



- Used 24 MPI cores for optimal RAM utilization within socket.
- Restricted 12 cores per CPU to avoid access of memory from another processor.
- Up to 38% reduction in simulation time.

# Cores	With NUMA
4096	6%
8192	15%
16384	17%
32768	38 %

Conclusion and outlook

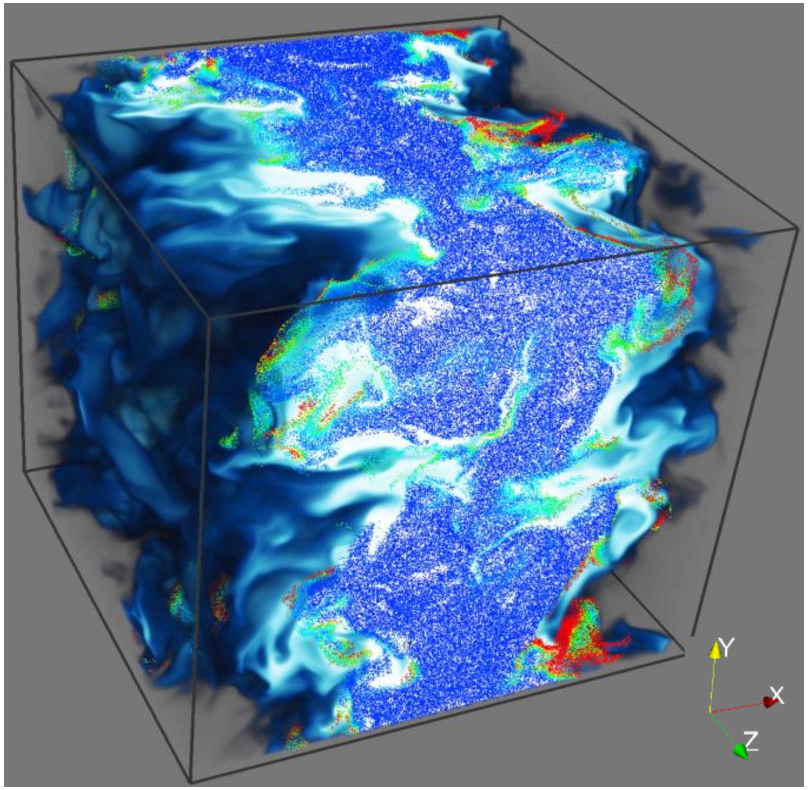
- Optimized DNS code using parallel I/O optimization.
- Four optimization techniques attempted.
  - ✓ Striping in Object Storage Targets (drastically reduced file processing time)
  - ✓ IOBUF optimization provided 14% reduction.
  - ✓ AVX2 added 5% more time reduction.
  - ✓ NUMA optimization gave 38% speedup.

- Overall scaling has shown linear speedup till 16K cores.

- Further experiment
  - ✓ Hyper threading
  - ✓ Multithreading
  - ✓ Use of advanced MPICH options
- Increased number of aggregators per OST
- Experiment on file system with more than 10 OSTs.

Acknowledgment:

- HPCS facility, IITM Pune, India.
- Director, IITM Pune.
- Manager, Cray Supercomputers, India



Thank you for your attention

Questions ?

Result: compiler level optimization

NUMA + AVX2

- Scaling simulation time NUMA and AVX2.
- Linear speedup till 16384 cores.