# Performance Advantages of Using a Burst Buffer for Scientific Workflows

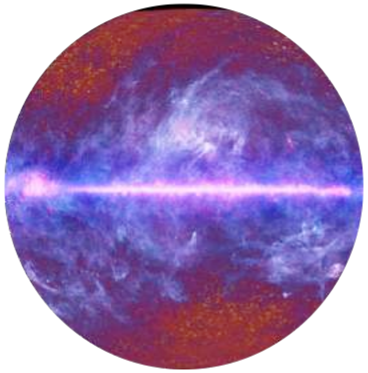
**Andrey Ovsyannikov**

NERSC, Lawrence Berkeley National Laboratory

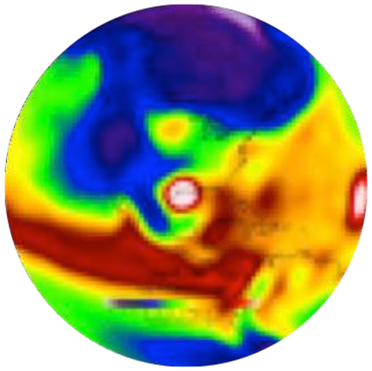
with David Trebotich, Brian Van Straalen (ANAG, LBNL)

BASCD-2016: Bay Area Scientific Computing Day
December 3, 2016. Stanford, CA

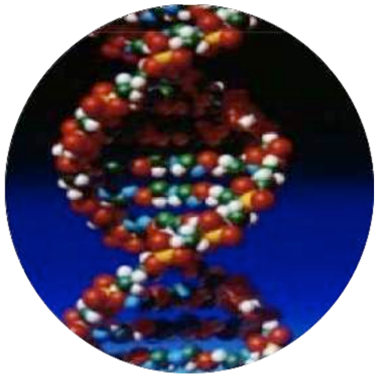
## Data-intensive science



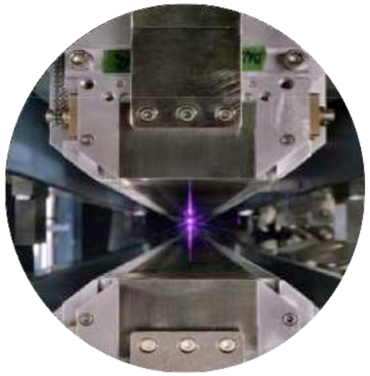
Astronomy



Climate



Genomics



Light Sources

- Applications analyzing data from experimental or observational facilities (telescopes, accelerators, etc.)
- Applications combining modeling/simulation with experimental/observational data
- Applications with complex workflows that require large amounts of data movement

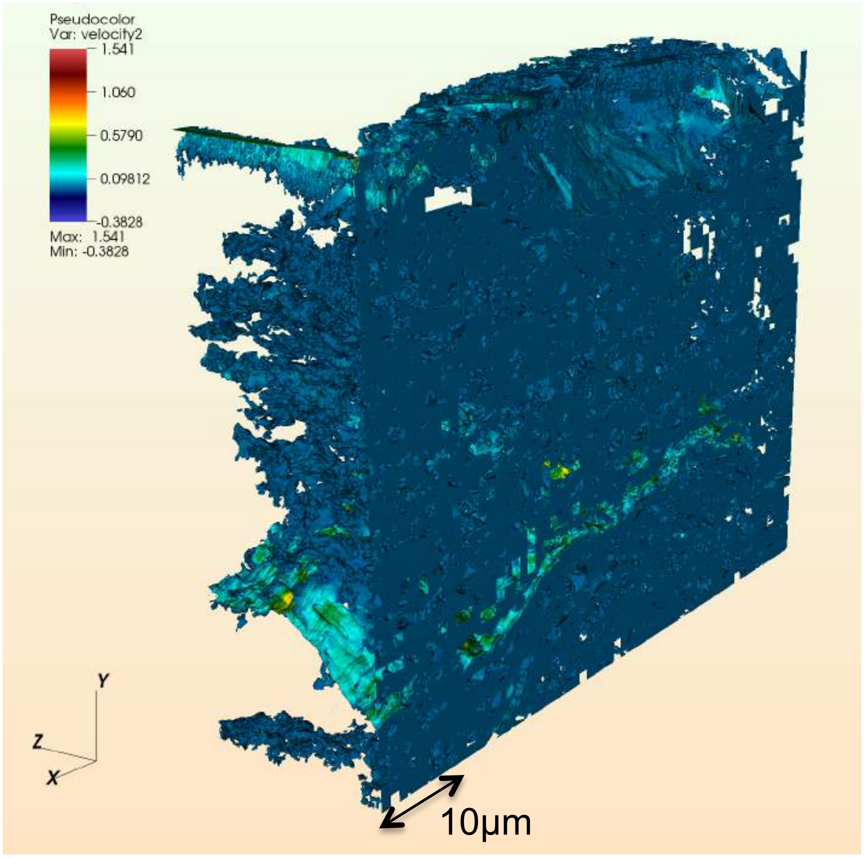
# Data-intensive simulation at scale

**Example**: Reactive flow in a shale

- Required computational resources: **41K cores**
- Space discretization: **2 billion cells**
- Time discretization: **~1μs; in total 3*10$^4$ timesteps**
- Size of 1 plotfile: **0.3TB**
- Total amount of data: **9PB***
- I/O: **61%** of total run time
- Time to transfer data:
  - to GlobusOnline storage: >**1000 days**
  - to NERSC HPSS: **120 days**

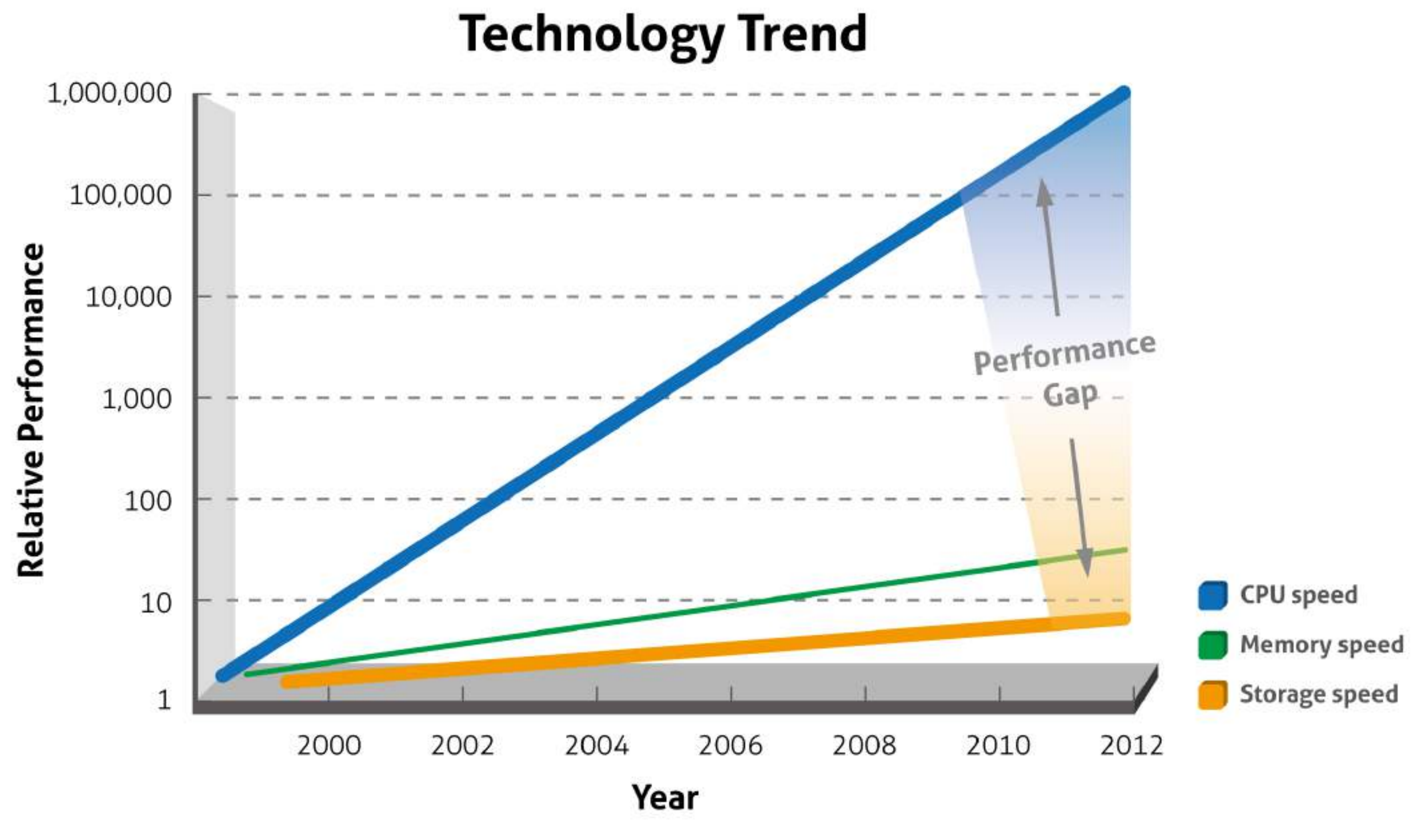
*Assuming that the plotfile is written at every timestep

Sample of California's Monterey shale



Complex workflow:
On-the-fly visualization/quantitative analysis
On-the-fly coupling of pore-scale simulation with continuum scale model

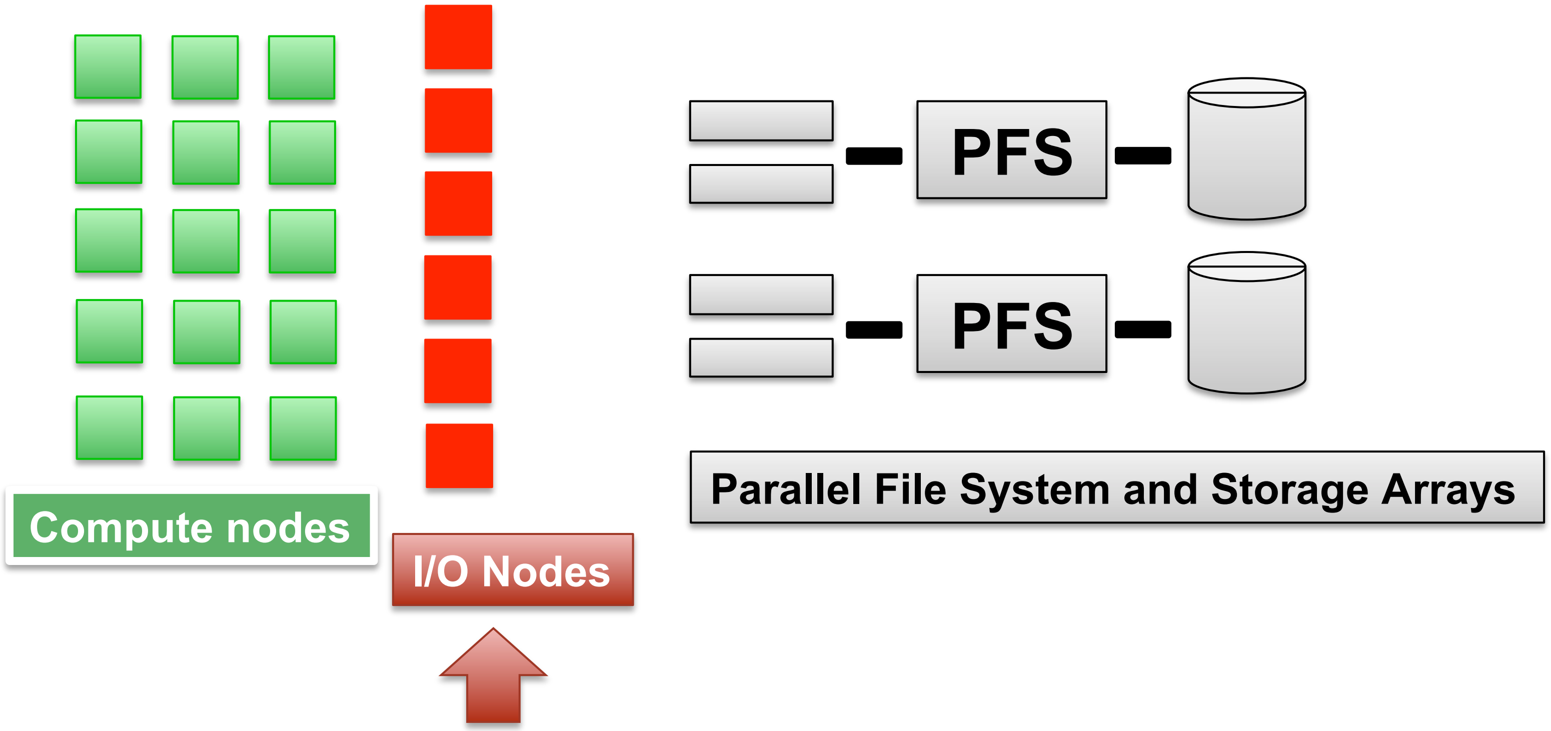
## Bandwidth gap



**Growing gap between computation and I/O rates.**
**Insufficient bandwidth of persistent storage media.**

# What is a burst buffer?

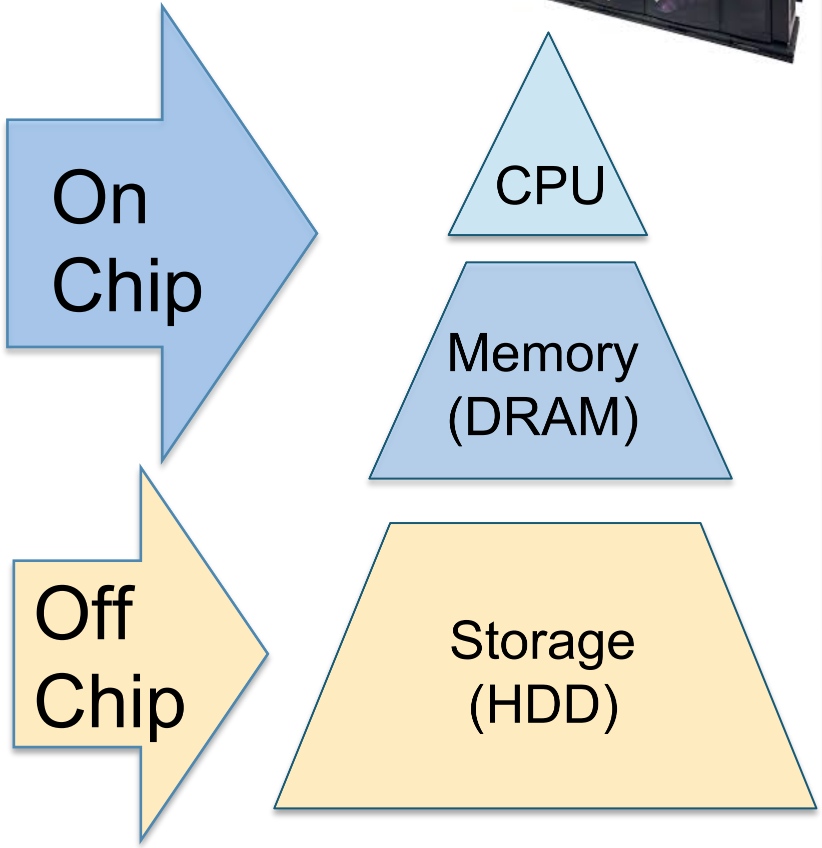
Layer of SSDs which resides between compute nodes and parallel file system



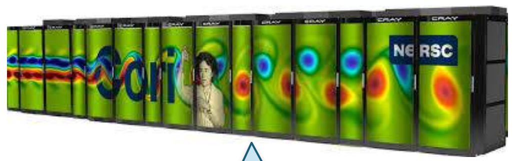
**SSD placement**

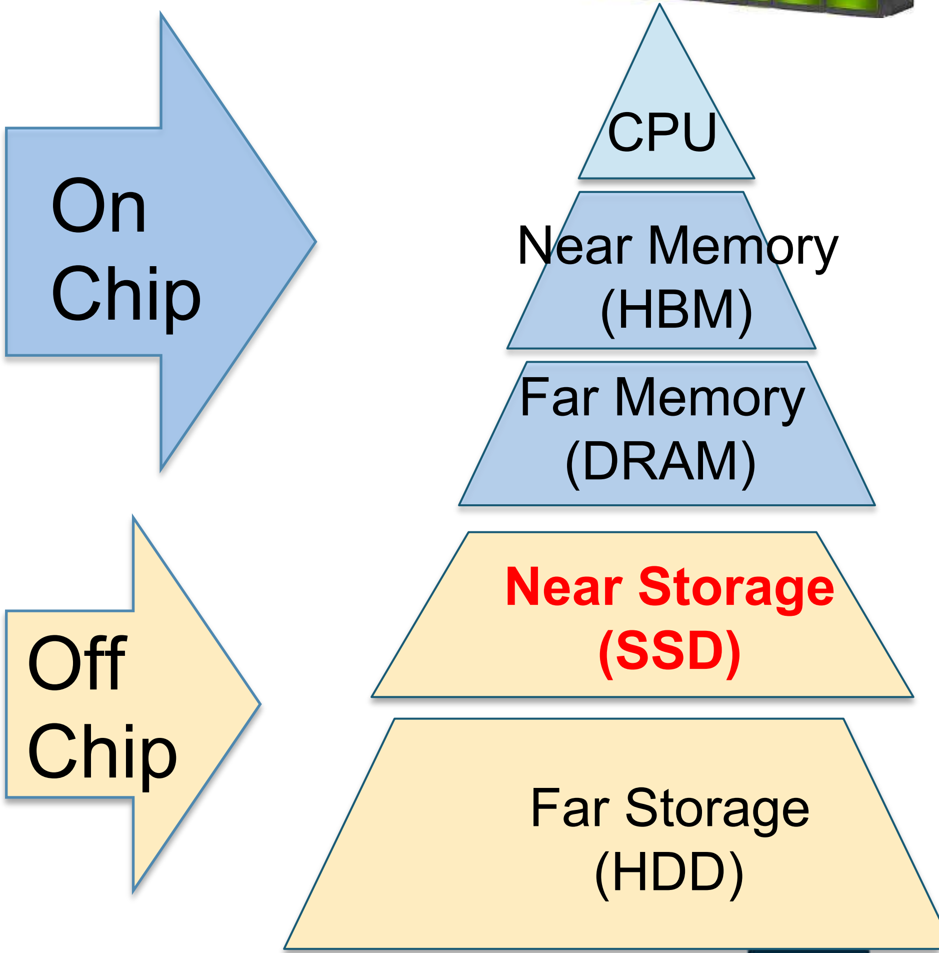
# HPC memory hierarchy

## Past



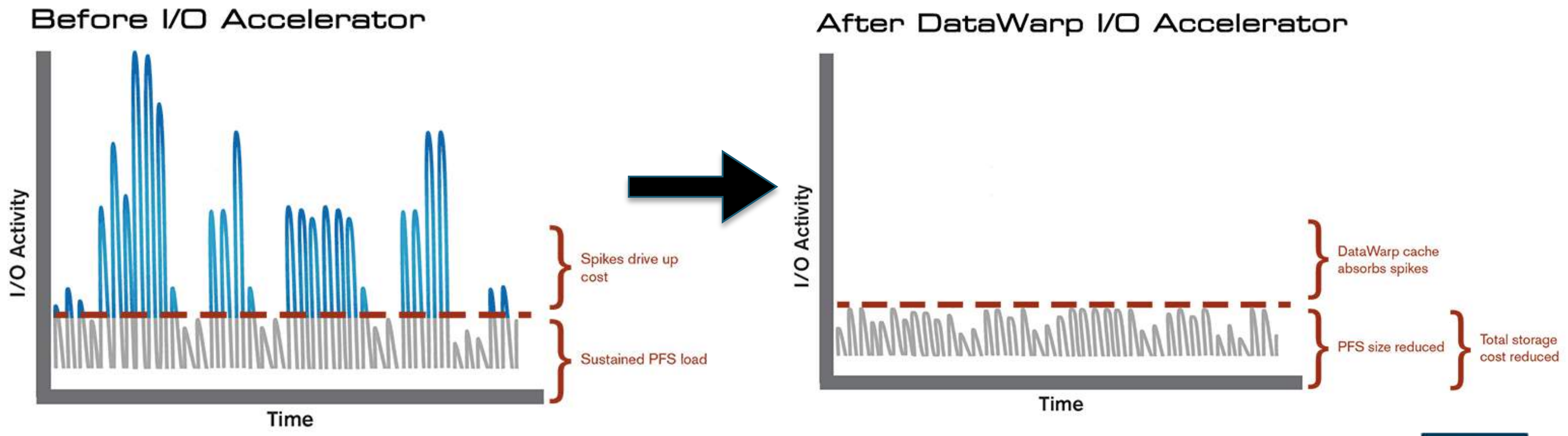
## Future





# Why a burst buffer?

- **HDD performance not increasing sufficiently**
  - More and more capacity to get required bandwidth
  - The bandwidth demand comes in ‘spikes’
- **For bandwidth HDD/PFS is more expensive than SSD**
- **Use NVRAM-based storage Burst Buffer**
  - Lower latency, higher bandwidth of flash-based Burst Buffer
  - Handle I/O bandwidth spikes without increasing size of PFS
  - File systems on demand scale better than large PFS



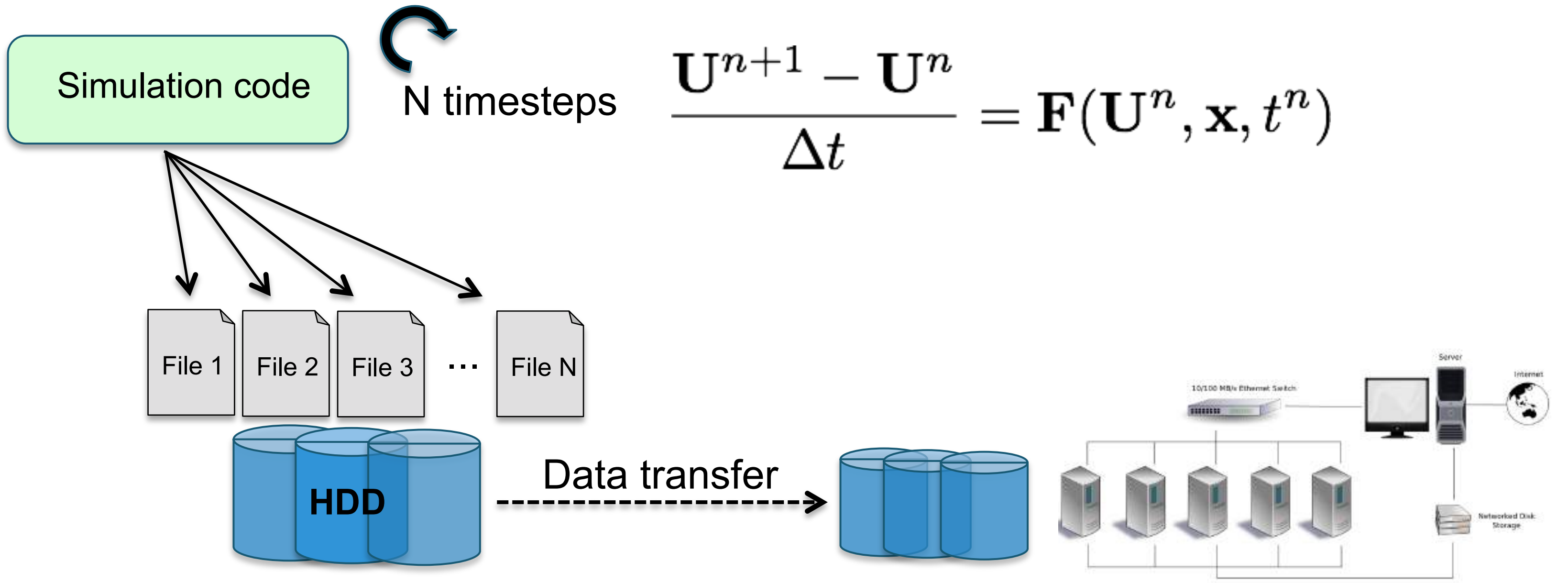
## Burst buffers at HPC centers

- **NERSC**: Cori (2016)
  - 288 BB nodes with 1.8PB total capacity (Cray DataWarp Burst Buffer)
- **LANL/Sandia**: Trinity (2016)
  - Similar architecture to NERSC/Cori
- **ANL**: Theta (2016)
  - 128GiB SSD per compute node
- **ANL**: Aurora (2018)
  - NVRAM per compute node and SSD burst buffers
- **ORNL**: Summit (2018)

Commonalities:

- Shorter path to compute nodes
- Handle latency-bound access patterns more effectively
- Solid state or NVRAM storage devices
- Limited capacity

# Computational physics and traditional post-processing

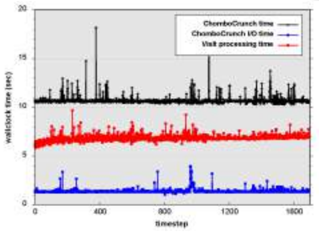


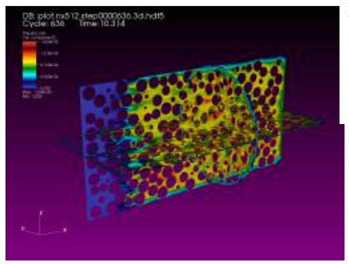
$$\frac{\mathbf{U}^{n+1} - \mathbf{U}^{n}}{\Delta t} = \mathbf{F}(\mathbf{U}^{n}, \mathbf{x}, t^{n})$$

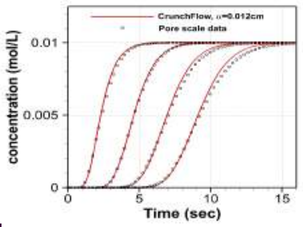
Remote storage: e.g. Globus Online, visualization cluster,...

Data analysis/ Visualization

Data transfer/storage and traditional post-processing is extremely expensive!





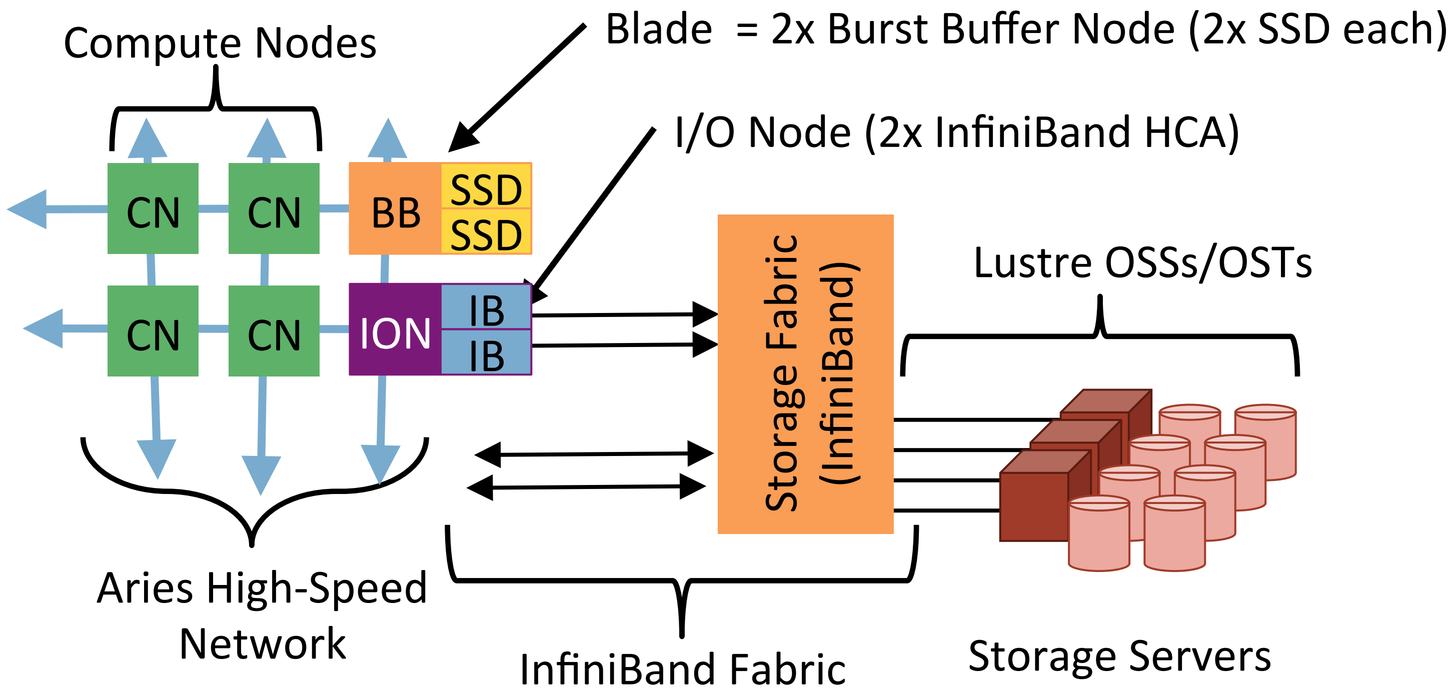


# Data processing methods

Data processing execution methods (Prabhat & Koziol, 2015)

| | **Post-processing** | **In-situ** | **In-transit** |
|---|---|---|---|
| **Analysis Execution Location** | Separate Application | Within Simulation | Burst Buffer |
| **Data Location** | On Parallel File System | Within Simulation Memory Space | Within Burst Buffer Flash Memory |
| **Data Reduction Possible?** | NO: All data saved to disc for future use | YES: Can limit output to only analysis products | YES: Can limit data saved to disk to only analysis products. |
| **Interactivity** | YES: User has full control on what to load and when to load data from disk | NO: Analysis actions must be pre-scribed to run within simulation | LIMITED: Data is not permanently resident in flash and can be removed to disk |
| **Analysis Routines Expected** | All possible analysis and visualization routines | Fast running analysis operations, statistical routines, image rendering | Longer running analysis operations bounded by the time until drain to file system. Statistics over simulation time |

# NERSC/Cray Burst Buffer Architecture



- Cori Phase 1 configuration: 920TB on 144 BB nodes (288 x 3.2 GB SSDs) 288 BB nodes on Cori Phase 2.
- DataWarp software (integrated with SLURM WLM) allocates portions of available storage to users per-job
- Users see a POSIX filesystem
- Filesystem can be striped across multiple BB nodes (depending on allocation size requested)

# Burst Buffer User Cases @ NERSC

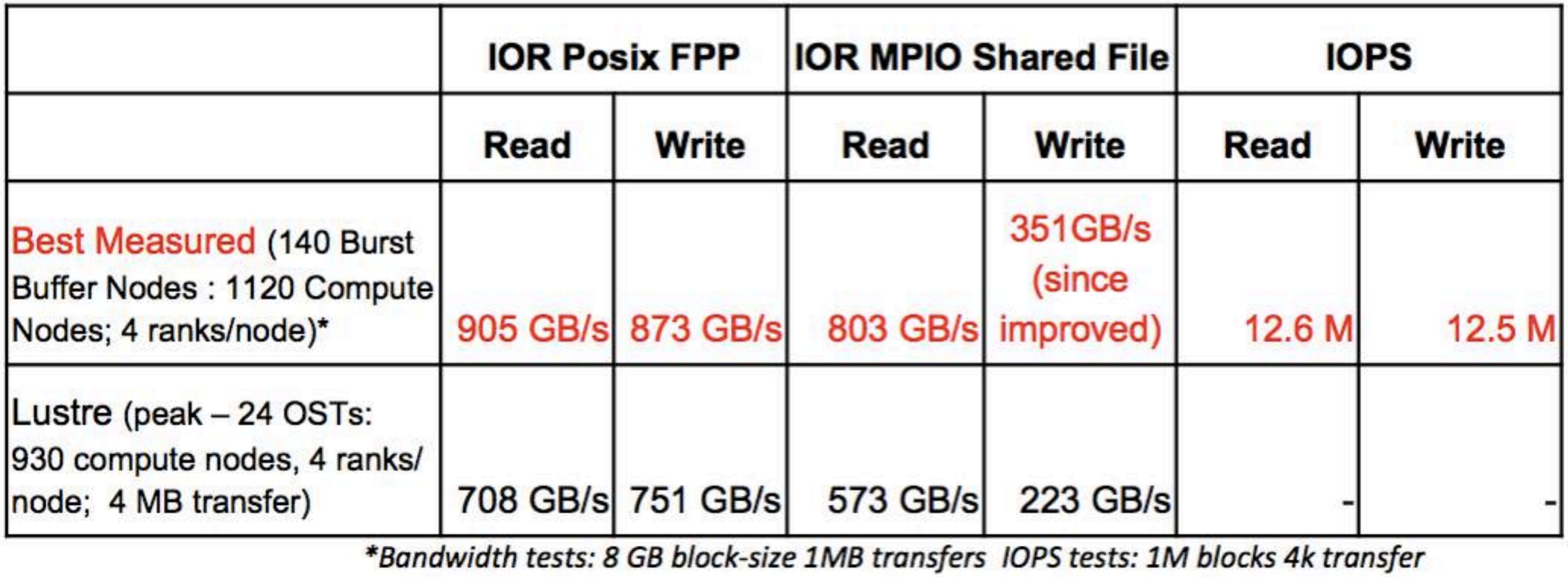
| **Burst Buffer User Cases** | **Example Early Users** |
|---|---|
| IO Bandwidth: Reads/ Writes | ● **Nyx/BoxLib**<br>● VPIC IO |
| Data-intensive Experimental Science - "Challenging/ Complex" IO pattern, eg. high IOPs | ● ATLAS experiment<br>● TomoPy for ALS and APS |
| Workflow coupling and visualization: in transit / in-situ analysis | ● **Chombo-Crunch / VisIt carbon sequestration simulation** |
| Staging experimental data | ● ATLAS and ALS SPOT Suite |

**Many others projects not described here (~50 active users).**

# Benchmark performance

- **Burst Buffer is doing well against benchmark performance targets**
  - Out-performs Lustre (in tests using half the full Burst Buffer and only a fraction of the full Cori compute load)

Details on use cases and benchmark performance in Bhimji et al, CUG 2016



| | IOR Posix FPP | | IOR MPIO Shared File | | IOPS | |
|---|---|---|---|---|---|---|
| | Read | Write | Read | Write | Read | Write |
| Best Measured (140 Burst Buffer Nodes : 1120 Compute Nodes; 4 ranks/node)* | 905 GB/s | 873 GB/s | 803 GB/s | 351GB/s (since improved) | 12.6 M | 12.5 M |
| Lustre (peak – 24 OSTs: 930 compute nodes, 4 ranks/node; 4 MB transfer) | 708 GB/s | 751 GB/s | 573 GB/s | 223 GB/s | - | - |

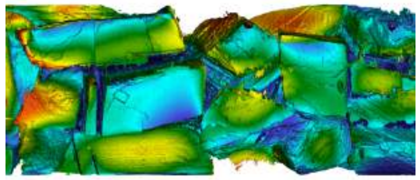
**Bandwidth tests: 8 GB block-size 1MB transfers IOPS tests: 1M blocks 4k transfer*

## Chombo-Crunch (ECP application)

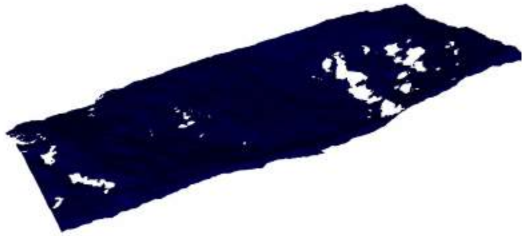
- **Simulates pore scale reactive transport processes associated with carbon sequestration**
- **Applied to other subsurface science areas:**
  - –Hydrofracturing (aka "fracking")
  - –Used fuel disposition (Hanford salt repository modeling)
- **Extended to engineering applications**
  - –Lithium ion battery electrodes
  - –Paper manufacturing (hpc4mfg)

***The common feature is ability to perform direct numerical simulation from image data of arbitrary heterogeneous, porous materials***

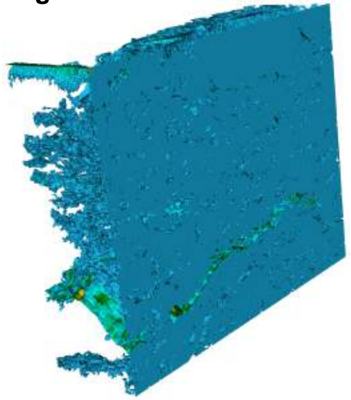
pH on crushed calcite in capillary tube



Transport in fractured dolomite

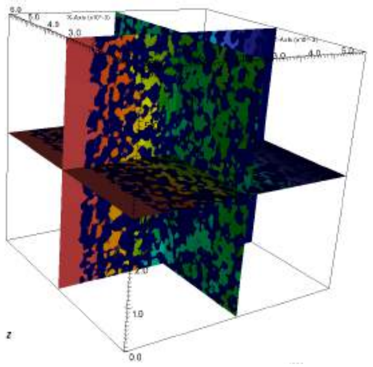


Flooding in fractured Marcellus shale

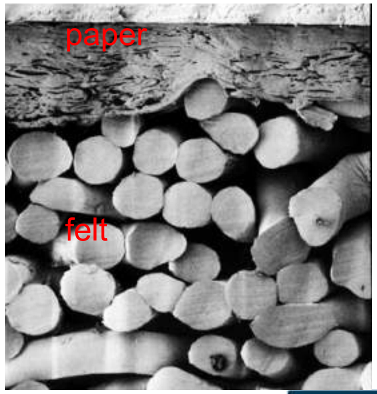


$O_2$ diffusion in Kansas aggregate soil

Electric potential in Li-ion electrode



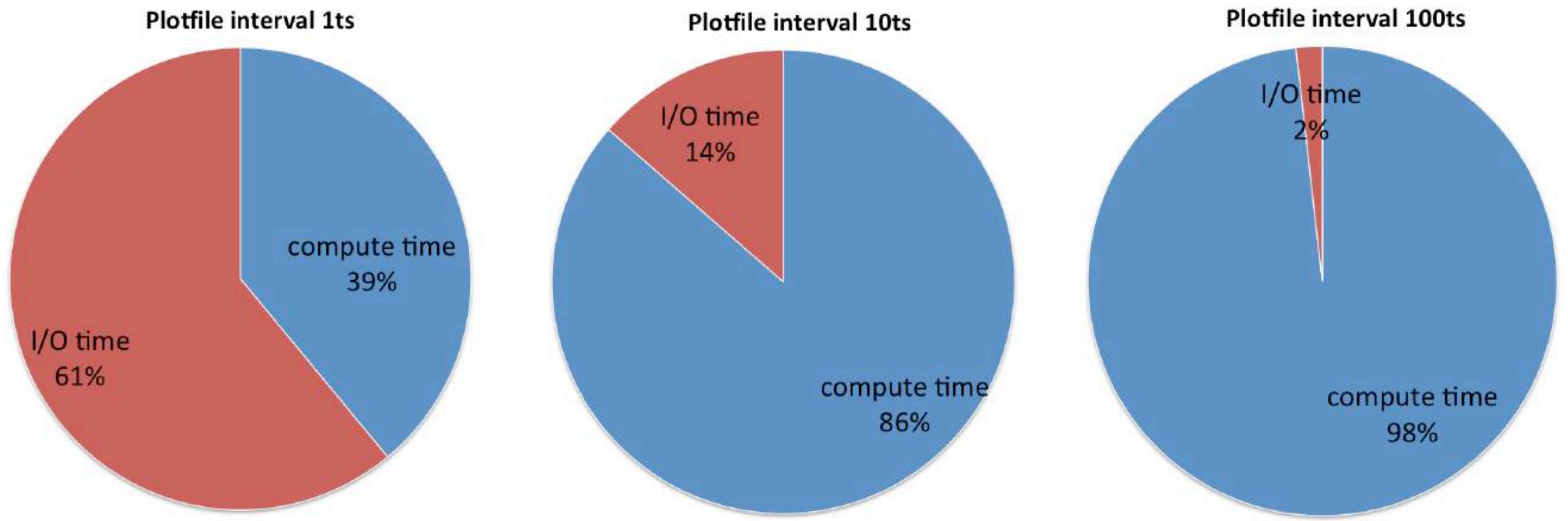
Paper re-wetting



## I/O constraint: common practice

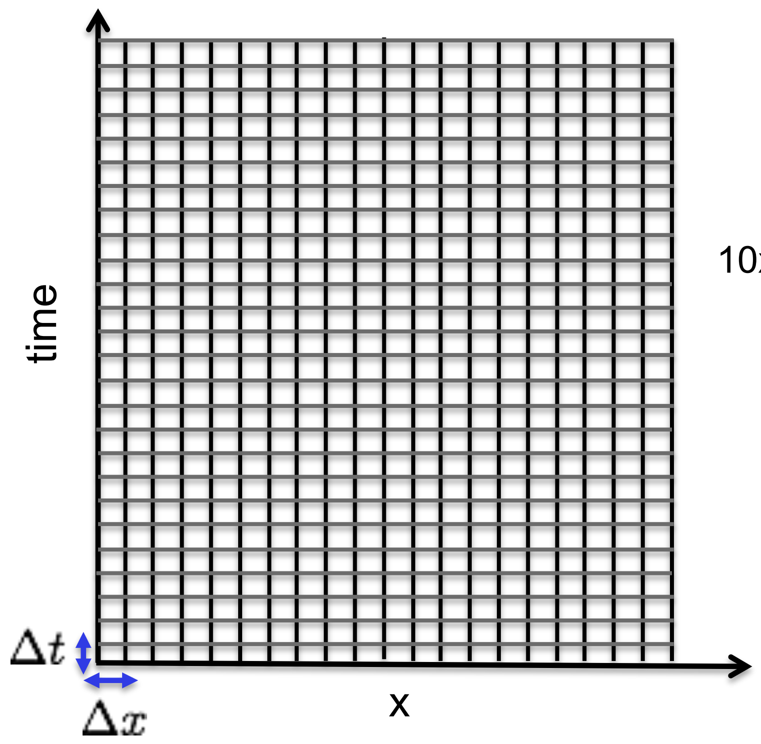
**Common practice: increase I/O (plotfile) interval by 10x, 100x, 1000x,...**

I/O contribution to Chombo-Crunch wall time at different plotfile intervals

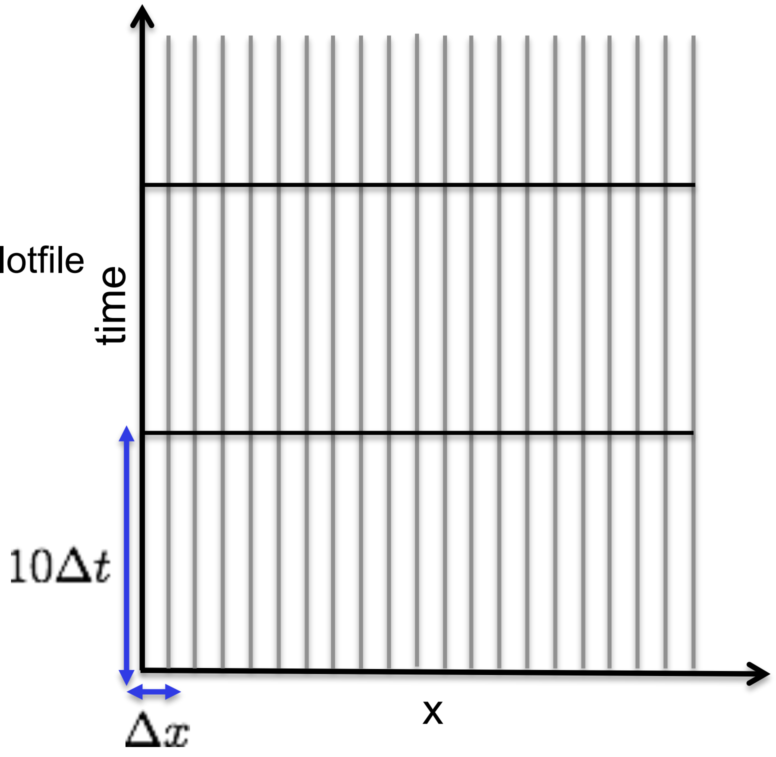


# Loss of temporal/statistics accuracy

Time evolution from 0 to T: $\frac{d\mathbf{U}}{dt} = \mathbf{F}(\mathbf{U}(x,t))$



10x increase of plotfile interval



**Pros**: less data to move and store
**Cons**: degraded accuracy of statistics (stochastic simul.)

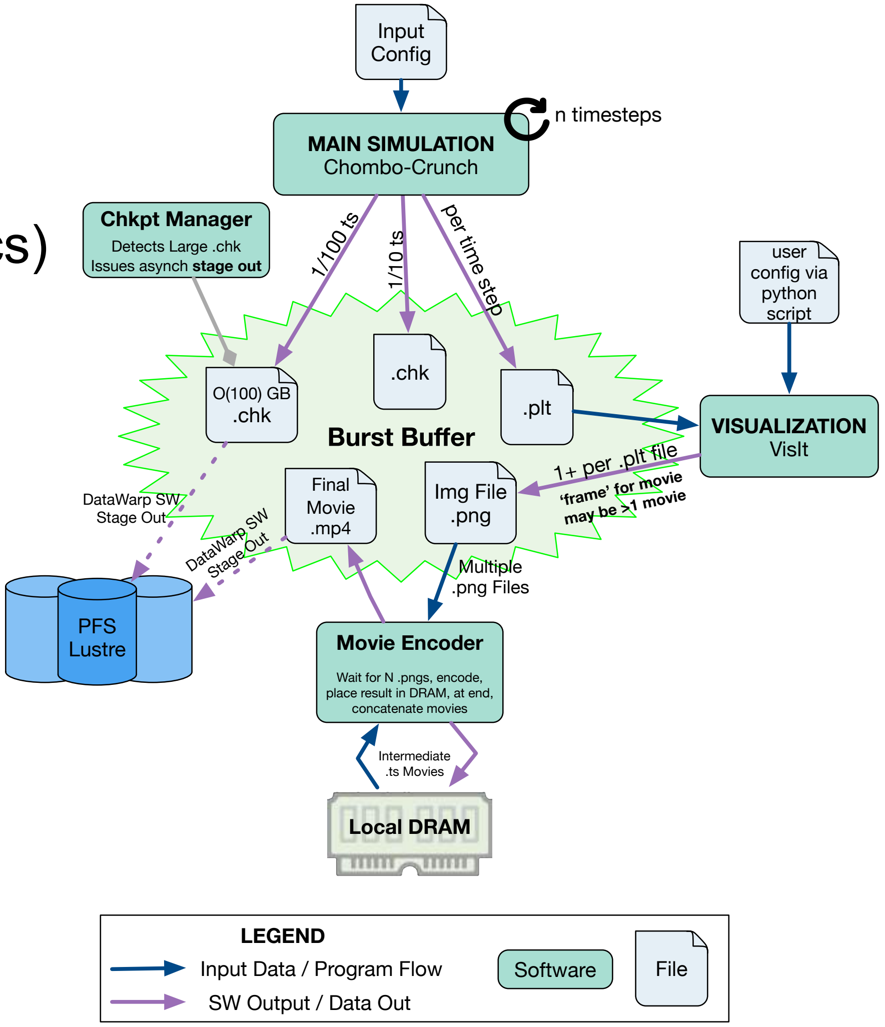
$$\boxed{\varepsilon \sim \frac{1}{\sqrt{N}},\ N \text{ is the sample size}}$$

# Proposed in-transit workflow

Workflow components:

- ❑ **Chombo-Crunch**
- ❑ **VisIt** (visualization and analytics)
- ❑ **Encoder**
- ❑ **Checkpoint manager**

I/O: HDF5 for checkpoints and plotfiles



# Straightforward batch script

allocate BB capacity

copy restart file to BB

run each component

transfer output product to persistent storage

```
#!/bin/bash
#SBATCH --nodes=1291
#SBATCH --job-name=shale
#DW jobdw capacity=200TB access_mode=striped type=scratch
#DW stage_in type=file source=/pfs/restart.hdf5 destination
     =$DW_JOB_STRIPED/restart.hdf5
### Load required modules
module load visit
ScratchDir="$SLURM_SUBMIT_DIR/_output.$SLURM_JOBID"
BurstBufferDir="${DW_JOB_STRIPED}"
mkdir $ScratchDir
stripe_large $ScratchDir
NumTimeSteps=2000
EncoderInt=200
RestartFileName="restart.hdf5"
ProgName="chombocrunch3d.Linux.64.CC.ftn.OPTHIGH.MPI.PETSC.
ex"
ProgArgs=chombocrunch.inputs
ProgArgs="$ProgArgs check_file=${BurstBufferDir}check
     plot_file=${BurstBufferDir}plot pfs_path_to_checkpoint=
     ${ScratchDir}/check restart_file=${BurstBufferDir}${
     RestartFileName} max_step=$NumTimeSteps"
### Launch Chombo-Crunch
srun -N 1275 —n 40791 $ProgName $ProgArgs > log 2>&1 &
### Launch VisIt
visit -l srun -nn 16 -np 512 -cli -nowin -s VisIt.py &
### Launch Encoder
./encoder.sh -pngpath $BurstBufferDir -endts $NumTimeSteps
     -i $EncoderInt &
wait
### Stage-out movie file from Burst Buffer
#DW stage_out type=file source=$DW_JOB_STRIPED/movie.mp4
     destination=/pfs/movie.mp4
```

# DataWarp API

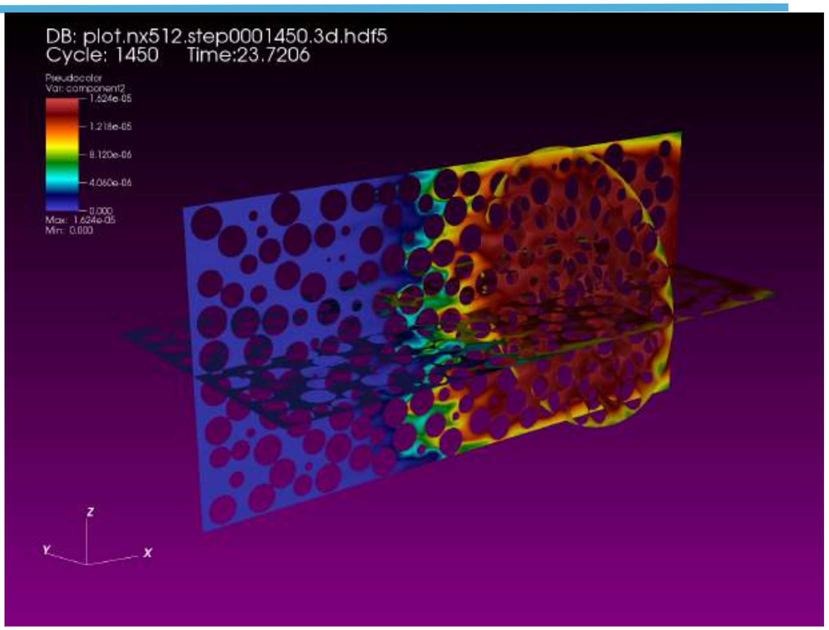
Asynchronous transfer of plot file/checkpoint from Burst Buffer to PFS

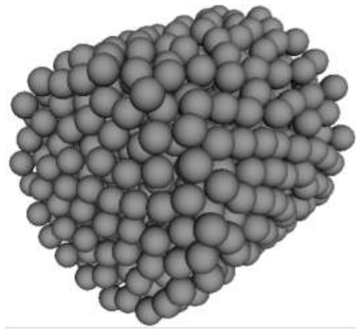
```
#ifdef CH_DATAWARP
// use DataWarp API stage_out call to move plotfile from BB to Lustre
   char lustre_file_path[200];
   char bb_file_path[200];
   if ((m_curStep%m_copyPlotFromBurstBufferInterval == 0) &&
(m_copyPlotFromBurstBufferInterval > 0))
   {
      sprintf(lustre_file_path, "%s.nx%d.step%07d.%dd.hdf5", m_LustrePlotFile.c_str(),
ncells, m_curStep, SpaceDim);
      sprintf(bb_file_path, "%s.nx%d.step%07d.%dd.hdf5", m_plotFile.c_str(), ncells,
m_curStep, SpaceDim);
      dw_stage_file_out(bb_file_path, lustre_file_path, DW_STAGE_IMMEDIATE);
   }
#endif
```

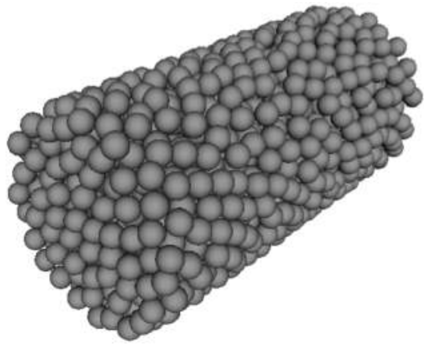
## Scaling study: Packed cylinder

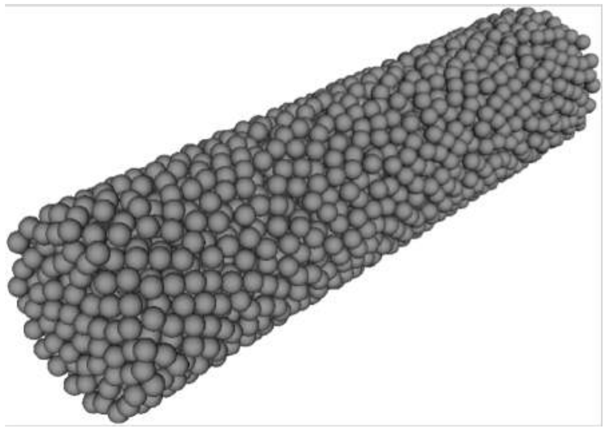
Weak scaling setup (*Trebotich&Graves,2015*)

- Geometry replication
- Number of compute nodes from 16 to 1024
- Ratio of number of compute nodes to BB nodes is fixed at 16:1
- Plotfile size: from 8GB to 500GB



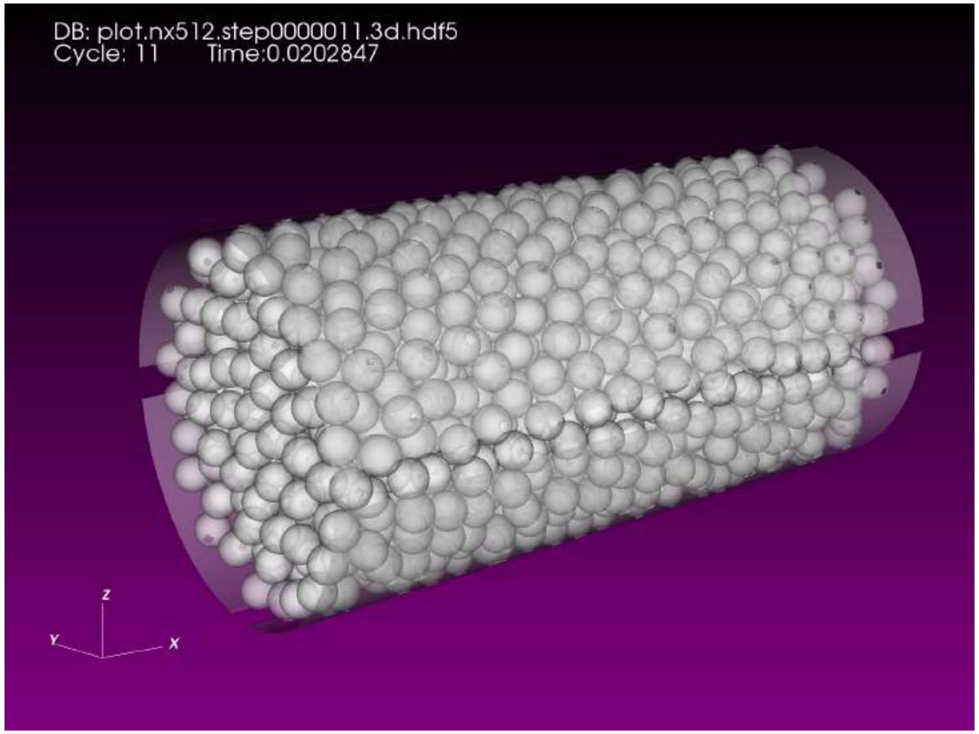


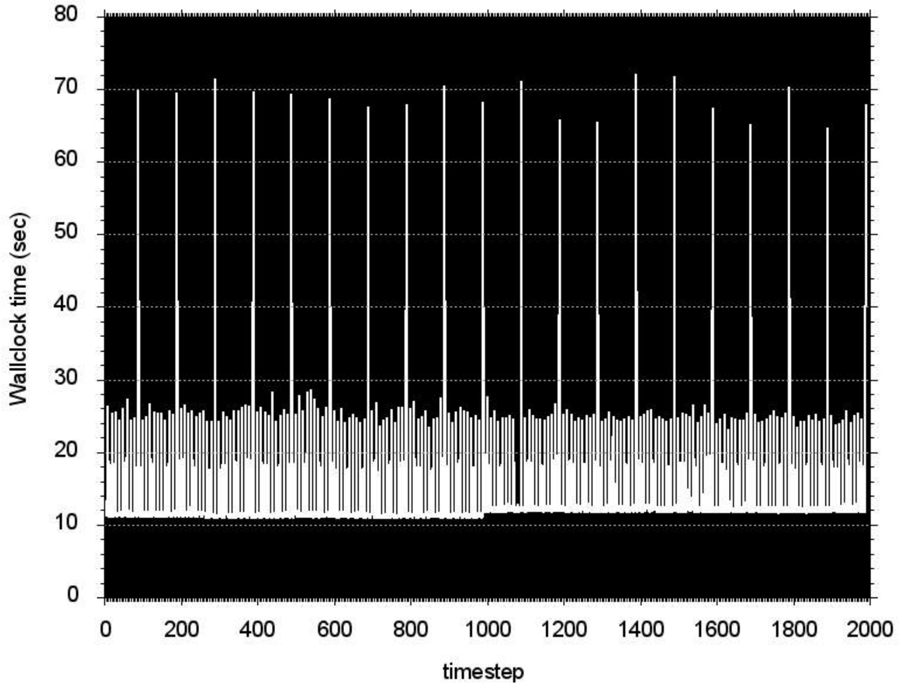




# Wall clock history: I/O to Lustre

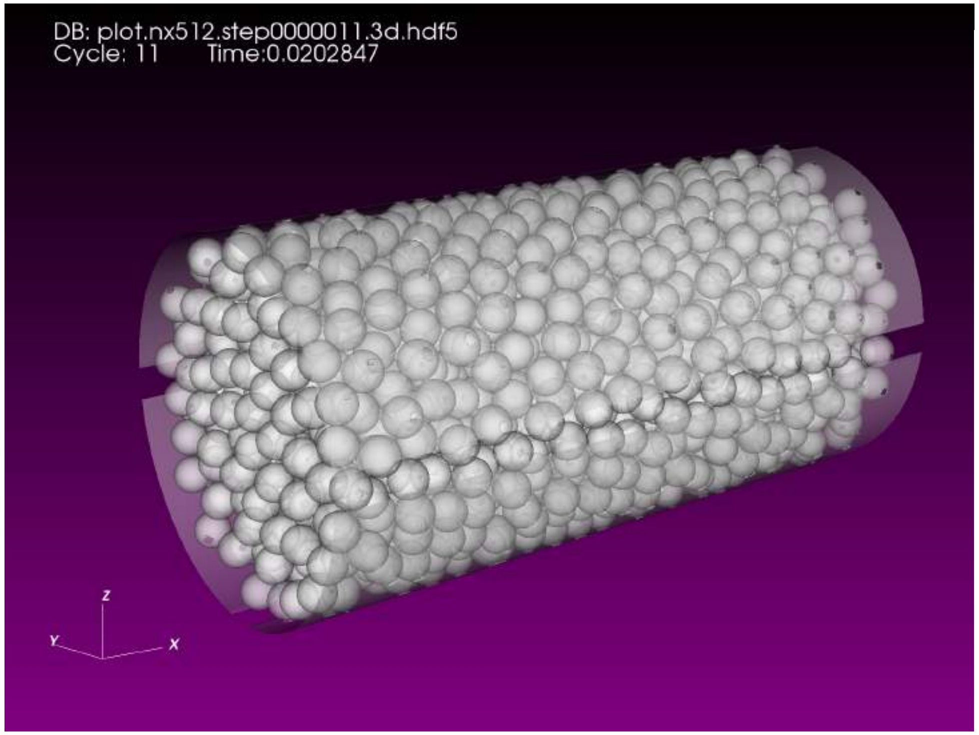
Reactive transport in packed cylinder: **256 compute nodes** (8192 cores) on Cori (HSW partition)
**72 OSTs** on Lustre (optimal for this file size). Peak I/O bandwidth: **5.6GB/sec**

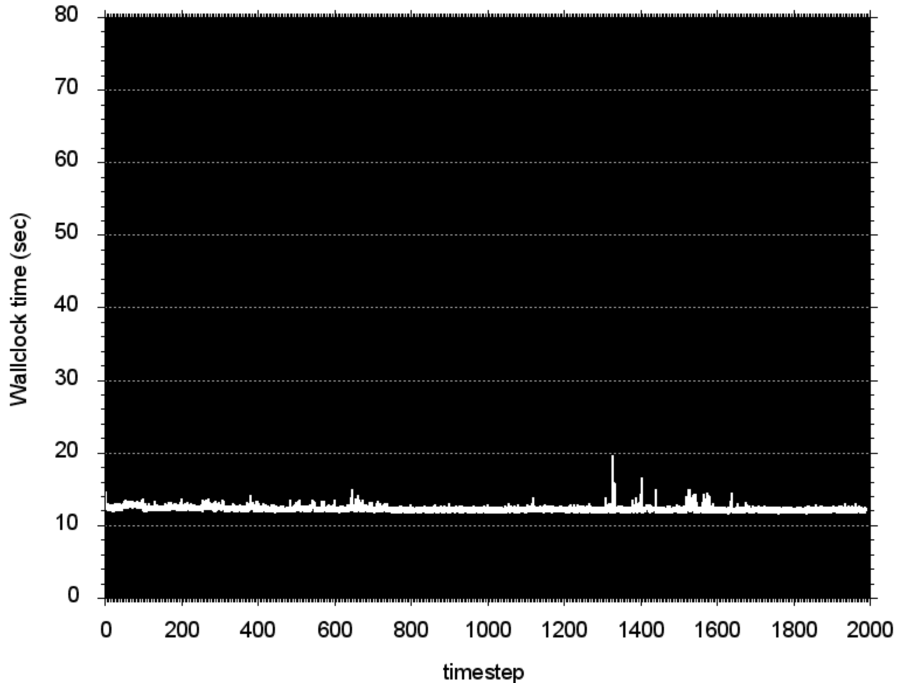




# Wall clock history: I/O to BB

Reactive transport in packed cylinder: **256 compute nodes** (8192 cores) on Cori (HSW partition)
**128 Burst Buffer nodes**. Peak I/O bandwidth: **70.2GB/sec**

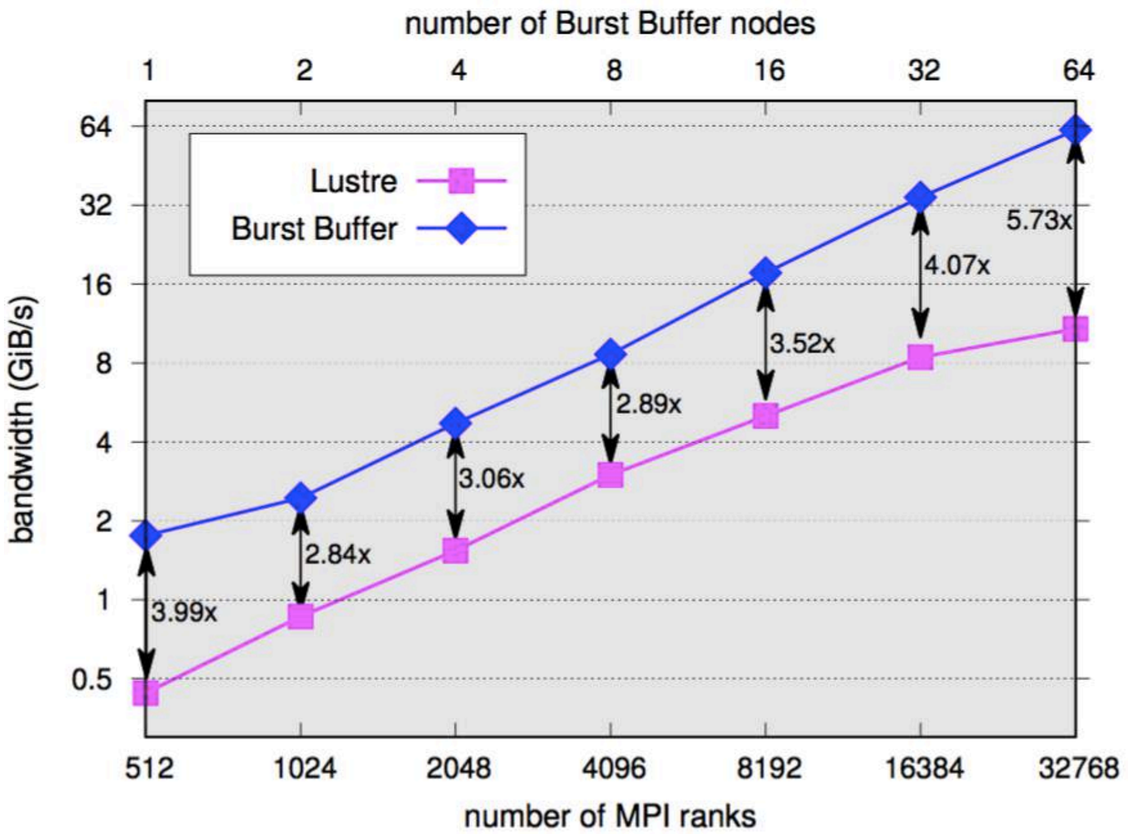




## I/O bandwidth study (1)

**Now: Number of compute nodes to BB nodes is fixed at 16:1**

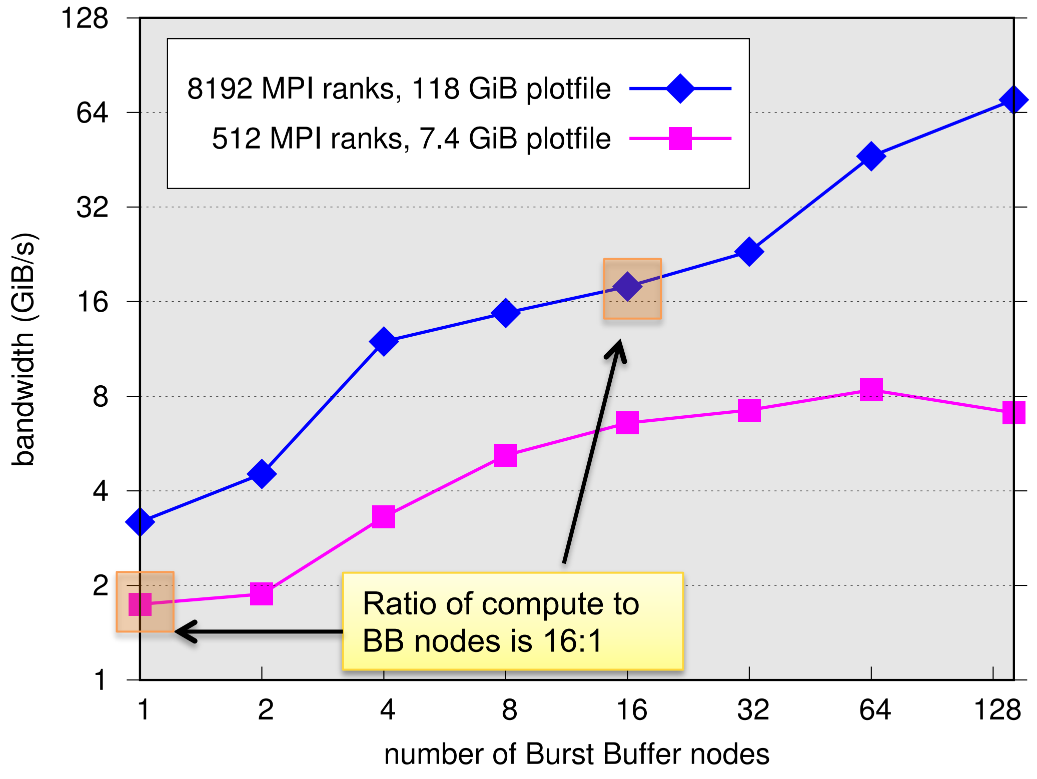
Collective write to shared file using HDF5 library



Scaling study for 16 to 1024 compute nodes on Cori Phase 1.

## I/O bandwidth study (2)

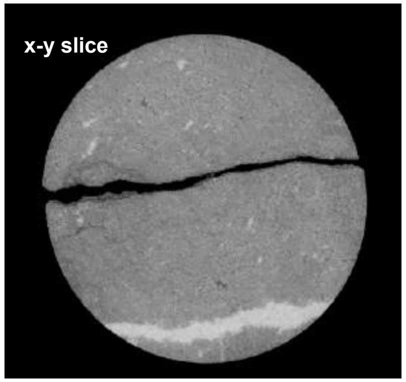
Collective write to shared file using HDF5 library

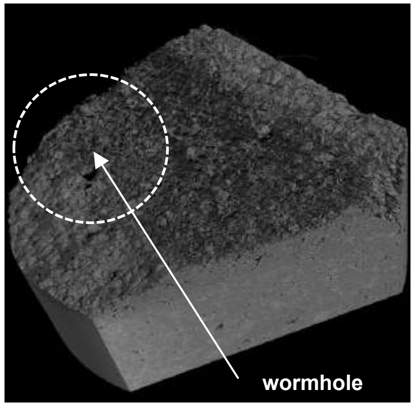


Write bandwidth study for 7.4GiB and 118GiB file sizes.

# In-transit visualization: show case 2

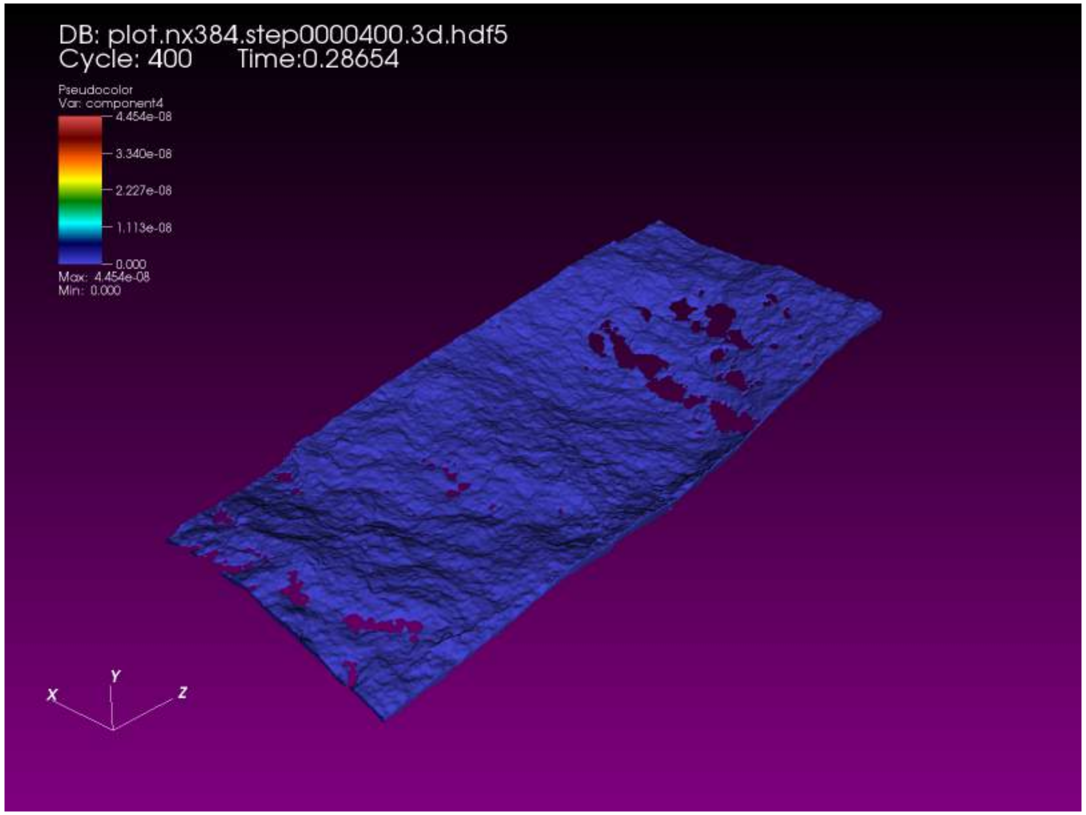
**Reactive transport in fractured mineral (dolomite)**: Simulation performed on Cori Phase 1: 512 cores (16 nodes) used by Chombo-Crunch, 64 cores (2 nodes) by VisIt, 128 Burst Buffer nodes for I/O.



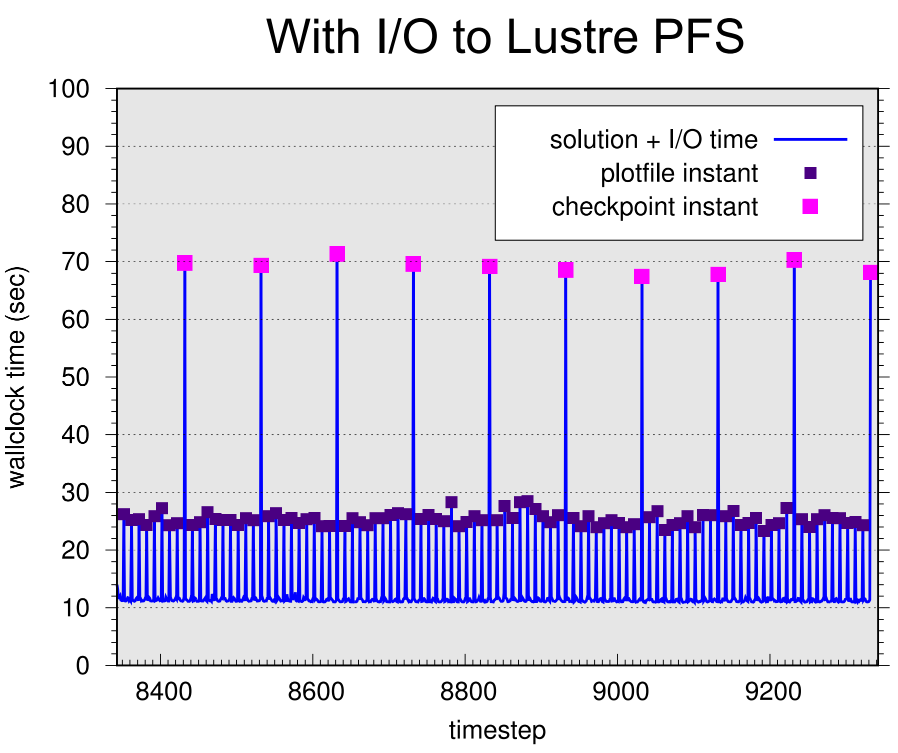


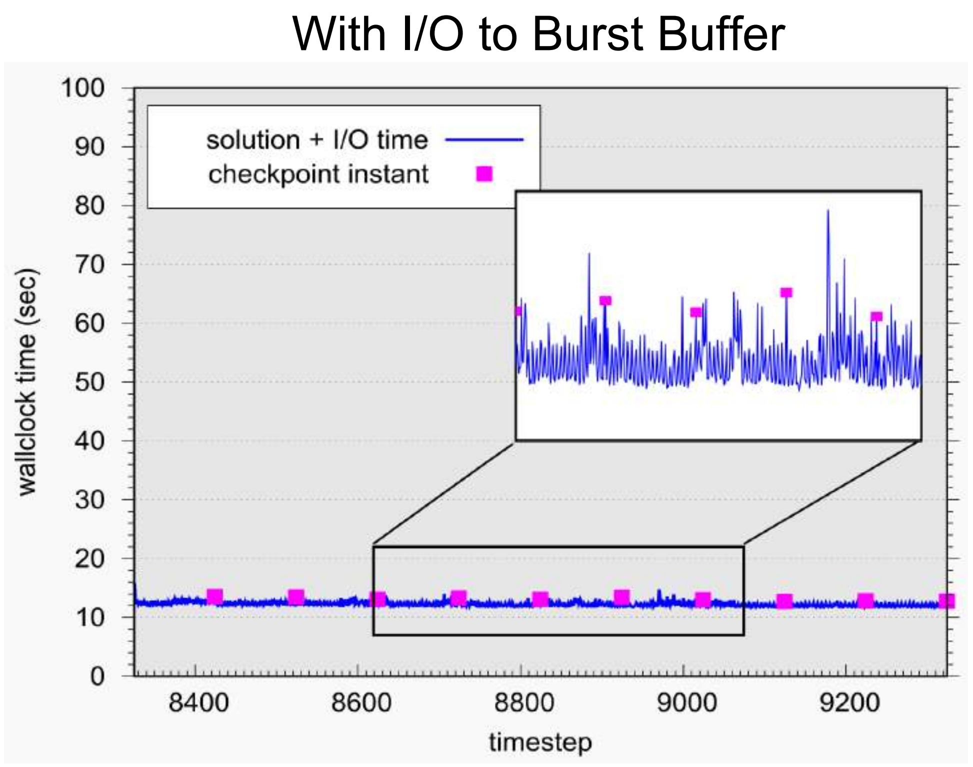
*Experimental images courtesy of Jonathan Ajo-Franklin and Marco Voltolini, EFRC-NCGC and LBNL ALS.*

$Ca^{2+}$ concentration



# Wall clock time history

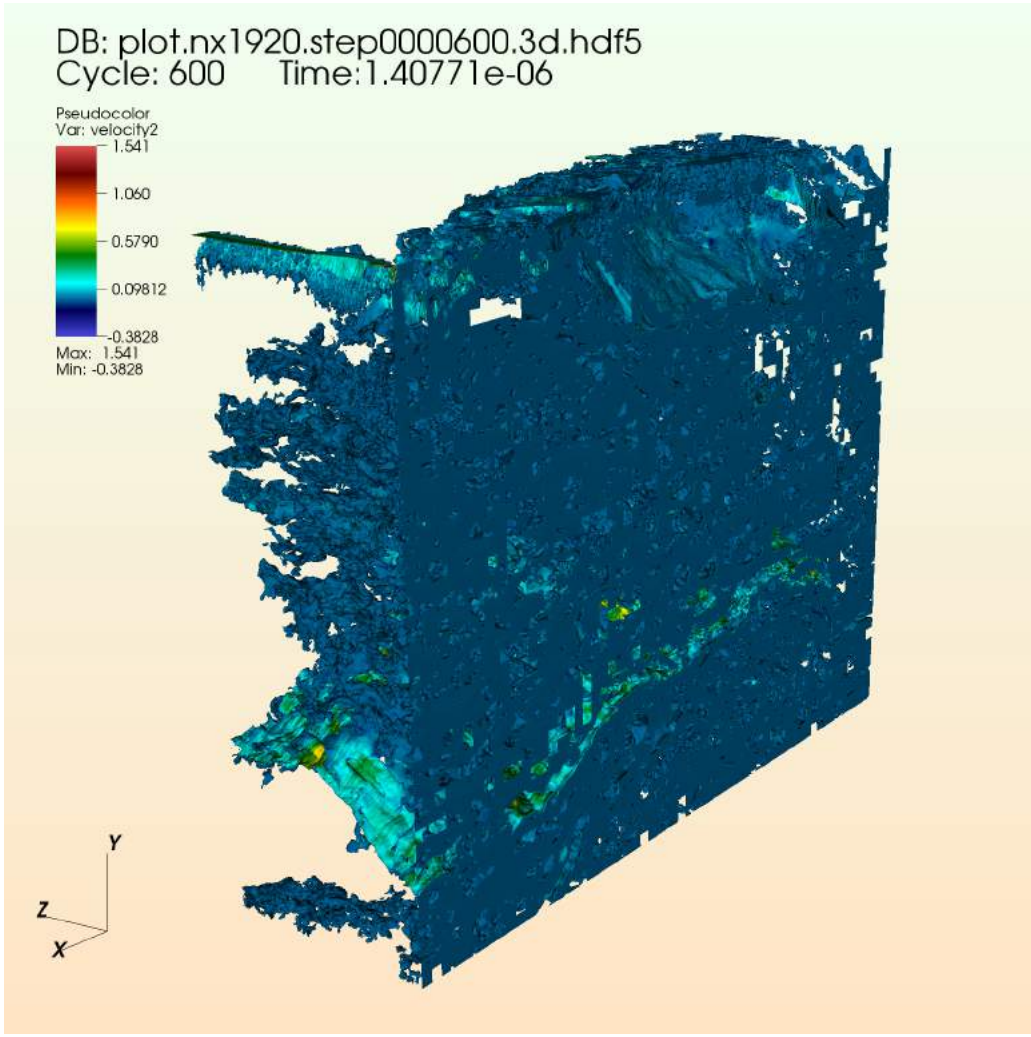




## In-transit visualization: show case 3

**Reactive Flow in Kahuna shale**

- 41K cores on NERSC's Cori system
- 100 micron block sample
- 48 nm resolution, 2 billion cells
- 16 nodes for VisIt
- 144 Burst Buffer nodes
- Plotfile size 290 GB (plotting interval 10 timesteps)
- **Total data set: 560 TB**

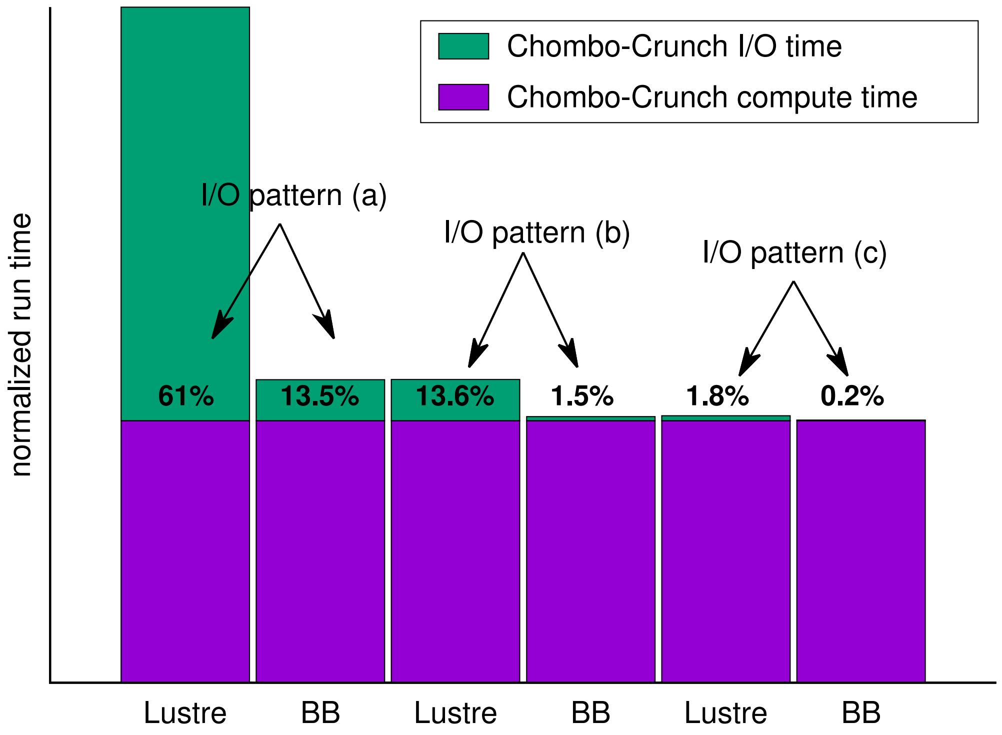


# Compute time vs I/O time

**(a) High intensity I/O**: plot file every timestep, checkpoint file every 10 timesteps

**(b) Moderate intensity I/O**: plot file every 10 timesteps, checkpoint file every 100 timesteps

**(c) Low intensity I/O**: plot file every 100 timesteps, checkpoint file every 500 timesteps



## Conclusions

- In-transit asynchronous workflow which couples simulation, visualization and quantitative analysis has been proposed. DataWarp Burst Buffer has been utilized.

- I/O speedup by utilizing Burst Buffer compared to Lustre file system:
  - **3-5x** for fixed ratio of compute nodes to BB nodes (16:1)
  - **13x** for peak performance (full BB capacity vs Lustre)

- Burst Buffer allowed Chombo-Crunch to move to every timestep of "data-processing" with minimal changes in the source code.

- **Remaining challenges and ongoing work:**
  - Run-time managing of BB capacity (limit per user will be ~20TB)
  - Dynamic component load balancing
  - Including additional components into workflow:
    - coupling pore-scale with reservoir scale simulation
    - extra VisIt sessions for quantitative analysis (computing flow statistics, reactions rates, pore graph extractor, ...)

## References

1. Ovsyannikov et al. "*Scientific Workflows at DataWarp Speed: Accelerated Data-Intensive Science Using NERSC's Burst Buffer*". In Proceeding of IEEE PDSW-DISCS 2016 Workshop, Supercomputing Conference, pp.1-6 (2016)
2. Bhimji et al. "*Accelerating Science with the NERSC Burst Buffer Early User Program*". In Proceedings of the Cray User Group (CUG'16), pp.1-15 (2016)
3. Liu et al. 2012 "*On the Role of Burst Buffers in Leadership-Class Storage Systems*". In Proceedings of the 2012 IEEE Conference on Massive Data Storage, pp.1-11 (2012)

## Thank you!

Contact: aovsyannikov@lbl.gov