

NERSC Overview

CSADS Workshop on PetaScale Applications and Performance Strategies

Katie Antypas
HPC Consultant

July 14, 2008



Overview

- Overview of NERSC Systems and Services
- Tips for running your code successfully at NERSC (or any other HPC Center)



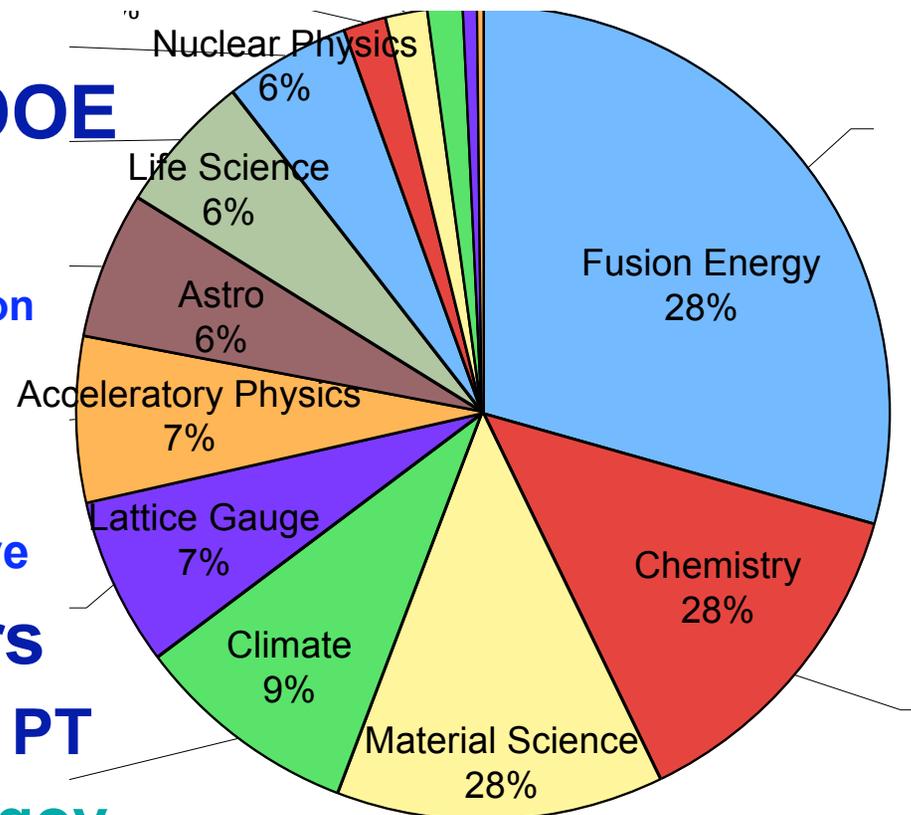
NERSC Mission

The mission of the National Energy Research Scientific Computing Center (NERSC) is to *accelerate the pace of scientific discovery* by providing high performance computing, information, data, and communications services for *all DOE Office of Science (SC) research*.



NERSC is the Production Facility for DOE

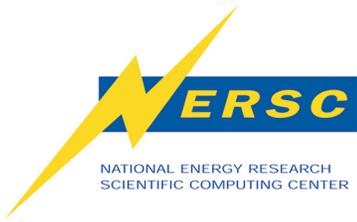
- **NERSC serves all areas**
 - ~3000 users, ~400 projects
- **Allocations managed by DOE**
 - 10% INCITE awards:
 - Large allocations, extra service
 - Used throughout SC; not just DOE mission
 - 70% Production (ERCAP) awards:
 - From 10K hour (startup) to 5M hour
 - Only available at NERSC
 - 10% each NERSC and DOE/SC reserve
- **8 consultants to help users**
 - Monday - Friday 8am - 5pm PT
 - email us at consult@nersc.gov
 - We are here to help!





Getting an HPC allocation

- **Not as hard as you might think**
 - If you have an abstract of your research goals applying will take you 30 min or so
- **A small allocation is stepping stone toward a large allocation when you need it. It helps you build a computing relationship with DOE and project reviewers.**
- **NERSC**
 - <https://nim.nersc.gov/newpi.php>
- **ANL**
 - <https://accounts.alcf.anl.gov/accounts/projects/intrepid.htm>
- **ORNL**
 - <http://www.nccs.gov/user-support/access/project-request>



Systems



NERSC 2008 Configuration

Large-Scale Computing System

Franklin (NERSC-5): Cray XT4

- 9,740 nodes; 19,480 cores
- 19 Tflop/s sustained SSP (100 Tflops/s peak)

Upgrading to QuadCore

- ~36 Tflops/s sustained SSP (355 Tflops/s peak)



Clusters



Bassi (NCSb)

- IBM Power5 (888 cores)

Jacquard (NCSa)

- LNXXI Opteron (712 cores)

PDSF (HEP/NP)

- Linux cluster (~1K cores)

NERSC Global Filesystem (NGF)

230 TB; 5.5 GB/s



HPSS Archival Storage

- 44 PB capacity
- 10 Sun robots
- 130 TB disk cache



Analytics / Visualization

- Davinci (SGI Altix)



Clusters at NERSC

- 2 clusters are available at NERSC for small and medium sized jobs
- NCS-a “Jacquard” (08/2005)
 - 722-processor (2.2 GHz Opteron) Linux Network Evolocity cluster
 - InfiniBand fat-tree
- NCS-b “Bassi” (01/2006)
 - 122 IBM p5-575 nodes (with 32GB)
 - 1.9 GHz POWER 5 processors
 - Dual plane Federation interconnect



Franklin Cray XT-4 System

NERSC-5 System is a Cray XT-4

- Parallelism: 9,740 nodes with 19,480 cores
 - 102 Node Cabinets, 16 KWs per cabinet
 - 39.5 TBs Aggregate Memory
- Peak performance: 100 Tflop/s
- Sustained performance: 19 Tflop/s
- Interconnect: Cray SeaStar2, 3D Torus
 - >6 TB/s Bisection Bandwidth
 - >7 GB/s Link Bandwidth
- Shared Disk: 345 TBs
- Upgrading to QuadCore
 - 355 Tflops/s peak
 - ~36 Tflops/s sustained

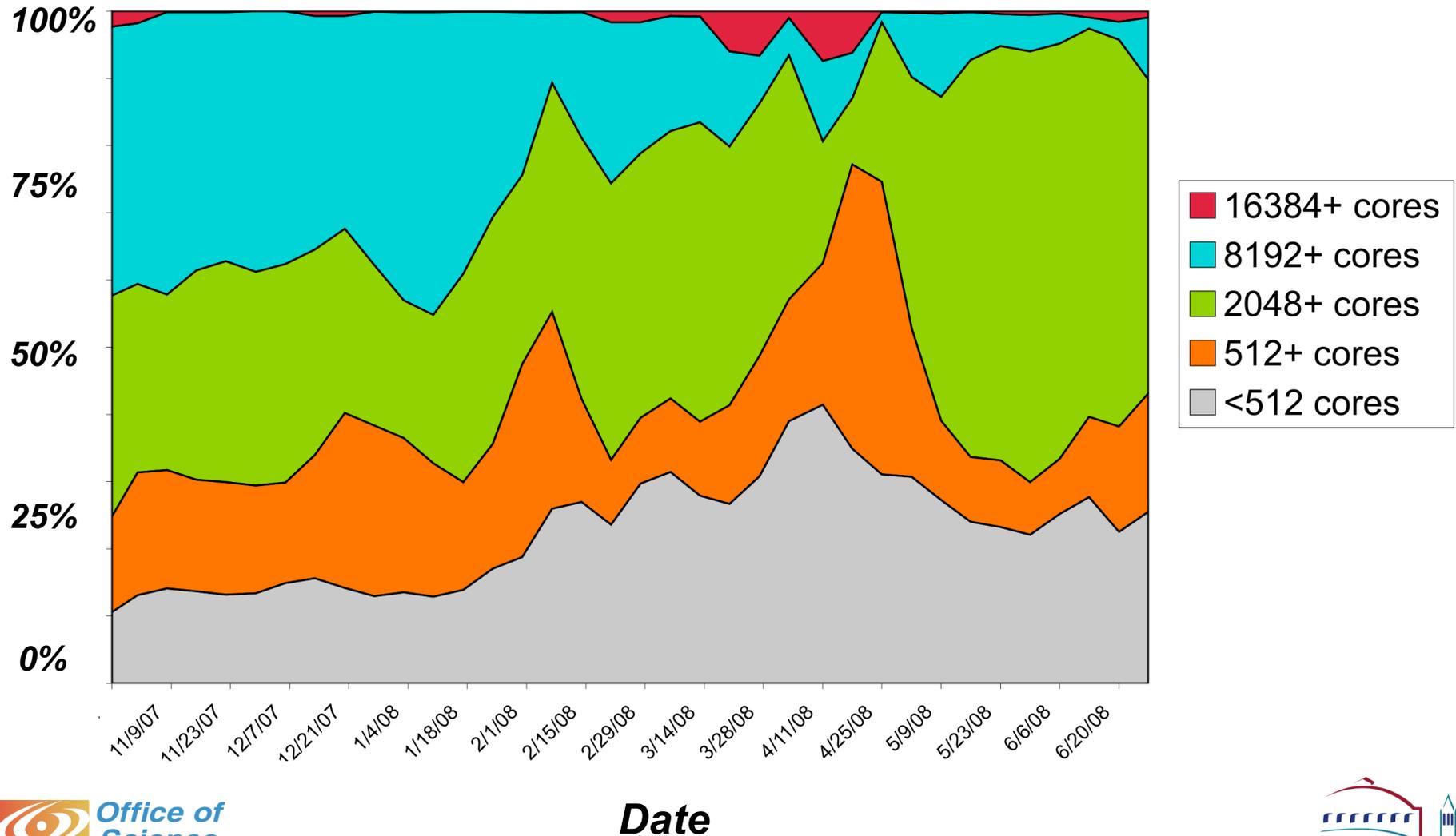


Benjamin Franklin, performed ground breaking work in energy efficiency, electricity, materials, climate, ocean currents, transportation, health, medicine, acoustics and heat transfer.



Computing at Scale

Concurrency on Franklin by % of Raw Hours





Franklin Programming Environment

- **Compilers (Fortran, C, C++)**
 - PGI
 - ! – PathScale
 - GNU
- **Parallel Programming Models: Cray MPICH2 MPI, Cray SHMEM, Open MP**
- **AMD Core Math Library (ACML): BLAS, LAPACK, FFT, Random number generators, GNU Fortran libraries**
- **LibSci scientific library: ScaLAPACK, BLACS, SuperLU**
- **Profiling tools CrayPat, Apprentice2, IPM, TAU**
- **Performance API (Papi)**
- **Modules**



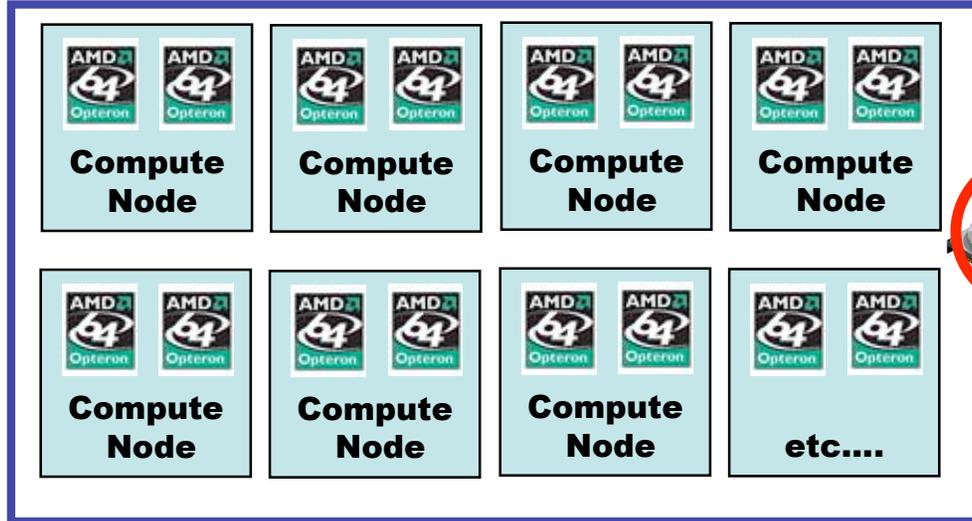
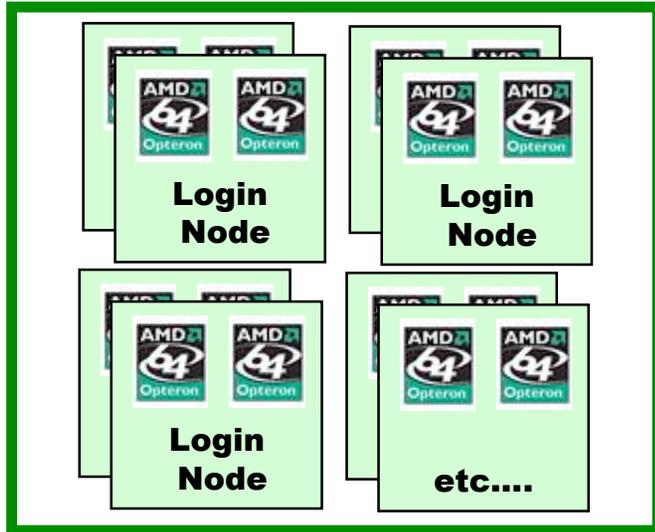
Extensive 3rd Party Software

- ! • **Check to make sure your application isn't already installed**
- **Use modules command to see software availability on all NERSC machines ("module avail")**
- **Math - acml, aztec, dfftpack, fftw, gsl, LibSci, parmetis, parpack, petsc, pspline, superlu, sprng**
- **I/O - hdf5, nco, netcdf, pnetcdf**
- **Chemistry/Mat Sci - amber, namd, nwchem, abinit, cpmd, lammps, quantum espresso, siesta, vasp**
- **Visualization - idl, gnuplot, visit, ncar**
- **Debuggers - Allinea's DDT**

Franklin Overview

Full Linux OS

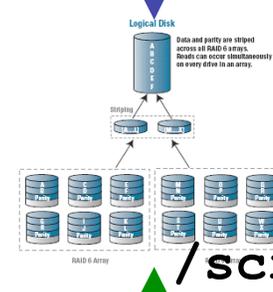
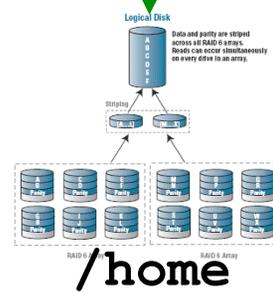
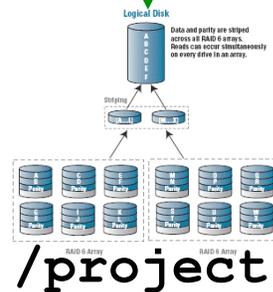
CNL (no logins)



No local disk

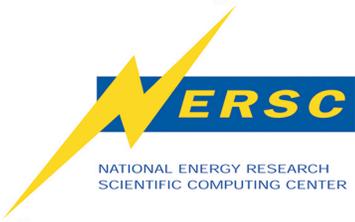


HPSS



What kind of OS?

- **Consider what kind of OS you are using**
 - **Limited OS**
 - **Depends on system but limited OS calls**
 - **Features which could be limited on compute nodes**
 - **Shared libraries**
 - **Scripting languages, python, perl**
 - **Process control (fork, exec)**
 - **Can't ssh from compute node to compute node**
 - **Can't call system() from Fortran parallel job**
 - **No Java on the compute nodes**
 - **No X-Windows support on compute nodes**



Memory Considerations

- Each Franklin compute node has 4GB of memory.
- CNL kernel, uses ~300 MB of memory.
- Lustre uses about 17 MB of memory
- Default MPI buffer size is about 72 MB.
- Single core MPI jobs have ~3.6 GB/task.
- Dual core MPI jobs have ~1.8 GB/task.
- Change MPI buffer sizes by setting certain MPICH environment variables.



User Questions and Problem Reports

NERSC Consulting Tickets Jan 1, 2007 to September 18, 2007

Profile of Incidents by Category

Category	Incidents
----------	-----------

Announcements	4
---------------	---

Files/Data Storage	361
--------------------	-----

Information Technology	55
------------------------	----

Network Access	56
----------------	----

Programming	194
-------------	-----

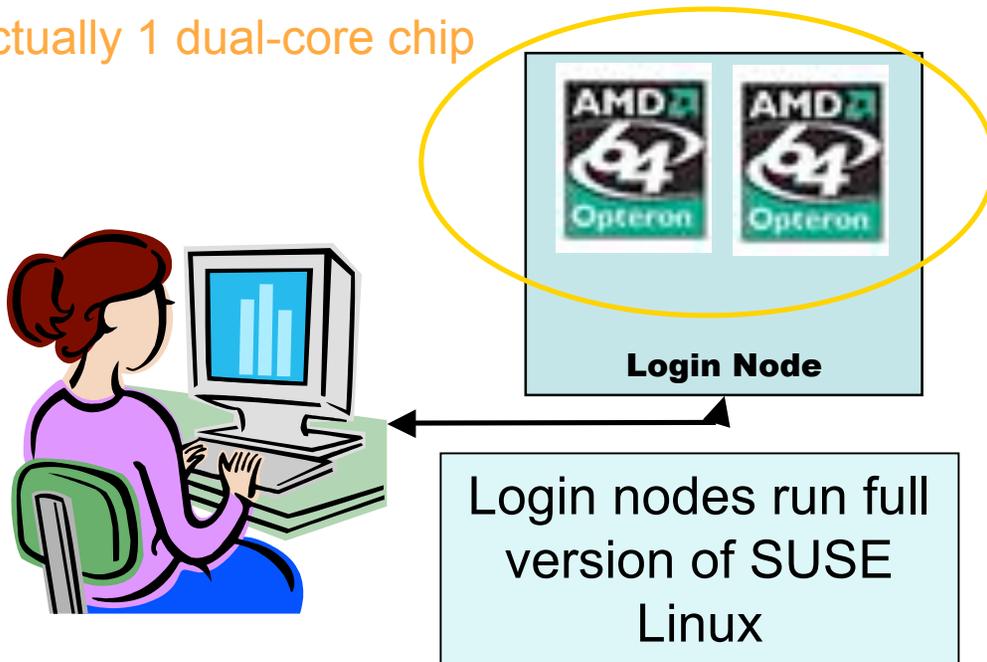
Running Jobs	1032
---------------------	-------------

Software	346
----------	-----

Record Count: 7

Running a Job on Franklin

Actually 1 dual-core chip



www.nersc.gov/nusers/status/queues/franklin/

NERSC Analytics server (DaVinci)

On a Franklin login node:

1. Log in from your desktop using SSH
2. Compile your code or load a software module
3. Write a job script
4. Submit your script to the batch system
5. Monitor your job's progress
6. Archive your output
7. Analyze your results



Batch Queues

- **At NERSC users submit jobs to a queue and wait in line to run**
- **Queue policies are set to:**
 - **Be fair**
 - **Accommodate needs**
 - **Users**
 - **DOE strategic**
 - **Encourage high parallel concurrency**
 - **Maximize scientific productivity**
- **Special requests always given consideration**
 - **Reservations**
 - **Emergencies**



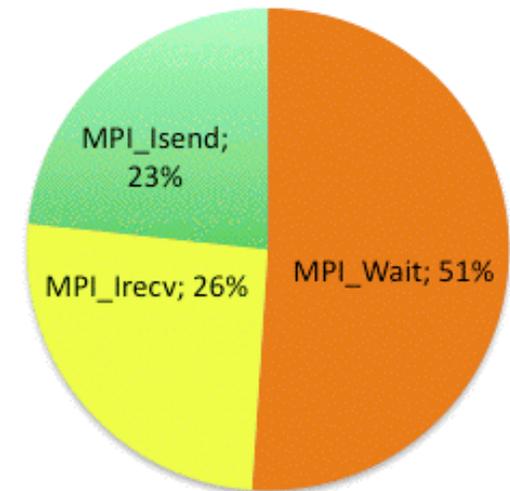
Batch Queues

- **regular: production runs**
- **debug: short, small test runs**
- **premium: I need it now, 2X charge**
 - Fast turn around on Franklin, not usually needed
- **low: I can wait a while: 1/2 charge**
- **special: unusual jobs by prior arrangement**
- **Interactive: implicit in qsub -I**

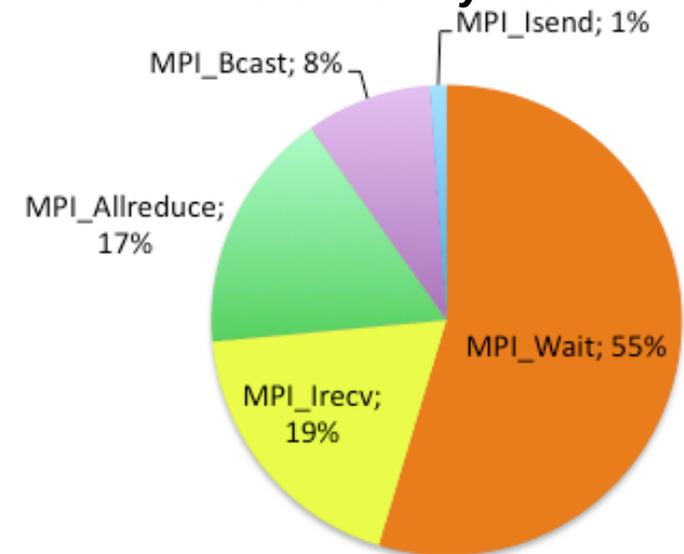
Code Profiling

- Profiling tools are too heavy
- Users want to do science, not spend months looking a code profiles
- But, profiling can still be valuable...
- ! • At the very least user should know
 - Amount of aggregate memory
 - Amount of memory per core
 - % MPI time
 - % Time in IO
- Franklin tools
 - CrayPat and Apprentice from Cray
 - IPM

MPI Calls by Count



MPI Calls by Time



Disk Space

- **Disk space is expensive and therefore limited and shared among users**
- **Every center must manage disk space in some way (purging, begging, quotas)**
- **Understand the disk usage policy at your center**
- **Be a courteous disk space user. We want you to run very large jobs, but then we want you to back up your files (quickly)**

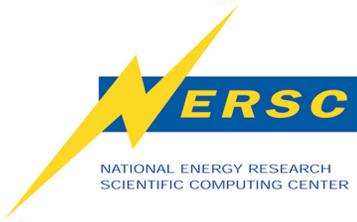
Disk Quotas

- **Each NERSC system has 2 local file systems**
 - **/home**
 - small (10s GB)
 - Backed up
 - Permanent
 - **/scratch**
 - larger (100s GB-TBs)
 - not backed up
 - not permanent
- **Projects needing larger disk quotas just need to ask**
- **Still disk space over subscribed, when usage gets high we resort to begging first, then purging old files**
- **NERSC Global File System can be a solution**

NERSC Global Filesystem (NGF)

- Seamless data access from NERSC's computational and analysis resources
- Single unified namespace makes it easier for users to manage their data across multiple system
- Goals: Functionality, Reliability, Performance





Large Storage Environment (HPSS)

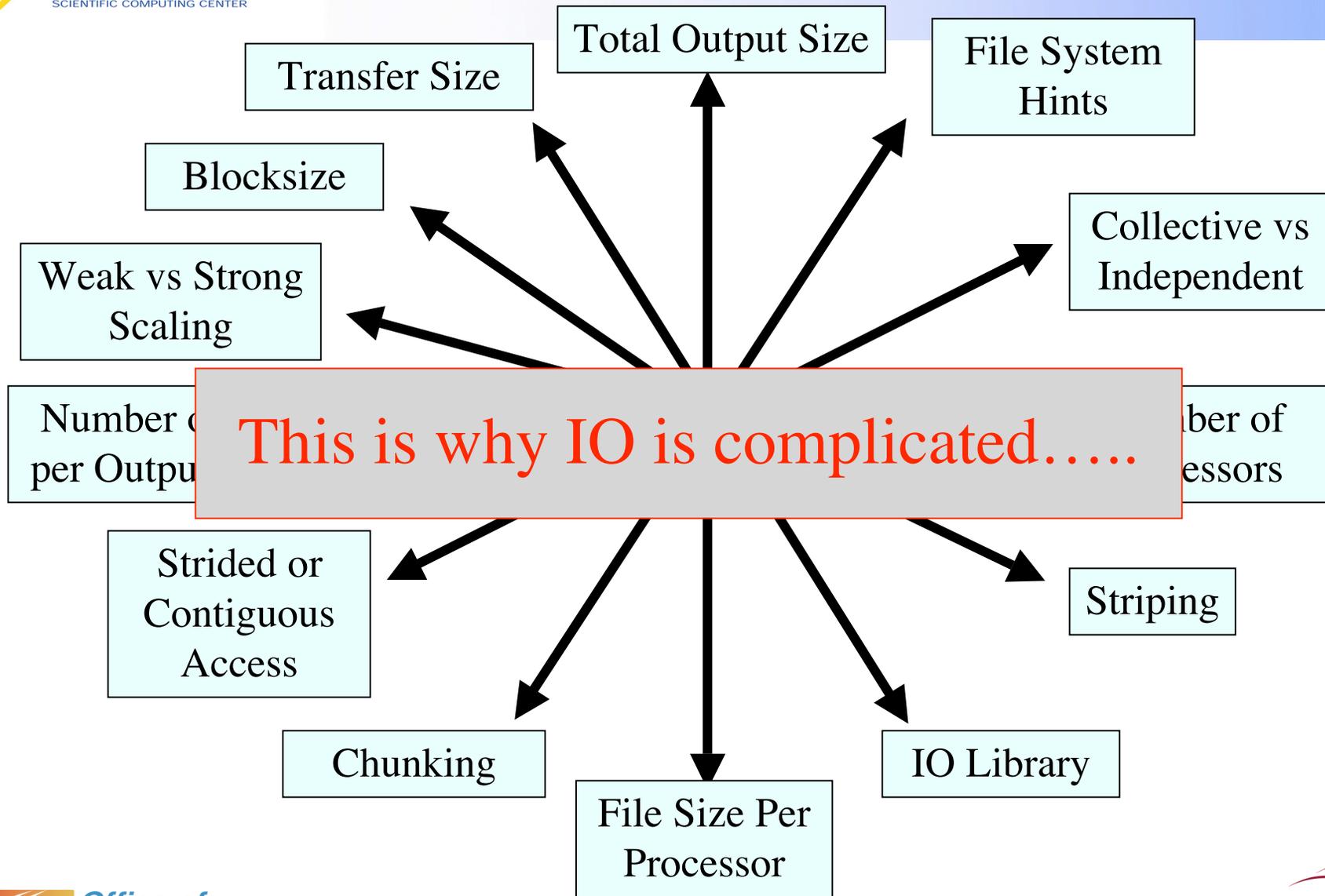
- Access HPSS from any NERSC resource with:
 - HIS
 - HTAR
 - ftp/pftp
- Outside NERSC must do a few extra steps
- 61+ million files
- 44 PB capacity





IO on Franklin

Axis of IO



- **User Wish List**

- Write data from multiple processors into a single file
- File can be read in the same manner regardless of the number of CPUs that read from or write to the file. (eg. want to see the logical data layout... not the physical layout)
- Do so with the same performance as writing one-file-per-processor (only writing one-file-per-processor because of performance problems)
- And make all of the above portable from one machine to the next

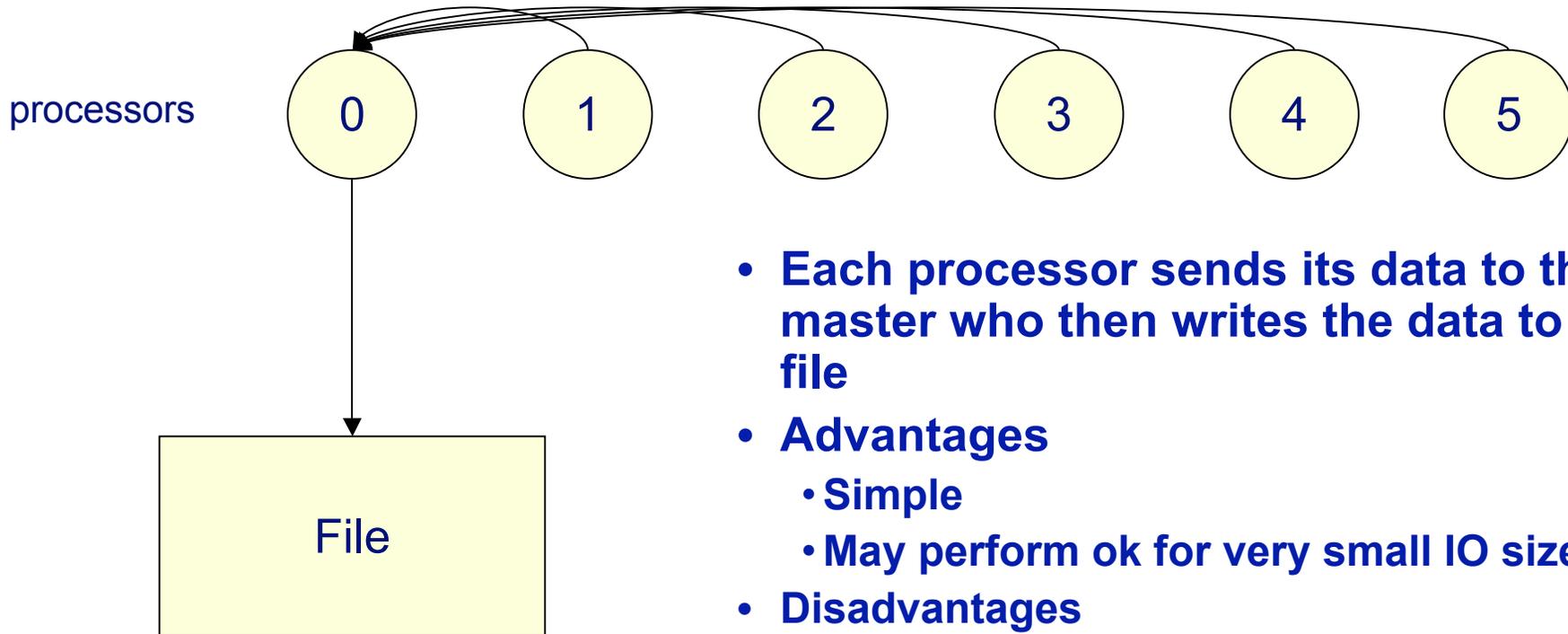
- **IO on Franklin**

- /home and /scratch file use Lustre File System
- Unfortunately users need to know about file system in order to get best performance



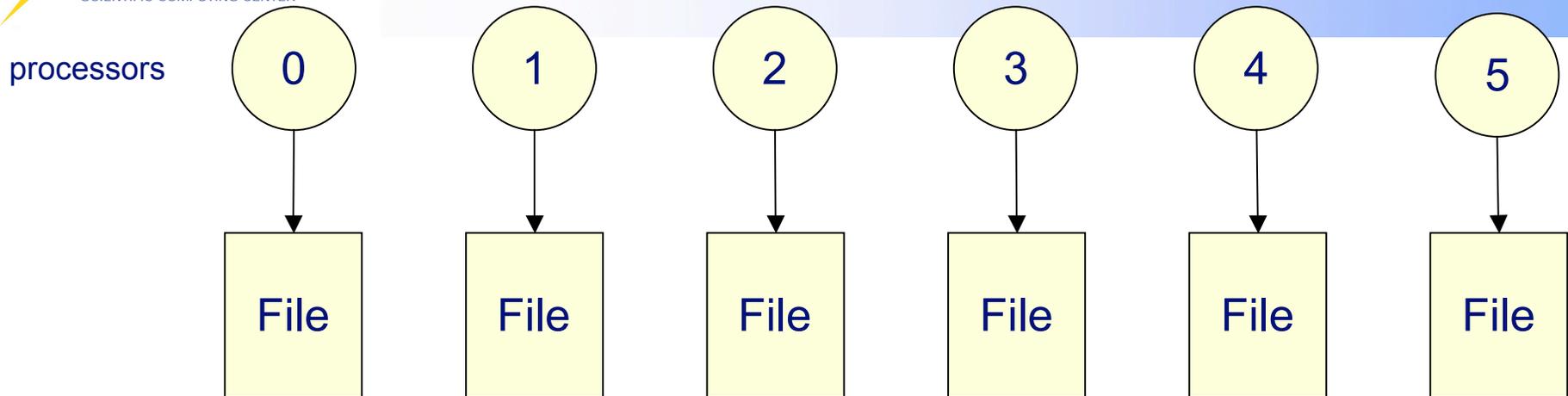
- Failures happen - recommend checkpointing code

Serial I/O



- **Each processor sends its data to the master who then writes the data to a file**
- **Advantages**
 - Simple
 - May perform ok for very small IO sizes
- **Disadvantages**
 - Not scalable
 - Not efficient, slow for any large number of processors or data sizes
 - May not be possible if memory constrained

Parallel I/O Multi-file



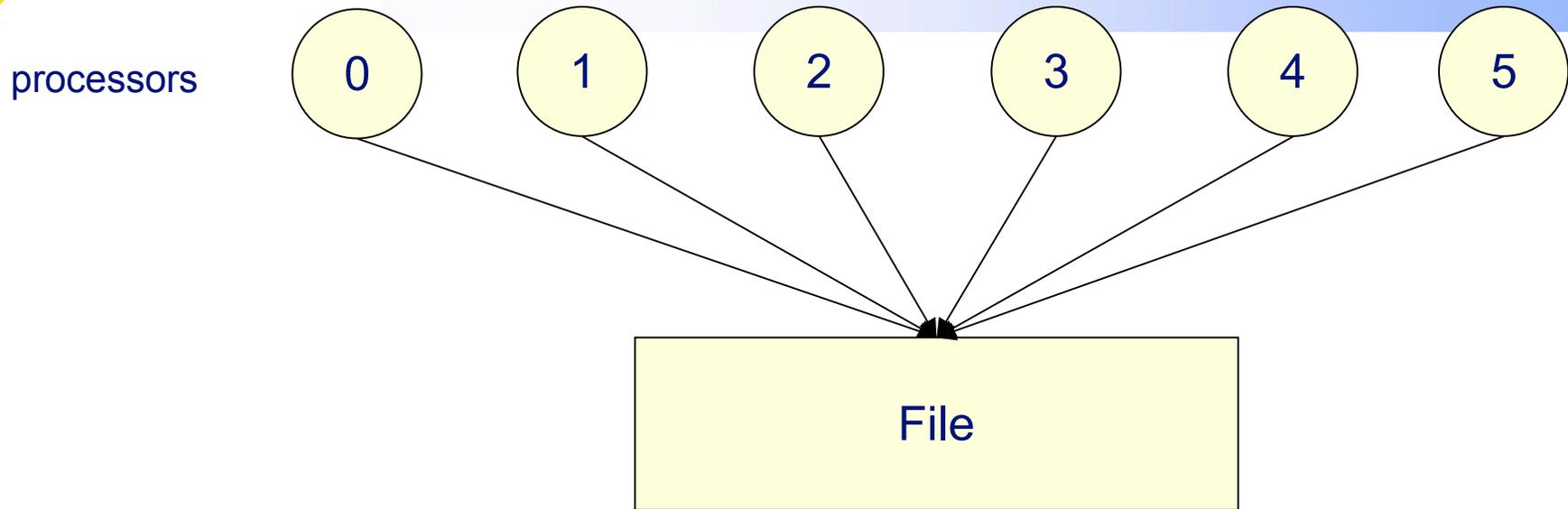
- **Each processor writes its own data to a separate file**
- **Advantages**
 - Simple to program
 - Can be fast -- (up to a point)
- **Disadvantages**
 - Can quickly accumulate many files
 - With Lustre, hit metadata server limit
 - Hard to manage
 - Requires post processing
 - Difficult for storage systems, HPSS, to handle many small files



Flash Center IO Nightmare...

- **Large 32,000 processor run on LLNL BG/L**
- **Parallel IO libraries not yet available**
- **Intensive I/O application**
 - **checkpoint files .7 TB, dumped every 4 hours, 200 dumps**
 - used for restarting the run
 - full resolution snapshots of entire grid
 - **plotfiles - 20GB each, 700 dumps**
 - coarsened by a factor of two averaging
 - single precision
 - subset of grid variables
 - **particle files 1400 particle files 470MB each**
- **154 TB of disk capacity**
- **74 million files!**
- **Unix tool problems**
- **Took 2 years to sift through data, sew files together**

Parallel I/O Single-file



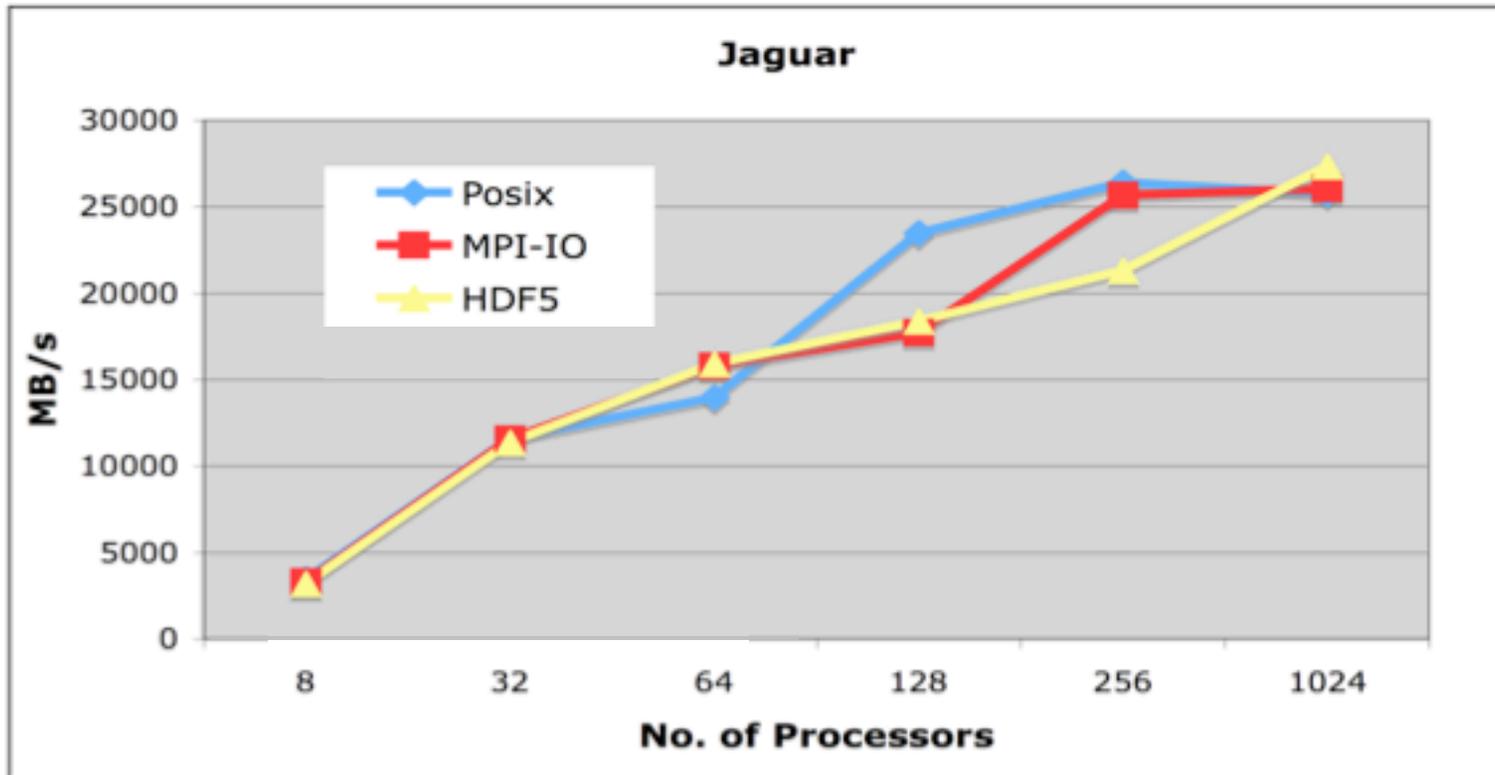
- **Each processor writes its own data to the same file using MPI-IO mapping**
- **Advantages**
 - Single file
 - Manageable data
- **Disadvantages**
 - Lower performance than one file per processor at some concurrencies

Common Storage Formats

- **ASCII:**
 - Slow
 - Takes more space!
 - Inaccurate
- **Binary**
 - Non-portable (eg. byte ordering and types sizes)
 - Not future proof
 - Parallel I/O using MPI-IO
- **Self-Describing formats**
 - NetCDF/HDF4, HDF5, Parallel NetCDF
 - Example in HDF5: API implements Object DB model in portable file
 - Parallel I/O using: pHDF5/pNetCDF (hides MPI-IO)
- **Community File Formats**
 - FITS, HDF-EOS, SAF, PDB, Plot3D
 - Modern Implementations built on top of HDF, NetCDF, or other self-describing object-model API

Many NERSC users at this level. We would like to encourage users to transition to a higher IO library

IO Library Overhead



Very little, if any overhead from HDF5 for one file per processor IO compared to Posix and MPI-IO

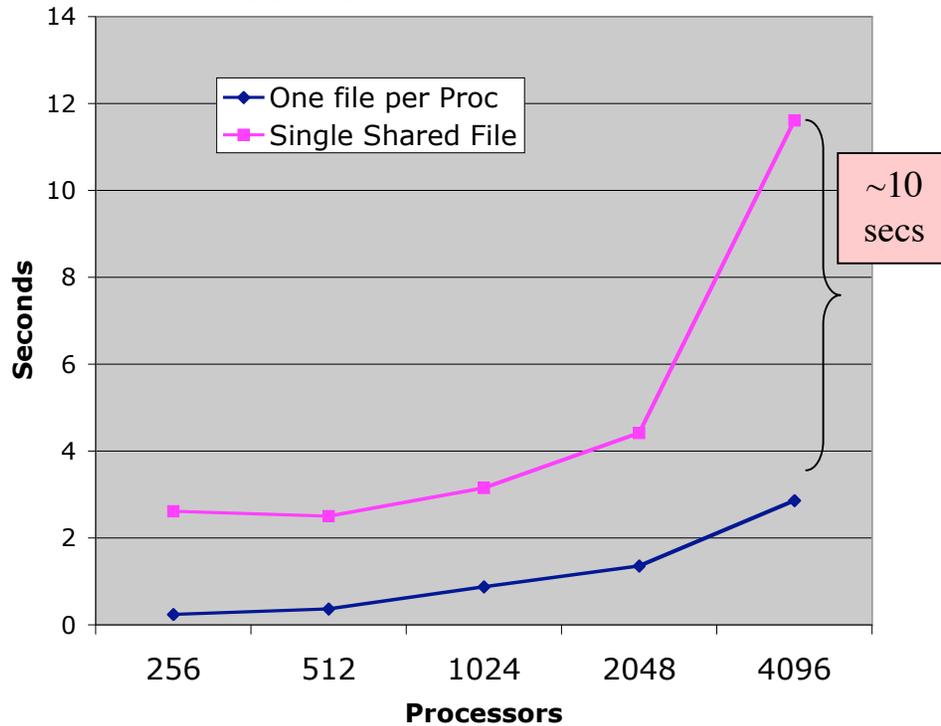


A Plug for Self Describing Formats ...

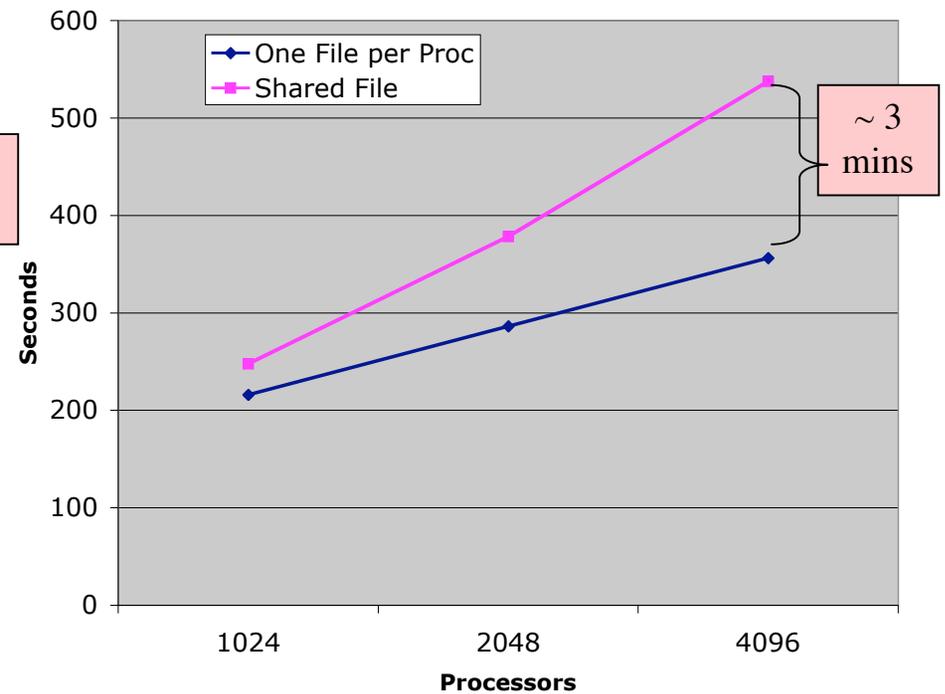
- **Application developers shouldn't care about about physical layout of data**
- **Using own binary file format forces user to understand layers below the application to get optimal IO performance**
- **Every time code is ported to a new machine or underlying file system is changed or upgraded, user is required to make changes to improve IO performance**
- **Let other people do the work**
 - **HDF5/pNetCDF can be optimized for given platforms and file systems by developers**
 - **User can stay with the high level**
- **But what about performance?**

File Per Processor vs Shared File Time

Aggregate File Size 1 GB



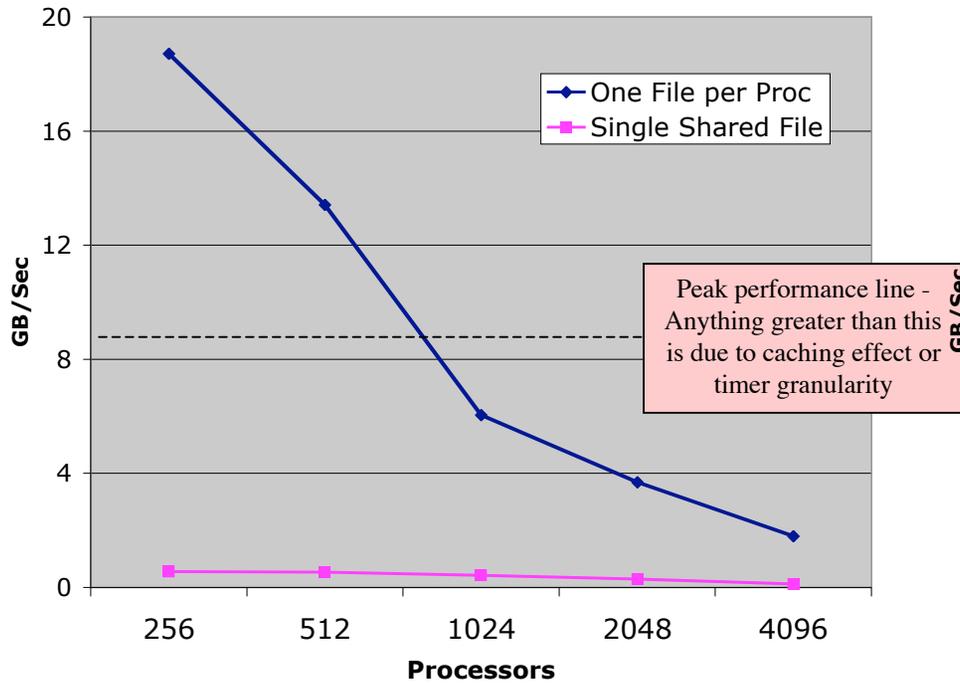
Aggregate File Size 1 TB



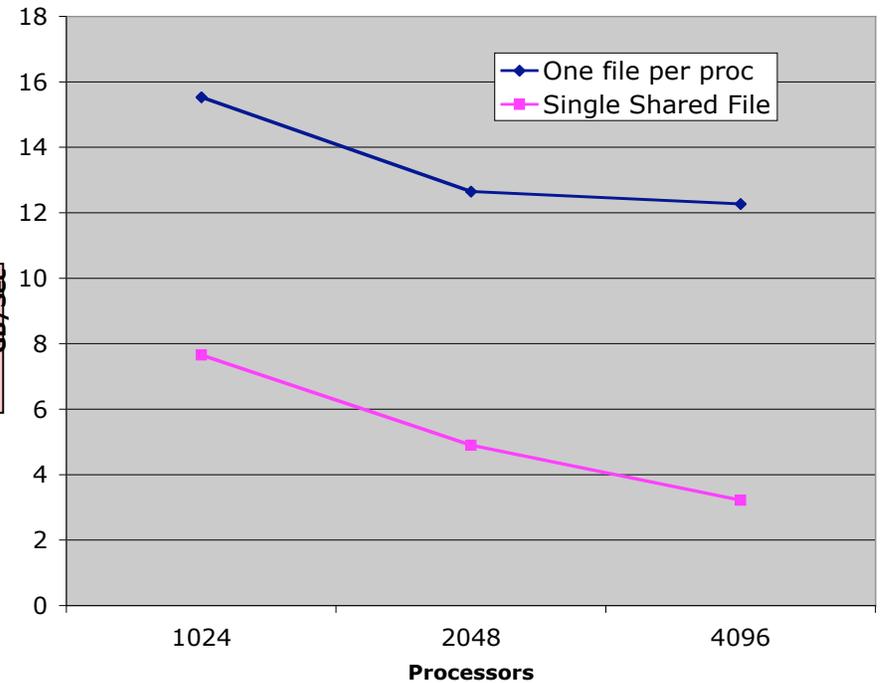
How much overhead can you afford?

File Per Processor vs Shared File -- Rate GB/Sec

Aggregate File Size 1 GB



Aggregate File Size 1 TB



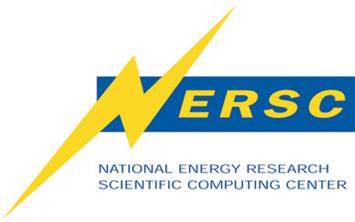
How much overhead can you afford?

What is Striping?

- **Lustre file system on Franklin made up of an underlying set of file systems calls Object Storage Targets (OSTs), essentially a set of parallel IO servers**
- **File is said to be striped when read and write operations access multiple OSTs concurrently**
- **Striping can be a way to increase IO performance since writing or reading from multiple OSTs simultaneously increases the available IO bandwidth**

What is Striping?

- **File striping will most likely improve performance for applications which read or write to a single (or multiple) large shared files**
- **Striping will likely have little effect for the following type of IO patterns**
 - **Serial IO where a single processor performs all the IO**
 - **Multiple node perform IO, but access files at different times**
 - **Multiple nodes perform IO simultaneously to different files that are small (each < 100 MB)**
 - **One file per processor**

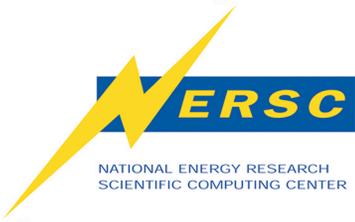


Striping Commands

- **Striping can be set at a file or directory level**
- **Set striping on an directory then all files created in that directory with inherit striping level of the directory**
- **Moving a file into a directory with a set striping will NOT change the striping of that file**

lfs setstripe <directory|file> <stripe size> <OST Offset> <stripe count>

- **stripe-size -**
 - Number of bytes in each stripe (multiple of 64k block)
- **OST offset -**
 - Always keep this -1
 - Choose starting OST in round robin
- **stripe count -**
 - Number of OSTs to stripe over
 - -1 stripe over all OSTs
 - 1 stripe over one OST

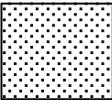


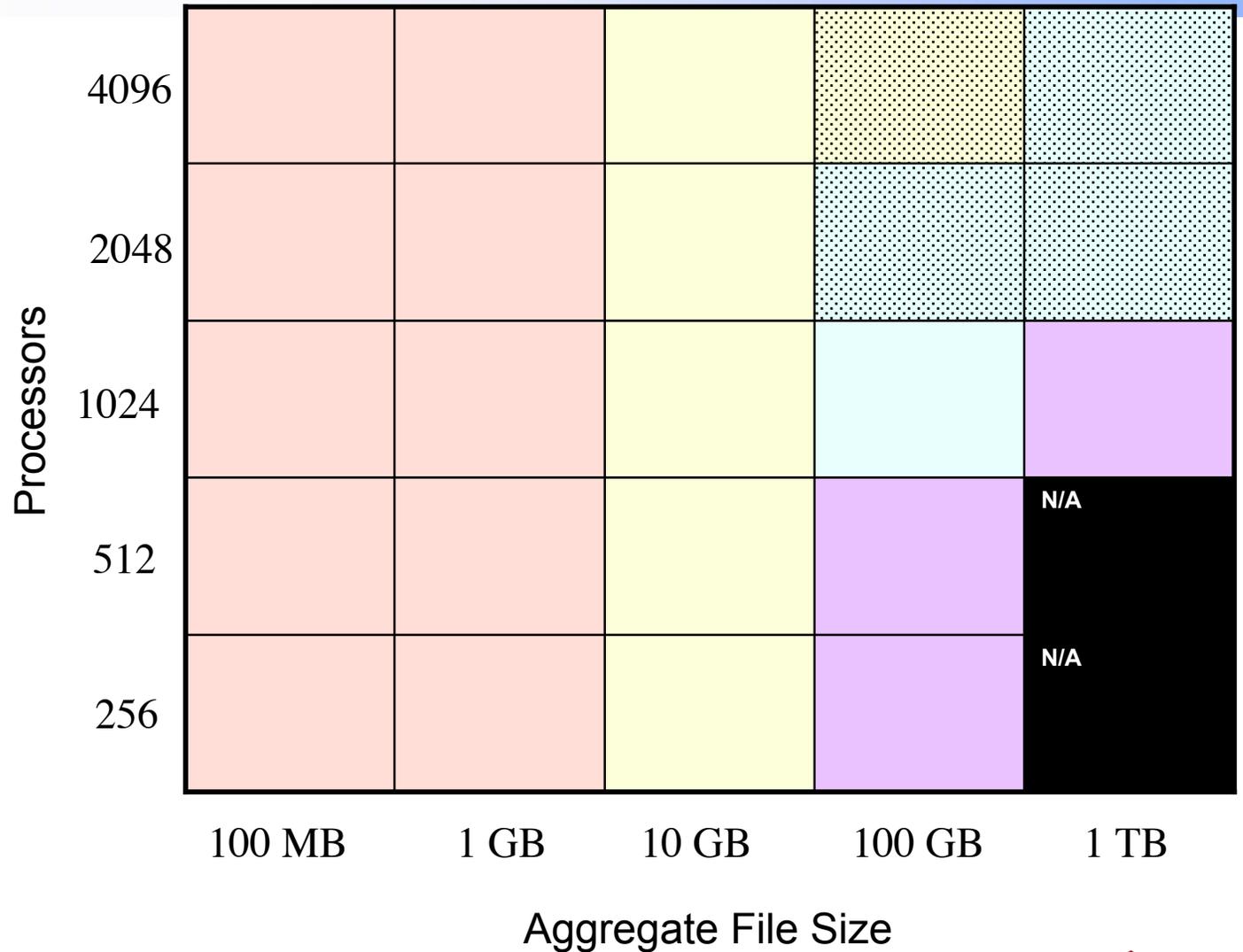
Stripe-Count Suggestions

- **Franklin Default Striping**
 - 1MB stripe size
 - Round robin starting OST (OST Offset -1)
 - Stripe over 4 OSTs (Stripe count 4)
- **Many small files, one file per proc**
 - Use default striping
 - Or 0 -1, 1
- **Large shared files**
 - Stripe over all available OSTs (0 -1 -1)
 - Or some number larger than 4 (0 -1 X)
- **Stripe over odd numbers?**
- **Prime numbers?**

Recommendations

Legend

	Single Shared File, Default or No Striping
	Single Shared File, Stripe over some OSTs (~10)
	Single Shared File, Stripe over many OSTs
	Single Shared File, Stripe over many OSTs OR File per processor with default striping
	Benefits to mod n shared files





Recommendations

- **Think about the big picture**
 - **Run time vs Post Processing trade off**
 - **Decide how much IO overhead you can afford**
 - **Data Analysis**
 - **Is there analysis you can do during your production run?**
 - **Portability**
 - **Longevity**
 - **H5dump/ncmpidump works on all platforms**
 - **Can view an old file with h5dump/ncmpidump**
 - **If you use your own binary format you must keep track of not only your file format version but the version of your file reader as well**
 - **Storability**



Recommendations

- **Use a standard IO format, even if you are following a one file per processor model**
- **One file per processor model really only makes some sense when writing out very large files at high concurrencies, for small files, overhead is low**
- **If you must do one file per processor IO then at least put it in a standard IO format so pieces can be put back together more easily**
- **Follow striping recommendations**
- **Consider the value of your time -- even if your advisor is not**
- **Ask the consultants, we are here to help!**



Portability and Flexibility

- **HPC machines change quickly**
 - **NERSC does technology refresh every 3 years**
 - **Always consider challenge for users moving to machine but we need to keep up with technology and market**
- **Codes become more robust when run on multiple platforms with multiple compilers**
- **Familiarity with different programming models, performance tools and debuggers gives you an advantage**
- **Be prepared to be able to move from center to center - go where the cycles are**

NERSC Science Over the Years



Please let us know what we can do to help!

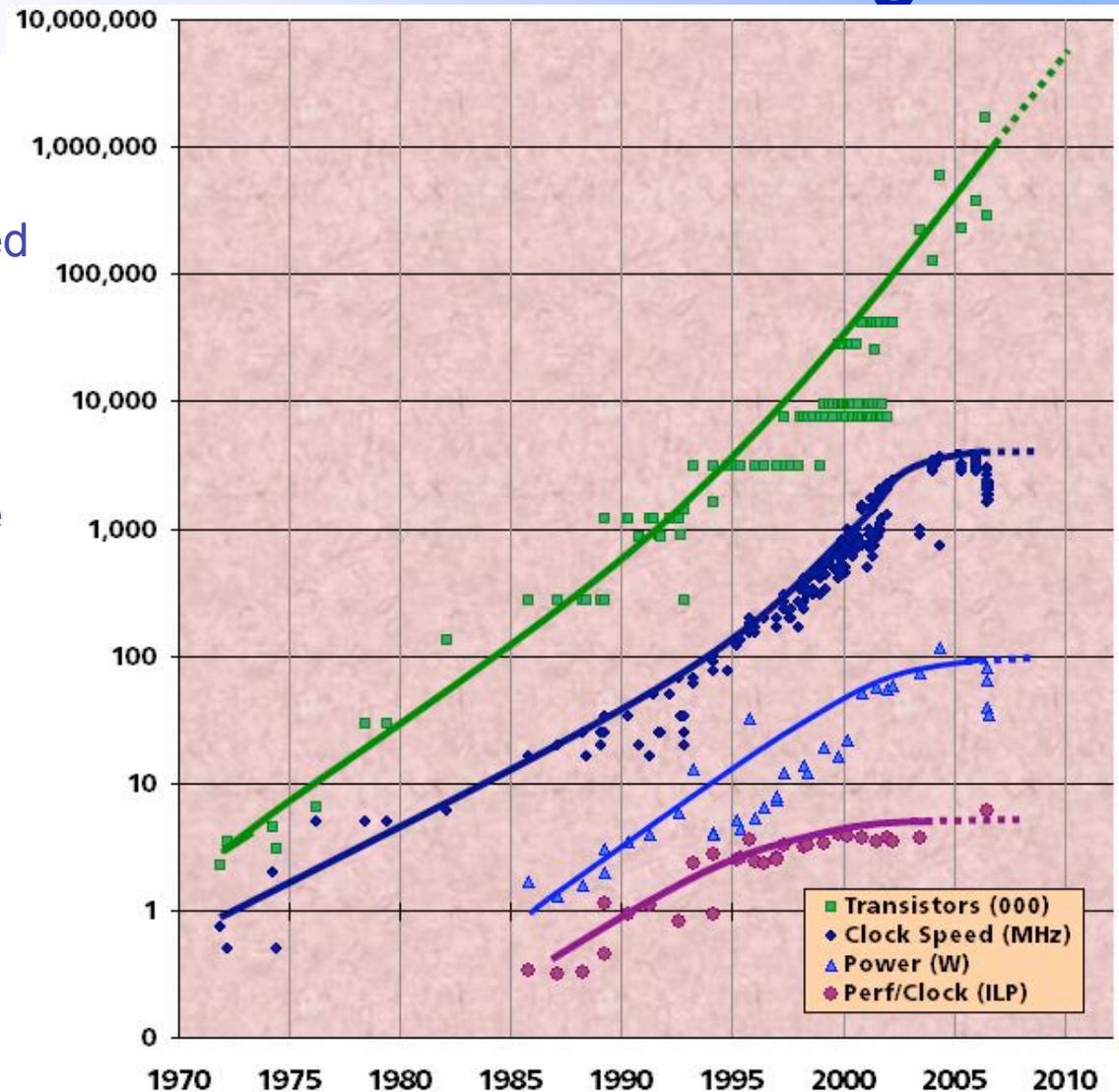
Consult@neresc.gov



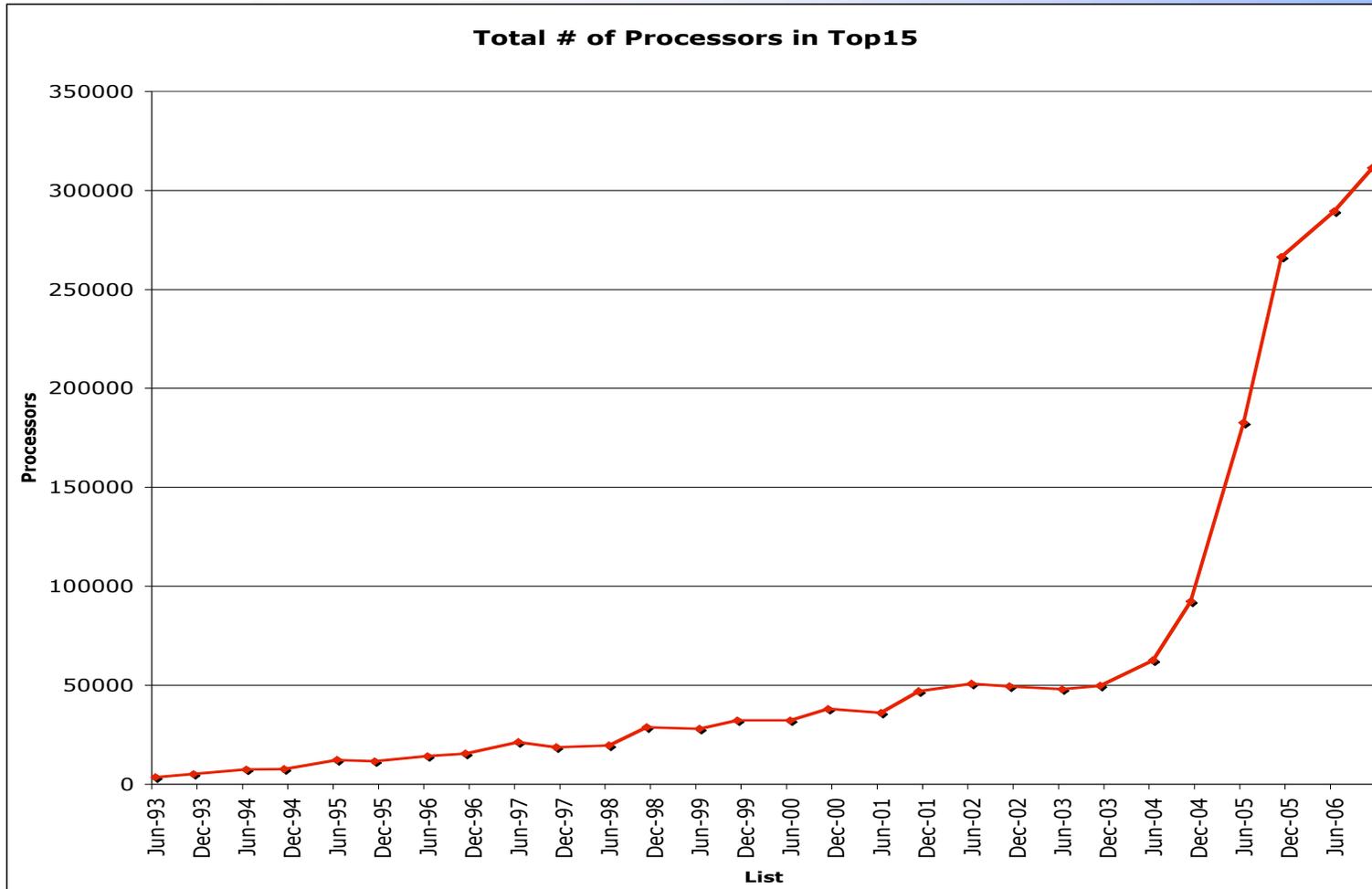
Tips for running successfully at NERSC (or any HPC center)

Traditional Sources of Performance Improvement are Flat-Lining

- **New Constraints**
 - 15 years of *exponential* clock rate growth has ended
- **But Moore's Law continues!**
 - How do we use all of those transistors to keep performance increasing at historical rates?
 - Industry Response: #cores per chip doubles every 18 months *instead* of clock frequency!



Growth in HPC System Concurrency



Must ride exponential wave of increasing concurrency for foreseeable future!

You will hit 1M cores sooner than you think!



Application Community's Response to Technology Trends

- **Parallel computing has thrived on weak-scaling for past 15 years**
- **Flat CPU performance increases emphasis on strong-scaling**
- **Workload Requirements will change accordingly**
 - Concurrency will increase proportional to system scale
 - Timestepping algorithms will be increasingly driven towards implicit or semi-implicit stepping schemes
 - Multiphysics/multiscale problems increasingly rely on spatially adaptive approaches such as Berger-Oliger AMR
 - Strong scaling will push applications towards smaller messages sizes – requiring lighter-weight messaging



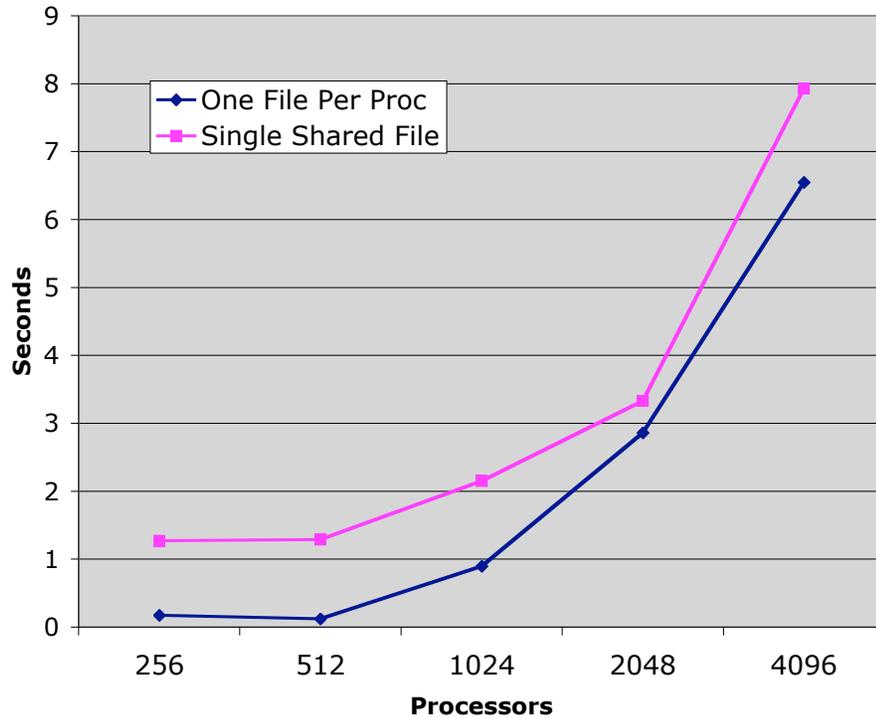
The Multicore Future

- **Start thinking about implications of multicore**
- **Don't need to have answers (nobody does)**
- **How can you increase parallelism?**
- **Will your code require algorithmic changes?**
- **Where are the bottlenecks in your code?**
- **Would a different language/communication construct help you?**

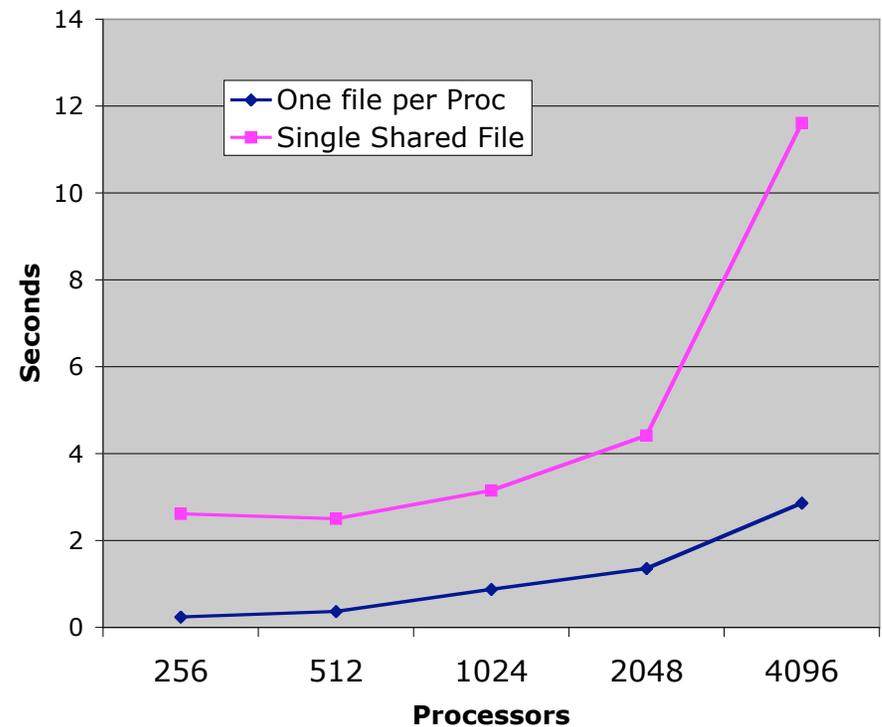
Small Aggregate Output Sizes 100 MB - 1GB

One File per Processor vs Shared File - Time

Aggregate File Size 100 MB



Aggregate File Size 1 GB



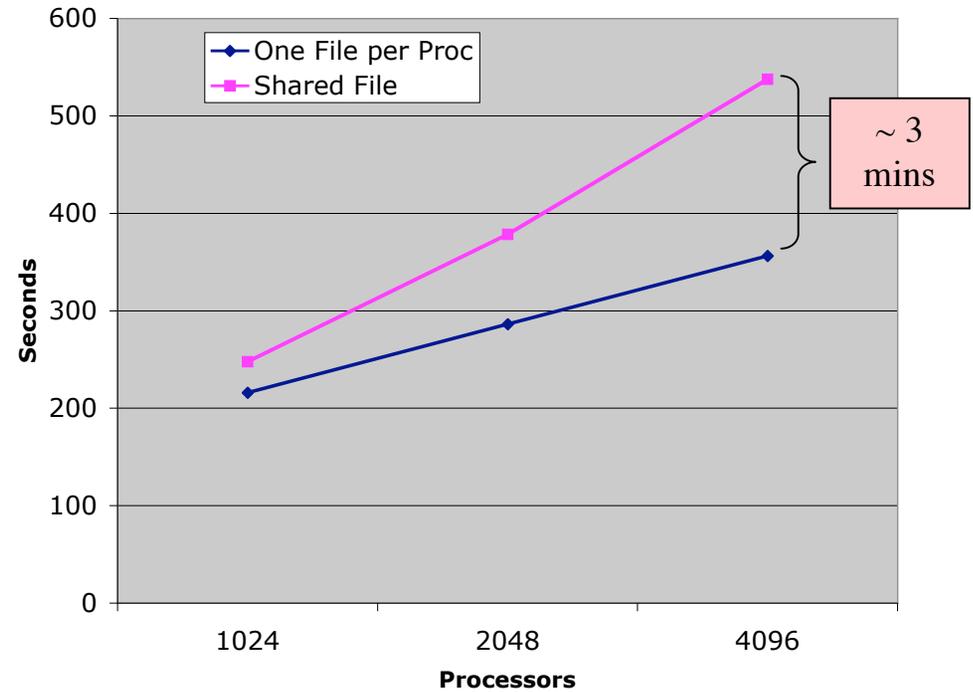
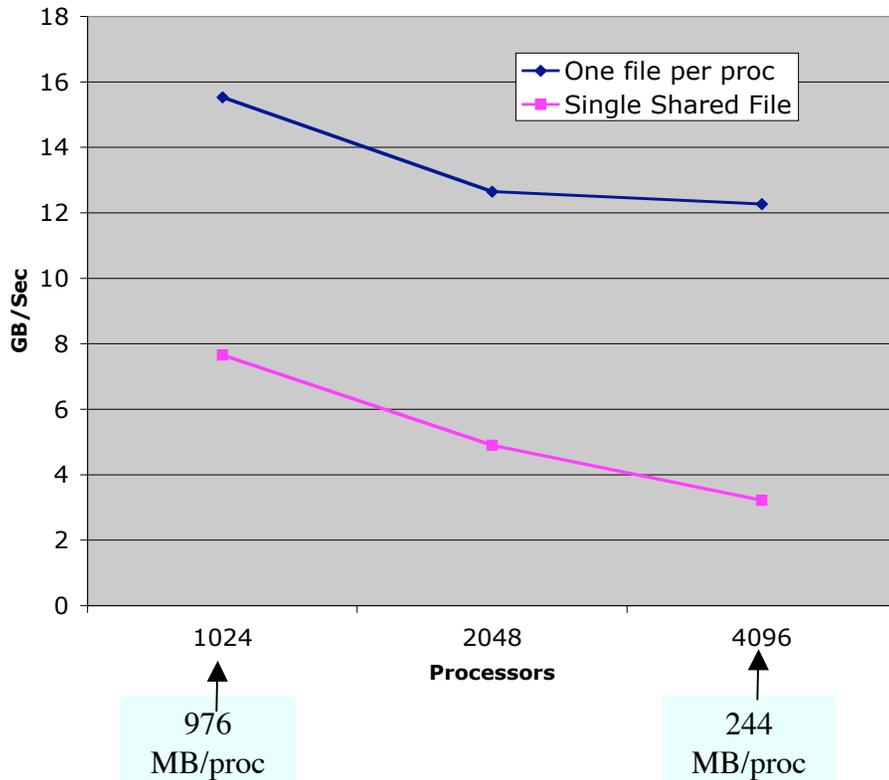
But when looking at absolute time, the difference doesn't seem so big...

Aggregate Output Size 1TB

One File per Processor vs Shared File

Rate: GB/Sec

Time: Seconds



Is there anything we can do to improve the performance of the 4096 processor shared file case ?



Science Stories



Simulation of a Low Swirl Burner Fueled with Hydrogen

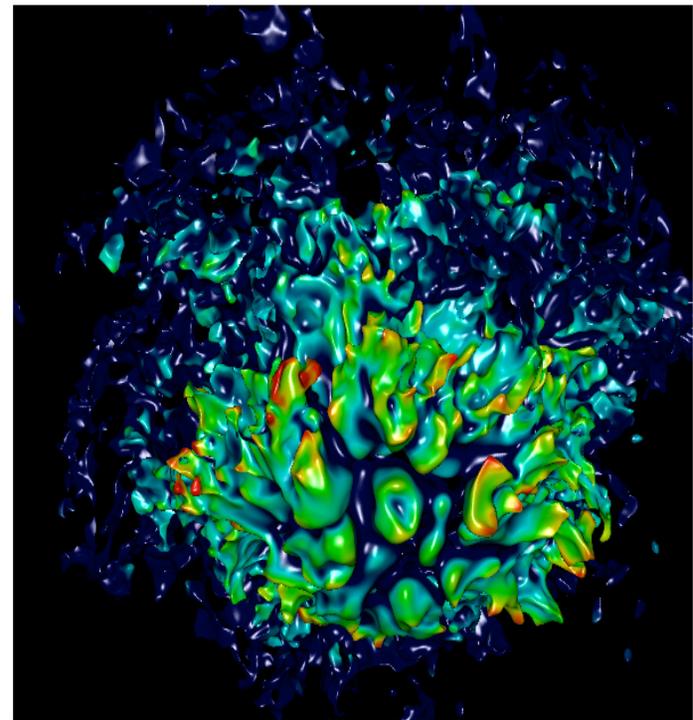
- **Calculation:** Numerical simulation of flame surface of an ultra-lean premixed hydrogen flame in a laboratory-scale low-swirl burner. Burner is being developed for fuel-flexible, near-zero-emission gas turbines.
- **PI: John Bell**

Science Result:

- Detailed transport and chemical kinetics using an adaptive low Mach number algorithm for reacting flow.

Scaling:

- Adaptive Mesh Refinement used to save memory and time.
- Scales to 6K cores, typically run at 2K
- Used 2.2M early science hours on Franklin



Nanoscience Calculations and Scalable Algorithms

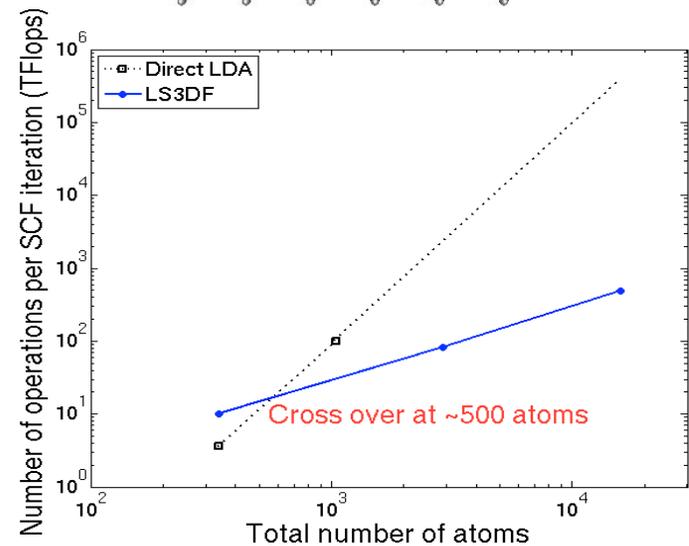
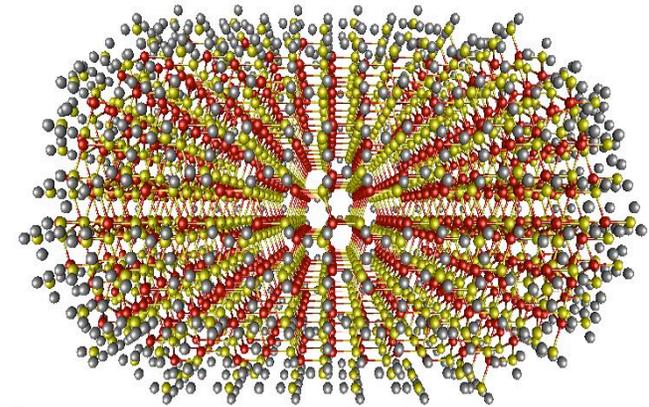
- **Calculation: Linear Scaling 3D Fragment (LS3DF). Density Functional Theory (DFT) calculation numerically equivalent to more common algorithm, but scales with $O(n)$ in number of atoms rather than $O(n^3)$**
- **PI: L.W. Wang, LBNL**

Science Results

- Calculated dipole moment on 2633 atom CdSe quantum rod, $\text{Cd}_{961}\text{Se}_{724}\text{H}_{948}$.

Scaling Results

- Ran on 2560 cores
- Took 30 hours vs many months for $O(n^3)$ algorithm
- Good parallel efficiency (80% on 1024 relative to 64 procs)



Middle Users Capable Large-Scale Computational Science

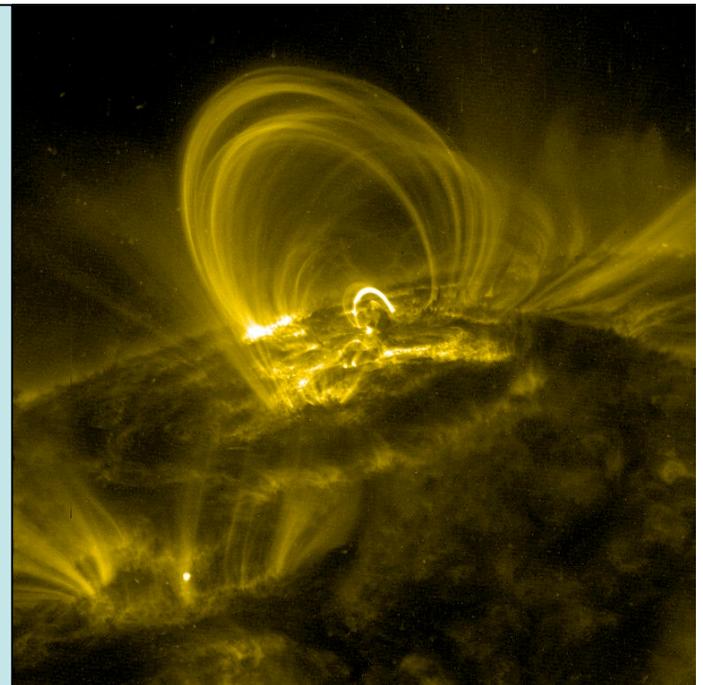
- **Calculations: AstroGK gyrokinetic code for astrophysical plasmas**
- **PIs: Dorland (U. of Maryland), Howes, Tatsuno**

- **Science Results**

- **Shows how magnetic turbulence leads to particle heating**

- **Scaling Results**

- **Runs on 16K cores**
- **Combines implicit and explicit methods**



Modeling Dynamically and Spatially Complex Materials for Geoscience

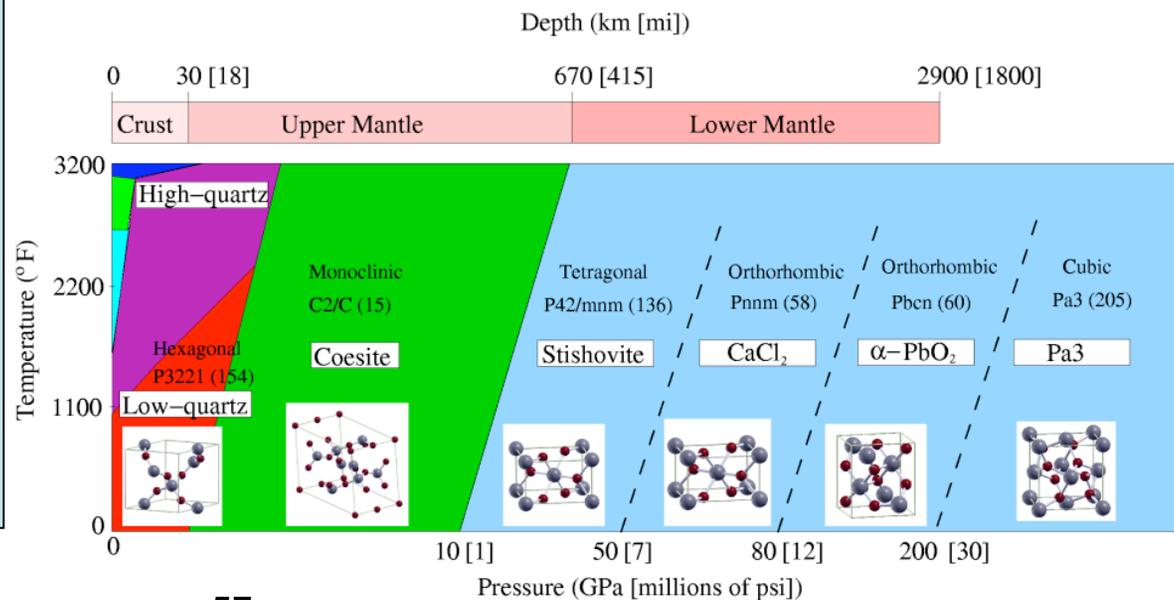
- **Calculation: Simulation of seismic waves through silicates, which make up 80% of the Earth's mantle**
- **PI: John Wilkins, Ohio State University**

• Science Result

- **Seismic analysis shows jumps in wave velocity due to structural changes in silicates under pressure**

• Scaling Result

- **First use of Quantum Monte Carlo (QMC) for computing elastic constants**
- **8K core vs. 128 on allocated time**

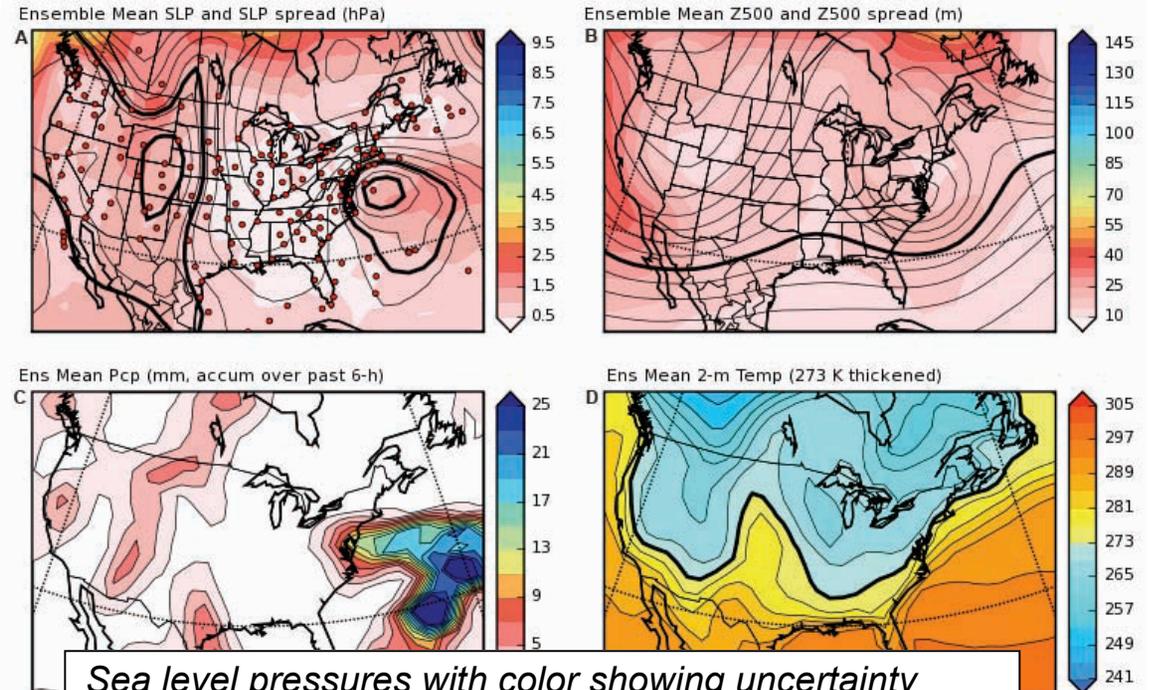


Validating Climate Models

- INCITE Award for “20th Century Reanalysis” using an Ensemble Kalman filter to fill in missing climate data since 1892

• Dr. G. Compo, U. Boulder

- **Science Results:**
 - Reproduced 1922 Knickerbocker storm
 - Data can be used to validate climate and weather models
- **Scaling Results:**
 - 3.1M CPU Hours in allocation
 - Scales to 2.4K cores
 - Switched to higher resolution algorithm with Franklin access



Sea level pressures with color showing uncertainty (a&b); precipitation (c); temperature (d). Dots indicate measurements locations (a).

Nuclear Physics

- **Calculation:** High accuracy *ab initio* calculations on O^{16} using no-core shell model and no-core full configuration interaction model
- **PI:** James Vary, Iowa State

- **Science Results:**

- Most accurate calculations to date on this size nuclei
- Can be used to parametrize new density functionals for nuclear structure simulations

- **Scaling Results:**

- 4M hours used; 200K allocated
- 12K cores; vs 2-4K before Franklin uncharged time
- Diagonalize matrices of dimension up to 1 billion

