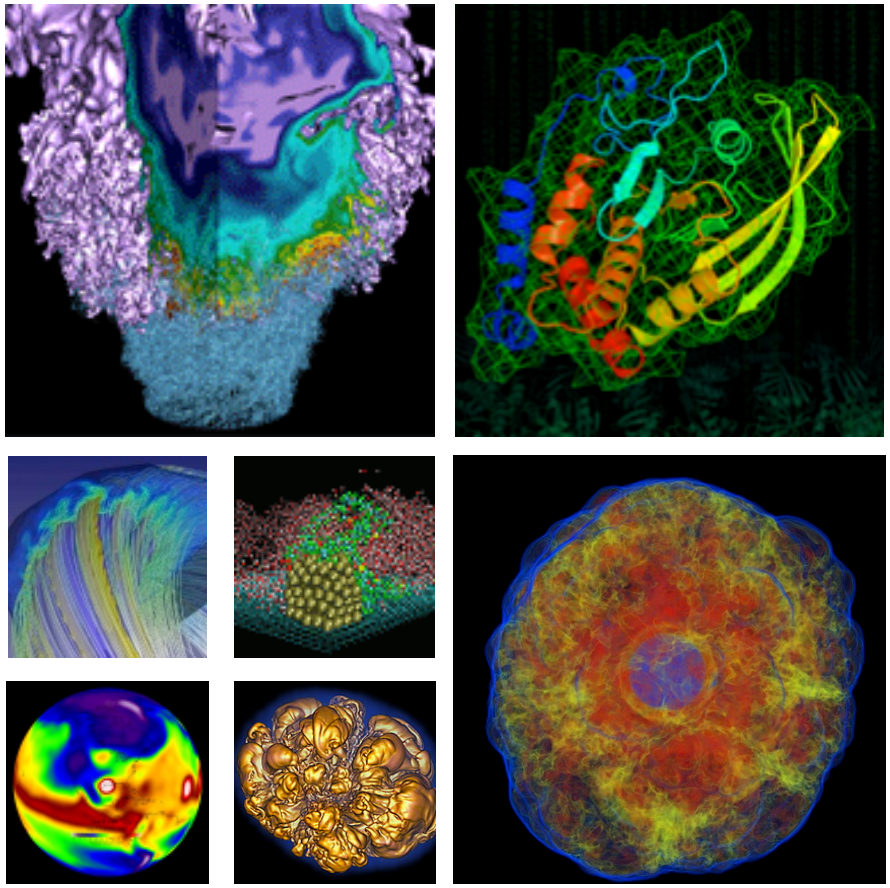
Getting Started at NERSC



Daniel Udwary

NERSC Data Science Engagement Group

June 22, 2016

U.S. DEPARTMENT OF ENERGY | Office of Science

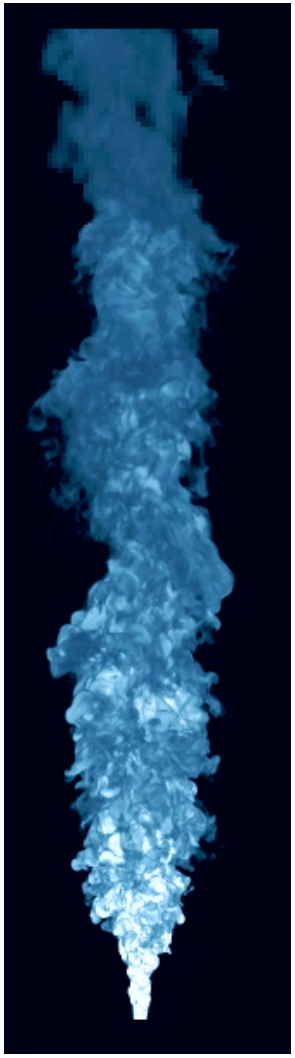
BERKELEY LAB

Purpose

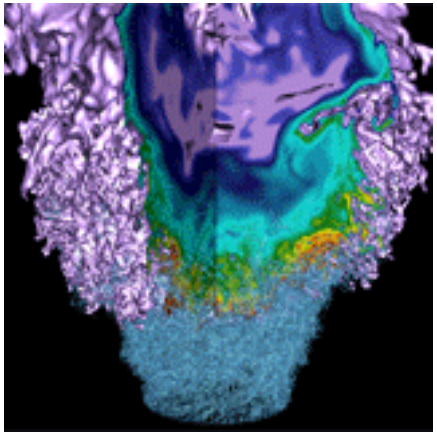
- **This presentation will help you get familiar with NERSC and its facilities**
  - Practical information
  - Introduction to terms and acronyms
- **This is not a programming tutorial**
  - But you will learn how to get help and what kind of help is available
  - We can give presentations on programming languages and parallel libraries – just ask

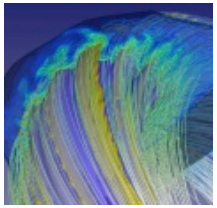
Outline

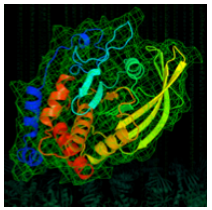
- NERSC and its computing systems
- How to get help
- Running and Monitoring Jobs
- Software Modules

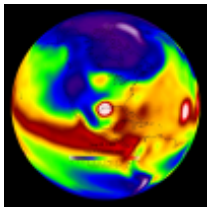


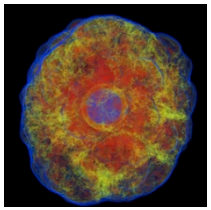
Computing Resources

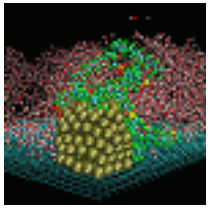












Current NERSC Systems

Large-Scale Computing Systems

Edison (NERSC-7): Cray Cascade

- Over 200 Tflop/s on applications, 2 Pflop/s peak

Cori (NERSC-8):

- "Phase I" is currently operational

NERSC-9 in planning

Midrange

>140 Tflops total

Mendel

PDSF (HEP/NP)

- ~1K core cluster

GenePool (JGI)

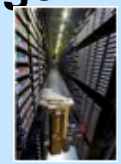
- ~8K core clusters
- 7.1 PB GPFS File System

NERSC Global Filesystem (NGF)

Uses IBM's GPFS

- 8.5 PB capacity
- 15GB/s of bandwidth

HPSS Archival Storage



- 240 PB capacity
- 5 Tape libraries
- 200 TB disk cache

Analytics & Testbeds

Babbage

Xeon Phi

NERSC is in a new building at LBL

- **All systems recently moved from Oakland to Berkeley**

NERSC Services

- **NERSC's emphasis is on enabling scientific discovery**
- **User-oriented systems and services**
  - We think this is what sets NERSC apart from other centers
- **Help Desk / Consulting**
  - Immediate direct access to consulting staff that includes many Ph.Ds
- **User group (NUG) has tremendous influence**
  - Monthly teleconferences & yearly meetings
- **Requirement-gathering workshops with top scientists**
  - One each for the six DOE Program Offices in the Office of Science
  - http://www.nersc.gov/science/requirements-workshops/
- **Ask, and we'll do whatever we can to fulfill your request**

Your NERSC Consultants

Kjiersten Fagnan
Applied Math
JGI's CIO
and Head of NERSC's
Data Science Engagement
Group (DSEG)

Dan Udwary
Bioorganic chemistry

Tony Wildish
High Energy Particle Physics

Office hours are W & Th, 10-12.
Stop by 400-413 if you have questions!
consult@nersc.gov

What do your Consultants do?

- **Software installation, troubleshooting, optimization**
- **Genepool account support**
  - Setting up project membership, disk quotas
- **Liaison for other NERSC departments**
  - Computing Services (CSG): keeps your servers maintained
    - Bhupender Thakur
  - Infrastructure Services (ISG): web and db services
    - Cory Snavely, Brian Yumae, Matt Dunford
  - Filesystem admins (NGF)
    - Ravi Cheema
  - Data and Analytical Services (DAS): tool support
    - NoMachine, ipython, Globus
- **Advice on software, computation, bioinformatics**
  - "Data Science Engagement"

Where to get started on getting help

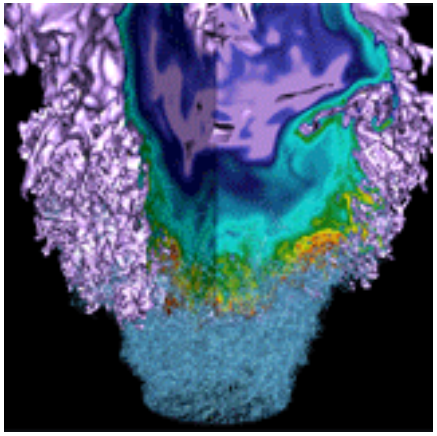
- **NERSC Genepool webpage**
  - https://www.nersc.gov/users/computational-systems/genepool/
- **Online Helpdesk – http://help.nersc.gov**
  - Create and monitor trouble tickets
  - Or email consult@nersc.gov
- **NERSC Information management (NIM) webpage**
  - https://nim.nersc.gov/ - change NERSC password
- **my.nersc.gov**
  - More information on your account and usage
- **Consulting line – 1-800-66-NERSC (menu option 3)**
  - Talk to a real live consultant 8-5, M-F

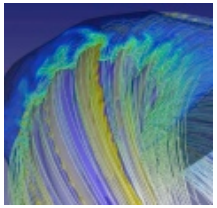
NERSC Information Management (NIM)

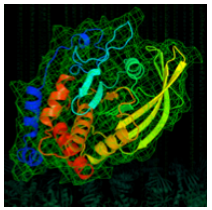
http://nim.nersc.gov

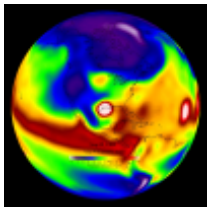
- **NERSC-wide management of passwords and other account information**
  - Same password used for all NERSC systems
    - 5 consecutive unsuccessful logins will lock you out
      - Log into NIM to reset failures, or reset password
  - ssh public keys should be deposited here
  - Be sure contact info (especially email) is kept up-to-date
  - Change your default shell (per system)
  - View NERSC account usage: compute time, quotas

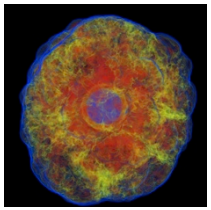
Resources provided by NERSC

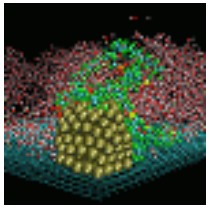




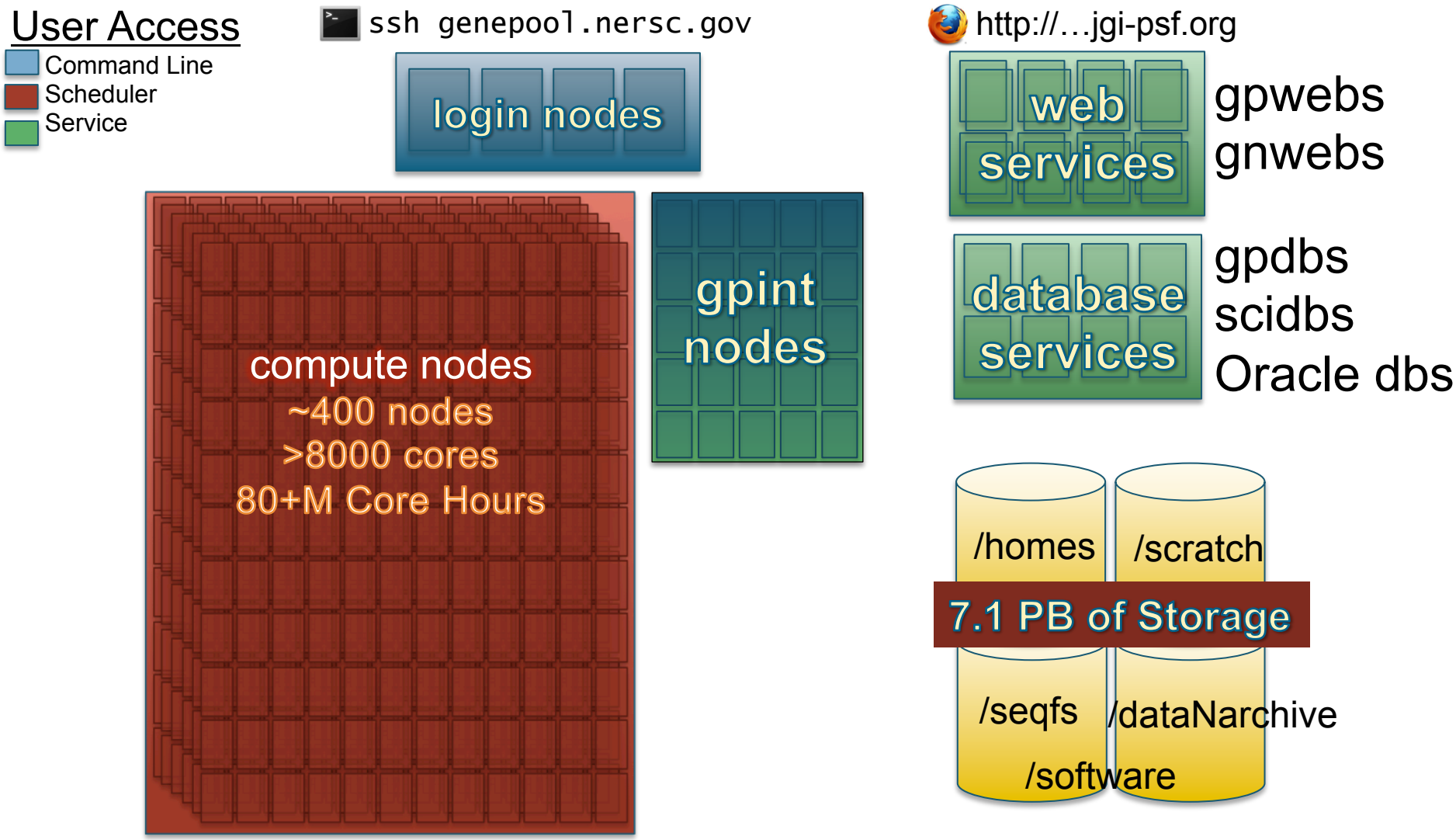








Structure of the Genepool system



JGI File Systems

Genepool is a heterogeneous computing environment

# Nodes	Cores/node	Memory/node	Local disk	Processor	Hostname	Vendor
2	32	1000GB	3.6TB	Xeon E5-4650L	mndlhm0205-ib,mndlhm0405-ib.nersc.gov	Appro/Cray
5	32	500GB	3.6TB	Xeon E5-4650L	mndlhm[01-05]03.nersc.gov	Appro/Cray
8	16	248GB	1.8TB	Xeon E5-2670	mc0xxx.nersc.gov	Appro/Cray
212	16	120GB	1.8TB	Xeon E5-2670	mc0xxx.nersc.gov	Appro/Cray
14	20	120GB			mc1322-33,mc1344	
36	32	120GB			mc1535-48,mc1601-22	
1	20	248GB			mc1359	
78	32	248GB			mc1637-48,mc1705-60	
12	32	500GB			mc1625-36	
1	80	2 TB	300 GB	Intel Xeon X7560 2.27 GHz	mndlmem03.nersc.gov	IBM

Interactive nodes (GPINTs)

Node Name	Assigned Group	Cores	Memory	Per-process Memory Limit
gpint200.nersc.gov	R & D	20	256G	208G
gpint201.nersc.gov	R & D	20	256G	208G
gpint202.nersc.gov	Plant	20	256G	208G
gpint203.nersc.gov	Plant	20	256G	208G
gpint204.nersc.gov	IMG	20	128G	101G
gpint205.nersc.gov	IMG	20	128G	101G
gpint206.nersc.gov	GBP	20	128G	101G
gpint207.nersc.gov	SDM	20	128G	101G
gpint208.nersc.gov	RQC	20	128G	101G
gpint209.nersc.gov	Assembly	20	128G	101G
gpint210.nersc.gov	Assembly	20	128G	101G
gpint211.nersc.gov	Vista	20	128G	101G
gpint212.nersc.gov	IMG	20	128G	101G
gpint213.nersc.gov	Fungal	20	128G	101G
gpint501.nersc.gov	IMG	20	512G	
gpint502.nersc.gov	Plant	20	512G	
gpint503.nersc.gov	Fungal	20	512G	

Connecting to Genepool

- **ssh genepool.nersc.gov**
  - Will take you to the least-utilized login node
  - ssh in MacOS, Linux. Putty commonly used in Windows

- **ssh gpint[xxx].nersc.gov**
  - If your group owns its own interactive node

- **Use NX for graphical (X-Windows) applications**
  - https://www.nersc.gov/users/connecting-to-nersc/using-nx/
  - Managed and supported by NERSC's DAS group

Home Directory

- When you log in you are in your "Home" directory.
- Permanent storage
  - Weeklong daily snapshots taken: $HOME/.snapshots
- The full UNIX pathname is stored in the environment variable $HOME

```
genepool04% echo $HOME
/global/homes/d/dudwary
```

- $HOME is a global file system
  - You see all the same directories and files when you log in to any NERSC computer.
- Your quota in $HOME is 40 GB and 1M inodes (files and directories).
- Use “myquota” command to check your usage and quota

Scratch Directories

- Each Genepool user may have, on request, a personal directory referenced by $SCRATCH and $BSCRATCH
  - on Genepool this points to /global/projectb/scratch/<username>
  - $SCRATCH is local on Edison and CORI (ie does not point to projectb).
- "Scratch" file systems are large, high-performance file systems, intended to be temporary.
  - Standard projectb scratch size: 20TB and 1M inodes
- Significant I/O from your compute jobs should be directed to $SCRATCH
- Data in $BSCRATCH is purged (12 weeks from last access)
- Always save data you want to keep to HPSS (see below)
- Data in $BSCRATCH is ***not*** backed up and could be lost if a file system fails.

Project Directories

- All NERSC systems mount the NERSC global "Project" file systems.
- Projectb is specific to the JGI, but is also accessible on Edison and Cori.
- "Project directories" are created upon request for projects (groups of researchers) to store and share data.
- Data in /projectb/projectdirs is not purged. This may change in the future, but for long term storage, you should use the archive.

IO Tips

- Use $SCRATCH for good IO performance
- Write large chunks of data (MBs or more) at a time
- Use a parallel IO library (e.g. HDF5)
- Read/write to as few files as practical from your code (try to avoid 1 file per MPI task)
- Use $HOME to compile unless you have too many source files or intermediate (*.o) files
- Do not put more than a few 1,000s of files in a single directory
- Save any and everything important to HPSS

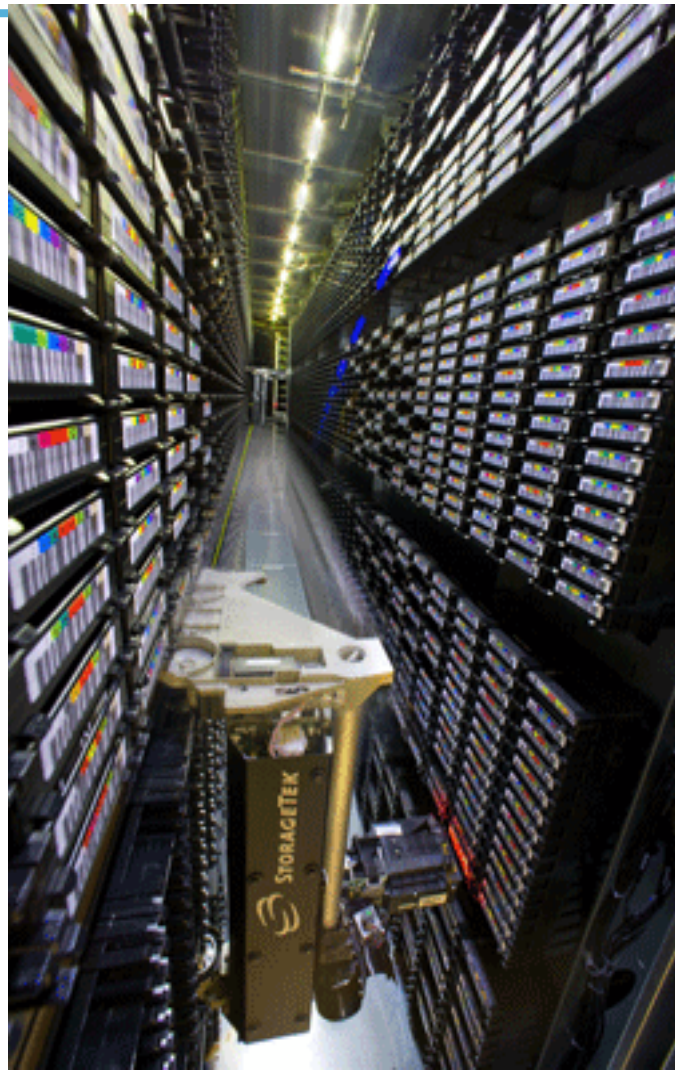
Archival Storage (HPSS)

- For permanent, archival storage
- Permanent storage is magnetic tape, disk cache is transient
  - 100PB data in >400M files written to 32k cartridges
  - Cartridges are loaded/unloaded into tape drives by sophisticated library robotics
- Front-ending the tape subsystem is 150TB fast-access disk

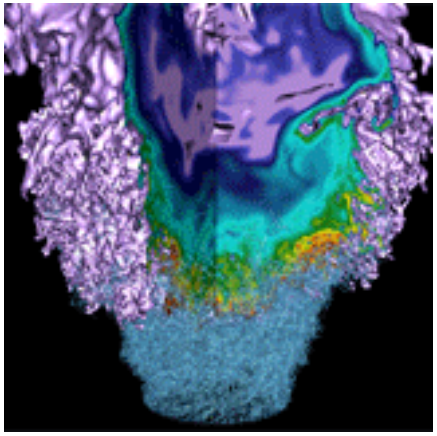
- **Hostname: archive.nersc.gov**
- **Over 100 Petabyes of data stored**
- **Data increasing by 1.7X per year**
- **150 TB disk cache**
- **8 STK robots**
- **44,000 tape slots**
- **Average data xfer rate: 100 MB/sec**

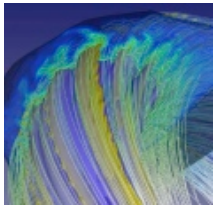
HPSS Clients

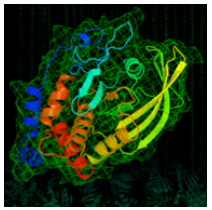
- **Parallel, threaded, high performance:**
  - HSI
    - Unix shell-like interface
  - HTAR
    - Like Unix tar, for aggregation of small files
  - PFTP
    - Parallel FTP
- **Non-parallel:**
  - FTP
    - Ubiquitous, many free scripting utilities
- **GridFTP interface (garchive)**
  - Connect to other grid-enabled storage systems

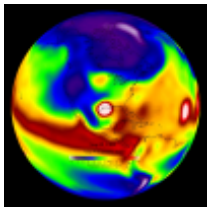


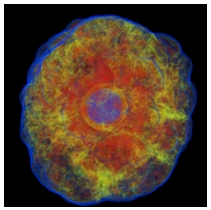
Running and Monitoring Jobs

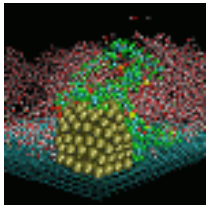












Types of Jobs on Genepool

- **Batch – Scheduled**
  - qsub
- **Interactive – Scheduled**
  - qlogin
- **Interactive – Unscheduled**
  - 2 login nodes, 17 gpint interactive nodes
  - direct login via ssh
- **Services – Unscheduled**
  - Web services
  - Database Services
  - Automated job submissions

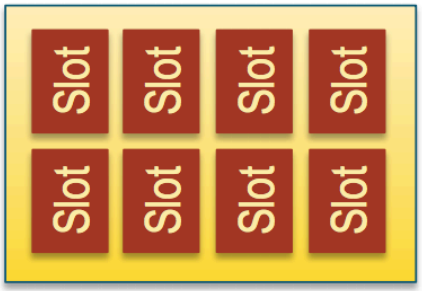
Basics of Batch Jobs

- Genepool is a shared resource
- Each calculation usually only takes a small portion of genepool
  - Every job is strictly limited on the consumption of genepool resources
  - The job description specifies the resource limits
- Univa GridEngine is used to schedule each calculation on genepool
  - The scheduler matches job resource limit requests with physical resources

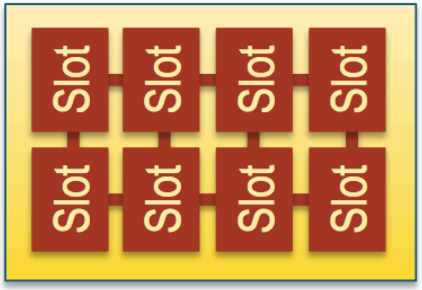
Basics of GridEngine

- **GridEngine schedules "slots"**
  - Not memory, nor processors, nor nodes
- **A *slot* is a portion of a node**
  - For most nodes on genepool, a slot is defined as a single processor plus ($ram.c_{nodeTotal}/n_{cores}$) memory
  - Some nodes are *exclusively scheduled* – all slots on the node are bonded together as one schedulable unit
- **Jobs are placed in *queues***
  - Queues manage the resources of disparate sets of nodes, and have distinct resource limits
    - normal.q has a 12 hour time limit
    - long.q has a 10 day time limit
- **Jobs are scheduled in order of a balance of:**
  - Resource availability
  - Job prioritization

Node

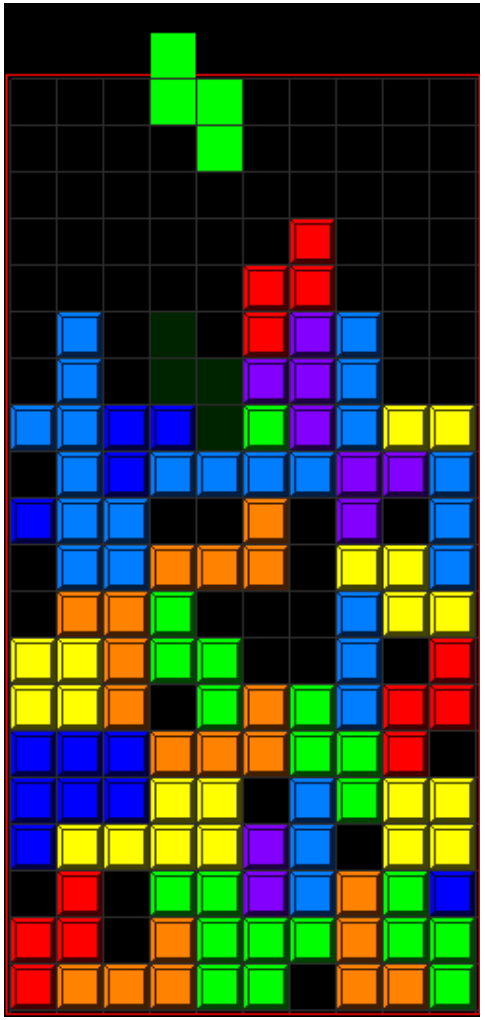


Exclusive Node



What is a JSV?

- Job Schedule Verifier
- Upon job submission, evaluates the requirements you provided (if you did), and sends the job to the right queue
- Occasionally adjusted by CSG staff at NERSC to accommodate changes in Genepool configuration



Basics of Batch Job Submission

Example Batch Script

```
#!/bin/bash
module load blast+
input=$1
database=$2
blastn -query $input -db $database <more options>
```

Submitting the example

```
genepool$ qsub -cwd example.sh queries.fa myDB
Your job 347283 ("example.sh") has been submitted.
```

- "qsub" submits the job for batch processing
- "-cwd" directs the job to work out of the present location in the filesystem
  - the current working directory
- Default resource limits will be applied, since none were specified
  - **1 slot**
  - **5.25GB memory/slot**
  - **12 hours**

Many examples on the NERSC webpage

https://www.nersc.gov/users/computational-systems/genepool/running-jobs/submitting-jobs/

qsub commands and options

UGE (Univa Grid Engine) is the batch system used for Genepool/Phoebe.

Action	How to do it	Comment
Submit a job	qsub *script*	In UGE you need to submit a script, not an executable.
Specify number of processors for a threaded job	qsub -pe pe_slots 8 ...	Request 8 cores on a single node for your job. Please specify as many processors as will be needed during your job.
Specify number of nodes and processors for an MPI job	qsub -pe pe_8 16 ...	Request 2 nodes with 8 processors per node. pe_1, pe_2, pe_4, pe_8, pe_16, and pe_32 are available.
Specify memory required per processor	qsub -l ram.c=4G ...	Specify how much memory is required *per processor* for your job. At present this is implemented by implicity setting h_vmem (a virtual memory limit), so you will need to account for all virtual memory needed by your application. Use of a program like memtime during your benchmarking ahead of production may be informative.
Specify a time limit for your job	qsub -l h_rt=6:00:00 ...	Specifies that your job will run for at most 6 hours. Default is 12 hours. If you request more than 12 hours, your job will enter the long queue, which has much fewer dedicated resources.
Submit a job to the high priority queue	qsub -l high.c *script*	The high.c complex is for small fast turn around jobs
Submit a job that depends on other jobs	qsub -hold_jid [job_ID|job_name] *script*	UGE just recognizes whether or not [job_ID|job_name] is finished before submitting your job. The newly submitted job will only start once all jobs in the hold_jid list are completed.
Submit a job to different project	qsub -P [project] *script*	By default your job runs as the project corresponding to your primary NERSC project repo. If qsub indicates you do not have access to the project you specify please file a ticket to get added to it.
Get e-mail from your job upon completion	qsub -m e -M <email address> ...	No email by default. UGE can also email at the beginning of a job with "-m b", or upon errors with "-m a".

Genepool Queues

Exclusive = all CPUs on a node

Queue Name	Purpose	User Requestable	Slot Limit	Wall Clock Limit
normal.q	Default queue	No	2460	12 hours
normal_excl.q		No	3348	12 hours
long.q	Workflows that need more than 12 hours	No	320	240 hours
long_excl.q		No	1548	240 hours
high.q	High priority jobs and debugging jobs	Yes	32	240 hours
high_exclusive.q		Yes	64	72 hours
xfer.q	Data Transfer Queue on genepool; Use this to transfer data to /global/dna	Yes, request "-l xfer.c".	2	72 hours

The number of nodes devoted to each queue is highly variable at the moment as we examine queue wait times

Pointers to avoid common mistakes

- **For new users, trust the JSV**
  - The JSV will do its best to place your job where it will run best given your specifications.

- **It helps to know 3 things:**
  - Max runtime: -l h_rt=6:00:00 run for up to 6 hours
  - Number of CPUs -pe pe_slots 8 give me 8 slots
  - Memory -l ram.c=120G I need 120G memory

- **Problems come in when memory and CPUs get combined.**
  - ram.c specifies memory PER CPU
    - so: qsub -pe pe_slots 8 –l ram.c=120G would only run on 1TB nodes

More pointers

- **Be aware of how many threads your software will use, and be sure you've requested the right number (typically with "qsub –pe pe_slots 8")**
  - hmmer is commonly problematic – by default will try to take all CPUs on a machine, unless otherwise specified
- **If at all possible, use 12 hours or less**
  - The long queue has few nodes, and usage is constrained
- **Use –cwd or –wd <directory> with qsub**
  - Writing output to your home directory can slow everyone down. Write to scratch!
- **For exclusive nodes:**
  - qsub –l exclusive.c

Check job status with qs (cached qstat)

```
dmj@phoebe:~$ qstat
job-ID  prior   name       user         state submit/start at     queue                          jclass                         slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------------------------------------
 336024 0.44577 testJob_1  dmj          r     02/11/2013 19:30:03 normal.q@sgi07a26.nersc.gov                                        1
 336025 0.39718 testJob_2  dmj          r     02/11/2013 19:30:03 normal.q@sgi07b08.nersc.gov                                        1
 336026 0.37289 testJob_3  dmj          r     02/11/2013 19:30:03 normal.q@sgi07b13.nersc.gov                                        1
 336027 0.00000 env        dmj          qw    02/11/2013 19:30:08                                                                    1
dmj@phoebe:~$
```

- **By default, qstat only shows *your* jobs**
- **To see others, qstat -u <username> or qstat -u ***
- **State:**
  - r: "running"
  - qw: "queue-wait"
  - R<state>: "rescheduled <basic state>"
  - E<state>: "error <basic state>"
  - h<state>: "hold <basic state>"

Why hasn't my job run yet?

sh /genepool/nsg/opt/scripts/daily_summary/show_summary.sh

```
MPP(SlotHours) By queue
##############################################################################
                                                              0  xfer.q
                                                            524  high.q
                                                            837  long.q
                                                           1661  high_excl.q
                                                          11741  long_excl.q
                                                          21115  normal.q
                                                          35241  normal_excl.q

                          Jobs      SlotHours Wall(hr)          Wait(hr)  Efficien. MaxVmem
                           #        (hr)      Mean, Med         Mean, Med Tcpu/MPP  Mean(GB)
high.q                    718       524.25      0.04, 0.00       0.24, 0.05    0.11      3.38
high_excl.q               30        1661.36     1.73, 0.23       4.94, 0.59    0.21      27.73
long_excl.q               226       11741.67    2.85, 0.85      21.58, 15.83   0.25      59.39
long.q                    4762      837.19      0.18, 0.01       0.04, 0.01    0.85      0.64
normal_excl.q             1275      35241.62    1.15, 0.43       5.30, 0.40    0.13      33.51
xfer.q                    67        0.98        0.01, 0.01       0.04, 0.01    0.80      2.09
normal.q                  132050    21115.89    0.13, 0.01       2.95, 0.01    0.69      1.24
----------------------------
ALL                       139128.0  71122.95    0.15, 0.01       2.89, 0.01    0.33      1.62
```

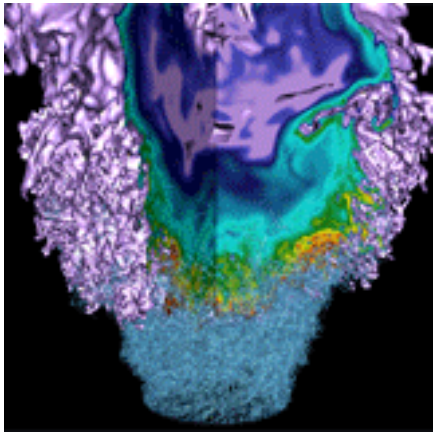
Investigating Completed Jobs

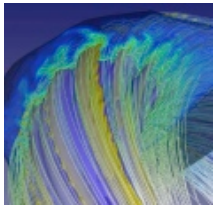
- **GridEngine saves accounting information for all completed and errored-out jobs**
- **These records reflect what your project has been billed for fair-share calculations**
- **Also show the total resource utilization figures**
  - Can be useful (but not perfect) when trying to understand why a job crashed

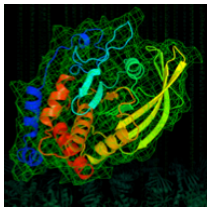
Investigating Completed Jobs

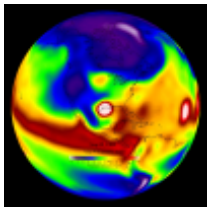
- **Check your jobs for the past 90 days:**
  - qqacct -D 90 -q 'user=="dmj"'
- **Just the jobs UGE thinks failed over past 3 days (default)**
  - qqacct -q 'user=="dmj" && failed != 0'
- **Just the jobs UGE thinks failed with time/memory info**
  - qqacct -q 'user=="dmj" && failed !=0' -c 'job_number,failed,memory(ppn*h_vmem),memory(maxvmem), h_rt,wall'
- **Always put query in single quotes – the shell is likely to try to parse many of the characters in the query**
- **"-c" overrides default output columns**

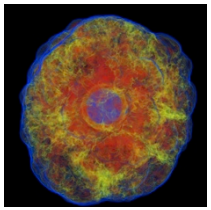
Modules

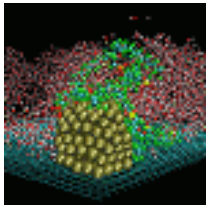












Using Software and the UNIX Environment

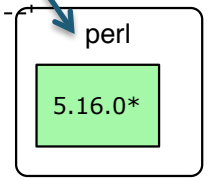
- **Providing large-scale installations of software for many different users on an HPC system presents a number of challenges:**
  - Different users need different software, use different shells
  - Some users need different specific versions, including older versions
  - All users need to access the software quickly and easily from "everywhere" [network-mounted, non-standard paths]
  - Providing a user interface for accessing that software can be challenging
    - Example: How would you use software installed in

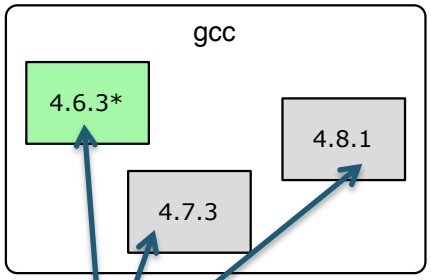
      /usr/common/jgi/aligners/blast+/2.2.28
    - Answer:
      - Add /usr/common/jgi/aligners/blast+/2.2.28/bin to PATH;
      - csh: setenv PATH /usr/common/jgi/aligners/blast+/2.2.28/bin:$PATH
      - bash: export PATH=/usr/common/jgi/aligners/blast+/2.2.28/bin:$PATH

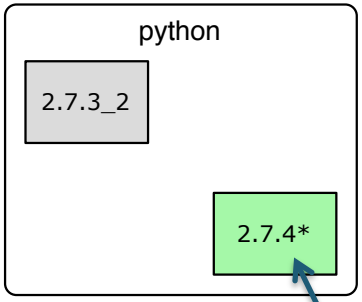
What are Modules?

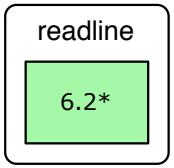
Modules have a name

A “module” is something that can be loaded or unloaded dynamically into the environment.









Modules have a version
can have *many versions*

Modules can have a *default* version

To refer to the *default version* of a module, use: <name>
e.g. module load gcc
To refer to a *specific version* of a module, use: <name>/<version>
e.g. module load gcc/4.8.1

Basic Modules Functionality

- **Modules manipulate the environment**
  - Loading can:
    - Set an environment variable (possibly by replacing)
    - Append (or prepend) to a compound environment variable
    - Unset an environment variable
    - *can* execute a command (not recommended if the command changes the state of the system)
  - 'module unload' reverses the effects of the 'module load'
  - Which effects of a module might be irreversible?
    - Answer:
      - setenv won't restore the environment to its original state
      - multiple modules calling 'setenv' or 'unsetenv' on the same variable might lead to an inconsistent state (those modules should conflict)
      - Executing system calls which change system state (e.g. xhost) are not trivially reversible by unloading the module

Modules: conflicting and swapping

- **Some modules are incompatible**
  - E.g. both wublast and blast+ provide different blastn, blastx, etc. executables
  - To prevent these modules from being simultaneously loaded, they conflict

```
dmj@genepool02:~$ module load wublast
dmj@genepool02:~$ module load blast+
blast+/2.2.26(25):ERROR:150: Module 'blast+/2.2.26' conflicts with the currently
loaded module(s) 'wublast/20060510'
```

- **Most of the time, only a single version of a module should be loaded at a time:**
  - e.g., doesn't make sense to load more than one version of gcc
  - Try:

```
module purge            ## cleans everything out
module load gcc
Module load gcc/4.8.1
```

  - Error? to change from gcc/4.6.3 (the default) to gcc/4.8.1 (the latest), swap!

```
module swap gcc gcc/4.8.1
```
-or-
```
module swap gcc/4.8.1
```

Common Environment Variables in Modules

- **Modules for software packages commonly set:**
  - PATH
  - LD_LIBRARY_PATH
  - PYTHONPATH
  - PERL5DIR

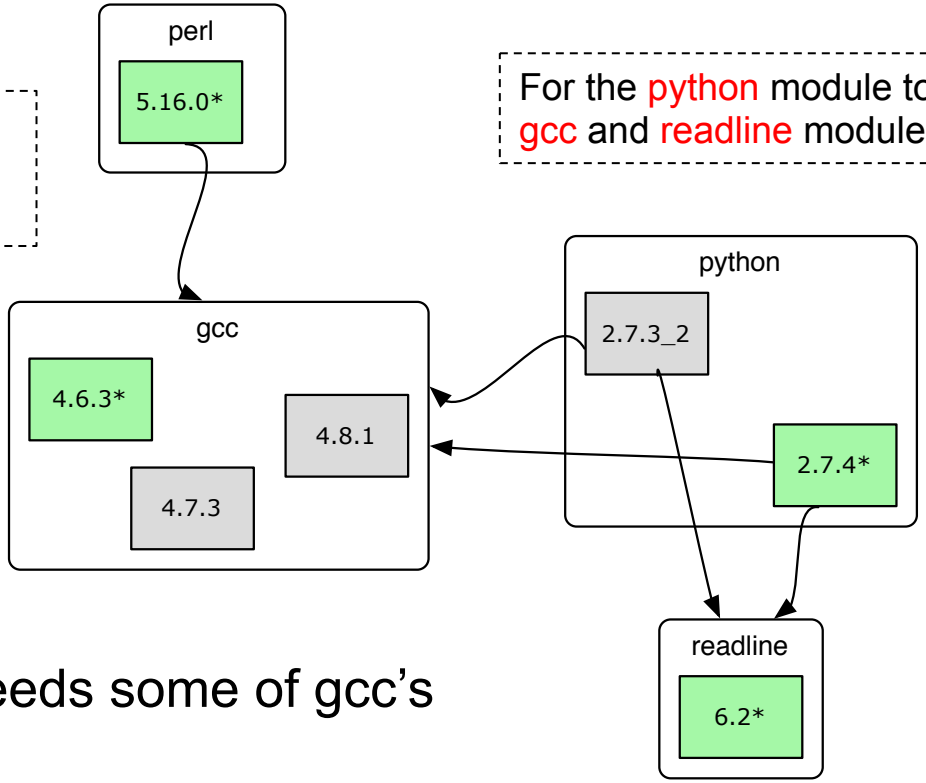
Be VERY careful about manipulating these environment variables!!!

- **Every usg/jgi module for software also sets an environment variable pointing to the base of the distribution:**
  - E.g. BOOST_ROOT, PERL_DIR, PYTHON_DIR, GIT_PATH
- **Exercise:**
  - Load the python module first
  - Use 'module info' to investigate the effects of:
    - graphviz
    - RSeQC
    - Smrtanalysis
  - Are there commonalities? Differences?

Modules may have dependencies

For the perl module to function, the gcc module needs to be loaded

For the python module to function, *both* the gcc and readline modules need to be loaded



- Python needs some of gcc's libraries
- Perl needs some of gcc's libraries
- Python also needs readline's libraries

Module commands reference

- **module list**
  - show all loaded modules
- **module avail <module name>**
  - list modules with <module name> that can be loaded
- **module load <module name>**
- **module unload**
- **module swap <current module> <new module>**
  - unload a loaded module and load the new one
- **module purge**
  - unload all modules (it's a good idea to start a batch script this way!)
- **module use <a directory>**
  - Use a different $MODULEPATH

For Genepool-wide installation of new modules, or software upgrades, contact your consultants!

Using Modules in Batch Scripts

```
#!/bin/bash –l
#$ -l ram.c=10G
#$ -l h_rt=8:00:00

set –e

module purge
module load PrgEnv-gnu/4.6
module load python/2.7.4

module use /path/to/my/groups/modulefiles
module load MyPipeline/1.0

#…. Run your programs here ….
```

Ensures login environment is initialized

UGE options

Kill script if any commands give non-zero exit status

Clear all the modules, load any needed variant-provider modules

Add your modulefiles to MODULEPATH (module use)
Load your pipeline module

What's coming up next for Genepool?

- **OS Upgrade this year**
  - Deb6 -> CentOS7?
  - Expecting lots of software re-installation
- **Move scheduler from UGE to SLURM (by Aug 2017)**
  - Standardization with other NERSC systems
- **Move more computation to Cori/Edison/NERSC9**
  - Large Cray systems designed around heavy parallelization
  - Use of Docker/Shifter containers
  - Adaptation of pipelines to workflow managers
    - Taskfarmer, NextFlow, FireWorks

NERSC

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB