Overview

How we can help
System overview
  Tech specs
  Signing on
Batch submission
Software environment
Interactive jobs
Next steps
We are here to help

We run a wide variety of systems
We can connect you to resources here and nationally
Today I’m going to focus on sherlock
Why Sherlock?

Sherlock is a shared platform for research at Stanford

A single node of Sherlock is more powerful than your laptop or desktop

Sherlock consists of many computers (nodes) which you can run your work on
System Overview

- **ln01**
- **ln02**
- Login nodes
- Interconnect
- Filesystems
- Compute nodes
- Generic nodes
- Specialized nodes

You
Login Nodes

Currently two machines
use: sherlock.stanford.edu

Login nodes = gateway to all other resources
Compute Nodes

Normal nodes
- 16 cores in two sockets
- 64GB RAM
- 100GB Solid State Disk
- Infiniband connection 56Gbps (2:1 oversubscribed)

GPU nodes (NVIDIA Kepler)

Big memory nodes (1.5TB RAM)
Storage

/home: $HOME (15GB), $PI_HOME (1TB)
small amount of space, backed up, small file friendly, accessible across all systems

/scratch: $SCRATCH, $PI_SCRATCH
large, high performance, not backed up, big file friendly, accessible across all systems

/local_scratch: $LOCAL_SCRATCH
SSD for duration of the job, local to a node
Storage Specs

/home
NFS
IOPS = ~16,200
Bandwidth = 125 MB/s based on connection

/scratch
Lustre
IOPS =~ 54,000
Bandwidth = 6.5 GB/s based on connection
total ~20 GB/s read; 13 GB/s write

/local_scratch
SSD
IOPS=100,000
Bandwidth = 500 MB/s read; 400 MB/s write
Sign On

Setup Kerberos
documented here:

Logon
2 steps: kinit then ssh
http://sherlock.stanford.edu/mediawiki/index.php/LogonCluster
Scheduling Jobs

Basic concept - tell the scheduler:
1. what you need
2. what it should do

Partitions
- sets of physical machines with comparable hardware

QOS
- sets of limitations (runtime, CPUs per user...)

No need to specify QOS or partition in most cases
A bit on scheduling

Sherlock currently uses fairshare scheduling
balances out usage across users and PIs

Uses backfill scheduling
short jobs can run before larger pending jobs

Accurate requirements help the scheduler
and you! (shorter wait times)

See different partitions/QOS at http://sherlock.stanford.edu
Condo model

General partition
  everybody can run there

Owners’ partitions
  owners have immediate access to their own nodes
  they can run on other owners’ nodes, but could be preempted by the rightful owner’s jobs
Slurm tools

Run
- `salloc` reserve nodes
- `srun` run a scheduled command
- `sbatch` submit a batch script
- `scancel` cancel a job

Check
- `squeue` check job status and queue
- `sstat/sacct` check job stats while/after completion
- `sprio` view job priority factors
- `sshare` view shares (fairshare scheduling)
#!/bin/bash
#SBATCH --job-name=R-hemisphere
#SBATCH --output=hemisphere.out
#SBATCH --error=hemisphere.err
#SBATCH --mail-type=ALL
#SBATCH --mail-user=<sunetid>@stanford.edu
#SBATCH --time=20:00
#SBATCH --nodes=1
#SBATCH --mem=4000
#SBATCH --ntasks-per-node=1

module load R

srun R --no-save < hemisphere.R
## Environment Modules

Allows to manage software versions

- `module avail` shows what is available
- `module load` loads a package

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
<th>Package</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>R/3.0.2</td>
<td></td>
<td>freesurfer/5.3.0</td>
<td>(D)</td>
</tr>
<tr>
<td>allinea/4.2</td>
<td>fsl/5.0.6-test</td>
<td>julia/gcc/0.3.0-pre</td>
<td></td>
</tr>
<tr>
<td>ansys/icemcfdf/15.0</td>
<td>fsl/5.0.6</td>
<td>matlab/R2012b</td>
<td></td>
</tr>
<tr>
<td>comsol/43b</td>
<td>gcc/4.8.1</td>
<td>matlab/R2013b</td>
<td></td>
</tr>
<tr>
<td>comsol/44</td>
<td>gcta/1.24.2</td>
<td>matlab/R2014a</td>
<td>(D)</td>
</tr>
<tr>
<td>cuda/5.5</td>
<td>intel/13sp1up1</td>
<td>nag/m24</td>
<td></td>
</tr>
<tr>
<td>freesurfer/5.3.0-test</td>
<td>intel/13sp1.2.144</td>
<td>openmpi/1.6.5/intel13sp1up1</td>
<td>(D)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>python/3.3.2</td>
<td>(D)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>openmpi/1.6.5/intel13sp1up1</td>
<td>(D)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>python/2.7.5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>tabix/0.2.6</td>
<td></td>
</tr>
</tbody>
</table>
Interactive Jobs for starting

Any resource can be interactive

A custom tool (sdev) makes interactive jobs easier
The dev partition allows jobs to start reasonably quickly
You might not be able to *run* for that long

To get a quick debug shell

```sh
$ sdev
```

To run interactively (not just for debugging)

```sh
$ srun --pty bash
```
Graphical interfaces

Example with Matlab

sherlock-ln01$ srun -p dev --qos dev --pty --x11 --time=1:0:0 --mem=8GB bash
sh-5-2$ ml load matlab/R2014a
sh-5-2$ matlab
Next Steps

Request an account - email research-computing-support@stanford.edu, cc your PI

Get started, try submitting some jobs
interactive jobs, batch scripts

Documentation (WiP)
http://sherlock.stanford.edu