Genome Scale Computing
How Do I Use the BioCluster?

• Command-line interface

• PLEASE ask questions!
Traditional Command-line Interface

```bash
ssh -p 734 stahl ircf missouri edu
```

`ssh` allows very few management options for running jobs
Unmanaged Connections

“Node 45” can quickly become overloaded
Distributed Resource Management Software

The BioCluster uses openlava software

http://www.openlava.org/home.html
Openlava Queuing System

stahl

User1
User2
User3
User4

openlava

1
2
3 + 4

node45
node48
node56
node65
node12
BioCluster Information

http://ircf.rnet.missouri.edu:8000/comp_hardware/biocluster

You must have an account to view page.

http://ircf.rnet.missouri.edu:8000/.register
Manage Your Jobs!

- CPU load
- RAM
- Disk space

```plaintext
  top
df
```
The top Command

top - 09:32:42 up 20 days, 22:22, 0 users, load average: 0.85, 0.43, 0.18
Tasks: 466 total, 2 running, 464 sleeping, 0 stopped, 0 zombie
Cpu0 : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu1 : 0.0%us, 0.3%sy, 0.0%ni, 96.3%id, 3.0%wa, 0.0%hi, 0.3%si, 0.0%st
Cpu2 : 95.0%us, 5.0%sy, 0.0%ni, 0.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu3 : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu4 : 0.0%us, 0.3%sy, 0.0%ni, 98.7%id, 1.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu5 : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu6 : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu7 : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu8 : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu9 : 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu10: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu11: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu12: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu13: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu14: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu15: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu16: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu17: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu18: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu19: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu20: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu21: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu22: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Cpu23: 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st

Mem: 528462176k total, 279308956k used, 249153220k free, 6999932k buffers
Swap: 57475064k total, 43520k used, 57431544k free, 270895416k cached

- 24 processors
- 512 GB RAM
- Pay attention to:
  - %CPU
  - %MEM
Use lstop to Run top on a Node

[03/19/13 11:47:14] stahl temp/$ bjobs
JOBID  USER   STAT  QUEUE   FROM_HOST  EXEC_HOST  JOB_NAME  SUBMIT_TIME
3260  sgivan  RUN   normal  stahl       compute-0-2  TEST      May 8 11:47

[03/19/13 11:47:24] stahl temp/$ lstop compute-0-2

Top - 11:47:26 up 61 days, 17:48, 0 users, load average: 1.00, 1.00, 1.00
Tasks: 517 total, 2 running, 515 sleeping, 0 stopped, 0 zombie
Cpu(s): 2.2%us, 0.1%sy, 4.0%ni, 93.6%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 1058953460k total, 71093444k used, 987860016k free, 324860k buffers
Swap: 65535992k total, 0k used, 65535992k free, 57078536k cached

PID USER   PR  NI VIRT  RES  SHR  S %CPU %MEM    TIME+ COMMAND
31521 sgivan 20 0 3117m 2.9g 944 R 99.6  0.3   0:10.15 test
13027 sgivan 20 0 15296 1444 832 R 2.0  0.0   0:00.01 top
### df -kh

```bash
[03/19/13 11:52:22] stahl temp/$ df -kh
```

<table>
<thead>
<tr>
<th>Filesystem</th>
<th>Size</th>
<th>Used</th>
<th>Avail</th>
<th>Use%</th>
<th>Mounted on</th>
</tr>
</thead>
<tbody>
<tr>
<td>/dev/sda3</td>
<td>16G</td>
<td>5.6G</td>
<td>9.1G</td>
<td>39%</td>
<td>/</td>
</tr>
<tr>
<td>tmpfs</td>
<td>16G</td>
<td>52K</td>
<td>16G</td>
<td>1%</td>
<td>/dev/shm</td>
</tr>
<tr>
<td>/dev/sda1</td>
<td>194M</td>
<td>97M</td>
<td>88M</td>
<td>53%</td>
<td>/boot</td>
</tr>
<tr>
<td>/dev/sda6</td>
<td>219G</td>
<td>8.1G</td>
<td>200G</td>
<td>4%</td>
<td>/export</td>
</tr>
<tr>
<td>/dev/sda5</td>
<td>3.9G</td>
<td>1.2G</td>
<td>2.5G</td>
<td>33%</td>
<td>/var</td>
</tr>
<tr>
<td>tmpfs</td>
<td>7.7G</td>
<td>214M</td>
<td>7.5G</td>
<td>3%</td>
<td>/var/lib/ganglia</td>
</tr>
<tr>
<td>nas-0-0:/export/home1/sgivan</td>
<td>3.6T</td>
<td>6.5G</td>
<td>3.4T</td>
<td>1%</td>
<td>/home/sgivan</td>
</tr>
<tr>
<td>nas-0-0:/export/ircf</td>
<td>1.8T</td>
<td>323G</td>
<td>1.4T</td>
<td>19%</td>
<td>/share/ircf</td>
</tr>
<tr>
<td>nas-0-0:/export/data1</td>
<td>30T</td>
<td>1.1T</td>
<td>29T</td>
<td>4%</td>
<td>/share/data</td>
</tr>
</tbody>
</table>
```

- **local and NFS file systems**
  - `/home/sgivan`
  - `~/data directory`
  - `/share/data/sgivan`

- **Pay attention to:**
  - `Avail`
  - `Use%`
Genome–Scale Mindset

• Repetitively apply algorithm to thousands of sequences
  – This potentially can ...
    • Generate thousands/millions of files
    • Rapidly fill disk space
    • Spawn thousands of processes
test, test, test!

• 10 – 100 – 1000
  – If you are testing a new algorithm, make sure it works on 10 sequences before testing it on 100 …
  – Test on 100 before testing on 1000 …

• This especially applies to large datasets!
Request/Reserve Resources

- particularly for **RAM** and **# of processors**

```bash
bsub -n 10 -q normal -R "rusage[mem=15] span[hosts=1]" cmd
```

this requests 10 processors and 15MB of RAM in the ‘normal’ queue
Ways to Create Test Files

- **head** – output lines from top of file
  - `head -n 10 input.fa > 10.fa`
  - outputs first 10 lines of file `input.fa` into `10.fa`
  - `head -n 100 input.fa > 100.fa`

- **tail** – output lines from bottom of file
  - `tail -n 100 input.fa > 100.fa`

- **sed** – print range of lines
  - `sed -n '10,1009p' input.fa > 1000.fa`
  - output lines 10 to 1009 of `input.fa` to `1000.fa`
Get Job Stats from openlava

[sgivan@lewis sgivan]$ bjobs -l

Job <467331>, User <sgivan>, Project <default>, Status <RUN>, Queue <bioq>, Interactive pseudo-terminal mode, Command <tcsh>
Tue Feb 15 09:26:13: Submitted from host <lewis>, CWD <$HOME/bin>;
Tue Feb 15 09:26:15: Started on <d18-29>;
Tue Feb 15 11:35:18: Resource usage collected.
The CPU time used is 47 seconds.

MEM: 6 Mbytes; SWAP: 156 Mbytes; NTHREAD: 5
PGID: 22982; PIDs: 22982
PGID: 22984; PIDs: 22984
PGID: 22986; PIDs: 22986
PGID: 23433; PIDs: 23433

SCHEDULING PARAMETERS:

<table>
<thead>
<tr>
<th>loadSched</th>
<th>loadStop</th>
</tr>
</thead>
<tbody>
<tr>
<td>r15s</td>
<td>r1m</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>
Useful File Commands*

- `ls -lh` displays file sizes in human-readable form
- `ls -lht` displays newest files at bottom of list
- `ls -l | wc -l` displays number of files in directory
- `grep -c '>' file.fasta` counts number of sequence in FASTA file

* For the genome scientist