An Introduction to High Performance Computing

What is High Performance Computing?

- Using more than your laptop or desktop (or high powered workstation)
- Designed with various hardware architectures that fit different problems
- Allows you to run analyses not possible with a typical computer
- High memory, high core count, bigdata

Before we launch in to ISU-HPC...

- Are you tired of typing:

```
$ ssh <netID>@hpc-class.its.iastate.edu
```

????

- `cd` into your root directory and try: `ls -a`
- You should see a hidden folder named `.ssh`
- `cd` into `.ssh` and create a file named "config":

```
$ touch config
```

Before we launch in to ISU-HPC...

- Now open the file `config`, add the following contents, and save the file:

```
Host hpc-class
HostName hpc-class.its.iastate.edu
User <net ID>
```

- Now try it out!

```
$ ssh hpc-class
```

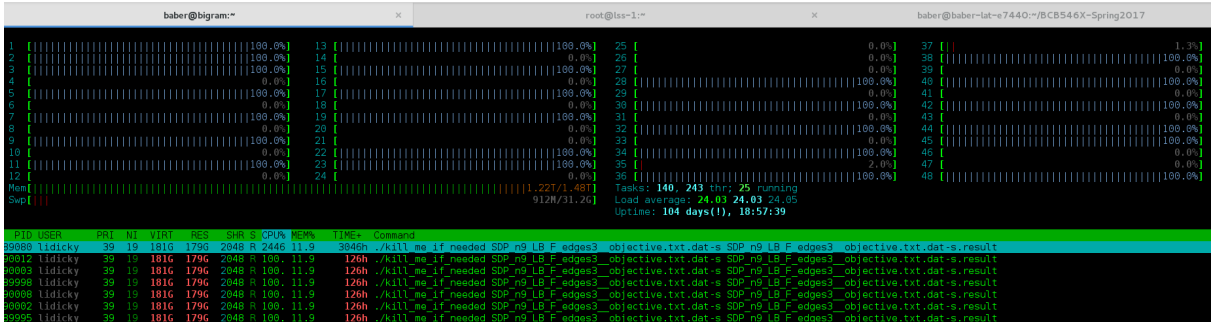
- And once you're in hpc-class, a few new tricks:

```
$ hostname

$ whoami
```

Performance Monitoring

- htop (or top)
- iostat
- iftop



There are other tools, but these will get you the basic info you need and are installed on most systems by default

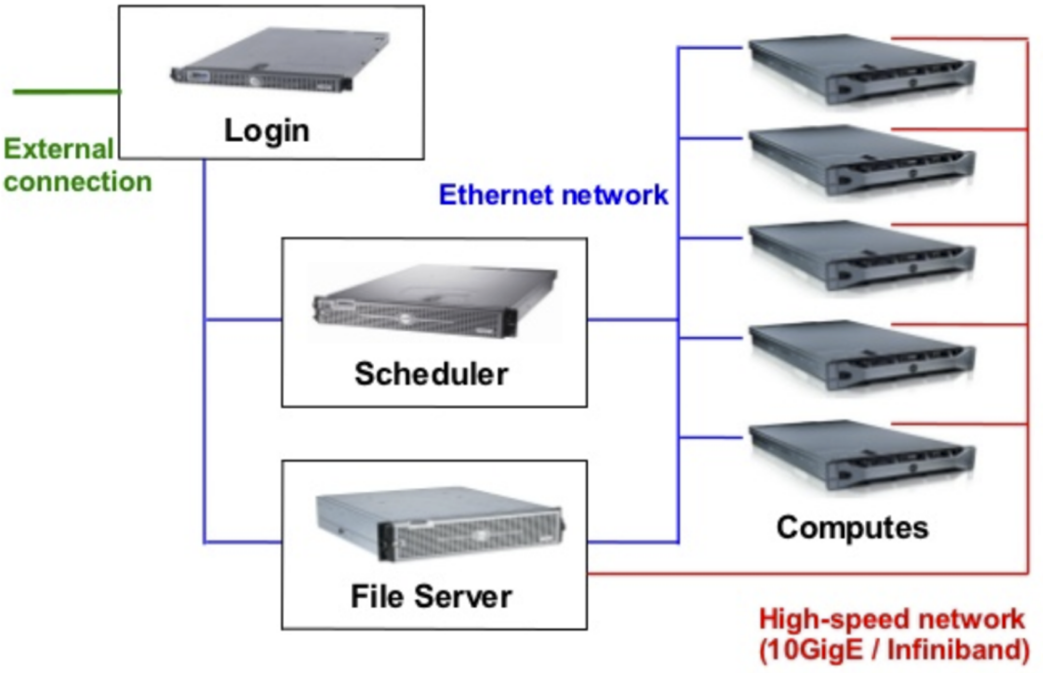
Terminology

- HPC terminology can be confusing, sometimes the same words have different contextual meaning (e.g. threads)

Terms

- Nodes: compute node, head node
- Processors (a cpu chip)
- Cores (physical cpus embedded on a single chip)
- Threads: hyper-threading (software 'trick' to let two processes share a core)
- Scheduler: Allocates access to resources to users (ISU uses Slurm)

Cluster Diagram



Compute Resources at ISU

Clusters

- hpc-class
  - For classes, not research
  - 48 nodes
    - 2.0GHz, 16 cores, 64GB RAM
- condo2017
  - Primarily for sponsored research
  - 192 Nodes
    - 2.6GHz, 16 cores, 128GB RAM
    - 3 High-memory nodes (1 and 2TB RAM)
  - Free-tier - 88 nodes
    - 2.0GHz, 16 cores, 128GB RAM
    - 1 High-memory node (1TB RAM)

Compute Resources at ISU

Clusters

- nova
  - Primarily for sponsored research
  - 100 Nodes
    - Fast Skylake processors, 36 cores, 192-384GB RAM
    - 1 High-memory node (3TB RAM)

Also check out these UNIX tutorials through ISU-HPC!

Compute Resources at ISU

Custom hardware

- BioCrunch (for multithreaded shared memory programs)
  - 2.4GHz, 80 threads, 768GB of RAM
- BigRAM (for large memory needs like de-novo assembly)
  - 2.6GHz, 48 threads, 1.5TB of RAM
- Speedy (for single threaded programs, like R)
  - 3.4GHz, 24 threads, 256GB of RAM
- Speedy2 (for single threaded programs, like R)
  - 3.2GHz, 32 threads, 256GB of RAM
- LASWIN01 (for Windows only software)
  - 2.6GHz, 24 threads, 64GB of RAM
- Legion (for massively parallel applications)
  - 4 nodes
  - 1.3GHz, 272 threads, 386GB of RAM (each)

Compute Resources

Xsede

- ISU researchers have access to the national supercomputer centers (e.g. TACC, PSC) via Xsede
- For problems that need to scale larger than our on-campus resources
- Contact campus champions: Jim Coyle or Andrew Severin

Cloud (AWS, Azure, etc.)

- Tempting introductory rates
- Be cautious of putting in a credit card (charges can accumulate quickly)
- Consider data transfer times and speed
- Consult with IT before purchasing - they have special negotiated rates

Software

Modules:

- Modules are used to allow multiple people to use the same software with consistent, reproducible results
- Modules keep your environment clean and free of conflicts (e.g. python2/3 or Java7/8)
- Think of modules as a software library. You need to check out the software before you can use it.
- Modules can be searched:

```
$ module avail
-------------------- /opt/rit/modules ---------------------------------
abyss/1.5.2              freesurfer/5.3.0         lib/htslib/1.2.1
abyss/1.9.0              gapcloser/1.12-r6        lib/htslib/1.3
afni/17.0.10             gapfiller/1-10           lib/htslib/1.3.2
albert/20170105          gatk/3.4-46              lib/ICE/1.0.9
...
```

Software

Modules

- To use a module:

```
$ module load <name of software>
```

- To delete a module:

```
$ module unload <name of software>
```

- Default behavior is to load the latest version if not specified
- Use 'module purge' to clear your environment before loading something different

Storage

Code

- Git (github, bitbucket, gitlab, etc.)

Datasets

- Storage on the clusters and servers should be treated as temporary
- Become familiar with the disk architecture on each machine
- Keep backups

Job Scheduler

- Scheduler assigns jobs from the queue to the compute nodes
- ISU uses SLURM
- Think about the scheduler like a hotel reservation - you're charged for the room whether you use it or not, and if you ask for an especially long or large reservation, the room may not be available when you want it.
- Basic info required: how many nodes, how long
- Script writer can get you started

Slurm Script Cheatsheat:

Lines starting with `#SBATCH` are for `SLURM` resource manager to request resources for HPC. Some important options are as follows:

Option	Examples	Description
`--nodes`	`#SBATCH --nodes=1`	Number of nodes
`--cpus-per-task`	`#SBATCH --cpus-per-task=16`	Number of CPUs per node
`--time`	`#SBATCH --time=HH:MM:SS`	Total time requested for your job
`--output`	`#SBATCH -output filename`	STDOUT to a file
`--error`	`#SBATCH --error filename`	STDERR to a file
`--mail-user`	`#SBATCH --mail-user user@domain.edu`	Email address to send notifications

Interactive session

To start a interactive session execute the following:

```
#this command will give 1 Node for a time of 4 hours

srun –N 1 –t 4:00:00 --pty /bin/bash
```

Sample SLURM script (mysbatch.sh)

```
#!/bin/bash
#SBATCH --time=1:00:00   # walltime
#SBATCH --nodes=2   # number of nodes in this job
#SBATCH --ntasks-per-node=16   # total number of processor cores in this job
#SBATCH --output=myout_%J.log
#SBATCH --error=myerr_%J.err
module load R
Rscript MyThing.R
```

Then submit:

```
$ sbatch mysbatch.sh
```

Slurm Job Management Cheatsheat:

Quick reference sheet for SLURM resource manager

Job scheduling commands

Commands	Function	Basic Usage	Example
`sbatch`	submit a slurm job	sbatch [script]	$ sbatch job.sub
`scancel`	delete slurm batch job	scancel [job_id]	$ scancel 123456
`scontrol hold`	hold slurm batch jobs	scontrol hold [job_id]	$ scontrol hold 123456
`scontrol release`	release hold on slurm batch jobs	scontrol release [job_id]	$ scontrol release 123456

Job management commands

Job Status	Commands
`sinfo -a`	list all queues
`squeue`	list all jobs
`squeue -u userid`	list jobs for userid
`squeue -t R`	list running jobs
`smap`	show jobs, partitions and nodes in a graphical network topology

Common stumbling blocks

- Over or under using resources
- Not taking advantage of the right machine for the problem
- Moving data through slow links
- Trying to scale up programs that weren't designed for large datasets

Support

- Research IT
  - Build software, run custom hardware, consult on performance improvements, etc.
  - Collaboration between LAS, CALS, IT, etc. http://rit.las.iastate.edu/people
  - researchit@iastate.edu
  - IRC (chat): #bitcom on freenode
- ISU IT HPC Team
  - Cluster Support team
  - hpc-help@iastate.edu