Research Computing Center

Edson Manners
September 10, 2019
Who we Are
RCC Resources - Applications

1. You need more computing power.
2. You need to share and collaborate on data (big or small).
3. You need to use specialized or specially-licensed software that is difficult to install, manage, and maintain.
4. You need help developing a data/computation workflow for your research.
RCC Resources - Applications

Mathematics

Bio-medicine

Chemistry

Physics

Earth Sciences

Engineering

Life Sciences
Systems Overview

GPFS Parallel Storage (~800TB)

Interactive Cluster

Compute Cluster
The Research Computing Center at FSU

**High Performance Computing (HPC)**

1. Large cluster (700+ nodes)
2. Run large and small jobs
3. C, C++, MPI, Python, Fortran, Map/Reduce and other applications
4. A single job can run from ~1sec to several months

**Interactive Computing (GPU)**

1. 64 x NVIDIA GTX1080 Ti
2. Connect graphically and run popular software (MATLAB, R, etc)
3. Access RCC storage and software resources

...and more.
Some Technical Bits

- From Mac or Linux, simply open the terminal.
- From Windows, download PuTTy and connect.

https://rcc.fsu.edu/docs/ssh

$ ssh cam02h@hpc-login.rcc.fsu.edu
.cam02h@hpc-login.rcc.fsu.edu's password:

Last login: Wed Jan 18 16:08:19 2017
[cam02h@hpc-login-35] $
High Performance Computing
HPC “Job”

Cluster

Output

Data

Program
Serial Workflow

Add 5 + 3  Add 10 + 6  Add 4 + 10

Problem

Processor Core

Output
Parallelized Task

- Add 4 + 10
- Add 10 + 6
- Add 5 + 3

Processor/Core

Output
Parallelized to multiple nodes

Problem/Job

out.txt
Resources in the HPC

1 Node x ~700
2-4 Processors
4-48 Cores (Per processor)
Resources in the HPC
Resources in the HPC
Scheduling Jobs on the Cluster

Submit

Schedule
The Cluster consists of heterogeneous hardware

- Intel Nodes
- AMD Nodes
- High Memory Nodes
- GPU Nodes
The HPC is broken up into logical “partitions”

“General Access” Partition

- Publicly Available/Community
- Jobs can run on any of the highlighted nodes.
- Jobs can run for a maximum of **two weeks**.
- Jobs can allocate up to 400 processor cores.
- Jobs can allocate up to 3.9GB/RAM per core.
- A user can submit a maximum of 400 jobs simultaneously.
“Backfill” Partition

- Any RCC user can use this
- Jobs can run on any of the highlighted nodes.
- Jobs can run for a maximum of **four hours**.
- Jobs can allocate up to 512 processor cores.
- Jobs can allocate up to 3.9GB/RAM per core.
- A user can submit a maximum of 500 jobs simultaneously.
The HPC is broken up into logical “partitions”

“QuickTest” Partition

- Any RCC user can use this
- Jobs can run on any of the highlighted nodes.
- Jobs can run for a maximum of 10 minutes.
- Jobs can allocate up to 8 processor cores.
- Jobs can allocate up to 3.9GB/RAM per core.
- A user can submit a maximum of 2 jobs simultaneously.
The HPC is broken up into logical “partitions”

“Chemistry” Partition

- Owner-based/Condominium
- Jobs can run on any of the highlighted nodes.
- Jobs can run for a maximum of five days.
- Jobs can allocate up to 440 processor cores.
- Jobs can allocate up to 3.9GB/RAM per core.
- A user can submit a maximum of 500 jobs simultaneously.
The HPC

- Edit & compile code
- Run tests
- Create submit scripts
- Work with data
- Examine job output
Connecting to the HPC

- Use SSH (Secure Shell Protocol)
- Instructions are at: https://rcc.fsu.edu/docs/hpc
- Mac or Linux clients can simply open the “terminal” app.
- Windows clients can download and use PuTTy or MobaXTerm

```
$ ssh cam02h@hpc-login.rcc.fsu.edu
cam02h@hpc-login.rcc.fsu.edu's password:

Last login: Wed Jan 18 16:08:19 2017
[cam02h@hpc-login-35] $
```
Script Generator

https://rcc.fsu.edu/submit-script-generator

- Interactively generate a slurm script
- Limited syntax checking
- Templates available for some software
- Submit jobs directly from website (future)
Job Script Generator

**Job Title**
MyProgram

Create a name for your job (alphanumeric, dashes, and spaces allowed)

**Executable Call**
test

Enter the program you wish to run for your job. If you pipe input or pass arguments, include those.

**Separate Results and Verbose Output**
- Separate Verbose Output from Results

**Email Notifications**
- On Job Start
- On Job End
- If Job Fails
- If Job Requeues

Please select the type of email notifications about your job you would like to receive.

**HPC Partition**
Backfill 2 Queue (backfill2) – General Access

**Number of Cores**
4

Select the number of processor cores your job will run on.

**Number of Nodes**
2

Select the number of compute nodes your job will run on. Note that adjusting this does not guarantee that processes will be evenly distributed across all nodes. The default is "No Preference" and the Number of Processes, #SBATCH --ntasks, are adjusted instead if Number of Nodes is set to "No Preference".

---

**SLURM Submission Script**

```bash
#!/bin/bash
#SBATCH --job-name=MyProgram

#SBATCH --mail-type=BEGIN,END,FAIL
#SBATCH -n 4

#SBATCH -N 2
#SBATCH -p backfill2
#SBATCH -t 04:00:00

module load gnu-openmpi/2.1.6

## Submit Script Generator automatically added
## slurm test
```

To use this script:
Jobs on the HPC: Step 1: Define Parameters

```bash
#!/bin/bash

#SBATCH --job-name="primes"
#SBATCH -n 4
#SBATCH -p genacc_q
#SBATCH -t 12:00:00
#SBATCH --mail-type=ALL

module load matlab

matlab < my_input.txt
```

- I shall name this job “primes”
- I need to allocate four processor cores
- I will submit this to the `genacc_q` partition
- This job will take no longer than 12 hours to run
- Email me when the job starts, stops, or errors
- Setup the environment (more on this later)
- Here is the program to run (and the input data)
Jobs on the HPC: Step 1: Define Parameters

I shall name this job “stuff”
I need to allocate 128 processor cores
Distribute these across 32 nodes (4 procs/node)
I will submit this to the genacc_q partition
This job will take four days and six hours to run
Email me when the job starts, stops, or errors

Run this program (it doesn’t require input, or it knows where to find its own input)
Careful! Just because you ask for resources...

...doesn’t mean that your job will use them.

It is **up to your program** to take advantage of the resources that you allocate via the Slurm Scheduler.

In this example, we ask for 16 cores on a single node, and we instruct our program to use the same number, so that we don’t waste any resources that we allocated.
Things to consider: Which partition?

Is this just a quick sub-10 minute test?
  ● Use quicktest

Will this job take less than four hours to run?
  ● Consider using backfill

Does this job require inter-node communication via MPI or OpenMP?
  ● Consider using condor

Have I purchased HPC resources?
  ● Use your dedicated partition to decrease wait time.
Things to consider: Processor configuration

#!/bin/bash

#SBATCH --job-name="stuff"
#SBATCH -N 10
#SBATCH --cpus-per-node=4
#SBATCH --cores-per-cpu=2
#SBATCH --mem-per-cpu=8G
#SBATCH -p genacc_q
#SBATCH -t 4-6:00:00
#SBATCH --mail-type=ALL

ffmpeg -threads 16 in.mpg
Things to consider: **Hardware Features**

```bash
#!/bin/bash

#SBATCH --job-name="stuff"
#SBATCH -n 10
#SBATCH --constraint="INTEL&2016"
#SBATCH -p genacc_q
#SBATCH -t 4-6:00:00
#SBATCH --mail-type=ALL

ffmpeg -threads 16 in.mpg
```
Jobs on the HPC: Step 2: Submit your Job

[you@hpc-login-35] $ sbatch my_job.sh
Submitted batch job 717997

[you@hpc-login-35] $ squeue -u `whoami`

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>NODELIST(REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>717997</td>
<td>condor</td>
<td>letscoun</td>
<td>cam02h</td>
<td>PD</td>
<td>0:00</td>
<td>1</td>
<td>(Priority)</td>
</tr>
</tbody>
</table>
Jobs on the HPC: Step 3: Wait
Quick Demonstration

1. Login to the HPC via SSH client or terminal.
2. Create a submission script.
3. Submit a job the HPC.
4. Check the status with `squeue`
5. Look at the output.
## Job States

<table>
<thead>
<tr>
<th>Code</th>
<th>Status</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PD</td>
<td>Pending</td>
<td>Your job is waiting in-queue to start.</td>
</tr>
<tr>
<td>CF</td>
<td>Configuring</td>
<td>The scheduler is preparing your nodes to run your job</td>
</tr>
<tr>
<td>R</td>
<td>Running</td>
<td>Job is running</td>
</tr>
<tr>
<td>CD</td>
<td>Completed</td>
<td>Your job is all done!  Woo hoo.</td>
</tr>
<tr>
<td>PR</td>
<td>Preempted</td>
<td>Your job was cancelled, because somebody else’s job took priority over the node (this can only happen in backfill)</td>
</tr>
<tr>
<td>S</td>
<td>Suspended</td>
<td>Resources have been allocated, but the user has suspended the job.</td>
</tr>
<tr>
<td>TO</td>
<td>Timeout</td>
<td>Your job hadn’t completed in the allotted time, so the scheduler killed it.</td>
</tr>
<tr>
<td>F</td>
<td>Failed</td>
<td>Your job ran, but failed somehow (exit code != 0)</td>
</tr>
<tr>
<td>NF</td>
<td>Node Failed</td>
<td>A node that your job was running on failed during execution</td>
</tr>
<tr>
<td>CA</td>
<td>Cancelled</td>
<td>The user or an administrator explicitly canceled the job</td>
</tr>
</tbody>
</table>
Jobs on the HPC: Some things that can go wrong

1. Job execution time exceeded specified time limit.
2. A compute node that the job was running on crashed or failed.
3. Your program crashed or failed (exit code != 0).
4. You cancelled your job.
5. The Slurm Scheduler failed.
Let’s talk Software. 4 basic choices:

1. Run one of the *many, many* software packages we already have installed on the HPC:  [https://rcc.fsu.edu/software](https://rcc.fsu.edu/software)

2. Ask us to install a new program on the HPC. We generally do!

3. Download the source code and compile it in your home directory on the HPC.

4. (compile?) and run your own code

<table>
<thead>
<tr>
<th>C/C++/Fortran</th>
<th>GNU Compiler</th>
<th>Java</th>
<th>MATLAB</th>
<th>100’s of custom packages...</th>
</tr>
</thead>
<tbody>
<tr>
<td>C/C++/Fortran</td>
<td>Intel Compiler</td>
<td>VASP</td>
<td>Strata</td>
<td></td>
</tr>
<tr>
<td>C/C++/Fortran</td>
<td>PGI Compilers</td>
<td>Python 3</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Gaussian</td>
<td>Charmm</td>
<td></td>
</tr>
</tbody>
</table>
Sometimes you have to load a module...

[cam02h@hpc-login-35 ~]$ nvcc --version
-bash: nvcc: command not found

[cam02h@hpc-login-35 ~]$ module load cuda

[cam02h@hpc-login-35 ~]$ nvcc --version
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2018 NVIDIA Corporation
Built on Tue_Jun_12_23:07:04_CDT_2018
Cuda compilation tools, release 9.2, V9.2.148
Sometimes you have to load a module...

```
[cam02h@hpc-login-35 ~]$ ls /etc/modulefiles
abyss       cuda       gromacs       nwchem       pgi-openmpi  tau
adf         desmond    gulp          openfoam     python27     tv
blender     g09test    intel         orca         python3      vasp
chapel      gaussian09 intel-mvapich2 pdtoolkit     R            xalt
charmm      gerris     intel-openmpi petsc-openmpi rosetta
plex_studio gnu-mvapich2 matlab       pg
crossbow    gnu-openmpi matlab_dcs     pg
```
You also have to load modules in your Slurm scripts.

```bash
#!/bin/bash

#SBATCH --job-name="primes"
#SBATCH -n 4
#SBATCH -p genacc_q
#SBATCH -t 12:00:00
#SBATCH --mail-type=ALL

module load matlab

matlab < my_input.txt
```
Spear: Interactive Computing
Spear Servers

- Several dozen physical Linux systems hosted at RCC.
- Perfect for computing applications where HPC workflows won’t work.

GPFS Parallel Storage (~800TB)

HPC
Spear Use Cases

MATLAB  
VMD  
Chimera  

Etcetera…
Connecting to Spear Graphically:

$ ssh -Y cam02h@spear-login.rcc.fsu.edu

Last login: Wed Jan 18 16:08:19 2017

[cam02h@hpc-login-35] $ matlab
Moving Data
Moving Data: Two Options

- Lots of good clients for Windows, Mac, and Linux
- Good for shorter, simpler transfers
- Quick and easy

- Official (free) clients for Windows, Mac, Linux
- Good for long transfers (big data) and setting up automatic, recurring transfers
- Some initial setup required.
Getting Access
The Research Computing Center at Florida State University enables research and education by developing a diverse campus cyberinfrastructure and by providing training opportunities and dedicated consulting.

What we Do

**Compute**

Our HPC Cluster provides over 182 TFLOPS of compute to researchers at FSU. We also provide an Interactive Cluster (Spear), and a Virtual Machine Cloud Platform.

**Storage**

We provide two high I/O parallel file systems, Panasas and Lustre. We also provide a general-purpose filesystem, NoileStor, for archival storage with the ability to replicate to partner institutions.

**Support**

In addition to comprehensive support for our core services, we also provide research consulting and assistance, and we also host many outreach and training events.
Getting Access

- Faculty accounts are activated within 24 hours.
- Students must find a faculty sponsor (*ANY* FSU Faculty is eligible).
- Non-FSU collaborators are welcome.
Getting Help

- support@rcc.fsu.edu
- https://rcc.fsu.edu/docs
- Come see us (150 Dirac)