Lecture 19 – Interacting with HPC Resources

Learning Objectives:

4. **Produce code that is reproducible and produces results that are replicable.**

    4.5 Use scripts to make command-line functions reproducible.

5. **Learn to interact with HPC resources.**

    5.1 Connect and navigate HPC via SSH and Bash shell.

    5.2 Setup your workspace by editing bash_rc and bash_profile.

    5.3 Understand the use of module files for loading specific programs.

    5.4 Execute scripts via command line.

    5.5 Execute scripts via SLURM job manager.

High-Performance Computational (HPC) Resources

- Starting in the late 1990's, the speed of single CPUs started to plateau.

  - People figured out that the job could be made faster by running processes on multiple CPUs at once (in parallel).

  - This saw the rise of "supercomputers" with multiple cores capable of tera-FLOP/s of processing (floating-point operations per second)

  - top500.org maintains a list of the largest and fastest supercomputers on Earth, updated twice a year.

  - Current (as of June 2021) largest/fastest supercomputer in the world is Supercomputer Fugaku: 7.63 million cores with a peak rate of 537k Tera FLOP/s

High-Performance Computational (HPC) Resources

- Shared HPC resources are both common and widespread in science and industry.

  - Not everyone needs 1,000 TFLOP/s machines sitting around not being used, so it's efficient to have a large system that is centrally maintained and available to many users.

  - The eXtreme Science and Engineering Discovery Environment (XSEDE) is an NSF-funded network of HPC resources which you can request resources through for research projects. xsede.org

  - In order for this to be effective, HPC resources or "clusters" must have management systems to:

SSH ▸ allow users to connect remotely and securely from around the world

modules ▸ allow users to use vastly different programs and applications

SLURM ▸ allow users to run programs in an equitable way

sftp ▸ allow users to transfer and store data on and off "site"

Securely Connecting via SSH

- Secure Shell (SSH) is a cryptographic network protocol that allows users to operate on a network securely and remotely.

    - Common on Unix-based platforms (which is most HPC resources).

    - Allows remote logins to login “nodes” (or special CPUs dedicated to typical user interactions and not computation)

    - Also allows for tunneling, forward TCP ports, and X11 connections, but those are a pain to set up sometimes.

General operation from Bash shell:

```
ssh username@resource-name.resourcelocation.ext
```

For Keck Research Cluster:

```
ssh chapmanemail@keckcluster.chapman.edu
```

How is Software Handled?

- Software is handled on many clusters using modulefiles, which are written to modify your PATH and LIB so that it finds specific software builds available to all users.

    - Example of a module file, my build of R 4.0.3, which is in my home directory in sfw folder.

    - Basic usage of module command:

`module avail`	**lists available modules**
`module load/add modulename`	**adds module file by name**
`module unload/rm modulename`	**removes module file by name**
`module list`	**lists currently loaded modules**

Set up your .bash_profile and .bash_rc

- We need to set these up such that you will be able to easily run R on the cluster

  - bashrc:

```
# Source global definitions
if [ -f /etc/bashrc ]; then
        . /etc/bashrc
fi
```

This has to be in your bashrc to see my module files!!

```
# define modulefiles path
MODULEPATH=${MODULEPATH}:/home/waldrop@chapman.edu/modulefiles
```

```
# Uncomment the following line if you don't like systemctl's auto-paging feature:
# export SYSTEMD_PAGER=

# User specific aliases and functions
module load gcc
module load slurm
module load R
```

  - bash_profile:

```
# .bash_profile
```

This makes sure you are loading the bashrc on login.

```
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
        . ~/.bashrc
fi
```

Secure file transfer protocol (SFTP)

- SFTP works on SSH and allows for secure file transfers.
  - Isn’t quite the same as bash, commands are a little different
  - There is a remote working directory and a local working directory
  - You can get files from the remote to your local, and put files from your local to the remote.

- Basic usage of sftp command:

`sftp login@resource.ext`	**login to sftp**
`pwd`	**retrieve remote working dir**
`lpwd`	**retrieve local working dir**
`get filename`	**move file from remote to local**
`put filename`	**move file from local to remote**

Running R on the Keck Cluster

- To enter command-line R, simply type **`R`** and hit enter!

- Use R like normal!

- To exit, type **`quit()`** and then **`n`**

- To run a script from bash, type:

```
[login ~]$  Rscript src/scriptname.R
```

Submitting a Job via SLURM

- Simple Linux Utility for Resource Management (SLURM) is a popular job scheduling and workload management tool for HPC resources.

    - SLURM will schedule your job on any available nodes and queue it if resources aren't immediately available.

    - Your job's position in the queue is determined by how many resources you have used recently, compared to other users' usages. This is a way of equitably distributing resources.

- Jobs can either be submitted via command line using **`srun`** or in simple scripts using **`sbatch`**.

- Example sbatch script in simpleR.job in AdvDiff/src

- Submit this job by entering: **`sbatch simpleR.job`** in the src folder

Assignment

Complete the AdvDiff full assignment by running the script in full on the HPC cluster. Download the results files and submit those alongside your code.