Introduction to Using R on ARC's Resources

CMDA 3634: Comp Sci Foundations for CMDA

Justin Krometis, Advanced Research Computing, 4 December 2017

1 Hardware

BlueRidge is a 408-node (408-computer) cluster built in 2012-4. Almost all of the nodes on BlueRidge have 16 cores and 64 GB memory. Instructions and examples for using BlueRidge are available at http://www.arc.vt.edu/blueridge

Descriptions and instructions for other ARC clusters are available at: http://www.arc.vt.edu/computing

2 Login

1. Log into the BlueRidge login node with your PID and password:
   (a) Mac/Linux: Open a terminal and type
   ```
   ssh YOURPID@blueridge2.arc.vt.edu
   ```
   (b) Windows: Download PuTTY and in the Host Name field type: `YOURPID@blueridge2.arc.vt.edu`
2. ARC's clusters are subject to two-factor authentication. One option is to simply type your password into the password field and then be *very* quick in approving the two-factor request when it comes in. However, the following is typically a better approach:
   (a) Open the Duo app on your phone
   (b) Click the key icon on the Virginia Tech line. The app will produce a six-digit code.
   (c) Log into ARC clusters
   (d) When prompted for your password, type `password,XXXXXX` where `XXXXXX` is the six-digit code produced by the Duo app

3 Getting Files

To do computations on ARC's resources, you need to migrate the files that you need to it:

- You can download files from the internet with `wget`, e.g.
  ```
  wget https://cran.r-project.org/src/contrib/Rmpi_0.6-6.tar.gz
  ```
- You can clone git repositories, e.g.
  ```
  git clone https://username@bitbucket.org/username/\dots
  ```
- You can copy files from your computer to ARC clusters (or back), with `scp` or `rsync`, e.g.
  ```
  scp file.txt username@blueridge2.arc.vt.edu:
  ```

Some key files for this class have been placed in a central location and can be copied to your home directory as follows:

```
cp -r /home/TRAINING/cmda3634 .
```

4 Software Stack

Software on ARC clusters are managed with a module environment. There are three layers of modules:

1. Base: Require neither a compiler or MPI stack.
2. Compiler: Require that a compiler (e.g. `gcc` or `intel`) be loaded.
3. MPI: Require that an MPI stack (e.g. `mvapich2`, `openmpi`, or `impi`) be loaded.

Some key commands include:

Command	Example	Meaning
`module list`	`module list`	List the modules currently loaded
`module avail`	`module avail`	List modules that are available to be loaded
`module show`	`module show cuda`	Print information about a module
`module load`	`module load cuda`	Load a module
`module unload`	`module unload cuda`	Unload a module
`module spider`	`module spider cuda`	Search for a module
`module swap`	`module swap intel gcc`	Replace one module with another & reload dependencies
`module purge`	`module purge`	Unload all modules

To load R on BlueRidge, you might use:

```
module purge
module load intel
module load mkl R/3.2.0
```

To load R-parallel, you may then want to use:

```
module load openmpi hdf5 netcdf R-parallel/3.2.0
```

4.1 Adding R Packages

ARC's R installations come with a variety of packages installed. However, we cannot provide everything that every user might need. To use packages that we do not offer centrally, you can install them in your Home directory and then tell R where to find them, as follows: http://www.arc.vt.edu/userguide/r/#packages

5 Submitting a Job

Before you can do computationally-intensive work on an ARC cluster, you need to move from the login node (a computer where lots of people may be doing basic tasks) to a compute node (a computer dedicated to computing). BlueRidge is a shared resource, so this is done by submitting a job to a schedule, a piece of software that tries to balance many users' needs. Note: This sometimes means that you will have to wait until the resources that you need are available (i.e. until other users are done)!

However, I have reserved resources for this hands on session so you should not have to wait today.

There are two main types of jobs: interactive and batch.

5.1 Interactive Job

You can use an interactive job to get a short(-ish) interactive session on a compute node. To do this, use the development queue (`dev_q`) as follows (here we request one node for one hour with the allocation `CMDA3634` created for this class):

```
interact -l nodes=1:ppn=16 -l walltime=1:00:00 -ACMDA3634
```

(Actually, the walltime and job size are the default, so simply `interact -ACMDA3634` would also suffice.)

Once the job starts, you will get a session on a new computer, e.g. `br145`. Now you can do computationally-intensive work without interfering with other users. Try running `mcpi_parallel_mc.r`:

```
module purge; module load intel mkl R/3.2.0
Rscript mcpi_parallel_mc.r
```

You can also try running the pbdMPI version:

```
module load openmpi hdf5 netcdf R-parallel/3.2.0
mpiexec -np $PBS_NP Rscript mcpi_pbdr.r
```

The variable `$PBS_NP` holds the number of cores assigned to a job. So in the interactive job requested above, `$PBS_NP=16`.

Once you are done with the interactive job, you can simply type

```
exit
```

to finish the job and be returned to the login node.

5.2 Batch Job

Batch jobs are used to launch non-interactive sessions to run larger and longer computationally-intensive programs. These jobs are typically submitted via a *submission script*. The submission script has two main parts:

1. The header, which describes the job, e.g. how many resources it needs and for how long. These lines begin with `#PBS`.
2. The body, which describes what to do once those resources have been obtained.

Here is an example submission script for submitting a one-node job for one hour to the production queue (`normal_q`), and to run several of the `mcpi` examples on those resources:

```
#!/bin/bash

#PBS -lnodes=1:ppn=16
#PBS -lwalltime=1:00:00
#PBS -qnormal_q
#PBS -Wgroup_list=blueridge
#PBS -ACMDA3634

cd $PBS_O_WORKDIR

#load R and R-parallel
module purge
module load intel mkl R/3.2.0
module load openmpi hdf5 netcdf R-parallel/3.2.0

#run the serial version
Rscript run_mcpi.r

#run the parallel version with parRapply
```

```
Rscript mcpi_parallel_apply.r

#run the parallel version with mclapply
Rscript mcpi_parallel_mc.r

#run the pbdr version
mpiexec -np $PBS_NP Rscript mcpi_pbdr.r
```

5.2.1 Submitting and Checking a Batch Job

This script can be submitted to the scheduler as follows:

```
qsub mcpi.qsub
```

This will return your job name of the form

```
181556.master.cluster
```

Here `181556` is the job number. Once a job is submitted to a queue, it will wait until requested resources are available within that queue, and will then run if eligible. Eligibility to run is influenced by the resource policies in effect for the queue.

To check a job's status, use the `checkjob` command:

```
checkjob -v 181556
```

You can also view all of your jobs using the `showq` or `qstat` commands:

```
showq -u $USER
qstat -u $USER
```

To check resource usage on the nodes available to a running job, use:

```
jobload 181556
```

5.2.2 Results

When your job has finished running, any outputs to stdout or stderr will be placed in the files .o and .e. These two files will be in the directory that you submitted the job from. For example, for a job submitted from `mcpi.qsub` and with job ID `181556`, the output would be in:

```
mcpi.qsub.o181556 #Output will be here
mcpi.qsub.e181556 #Any errors will be here
```

6 References

- http://www.arc.vt.edu/r describes how to use R on ARC's systems
- http://www.arc.vt.edu/computing describes ARC's hardware, usage policies, and software, and provides some step-by-step examples
- http://www.arc.vt.edu/storage describes ARC's storage systems
- http://www.arc.vt.edu/software describes software installed on ARC's systems
- http://www.arc.vt.edu/modules describes how to use ARC's software modules
- http://www.arc.vt.edu/scheduler describes how to interact with the scheduler - submit and check jobs, view output, etc.
- http://www.arc.vt.edu/faq provides answers to some frequently asked questions