Engineering Software

Tyson Whitehead and Doug Roberts

February 7, 2024

COURSE

CHAPTER

ONE

SUPERCOMPUTING

1.1 Overview

1.1.1 What isn't a supercomputer

A super fast computer that just runs your program faster.

1.1.2 What is a supercomputer

Graham (latest SHARCNET supercomputer)

- 1,327 computers (nodes)
- 44,444 cores
- 197 terrabytes (TB) of RAM (most 4 gigabyte (GB)/core)
- 19 petabytes (PB) of disk storage (home, project, and scratch)
- 80 petabytes (PB) of transparent tape storage (nearline)
- 534 NVIDIA GPUs (P100 Pascal, V100 Volta, and T4 Turing)
- EDR (56G/s) and FDR (100Gb/s) InfiniBand

1.1.3 How do you use a supercomputer

serial ~ (easy) many computers means you can run many programs indepedently at the same to solve many independent problems

parallel ~ (hard) many computers can (sometimes) be programmed to all collaborate and solve a single problem together

cloud ~ virtual computer on the internet (like Amazon's Elastic Compute Cloud, Microsoft's Azure cloud, or Google's Compute Engine)

1.2 Canada

1.2.1 The players

National

- Digital Research Alliannce of Canada

Regional

- Compute Ontario, Calcul Québec, ACENET, BC DRI Group, and Prairies DRI

Ontario

- Center for Advanced Computing (CAC), HPC4Health, SciNet, and SHARCNET

1.2.2 Getting an accounts

Cost

- no cost

Who

- faculty or those sponsored by faculty that have an account

How

- https://ccdb.alliancecan.ca
- apply for a Digital Research Alliance account
- pick the desired regional consortia accounts

1.2.3 What software is available

Operating system

- Linux (CentOS 7/Rock 8)

Programming languages

- C/C++, Fortran, MATLAB/Octave, Python, R, Julia, etc.

Parallel development support

- pthreads, MPI, OpenMP, CUDA, OpenACC, OpenCL

Other

- common open source and commerical packages (e.g., OpenFOAM, Fluent, and STAR-CCM+)

1.2.4 How to use

- resources are scheduled to avoid collisions and ensure fair access
- access from anywhere on the internet using secure shell (SSH) to enter commands and a secure file transfer (SFTP) to transfer files
- tell the supercomputer what progam (command) you want it to run and then do something else till it does it

1.2.5 Typical workflow

1. transfer your data and/or programs to the supercomputer using SFTP
2. login (bring up a window in which you can enter commands) to the supercomputer using SSH
3. enter commands to tell the scheduler what you want it to run when the required resources are avaialable
4. do something else until you get notification that your commands have completed running
5. transfer the resulting data from the supercomputer to your computer

CHAPTER

TWO

USING

2.1 Required Software

Clusters are accessed via the secure shell programs

ssh ~ used to run commands (secure shell)

scp ~ used to copy files (secure copy)

sftp ~ alternative to copy files (secure file transfer protocol)

from anywhere on the internet using the address `CLUSTER.alliancecan.ca` (capitalized terms indicate you should substitute something appropriate).

2.1.1 Windows

Most common options are the GUI

- mobaxterm is a good freeware GUI choice that provides everything in one package including a graphical sftp, X11 server, and local shell
- PuTTY is a basic open-source ssh-only option

Windows 10 has native versions of the command-line tools too

- Windows subsystem for Linux provides full linux text environment including secure shell
- OpenSSH installable component provides secure shell inside Windows command or power shell

2.1.2 MacOS X and Linux

Command-line versions are builtin and available in the shell

- search for and start a *terminal* application

2.2 Accessing

2.2.1 Logging in

`ssh USER@CLUSTER.alliancecan.ca` ~ login to CLUSTER as USER and start a remote shell session

- for GUI ones (e.g., mobaxterm, putty, etc.), enter `USER` into username field, `CLUSTER.alliancecan.ca` into host name field, and connect

2.2.2 Copying files

`scp USER@CLUSTER.alliancecan.ca:REMOTE LOCAL` ~ login to CLUSTER as USER and copy REMOTE file to LOCAL file

`scp LOCAL USER@CLUSTER.alliancecan.ca:REMOTE` ~ login to CLUSTER as USER and copy LOCAL file to REMOTE file

- For GUI ones (e.g., mobaxterm, etc.) enter `USER` and `CLUSTER`, connect, and then drag and drop files to transfer

2.3 Shell

Commands, entered interactively or ran from a file, tell computer what to do

- a bit of an initial learning curve, but
- much easier to automate, document, and share

2.3.1 Basic command

```
PROGRAM ARGUMENTS...
```

- arguments usually options (switches) followed by strings (e.g., file names, etc.)
- short options are a single dashes followed by letter for each switch (e.g., `rm -fr mydir`)
- long options are a double dash followed by a descriptive string (e.g., `rm --force --recurse mydir`)

Help

`help COMMAND` ~ show help for builtin command (like "cd")

`man COMMAND` ~ show the manual page for command ('q' to quit)

Getting around

`ls` ~ list files

`pwd` ~ what folders you are in

`cd DIRECTORY` ~ change to directory named directory (use ".." for previous one)

Copying/renaming/moving/removing files and directories

`cp SRC DEST` ~ copy file/directory (add `-r` for directory)

`mv SRC DEST` ~ move/rename file/directory

`rm PATH` ~ remove file (add `-r` for directory)

Editing files

`nano PATH` ~ edit file named file

Digital Research Alliance of Canada software (scientific packages)

`module load CcEnv StdEnv` ~ enable the software stack on VDI systems (Cc for Compute Canada for historical reasons)

`module list` ~ show what software is load environment

`module avail` ~ show what software you can load

`module spider PACKAGE` ~ search for package

`module load PACKAGE` ~ load package

`module unload PACKAGE` ~ unload the package

Software via Nix (utilities)

`module load nix` ~ enable personal Nix software environment

`nix search PACKAGE` ~ search for PACKAGE

`nix run ATTRIBUTES ...` ~ start sub-shell with ATTRIBUTES available`

`nix-env -q` ~ list software installed in personal environment

`nix-env -iA ATTRIBUTE` ~ install ATTRIBUTE into personal environment

`nix-env -e NAME` ~ remove NAME from personal environment

`nix-env --rollback` ~ undo change to personal environment

Running/submitting, viewing, and killing jobs

`salloc` ~ start interactive job

`sbatch SCRIPT` ~ queue batch job

`squeue` ~ list jobs

`skill JOBID` ~ kill job

2.4 Filesystems

Storage Folder	Group	Expiration	HPC	Size	Files
/home/USER	No	No	No	50GB	500K
/scratch/USER	No	60 days	Yes	20TB	1,000K
/project/PROJECT	Yes	No	Yes	1TB	500K

- quick `scratch`, `project` (default), and `projects` (all) links under `home`

2.5 Jobs

Check documentation page for desired software on docs.alliancecan.ca

- program specifc instructions
- licensing details for commerical software

All significant computations (> few minutes) run as SLURM jobs

- only way to access the 1,000+ compute nodes
- initial interative type-in-commands test runs with `salloc`
- final non-interactive put-commands-in-file runs with `sbatch`

Batch file for `sbatch` have the general format

```
##!/bin/bash
##SBATCH OPTION1
##SBATCH OPTION2
...

COMMANDS TO RUN
...
```

2.5.1 Job options

Specify the resources and accounting required to run a job

- as options to `salloc` and `sbatch` commands, and/or
- in start of job files using special `#SBATCH ...` lines

Basic options

`--account def-SPONSOR` ~ account to allocate usage to

`--time D-HH:MM:SS` ~ maximum runtime before killing (add a bit of buffer)

`--mem-per-cpu N` ~ memory per CPU (suffix with `K`, `M`, `G`, or `T` for units)

Multi-threaded jobs

- limited to maximum number of CPUs in one computer (generally 32)
- may need `export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK` in job script

`--cpus-per-task N` ~ number of threads per process (threaded)

Multi-processor jobs (MPI)

- need to start the program with the `mpirun` or `srun` commands

`--ntasks` ~ number of processes (MPI)

GPU jobs

`--gpus-per-task TYPE:N` ~ number of GPUs per process

2.5.2 Breakpoints

`partition_stats` ~ show paritioning of compute nodes eligible for jobs

Exceeding these resources decreases the number of machines that will run a job

- runtime > 3, 12, 24, 72, 168, and 672hrs
- memory/cpu > 12MB

CHAPTER

THREE

OPENFOAM

3.1 Enabling

Search through the available modules to see what OpenFOAM ones exist

```
$ module avail openfoam
```

This only shows versions that are compatible with other loaded modules. All versions can be found with the `spider` command along with directions on what else needs to be loaded to load them

```
$ module spider openfoam
$ module spider openfoam/11
```

Note that there two releases of OpenFOAM. One is openfoam.com and uses vYYMM (year and month) version numbering. The other is openfoam.org and uses X and X.Y version numbering. With the exception of a some specific features, the two are largely similar.

Load the openfoam module (not specifying a version uses the default). Note that `spider` may shom additional modules that you need to load first.

```
$ module load gcc
$ module load openfoam
```

Can use the `show` command to see what this does. The environment variables that it sets can be accessed using the `$VARIABLE` syntax (substuting `VARIABLE` for the name of the variable).

```
$ module show openfoam
```

Create the top-level user OpenFOAM directory and change to it (not strictly required, but recommended by the tutorials)

```
$ mkdir -p $FOAM_RUN
$ cd $FOAM_RUN
```

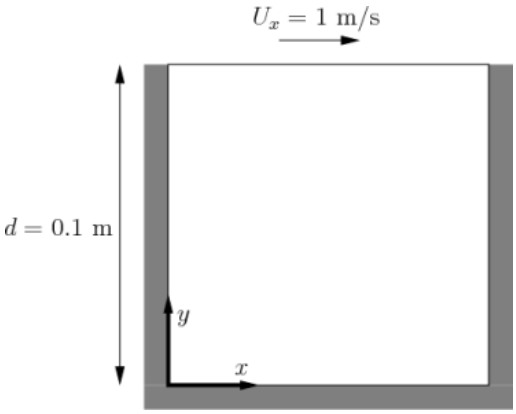
3.2 Tutorial

OpenFOAM comes with several tutorials in the `$FOAM_TUTORIALS` directory. This includes the ones in the user manual.

```
$ ls $FOAM_TUTORIALS
```

Until version 11, the standard first tutorial was a cavity flow problem. It is still available in the *legacy* directory though, and it is one of the simplest examples, so we will use it.

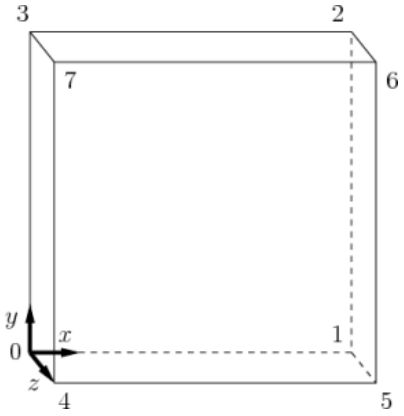
```
$ cp -r $FOAM_TUTORIALS/legacy/incompressible/icoFoam/cavity/cavity .
$ cd cavity
```



3.3 Generate mesh

Generate the mesh data files in *constant/polyMesh* from the higher-level description file *system/blockMeshDict*

```
$ blockMesh
```



3.4 Interactive run

Run the simulation (incompressible flow using the PISO algorithm) in an interactive session to test

```
$ salloc --mem-per-cpu=500M --ntasks=1 --account=def-SPONSOR
```

```
$ icoFoam
$ exit
```

3.5 Batch run

Create a *run.sh* (name not important) using nano to do the same

```
$ nano run.sh
```

```
#!/bin/bash
#SBATCH --time=00:05:00
#SBATCH --mem-per-cpu=500M
#SBATCH --ntasks=1
#SBATCH --account=def-SPONSOR
#SBATCH --output=icoFoam-%J.log
icoFoam
```

Submit the batch file to the cluster to run

```
$ sbatch run.sh
```

3.6 New case

Make a new *cavityMPI* case based on the *cavity* case setup

```
$ cd ..
$ foamCloneCase cavity cavityMPI
$ cd cavityMPI
```

3.7 Run in parallel

Basic four processor *system/decomposeParDict* decomposition setup copied from damBreak tutorial

```
$ cp $FOAM_TUTORIALS/incompressibleVoF/damBreakLaminar/damBreak/system/decomposeParDict
```

Decompose the cavity across the four processors

```
$ decomposePar
```

Run the simulation in parallel using four processors

```
$ salloc --mem-per-cpu=500M --ntasks=4 --account=def-SPONSOR
```

```
$ mpirun icoFoam -parallel
$ exit
```

Collect the output back into a single file

```
$ reconstructPar
```

3.8 Prep for ParaView

Create a `NAME.foam` file to give something to open in ParaView (name doesn't matter – just identifies it as an OpenFOAM directory)

```
$ touch cavity.foam
```

Can also generate VTK input files

```
$ foamToVtk
```

CHAPTER

FOUR

PARAVIEW

4.1 VDI nodes

Two large memory graham remote desktop machines pre/post-processing

- access to all files (don't have to transfer data back and forth)
- full details on docs.alliancecan.ca

4.1.1 Accessing

Connect via VNC (TigerVNC viewer recommended) to `gra-vdi.alliancecan.ca`

Windows and OS X

Download latest version from TigerVNC website

- releases (at top) -> binaries are available from bintray (end of changes list)

Linux

Install with your package manager

- Debian and Ubuntu: `sudo apt-get install tigervnc-viewer`
- Fedora: `sudo yum install tigervnc`

4.1.2 Software

- Compute Canada stack is the same as is on cluster
- Nix stack lets you install and customize your own packages
- Limited number of local-only packages: `module load clumod`

4.2 Software

4.2.1 Personal computer

Available for Windows, OS X, and Linux

Windows and OS X

Download latest version from ParaView website

- older version may if newer fails on older OS or graphics card

Linux

Install with your package manager (can also download as above)

- Debian and Ubuntu: `sudo apt-get install paraview`
- Fedora: `sudo yum install paraview`

4.2.2 VDI

Enable the Compute Canada stack and load the `paraview` module

```
$ module load CcEnv StdEnv
$ module load paraview
```

or install in your personal Nix environment

```
$ module load nix
$ nix-env -iA nixpkgs.paraview
```

4.3 Tutorial

Working through the ParaView tutorial is one of the quickest and easiest ways to get up-to-speed on ParaView.

- basic usage (what we will be going over)
- batch python scripting
- visualizing large models

```
$ cp -r /home/tyson/ParaView .
```

4.3.1 Basis of visualization

- map raw data to visual data
- spacial and temporal data
- topology and types of grids

4.3.2 User interface

- menu bar
- tool bars
- pipeline browser
- properties panel
- view

4.3.3 Basic interface

- creating a source
- interacting with a 3d view
- modifying visualization paramaters (filter, display, view)
- undo and redo (regular vs camera)

4.3.4 Loading data

- opening file (selecting which variables to load)
- representation and field coloring
- scaling

4.3.5 Filters

- selecting filters (toolbar, menu, and search)
- applying a filter (contours, slices)

4.3.6 Multiview

- creating a multiple views
- linking cameras
- re-arranging the views

4.3.7 Vector visualization

- streamlines
- tubes and glyphs
- surface LIC

4.3.8 Volume rendering

- Enabling
- Transfer function

4.3.9 Plotting

- histogram plot
- plot over a line in space
- plot over a curve
- plot over time

4.3.10 Selections

- Query and view based selections
- Data vs spatial selections
- Selection labels
- Extract selection and spreadsheet

CHAPTER

FIVE

ANSYS

5.1 Licensing

Types:

- CMC/SHARCNET/Campus
- Server/Floating
- License File
- Purchase/Renewal

5.2 Wiki Documentation

- Abaqus
- Ansys
- Comsol
- Starccm

https://docs.alliancecan.ca/

5.3 Ansys Online Help

- Release Notes
- Users Guides
- Analysis Guides
- Tutorials & Videos
- Optimization & Performance Guides
- Material Reference
- Command Reference

https://docs.alliancecan.ca/wiki/Ansys#Online_documentation

5.4 Ansys Electronics Desktop (AEDT)

- Ansys Help - HFSS (high-frequency simulation software) - Videos:
  - The ANSYS Electronics Desktop Environment
  - ANSYS HFSS: Overview of Antenna Simulation - Part 1
- Demo Interactive (salloc) AND Queue (sbatch)

5.5 ANSYS Tutorials

- Meshing - Meshing Tutorial Guide
  - Can Combustor
- Fluent - Fluent Tutorials
  - Fluid Flow in an Exhaust Manifold
- Fluent - Fluent Workbench Tutorial
  - Introduction to Using Ansys Fluent in Ansys Workbench: Fluid Flow and Heat Transfer in a Mixing Elbow
- Mechanical APDL - Introductory Tutorials
  - Structural Tutorial - Static Analysis of a Corner Bracket
- Mechanical Application - Mechanical Tutorials
  - Rigid Dynamics Analysis of an Actuator Mechanism using Joints and a Spring

CHAPTER

SIX

SEARCH

- search