Week 2 Lab Session

Adam Wu, Wonje Yun

March 29, 2024

University of Chicago

Contents

Contents

Accessing Midway

Using SSH - On Terminal

The main and stable method of accessing Midway3 is through SSH. Open *terminal* for Mac/Linux and type:

```
ssh <CNetID>@midway3.rcc.uchicago.edu
```

- Type yes when asked
- Enter your password
- Use whatever verification method to get in (Duo, text, call)
- type *logout* in terminal to disconnect.
- See more: RCC User Guide for SSH

Using SSH - For Windows Users

There are couple of options for Windows users to access Midway with SSH.

- **Git Bash**: Use the same command as in the previous slide by opeing *`git bash`*.
- **WSL2**: Use the same command as in the previous slide (need to install WSL2 first) WSL2 Installation Guide.
- **Vscode(Mac/Linux as well)**: Use the Remote-SSH extension to connect to Midway (explained later, needs SSH client pre-installed for Windows).
- **Powershell**: Use the same command as in the previous slide with *`powershell`* app.

Using ThinLinc

ThinLinc is a graphics interface for the Midway server. It is more unstable, so if you fail SSH login or keep failing with your job submission (usually Windows), then try this way.

- Type in *https://midway3.rcc.uchicago.edu* to access.
- Type in your CNet ID and password and do the verification
- Open the Menu and open terminal You'll see the same interface as your SSH.
- **DO NOT shut down the system!!** Just close the tab or click the disconnect button.

Setting up Github on Midway3

- We'll be using Github to manage our assignment files (please let the TAs know if you have problem with this process).
- The process of setting up can be found on Canvas Modules or you can click this link: Setting up Github on Midway3

Setting up Assignment Files on Midway3

- From your assignment repository, git clone to your login node.
- Then you'll see your assignment files in the login node using `ls` command.
- Remember to git push your codes when changes are made.
- The results for the assignments should be addressed in your README.md file (**Read Carefully for Instructions in the Assignment Description**)

Editing Assignment Files on Midway3

There are several ways to edit your code, choose whatever you prefer.

- **Vim/Nano**: Edit code directly on Midway terminal using the pre-installed code editor.
- **Vscode**: Use SSH remote connection to connect with Midway and access the files as in Vscode style. See more: Setting up Remote-SSH in Vscode
- **Github**: Edit directly on github and commit/push, then git pull on Midway to access files or submit jobs.
- **Local**: Edit the files on your local machine and transfer the files to Midway3 using *scp* command. See more: Stanford User Guide for SCP

Using SCP to Transfer Files

- **SCP** is a command to transfer files between your **local** machine and the server.
- *Note: For Windows users, *`cd`* to the directory where the file is located for stable transfer.
- *Note: For Windows users run *`dos2unix <filename>`* on your terminal before submitting jobs.
- Usage for transferring folders to/from Midway3 is as follows:

```
scp -r <local_folder> <CNetID>@midway3.rcc.uchicago.edu:</path/>
scp -r <CNetID>@midway3.rcc.uchicago.edu:</path/> <local_folder>
```

See more: Stanford User Guide for SCP

Submitting Jobs

.sbatch Script Parameters

- .sbatch files are required to submit your codes/jobs to the HPC server for computation. Login nodes are not for computation!!
- The following code is some of the parameters that needs to be written at the top of your .sbatch file.

```
#!/bin/bash

#SBATCH --job-name=<job_name>
#SBATCH -o <output_file_name>.out
#SBATCH -e <error_file_name>.err
#SBATCH --partition=caslake
#SBATCH --ntasks=3
#SBATCH --nodes=1
#SBATCH --mail-type=ALL
#SBATCH --mail-user=<CNetID>@rcc.uchicago.edu
#SBATCH --account=macs30123
```

.sbatch Script Parameters - Explained

- **job-name**: the name for the job you submit
- **o**: the file to save your output
- **e**: the file to save your error output
- **partition**: the type of server you want to use
- **ntasks**: the number of CPU cores to utilize
- **nodes**: the number of nodes to use
- **mail-type/mail-user**: the email to send the job status
- **account**: the Midway account we are using

These are the common parameters that is used in Assignment 1. There are more parameters needed to run your assignments depending on GPU jobs or CPU jobs, which will be discussed in class.

.sbatch Script Commands

- Load necessary modules (python, mpich)
- Type in the terminal command that runs your file
- The code that should be written here must be in bash language.
- If you want to run the same files in a loop, use the bash loop code

```
#Load Python module to run the code
module load python mpich

#Running python programs
python3 <file_name>.py
for i in {1..3}
    mpirun -n $i python3 <another_file_name>.py
done
```

.sbatch Script as a Whole

```
#!/bin/bash

#SBATCH --job-name=<job_name>
#SBATCH -o <output_file_name>.out
#SBATCH -e <error_file_name>.err
#SBATCH --partition=caslake
#SBATCH --ntasks=3
#SBATCH --nodes=1
#SBATCH --mail-type=ALL
#SBATCH --mail-user=<CNetID>@rcc.uchicago.edu
#SBATCH --account=macs30123

module load python mpich

for i in {1..3}
    mpirun -n $i python3 <file_name>.py
done
```

SLURM Commands

1. After writing your .sbatch file, you need to submit the file to the server to run the job.

```
sbatch ./<your_sbatch_file>.sbatch
```

2. To check the status of your job:

```
squeue --user=<CNetID>
```

3. To cancel your job (job id can be obtained by checking your status)

```
# to cancel all the jobs you submitted
scancel --user=<CNetID>
# to cancel a single job you submitted
scancel <job_ID>
```

4. More commands: RCC User Guide for SLURM

Try Yourself!

- If you have not installed packages as in the course-material repo, please do it prior.
- Copy the provided code *`mpi_rand_walk.py`* and save it in your login node (using your preferred editor)
- Also, generate the *`lab_wk2.sbatch`* that would submit the python file and run parallel on 3 cores.
- Add any necessary parameters to the .sbatch file from the RCC user guide.
- Submit the job using the SLURM command, and try *`squeue`*, *`scancel`* commands to see what happens.
- (Extra) Try to run the same code in a loop for 1 to 3 cores (codes are in the previous slides).

Try Yourself!

```
#!/bin/bash

#SBATCH --job-name=mpi
#SBATCH --output=mpi.out
#SBATCH --ntasks=3
...
#SBATCH --account=macs30123
#SBATCH --partition=caslake

# Load Python and MPI modules
module load python mpich

# mpirun will automatically figure out the best configuration
    from the
# Slurm environment variables.
mpirun python3 ./mpi_rand_walk.py
```

Assignment Walk-through

Question 1

- **Objective**: Testing speed-up of Numba and MPI by applying it to a given code.
- (a): Implement the given code in Numba using **pycc** (**not numba.jit**).
- (b): Multi-process the given code using MPI and check the computation time using different number of cores.
- (c): Reason of non-linear speed-up in MPI.

Question 2

- **Objective**: Using large-scale computation (MPI) to tackle a problem.
- (a): Improve the given code to make it solve the probelm and use multiprocessing to handle the computation.
- (b): Result report using plot. (read the output file by line to a list to plot)
- (c): Result interpretation.

QnA

Why you need all this?

- There are many data services offering cloud servers.
- Eg) WRDS Cloud: you can exactly and access their HPC via SSH to query or compute using their data.
- Eg) Midway SSD for simulating your thesis experiment.
- Eg) Private companies or universities may have own HPC servers, where you may have chance to use in the future.

QnA

Any questions?

Further Reference

- Main reference for this slide is from UChicago Research Computing Center's User Guide [1]
- For further reference or documentation, visit *https://rcc-uchicago.github.io/user-guide/*
- If the code doesn't run even if you are using the reference from user guide, look for documentations from other HPC servers (other universities such as Princeton, Stanford, etc.)

References i

- Research Computing Center. ”RCC User Guide.” Accessed March 20, 2024. https://rcc-uchicago.github.io/user-guide/.
- Microsoft. ”How to install Linux on Windows with WSL.” Accessed March 20, 2024. https://docs.microsoft.com/en-us/windows/wsl/install-win10.
- Visual Studio Code. ”Remote Development using SSH.” Accessed March 20, 2024. https://code.visualstudio.com/docs/remote/ssh.
- Sherlock (Stanford University). ”Data Transfer - SSH based Protocols.” Accessed March 20, 2024. https://www.sherlock.stanford.edu/docs/storage/data-transfer/#scp-secure-copy.