MTU MRI GPU Cluster Blizzard User Manual

Tao Liu, Hamid Ismail, Joshua Kemppainen, Dukka KC

04/04/2024

Get an account to use Blizzard

Send an email to Dr. Dukka KC (dbkc@mtu.edu) to request an account.

Description of the cluster

DeepBlizzard was acquired through an NSF MRI grant () and it has the following nodes. Deepblizzard has the following nodes:

Qty 7 - GPU Nodes - 64 Cores (AMD EPYC) CPU, NVIDIA HGX A100 (4x A100 SXM4 80GB), 512GB Ram
Qty 3 - Large memory nodes - 64 Cores (AMD EPYC) CPU, 1T memory
Qty 2 - compute nodes - 64 Cores (AMD EPYC) CPU, NVIDIA A30 24GB

Hence the partitions: mrigpu, mrilargemem, and mrijobs for each of these nodes. You can use the *sinfo* command to see the working status of each partition.

1. Connecting to the cluster

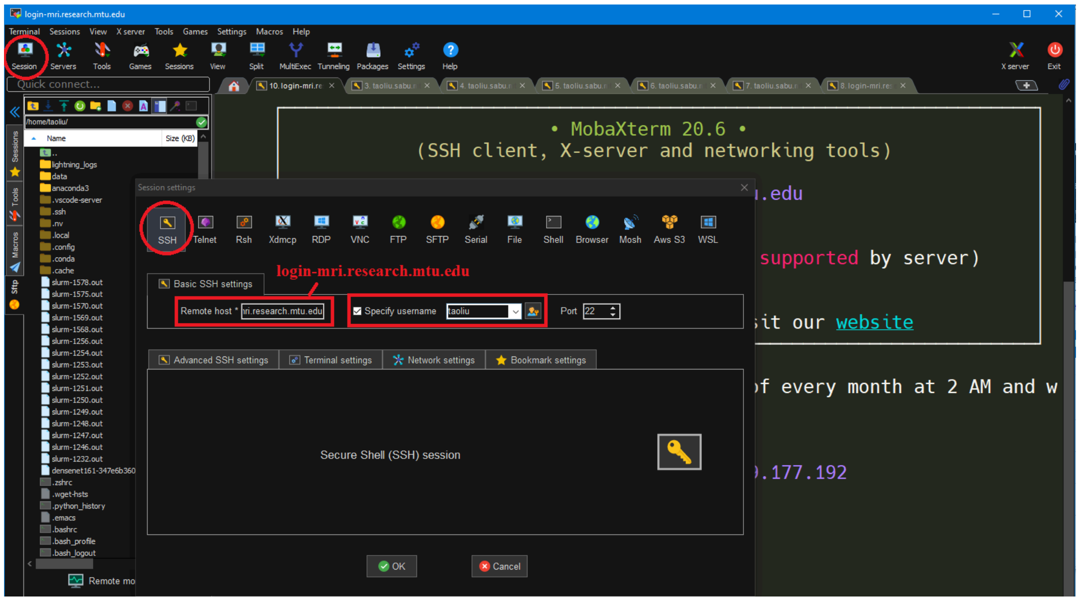
Several options are available to log on to your account using Secure Shell (SSH). The users of old Windows versions can install MobaXterm (https://mobaxterm.mobatek.net/download.html) or Putty (https://www.putty.org/), which are open-source software and can be installed for free. In Linux and MacOS, use the terminal to connect to the server. In the following, we describe the use of each method to connect to the Blizzard server.

A- MobaXterm

To connect to the server follow these steps:

- Launch MobaXterm (Fig. 1).
- Click the Session icon on the upper left.
- On the session setting dialog window, click “SSH” on the upper left.
- On the remote host field enter “login-mri.research.mtu.edu”
- Check the checkbox next to “Specify username” and enter your MTU email username without “@mtu.edu”.
- Leave the default port number 22 and click “OK”.
- The terminal will prompt you to enter your password. Enter your MTU SSO password and press enter.
- A dialog box will pop up asking whether you want to save your password or not. Press “Yes” or “No” to log on.

Once you log on, the terminal with the command line prompt appears as in Fig. 2.



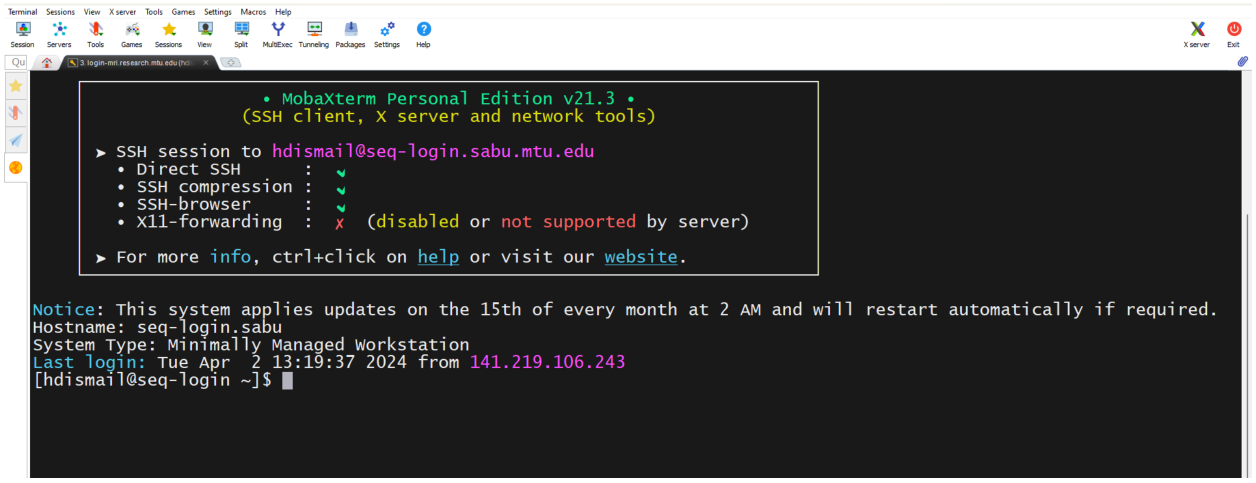
login-mri.research.mtu.edu

Remote host * ri.research.mtu.edu
[✓] Specify username taoliu
Port 22

Secure Shell (SSH) session

```
• MobaXterm 20.6 •
(SSH client, X-server and networking tools)
```

Fig. 1: This figure shows you how to use the session dialog window to enter the hostname and username.



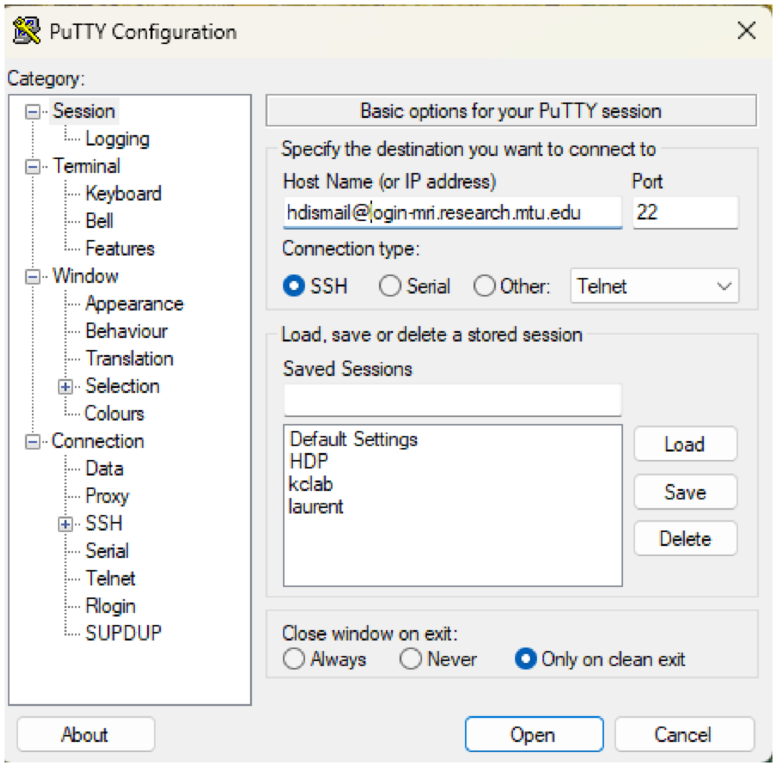
```
• MobaXterm Personal Edition v21.3 •
(SSH client, X server and network tools)

➤ SSH session to hdismail@seq-login.sabu.mtu.edu
  • Direct SSH       :  ✓
  • SSH compression  :  ✓
  • SSH-browser      :  ✓
  • X11-forwarding   :  ✗  (disabled or not supported by server)

➤ For more info, ctrl+click on help or visit our website.

Notice: This system applies updates on the 15th of every month at 2 AM and will restart automatically if required.
Hostname: seq-login.sabu
System Type: Minimally Managed Workstation
Last login: Tue Apr  2 13:19:37 2024 from 141.219.106.243
[hdismail@seq-login ~]$
```

Fig. 2: The screen shows the shell command line prompt when you log on.

B- PuTTy

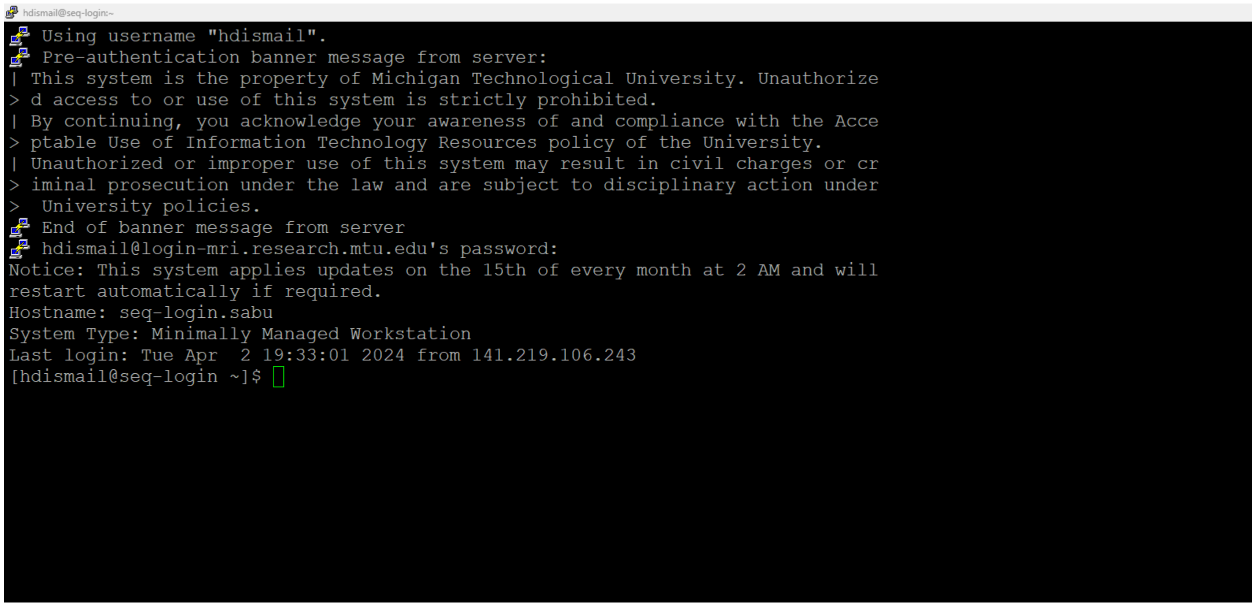
To connect to the server using PuTTy:

- Launch PuTTy
- From the Category tree items on the left side, select “Session”.
- On the host field, you can enter your user name + “@” + the hostname as shown in Fig 3 (e.g. yourusername@login-mri.research.mtu.edu) and click “Open”.
- A dialog box may pop up when you log on for the first time, telling you that the host key is not cached for this server. Click “Accept”. Enter your password and press enter. The command line prompt will show up (Fig. 4).



Specify the destination you want to connect to
Host Name (or IP address): hdismail@login-mri.research.mtu.edu
Port: 22
Connection type: [✓ SSH] Serial Other: Telnet
Load, save or delete a stored session
Saved Sessions
Default Settings
HDP
kclab
laurent
Close window on exit: Always Never [✓ Only on clean exit]

Fig 3: Connecting to the server using PuTTy



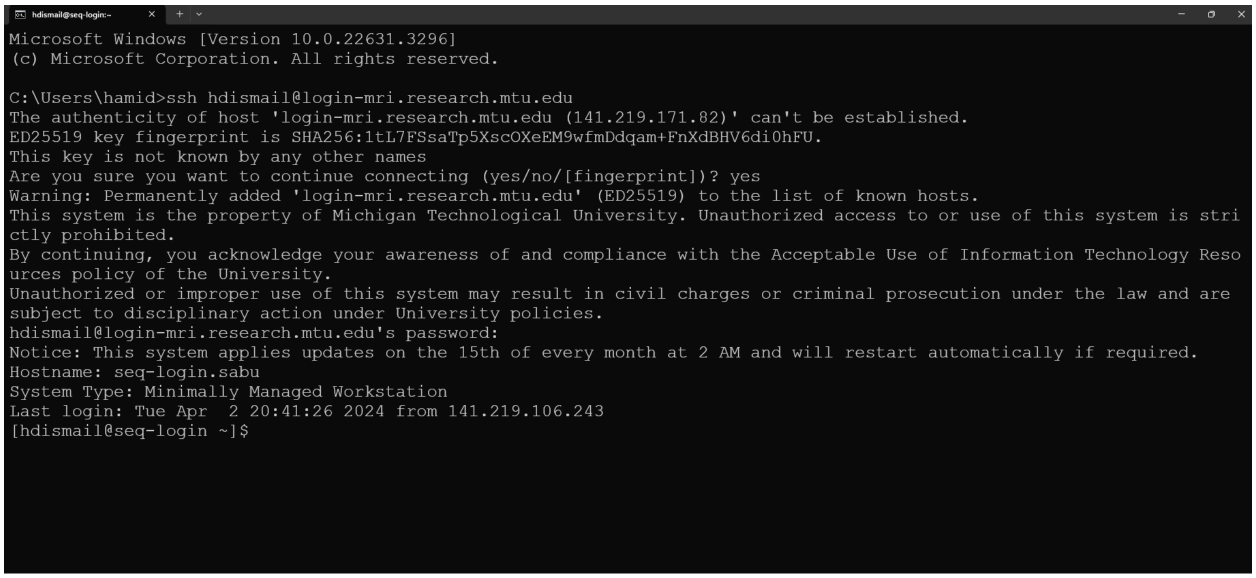
```
hdismail@seq-login:~
Using username "hdismail".
Pre-authentication banner message from server:
| This system is the property of Michigan Technological University. Unauthorize
> d access to or use of this system is strictly prohibited.
| By continuing, you acknowledge your awareness of and compliance with the Acce
> ptable Use of Information Technology Resources policy of the University.
| Unauthorized or improper use of this system may result in civil charges or cr
> iminal prosecution under the law and are subject to disciplinary action under
>  University policies.
End of banner message from server
hdismail@login-mri.research.mtu.edu's password:
Notice: This system applies updates on the 15th of every month at 2 AM and will
restart automatically if required.
Hostname: seq-login.sabu
System Type: Minimally Managed Workstation
Last login: Tue Apr  2 19:33:01 2024 from 141.219.106.243
[hdismail@seq-login ~]$
```

Fig. 4: The command line prompt when you use PuTTy.

C- Windows Prompt

- On search at the bottom, type “Command Prompt” and launch it.
- On the command prompt type
  `ssh yourusername@login-mri.research.mtu.edu`
  and press enter.

- You will be asked if are sure that you want to connect. Type “yes” and enter.
- You will be prompted to enter your password. Once you enter it and press the enter key, the Linux command line prompt will show up as shown in Fig. 5.



```
Microsoft Windows [Version 10.0.22631.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hamid>ssh hdismail@login-mri.research.mtu.edu
The authenticity of host 'login-mri.research.mtu.edu (141.219.171.82)' can't be established.
ED25519 key fingerprint is SHA256:1tL7FSsaTp5XscOXeEM9wfmDdqam+FnXdBHV6di0hFU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'login-mri.research.mtu.edu' (ED25519) to the list of known hosts.
This system is the property of Michigan Technological University. Unauthorized access to or use of this system is stri
ctly prohibited.
By continuing, you acknowledge your awareness of and compliance with the Acceptable Use of Information Technology Reso
urces policy of the University.
Unauthorized or improper use of this system may result in civil charges or criminal prosecution under the law and are
subject to disciplinary action under University policies.
hdismail@login-mri.research.mtu.edu's password:
Notice: This system applies updates on the 15th of every month at 2 AM and will restart automatically if required.
Hostname: seq-login.sabu
System Type: Minimally Managed Workstation
Last login: Tue Apr  2 20:41:26 2024 from 141.219.106.243
[hdismail@seq-login ~]$
```

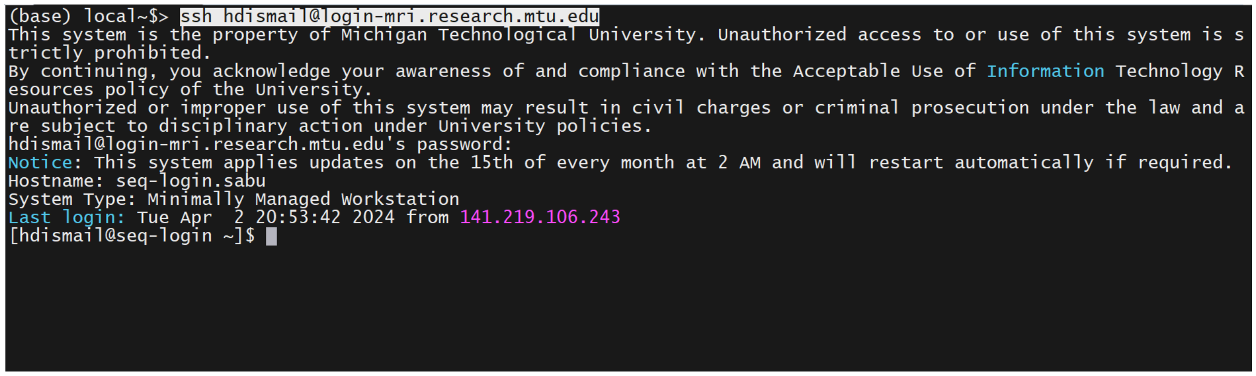
Fig. 5: Connecting to the server using Windows Command Prompt.

D- Linux and MacOS

To connect to the server from the Linux or Linux-like operating system:

- launch the Terminal that comes with the system. Then, connect to the server by typing the following and pressing enter:

  ssh yourusername@login-mri.research.mtu.edu
- The command line prompt on the server will show up as shown in Fig. 6.



```
(base) local~$> ssh hdismail@login-mri.research.mtu.edu
This system is the property of Michigan Technological University. Unauthorized access to or use of this system is s
trictly prohibited.
By continuing, you acknowledge your awareness of and compliance with the Acceptable Use of Information Technology R
esources policy of the University.
Unauthorized or improper use of this system may result in civil charges or criminal prosecution under the law and a
re subject to disciplinary action under University policies.
hdismail@login-mri.research.mtu.edu's password:
Notice: This system applies updates on the 15th of every month at 2 AM and will restart automatically if required.
Hostname: seq-login.sabu
System Type: Minimally Managed Workstation
Last login: Tue Apr  2 20:53:42 2024 from 141.219.106.243
[hdismail@seq-login ~]$
```

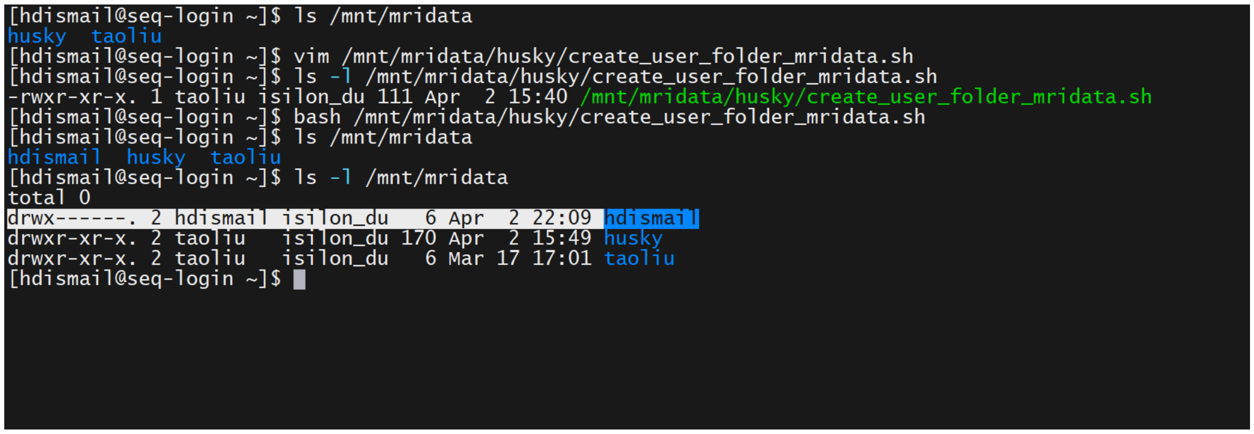
Fig. 6: Connecting to the server using a terminal on a Linux or MacOS computer.

2- Creating a working directory

When you log on, you will be in your home directory. Use the Linux command “pwd” to display the directory that you are currently in. It will be “/home/yourusername”. Please avoid storing your files in the home folder. Instead, you need to create a working directory with your username in “/mnt/mridata” or “/mnt/mridatafast”. To create a working directory in those two directories, please avoid using the mkdir command directly. Instead, you need to run the bash script in "/mnt/mridata/husky/create_user_folder_mridata.sh". From your default working directory run the following on the command line prompt:

```
bash /mnt/mridata/husky/create_user_folder_mridata.sh
```

A directory with your username will be created in "/mnt/mridata" and you will be the owner of that directory as shown in Fig. 7:



```
[hdismail@seq-login ~]$ ls /mnt/mridata
husky  taoliu
[hdismail@seq-login ~]$ vim /mnt/mridata/husky/create_user_folder_mridata.sh
[hdismail@seq-login ~]$ ls -l /mnt/mridata/husky/create_user_folder_mridata.sh
-rwxr-xr-x. 1 taoliu isilon_du 111 Apr  2 15:40 /mnt/mridata/husky/create_user_folder_mridata.sh
[hdismail@seq-login ~]$ bash /mnt/mridata/husky/create_user_folder_mridata.sh
[hdismail@seq-login ~]$ ls /mnt/mridata
hdismail  husky  taoliu
[hdismail@seq-login ~]$ ls -l /mnt/mridata
total 0
drwx------. 2 hdismail isilon_du   6 Apr  2 22:09 hdismail
drwxr-xr-x. 2 taoliu   isilon_du 170 Apr  2 15:49 husky
drwxr-xr-x. 2 taoliu   isilon_du   6 Mar 17 17:01 taoliu
[hdismail@seq-login ~]$
```

Fig. 7: Creating a working directory using a bash script "create_user_folder_mridata.sh".

Since your working directory will not be in your default directory when you log on, the absolute path of your working directory, where you store your files, is "/mnt/mridata/yourusername/". The following command may help you to navigate your working directory:

To display the content of your working directory, run:

```
ls -l /mnt/mridata/yourusername
```

To move to your working directory, run:

```
cd /mnt/mridata/yourusername
```

To move back to your default directory, run:

```
cd ~
```

Use the Linux command "pwd" to know where you are.

3- Using scratch directory

The scratch directory is available at "/scratch". This directory is typically used for temporary storage of data and intermediate files during computational tasks or jobs running on the cluster. Therefore, as the cluster user, use the scratch directory "/scratch" for temporary storage of data that is only needed during a job or computation. Using this directory for temporary storage during computing is efficient as the storage drive (where this directory is found) is optimized for high throughput and low latency, making it well-suited for handling input/output-intensive workloads in parallel computing environments. The scratch directory is often configured to automatically delete data after a job has been completed or after a certain period has elapsed. Therefore, use it only for temporary storage. For permanent storage, use your working directory.

To use the scratch directory, save the following bash script in a file say "myscratch.sh":

```
#!/bin/bash
if [ ! -d /scratch/$USER ]; then
    mkdir /scratch/$USER;
    chmod 700 /scratch/$USER;
fi
```

Then, run the script file using:

```
bash myscratch.sh
```

This will create a directory with your username in the scratch directory. So, the path to your scratch directory will be as follows:

```
/scratch/yourUserName
```

4. Copying files

You may need to copy files from or to your working directory. There are two ways to do this: (i) using the Linux command "scp" on the command line prompt or (ii) using WinSCP if you are using Windows. Remember that your working directory is a container for all your files. So, it is better to organize your files in directories inside this working directory.

i- Using "scp" command

The "scp" command can be used from any Linux and MacOS terminal and the Command Prompt of the late versions of Windows.

A- Copying files from your local computer to your working directory

The general syntax for "scp" command to copy a file from a local computer to your working directory is as follows:

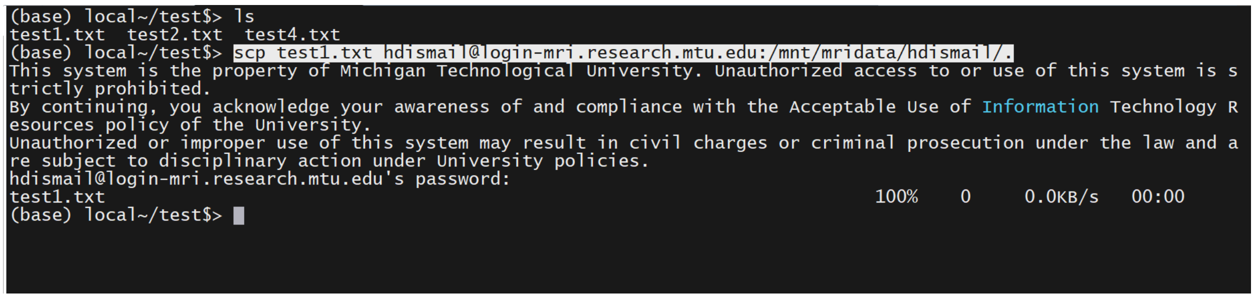
```
scp yourFileName yourusername@mri.research.mtu.edu:/mnt/mridata/yourUserName/.
```

You should provide the absolute or relative path of your file.

For example, to copy "test1.txt" from a local directory to my working directory, I will use this command:

```
scp test1.txt hdismail@login-mri.research.mtu.edu:/mnt/mridata/hdismail/.
```

When you enter, you will be asked to enter your password. Once you enter it and press enter, the single file will be copied to your remote working directory as shown in Fig. 8.



```
(base) local~/test$> ls
test1.txt  test2.txt  test4.txt
(base) local~/test$> scp test1.txt hdismail@login-mri.research.mtu.edu:/mnt/mridata/hdismail/.
This system is the property of Michigan Technological University. Unauthorized access to or use of this system is s
trictly prohibited.
By continuing, you acknowledge your awareness of and compliance with the Acceptable Use of Information Technology R
esources policy of the University.
Unauthorized or improper use of this system may result in civil charges or criminal prosecution under the law and a
re subject to disciplinary action under University policies.
hdismail@login-mri.research.mtu.edu's password:
test1.txt                                                        100%    0     0.0KB/s   00:00
(base) local~/test$> 
```

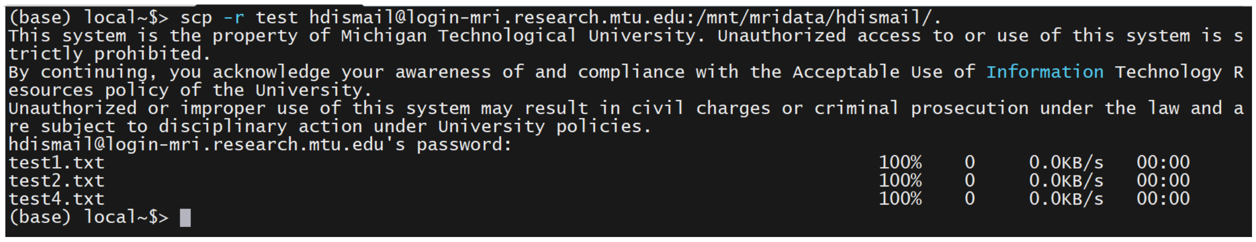
Fig. 8: Copying a single file from a local computer to a remote working directory.

For multiple files with some common characters, you can use wild cards (*, ?, []). For example, since I have the files “test1.txt, test2.txt, test3.txt, and test4.txt” I can use the following command:

```
scp test*.txt hdismail@login-mri.research.mtu.edu:/mnt/mridata/hdismail/.
```

To copy a directory, use “scp -r”. For example, to copy the directory “test” from my local computer to my working directory:

```
scp -r test hdismail@login-mri.research.mtu.edu:/mnt/mridata/hdismail/.
```

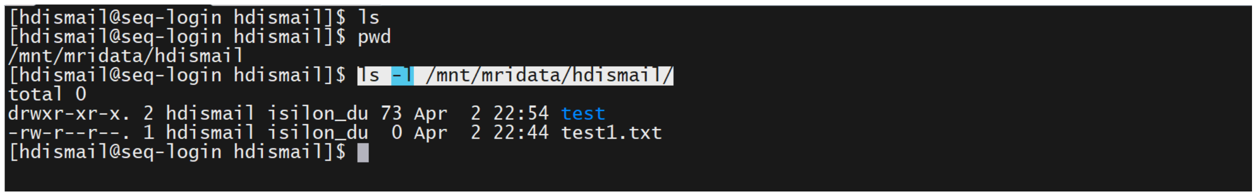


```
(base) local~$> scp -r test hdismail@login-mri.research.mtu.edu:/mnt/mridata/hdismail/.
This system is the property of Michigan Technological University. Unauthorized access to or use of this system is s
trictly prohibited.
By continuing, you acknowledge your awareness of and compliance with the Acceptable Use of Information Technology R
esources policy of the University.
Unauthorized or improper use of this system may result in civil charges or criminal prosecution under the law and a
re subject to disciplinary action under University policies.
hdismail@login-mri.research.mtu.edu's password:
test1.txt                                                    100%    0     0.0KB/s   00:00
test2.txt                                                    100%    0     0.0KB/s   00:00
test4.txt                                                    100%    0     0.0KB/s   00:00
(base) local~$>
```

Fig. 9: Copying a directory from a local computer to a remote working directory.

In your remote working directory, you can display the copied file and directory using as shown in Fig. 10.:

```
ls -l /mnt/mridata/yourusername/
```



```
[hdismail@seq-login hdismail]$ ls
[hdismail@seq-login hdismail]$ pwd
/mnt/mridata/hdismail
[hdismail@seq-login hdismail]$ ls -l /mnt/mridata/hdismail/
total 0
drwxr-xr-x. 2 hdismail isilon_du 73 Apr  2 22:54 test
-rw-r--r--. 1 hdismail isilon_du  0 Apr  2 22:44 test1.txt
[hdismail@seq-login hdismail]$
```

Fig. 10: Displaying the file and directory copied to the working directory.

A- Copying files from your working directory to a local computer

To copy a file from your working directory to a local computer, you need to open the terminal or Command prompt on your local computer

```
scp yourusername@mri.research.mtu.edu:/mnt/mridata/yourUserName/yourFileName filePath/.
```

If you are inside the directory on the local computer where you want to copy the file, you replace “filePath/.” with “.”:

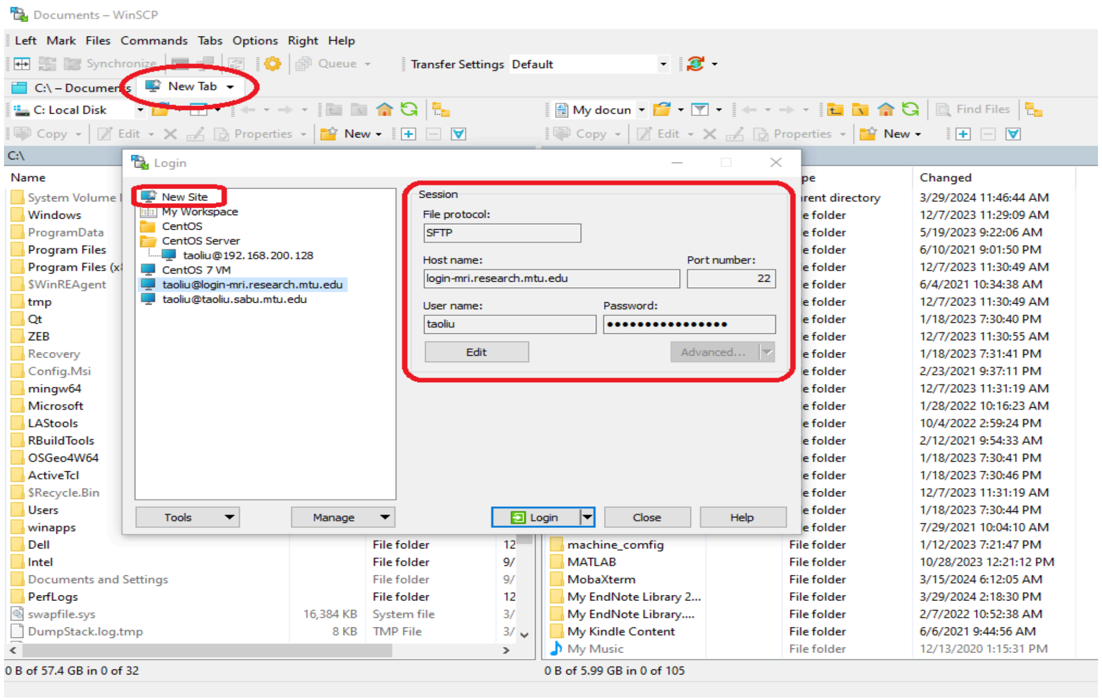
```
scp yourusername@mri.research.mtu.edu:/mnt/mridata/yourUserName/yourFileName .
```

Once you press enter, you will be asked for the password to copy the file.
For a group of files with common characters or a file extension, you can use the right wildcard [*,?, or []).
To copy a directory, use “scp -r”:

```
Scp yourusername@mri.research.mtu.edu:/mnt/mridata/yourUserName/yourDirName filePath/.
```

ii- Using “WinSCP”

WinSCP is an open-source software (for Windows) used to copy files/directories from or to the remote working directory. If you are using Windows, you can download this software from “https://winscp.net/eng/index.php”. WinSCP provides an easy way to transfer files from a local computer to a remote working directory or from a remote directory to a local computer by dragging the file or directory. Once you install it, launch it, and connect to the server as shown in Fig. 11. On the session dialog box, you will be asked to enter the file protocol, hostname, port number, username, and password. For the file protocol, you can either provide “scp” or “sftp” (pay attention that sftp protocol may be blocked by a firewall). The tree of files and directories are shown on both the local computer and the remote computer. Use the mouse to navigate to the right directory or file and drag it to its destination.



Login

New Site
My Workspace
CentOS
CentOS Server
taoliu@192.168.200.128
CentOS 7 VM
taoliu@login-mri.research.mtu.edu
taoliu@taoliu.sabu.mtu.edu

Session
File protocol: SFTP
Host name: login-mri.research.mtu.edu
Port number: 22
User name: taoliu
Password: ••••••••••••••••

Name	Size	Type
System Volume I		
Windows		
ProgramData		
Program Files		
Program Files (x		
$WinREAgent		
tmp		
Qt		
ZEB		
Recovery		
Config.Msi		
mingw64		
Microsoft		
LAStools		
RBuildTools		
OSGeo4W64		
ActiveTcl		
$Recycle.Bin		
Users		
winapps		
Dell		File folder
Intel		File folder
Documents and Settings		File folder
PerfLogs		File folder
swapfile.sys	16,384 KB	System file
DumpStack.log.tmp	8 KB	TMP File

0 B of 57.4 GB in 0 of 32

Name	Type	Changed
	rent directory	3/29/2024 11:46:44 AM
	e folder	12/7/2023 11:29:09 AM
	e folder	5/19/2023 9:22:06 AM
	e folder	6/10/2021 9:01:50 PM
	e folder	12/7/2023 11:30:49 AM
	e folder	6/4/2021 10:34:38 AM
	e folder	12/7/2023 11:30:49 AM
	e folder	1/18/2023 7:30:40 PM
	e folder	12/7/2023 11:30:55 AM
	e folder	1/18/2023 7:31:41 PM
	e folder	2/23/2021 9:37:11 PM
	e folder	12/7/2023 11:31:19 AM
	e folder	1/28/2022 10:16:23 AM
	e folder	10/4/2022 2:59:24 PM
	e folder	2/12/2021 9:54:33 AM
	e folder	1/18/2023 7:30:41 PM
	e folder	1/18/2023 7:30:46 PM
	e folder	12/7/2023 11:31:19 AM
	e folder	1/18/2023 7:30:44 PM
	e folder	7/29/2021 10:04:10 AM
machine_comfig	File folder	1/12/2023 7:21:47 PM
MATLAB	File folder	10/28/2023 12:21:12 PM
MobaXterm	File folder	3/15/2024 6:12:05 AM
My EndNote Library 2...	File folder	3/29/2024 2:18:30 PM
My EndNote Library....	File folder	2/7/2022 10:52:38 AM
My Kindle Content	File folder	6/6/2021 9:44:56 AM
My Music	File folder	12/13/2020 1:15:31 PM

0 B of 5.99 GB in 0 of 105

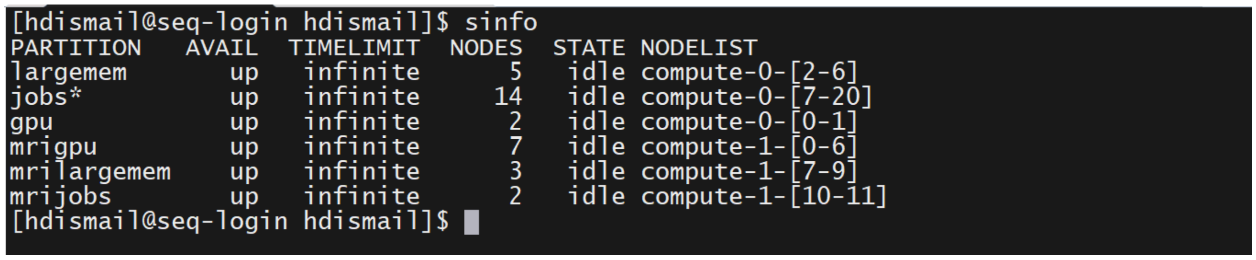
Fig. 11: Using WinSCP to transfer files and directories from or to a remote working directpry.

5. Using the cluster computational resources

A- Listing and loading the available models

As a computer cluster, the programs installed are managed by the SLURM (Simple Linux Utility for Resource Management). Therefore, the available programs on the servers must be loaded first before being used. When you use a loaded program, you can use it interactively like a standalone computer and this is not recommended as it may not be efficient for your computing or you can use it non-interactively by allocating the needed number of processors, memory, time, and then submitting your jobs to the queue, where it is executed based on priority and availability of the computational resources. Submitting jobs to the cluster queue will be discussed in detail. Here are some useful commands that you can use to get some information about the cluster.

- `sinfo`: the sinfo command in SLURM is used to display information about available compute resources in the cluster managed by SLURM. This command provides an overview of the nodes, their states, partitions, and other relevant details. Run “sinfo --help” to display the options used with this command. However, you can use this command simply by running “sinfo” as shown in Fig 12.

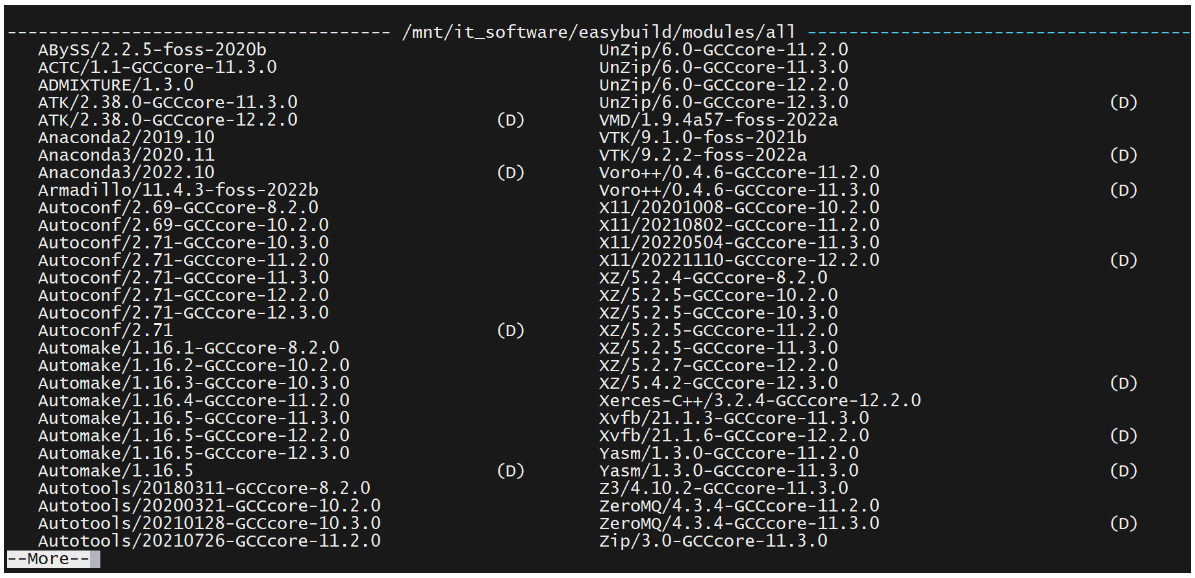


```
[hdismail@seq-login hdismail]$ sinfo
PARTITION    AVAIL  TIMELIMIT  NODES  STATE NODELIST
largemem        up   infinite      5   idle compute-0-[2-6]
jobs*           up   infinite     14   idle compute-0-[7-20]
gpu             up   infinite      2   idle compute-0-[0-1]
mrigpu          up   infinite      7   idle compute-1-[0-6]
mrilargemem     up   infinite      3   idle compute-1-[7-9]
mrijobs         up   infinite      2   idle compute-1-[10-11]
[hdismail@seq-login hdismail]$ 
```

Fig. 12: The output of the sinfo command. The figure shows the partitions, the number of nodes in each partition, and the distribution of the compute nodes.

- `module avail`: A shortcut for this command is “ml avail”. This command will output a list of available modules along with their versions as shown in Fig 13. These modules typically include software packages, libraries, compilers, and other tools that can be loaded into your environment to customize it for your needs. However, to list the modules on this server, you need to run the following commands:

```
module use /mnt/it_software/easybuild/modules/all
module avail
```



```
------------------------------------ /mnt/it_software/easybuild/modules/all -------------------------------------
   ABySS/2.2.5-foss-2020b                                UnZip/6.0-GCCcore-11.2.0
   ACTC/1.1-GCCcore-11.3.0                               UnZip/6.0-GCCcore-11.3.0
   ADMIXTURE/1.3.0                                       UnZip/6.0-GCCcore-12.2.0
   ATK/2.38.0-GCCcore-11.3.0                             UnZip/6.0-GCCcore-12.3.0                     (D)
   ATK/2.38.0-GCCcore-12.2.0                     (D)     VMD/1.9.4a57-foss-2022a
   Anaconda2/2019.10                                     VTK/9.1.0-foss-2021b
   Anaconda3/2020.11                                     VTK/9.2.2-foss-2022a                         (D)
   Anaconda3/2022.10                             (D)     Voro++/0.4.6-GCCcore-11.2.0
   Armadillo/11.4.3-foss-2022b                           Voro++/0.4.6-GCCcore-11.3.0                  (D)
   Autoconf/2.69-GCCcore-8.2.0                           X11/20201008-GCCcore-10.2.0
   Autoconf/2.69-GCCcore-10.2.0                          X11/20210802-GCCcore-11.2.0
   Autoconf/2.71-GCCcore-10.3.0                          X11/20220504-GCCcore-11.3.0
   Autoconf/2.71-GCCcore-11.2.0                          X11/20221110-GCCcore-12.2.0                  (D)
   Autoconf/2.71-GCCcore-11.3.0                          XZ/5.2.4-GCCcore-8.2.0
   Autoconf/2.71-GCCcore-12.2.0                          XZ/5.2.5-GCCcore-10.2.0
   Autoconf/2.71-GCCcore-12.3.0                          XZ/5.2.5-GCCcore-10.3.0
   Autoconf/2.71                                 (D)     XZ/5.2.5-GCCcore-11.2.0
   Automake/1.16.1-GCCcore-8.2.0                         XZ/5.2.5-GCCcore-11.3.0
   Automake/1.16.2-GCCcore-10.2.0                        XZ/5.2.7-GCCcore-12.2.0
   Automake/1.16.3-GCCcore-10.3.0                        XZ/5.4.2-GCCcore-12.3.0                      (D)
   Automake/1.16.4-GCCcore-11.2.0                        Xerces-C++/3.2.4-GCCcore-12.2.0
   Automake/1.16.5-GCCcore-11.3.0                        Xvfb/21.1.3-GCCcore-11.3.0
   Automake/1.16.5-GCCcore-12.2.0                        Xvfb/21.1.6-GCCcore-12.2.0                   (D)
   Automake/1.16.5-GCCcore-12.3.0                        Yasm/1.3.0-GCCcore-11.2.0
   Automake/1.16.5                               (D)     Yasm/1.3.0-GCCcore-11.3.0                    (D)
   Autotools/20180311-GCCcore-8.2.0                      Z3/4.10.2-GCCcore-11.3.0
   Autotools/20200321-GCCcore-10.2.0                     ZeroMQ/4.3.4-GCCcore-11.2.0
   Autotools/20210128-GCCcore-10.3.0                     ZeroMQ/4.3.4-GCCcore-11.3.0                  (D)
   Autotools/20210726-GCCcore-11.2.0                     Zip/3.0-GCCcore-11.3.0
--More--
```

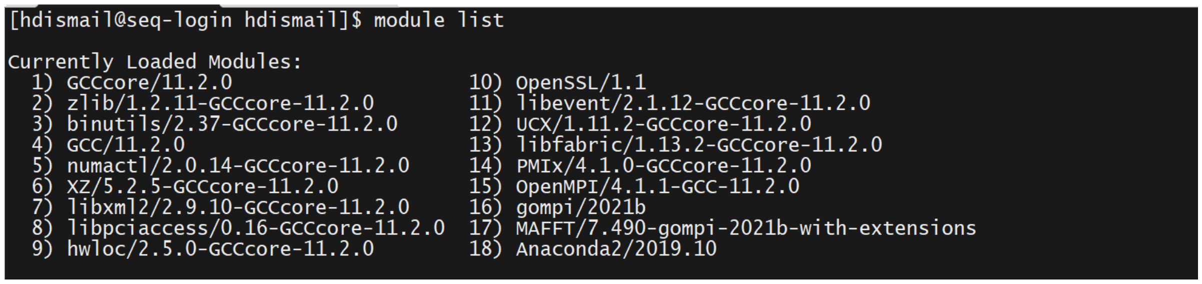
Fig. 13: Displaying available modules managed by SLURM using “module avail” command.
Press the enter key to scroll down to display more modules. Press the “q” key to exit.

- `module load`: This command would load a module, making it and its associated tools and libraries available for your use within the current environment. To load a module select any of the modules shown above e.g. Anaconda2/2019.10 and the command as follows:

```
module load Anaconda2/2019.10
```

- `module list`: This command will display the list of the currently loaded modules as shown in Fig. 14.

  `module list`



```
[hdismail@seq-login hdismail]$ module list

Currently Loaded Modules:
  1) GCCcore/11.2.0                      10) OpenSSL/1.1
  2) zlib/1.2.11-GCCcore-11.2.0          11) libevent/2.1.12-GCCcore-11.2.0
  3) binutils/2.37-GCCcore-11.2.0        12) UCX/1.11.2-GCCcore-11.2.0
  4) GCC/11.2.0                          13) libfabric/1.13.2-GCCcore-11.2.0
  5) numactl/2.0.14-GCCcore-11.2.0       14) PMIx/4.1.0-GCCcore-11.2.0
  6) XZ/5.2.5-GCCcore-11.2.0             15) OpenMPI/4.1.1-GCC-11.2.0
  7) libxml2/2.9.10-GCCcore-11.2.0       16) gompi/2021b
  8) libpciaccess/0.16-GCCcore-11.2.0    17) MAFFT/7.490-gompi-2021b-with-extensions
  9) hwloc/2.5.0-GCCcore-11.2.0          18) Anaconda2/2019.10
```

Fig. 14: Displaying the loaded list of modules by using the "module list" command.

B- Submitting your job to the SLURM queue

The programs managed by SLURM are usually unloaded until the user loads them using the above command "module load". As a user, **you must submit your jobs to the queue**, since other users may be using computational resources like processors and memory. Therefore, submitting your job to the queue allows the jobs to be executed based on priority and availability of the resources. A bash script will be used to choose the number of processors, memory, and time required to execute your computing job. Use the resources wisely namely do not allocate resources more than what your b needs. That may affect the job executions of other users waiting in the queue. In the following, we will show you how to submit your job to the SLURM queue. Assume that you want to run a Python program named "mypyprog.py", which contains the following script that sums the number from 0 to 10000000 and writes the sum to a text file named "output.txt":

```
outFile = open("output.txt", "w")
sum = 0
for i in range(10000000):
    sum = sum + i
outFile.write(str(sum))
outFile.close()
```

To submit this script to the queue to be executed, you need to create a bash script in a file e.g. "myslurm.sh". The script in the file can be as follows:

```
#!/bin/bash
#SBATCH --job-name=myjob            # Job name
#SBATCH --nodes=1                   # Number of nodes
#SBATCH --ntasks-per-node=1         # Number of tasks per node
#SBATCH --cpus-per-task=1           # Number of CPU cores per task
#SBATCH --mem=4G                    # Memory per CPU core
#SBATCH --time=00:10:00             # Time limit hrs:min:sec
```

```
#SBATCH --partition=mrijobs          # Partition (queue) name
# Load any necessary modules
module use /mnt/it_software/easybuild/modules/all
module load Python/3.9.5-GCCcore-10.3.0
# Change directory to your working directory
cd /mnt/mridata/hdismail
# Execute your program or command
python mypyprog.py
```

In the bash script, the SLURM command begins with "#SBATCH" followed by a parameter and its value. The value of the parameters are set by the users. Here are the descriptions of those parameters:
`--job-name=myjob`: The job name, which is an identifier and can be any name chosen by the user to describe the nature of the job submitted.
`--nodes=1`: The number of nodes needed for computing. The number of nodes is determined by the workload. A single node may be enough for many users, however, for heavy scientific computing, in which we use Message Passing Interface (MPI) or OpenMP, we may need to allocate multiple nodes for parallel computing. Tasks can also be parallelized using the loops (e.g. for loop) in multiprocessing programming. The user may need to learn how to benefit from the computer cluster by running multiple tasks simultaneously. This is usually done through programming.
`--ntasks-per-node=1`: This parameter determines the number of tasks. This parameter is particularly important in multithreading programming such as the use of "pthread" and OpenMP library in C/C++, and multithreading in Python. Again, choosing the right number of tasks requires programming knowledge.
`--cpus-per-task=1`: The number of CPUs for each task. This parameter is important for multithreading as well.
`--mem=4G`: This parameter is for memory allocation. Choose the memory amount that is sufficient for your computing task. This can be determined by the size of data and the data type.
`#SBATCH --time=00:10:00`: The time policy is not enforced in this computer cluster (No time limit). In case the time policy is enforced, the time chosen will be part of the resource allocation (i.e. the job will be killed if it takes more than the requested time).
`--partition=jobs`: The partition name where the compute nodes will be used (the jobs will be sent to the queue of this partition). You can run the "sinfo" command to list the partition. Each partition may include compute nodes for specific tasks. For example, if you need to use the GPU you can choose the partition with GPU compute nodes.

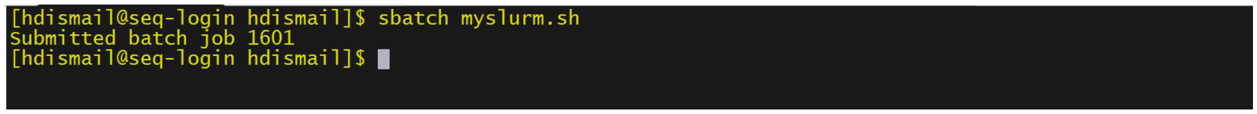
The cluster has six partitions as shown in Table 1. Each partition is dedicated for a specific purpose as shown in the description table.

Table 1: The descriptions of the partitions on the cluster.

Partition	# nodes	Node list	Description
mrigpu	7	compute-1-[0-6]	For programs that use GPUs
mrilargemem	3	compute-1-[7-9]	
mrijobs	2	compute-1-[10-11]	

`module load Python/3.9.5-GCCcore-10.3.0`: You have to load the modules that you will use. Use the "module avail" commands to list the available modules as shown above. Here, we loaded the show python version to be used.
`cd /mnt/mridata/hdismail`: It is recommended to use your working directory to store your files as discussed above. This command will change the directory to your working directory if it is not already there.
`python mypyprog.py`: The line will execute the Python script in the "`mypyprog.py`". In this line, type the command as you type it on the command line.

So far, we have the above bash script saved in "myslurm.sh". To submit the job to the queue of the chosen partition use "sbatch" command as follows:
```
sbatch myslurm.sh
```
When you run this on the command line, your job will be placed in the queue and a batch job ID number will be generated as shown in Fig. 15.



```
[hdismail@seq-login hdismail]$ sbatch myslurm.sh
Submitted batch job 1601
[hdismail@seq-login hdismail]$ 
```

Fig. 15: Submitting a job to the queue using "sbatch" command.

The time taken to execute your job depends on the number of jobs waiting in the queue and the time complexity of your tasks. You can run "`squeue`" command to check the status of your job. You can also use "`skill` or `scancel`" with the job ID to kill or cancel the job if you want to cancel its execution.

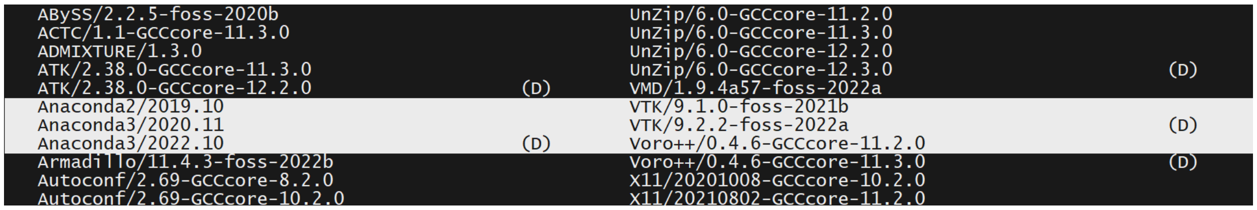
When the job is completed, a file will be written to the directory where you run the script. By default, the file name is "slurm-ID.out", ID is the batch job ID generated upon job submission. It is good practice to open this file and study its content to see if your program has been executed without error.

C- Submitting your job to the GPU compute nodes
You may need to benefit from the computing power of the GPUs in your computing. The GPUs can be used in different programs like Python (PyTorch and TensorFlow) for deep learning and tensor computing or C/C++ for CUDA programming. In the case of Python, it is straightforward to use a GPU if the GPU driver and API are configured by the administrator. You only need to

choose the partition where the GPU nodes are found. For CUDA programming, make sure that the NVIDIA toolkit is installed by the administrator. After creating your program, you need to compile it interactively using the nvcc compiler and when you want to run it, you can run it using the above batch script, however, you need to choose a partition with GPUs.

6. Creating an anaconda environment

Anaconda is a popular open-source distribution of the Python and R programming languages for scientific computing, data science, and machine learning tasks. It aims to simplify package management and deployment by providing a comprehensive set of pre-installed libraries and tools commonly used in these fields. Anaconda allows users to create isolated environments, each with its own set of packages, dependencies, and Python versions. This makes it easier to manage different projects, each with specific requirements.
When you have a specific project using Python with specific package requirements, you can create an Anaconda environment for that project. First, you need to load the right Anaconda module. Use the following commands to mount and display the list of the available modules; use the enter key to scroll down until you find the right Anaconda module (see Fig. 16).



```
ABySS/2.2.5-foss-2020b                                UnZip/6.0-GCCcore-11.2.0
ACTC/1.1-GCCcore-11.3.0                               UnZip/6.0-GCCcore-11.3.0
ADMIXTURE/1.3.0                                       UnZip/6.0-GCCcore-12.2.0
ATK/2.38.0-GCCcore-11.3.0                             UnZip/6.0-GCCcore-12.3.0                 (D)
ATK/2.38.0-GCCcore-12.2.0                      (D)    VMD/1.9.4a57-foss-2022a
Anaconda2/2019.10                                     VTK/9.1.0-foss-2021b
Anaconda3/2020.11                                     VTK/9.2.2-foss-2022a                     (D)
Anaconda3/2022.10                              (D)    Voro++/0.4.6-GCCcore-11.2.0
Armadillo/11.4.3-foss-2022b                           Voro++/0.4.6-GCCcore-11.3.0              (D)
Autoconf/2.69-GCCcore-8.2.0                           X11/20201008-GCCcore-10.2.0
Autoconf/2.69-GCCcore-10.2.0                          X11/20210802-GCCcore-11.2.0
```

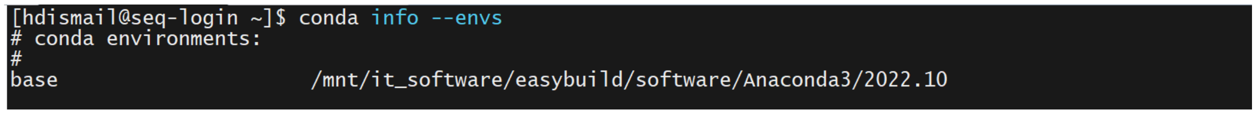
Fig. 16: Finding the right Anaconda module using "module avail".

As shown in Fig. 16, there are three versions of Anaconda modules. You may decide to use the latest version in the list, which is "Anaconda3/2022.10". Load this module with the following command:

```
module load Anaconda3/2022.10
```

Before you create a new environment, you may need to display the list of the environments created before, using the following commands:

```
conda info --envs
```



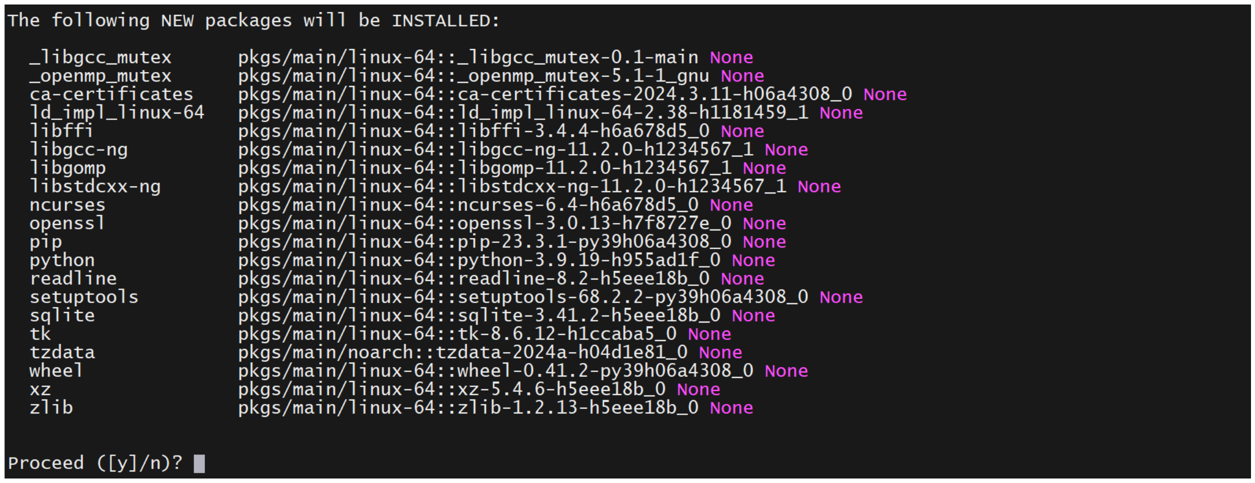
```
[hdismail@seq-login ~]$ conda info --envs
# conda environments:
#
base                     /mnt/it_software/easybuild/software/Anaconda3/2022.10
```

Fig. 17: Displaying the list of the Anaconda environments.

As shown in Fig. 17, there is only the default environment, which is "base".
Assume that you want to create a new Anaconda environment named "bio" for a new bioinformatics project that requires specific Python packages. Use the following command to create that new environment with a specific Python version:

```
conda create --name bio python=3.9
```

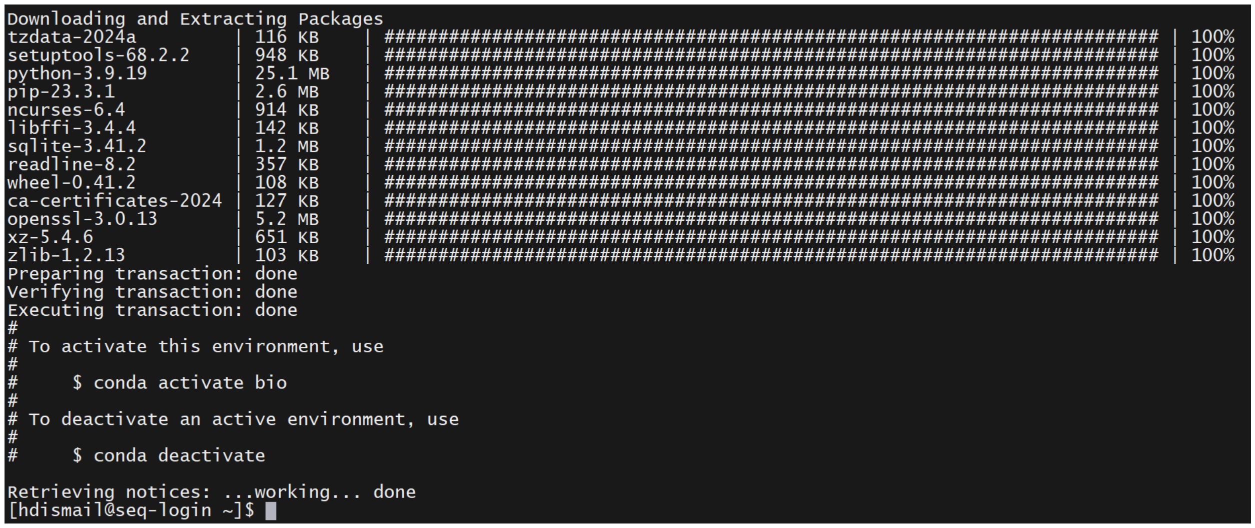
You may be asked whether you want to proceed or not as shown in Fig. 18. Type "y" and press enter to start downloading and installing the essential packages for the new environment.



```
The following NEW packages will be INSTALLED:

  _libgcc_mutex      pkgs/main/linux-64::_libgcc_mutex-0.1-main None
  _openmp_mutex      pkgs/main/linux-64::_openmp_mutex-5.1-1_gnu None
  ca-certificates    pkgs/main/linux-64::ca-certificates-2024.3.11-h06a4308_0 None
  ld_impl_linux-64   pkgs/main/linux-64::ld_impl_linux-64-2.38-h1181459_1 None
  libffi             pkgs/main/linux-64::libffi-3.4.4-h6a678d5_0 None
  libgcc-ng          pkgs/main/linux-64::libgcc-ng-11.2.0-h1234567_1 None
  libgomp            pkgs/main/linux-64::libgomp-11.2.0-h1234567_1 None
  libstdcxx-ng       pkgs/main/linux-64::libstdcxx-ng-11.2.0-h1234567_1 None
  ncurses            pkgs/main/linux-64::ncurses-6.4-h6a678d5_0 None
  openssl            pkgs/main/linux-64::openssl-3.0.13-h7f8727e_0 None
  pip                pkgs/main/linux-64::pip-23.3.1-py39h06a4308_0 None
  python             pkgs/main/linux-64::python-3.9.19-h955ad1f_0 None
  readline           pkgs/main/linux-64::readline-8.2-h5eee18b_0 None
  setuptools         pkgs/main/linux-64::setuptools-68.2.2-py39h06a4308_0 None
  sqlite             pkgs/main/linux-64::sqlite-3.41.2-h5eee18b_0 None
  tk                 pkgs/main/linux-64::tk-8.6.12-h1ccaba5_0 None
  tzdata             pkgs/main/noarch::tzdata-2024a-h04d1e81_0 None
  wheel              pkgs/main/linux-64::wheel-0.41.2-py39h06a4308_0 None
  xz                 pkgs/main/linux-64::xz-5.4.6-h5eee18b_0 None
  zlib               pkgs/main/linux-64::zlib-1.2.13-h5eee18b_0 None


Proceed ([y]/n)?
```

Fig. 18: Creating a new Anaconda environment.

The new environment will be created, when the screen shows what is displayed in Fig. 19.

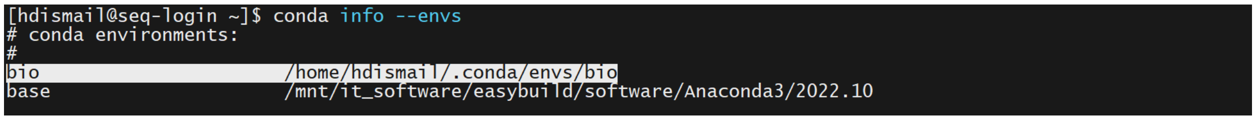


```
Downloading and Extracting Packages
tzdata-2024a         | 116 KB    | ######################################################################### | 100%
setuptools-68.2.2    | 948 KB    | ######################################################################### | 100%
python-3.9.19        | 25.1 MB   | ######################################################################### | 100%
pip-23.3.1           | 2.6 MB    | ######################################################################### | 100%
ncurses-6.4          | 914 KB    | ######################################################################### | 100%
libffi-3.4.4         | 142 KB    | ######################################################################### | 100%
sqlite-3.41.2        | 1.2 MB    | ######################################################################### | 100%
readline-8.2         | 357 KB    | ######################################################################### | 100%
wheel-0.41.2         | 108 KB    | ######################################################################### | 100%
ca-certificates-2024 | 127 KB    | ######################################################################### | 100%
openssl-3.0.13       | 5.2 MB    | ######################################################################### | 100%
xz-5.4.6             | 651 KB    | ######################################################################### | 100%
zlib-1.2.13          | 103 KB    | ######################################################################### | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate bio
#
# To deactivate an active environment, use
#
#     $ conda deactivate

Retrieving notices: ...working... done
[hdismail@seq-login ~]$
```

Fig. 19: The screen shows that the new environment has been created and it can be activated using the "`conda activate bio`" command.

You display the list of the Anaconda environments to be sure that your environment has been created as shown in Fig. 20:

```
conda info --envs
```



```
[hdismail@seq-login ~]$ conda info --envs
# conda environments:
#
bio                      /home/hdismail/.conda/envs/bio
base                     /mnt/it_software/easybuild/software/Anaconda3/2022.10
```

Fig. 20: Displaying the Anaconda environment.

i-Installing new Python package on your Anaconda environment

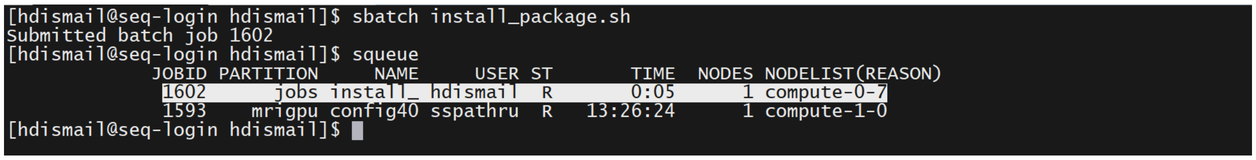
You may need to install some packages to your new environment. Package installation in an Anaconda environment on a cluster is different from that on a standalone computer. Here, you have an SLURM script in a bash file, say “install_package.sh”. Assume that you want to install the BioPython package on your “bio” environment, so you can use this script:

```
#!/bin/bash
#SBATCH --job-name=install_packages
#SBATCH --output=install_packages.log
#SBATCH --partition=jobs
#SBATCH --nodes=1
#SBATCH --ntasks-per-node=1
#SBATCH --cpus-per-task=1
#SBATCH --time=1:00:00
# Load Anaconda
module use /mnt/it_software/easybuild/modules/all
module load Anaconda3/2022.10
# Activate Conda Environment
conda activate bio
# Install Package
conda install anaconda::biopython
```

Then, run the script using:

```
sbatch install_package.sh
```

If you use the “squeue” command, you can check the job name and id on the queue as shown in Fig 21.



```
[hdismail@seq-login hdismail]$ sbatch install_package.sh
Submitted batch job 1602
[hdismail@seq-login hdismail]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
              1602      jobs install_ hdismail  R       0:05      1 compute-0-7
              1593    mrigpu config40 sspathru  R   13:26:24      1 compute-1-0
[hdismail@seq-login hdismail]$
```

Once package installation is complete, check the log file “install_packages.log” as named above to check if the installation has been completed successfully.

i-Using Anaconda environment on the cluster

Executing a Python program on an Anaconda environment on the computer cluster is not like executing it on a standalone machine. Use the SLURM scrip discussed above under “**Submitting your job to the SLURM queue**”. However, you should load Anaconda, activate the environment, and then run the program.

For example, you can save this Python script in a file e.g. mypyprog2.py”:

```
import os
outFile = open("output.txt", "w")
sum = 0
for i in range(10000000):
    sum = sum + i
```

```
cmd = "which python"
os.system(cmd)
outFile.write(str(sum))
outFile.close()
```

The SLURM script (e.g. myslurm2.sh) can be as follows:

```
#!/bin/bash
#SBATCH --job-name=myjob            # Job name
#SBATCH --nodes=1                   # Number of nodes
#SBATCH --ntasks-per-node=1         # Number of tasks per node
#SBATCH --cpus-per-task=1           # Number of CPU cores per task
#SBATCH --mem=4G                    # Memory per CPU core
#SBATCH --time=00:10:00             # Time limit hrs:min:sec
#SBATCH --partition=jobs            # Partition (queue) name
# Load any necessary modules
module use /mnt/it_software/easybuild/modules/all
module load Anaconda3/2022.10
# Activate Conda Environment
source activate bio
# Change directory to your working directory
cd /mnt/mridata/hdismail
# Execute your program or command
python mypyprog2.py
```

Use the “sbatch” command to submit the job to the queue:

```
sbatch myslurm2.sh
```

Check the “slurm-ID.out” (where ID is the job ID) to be sure that the Python used is one installed on your Anaconda environment.

Acknowledgments

We would like to acknowledge Jarrod Karrau from the IT department for his help with the management of the cluster. Judy Long, Revanth Mattey, and Shashank Pathrudkar also contributed to making this user guide possible by testing the Blizzard system during its development stage.

Funding

Please don’t forget to acknowledge the NSF MRI grant in your publications and grants.
The GPU cluster was funded by NSF Major Research Instrumentation (MRI) Grant Award # 221734, MRI: Acquisition of a GPU-accelerated cluster for research, training, and outreach.