BHARATHIAR UNIVERSITY
COIMBATORE - 641046

HIGH PERFORMANCE COMPUTATIONAL FACILITY

USER MANUAL

Prepared by

Dr. R. Rajeswari
Assistant Professor
Department of Computer Applications
Bharathiar University
Coimbatore - 641046

AUGUST 2018

TABLE OF CONTENTS

Preface

This manual is intended for Faculty Members, Research Scholars and Students of Bharathiar University who will use the computing facility provided by the High Performance Computational Facility, Bharathiar University. This manual gives an introduction to the computing hardware, applications, operating system, how to connect to the computers, and how to run jobs. This High Performance Computational Facility has been equipped with advanced technologies to perform the high-end computations for the academic community. This facility will be able to address and catalyze the research and promises to be a boon to the academic community. Additionally, it will help in creating a workforce that is aware of High Performance Computing skills (capacity building) and promote research and teaching by integrating leading-edge emerging technologies.

Prof. Dr. T. DEVI
Principal Co-ordinator, High Performance Computational Facility
Professor and Head, Department of Computer Applications
Bharathiar University

Chapter 1: Introduction

Bharathiar University is an institution which is known for its quality and diversity of research with over 249 faculty members and 1500 research scholars and 3176 post-graduate students in University Departments and 4500 research scholars and 70000 post-graduate students in affiliated colleges. The University provides ample opportunities for research-minded students to hone their research skills and participate actively in pioneering research studies through M. Phil and Ph. D programmes. The faculty of science, humanities and management departments and research scholars do active research in frontier areas, which results in highly acclaimed publications in International and National Journals and patents. A large number of sponsored research projects are funded by funding agencies such as UGC, DST, DRDO, ICSSR, DBT and ICMR. Technology is rapidly changing and it is very essential for academic Institutions to have state-of-art equipments to carry out the research and reach greater heights at Global level. Research is being carried out on areas such as knowledge engineering, data analytics, bio-inspired computing, image processing, medical imaging, computational intelligence, security, statistical quality control, statistical inference, financial Services, marketing management, human genetics, toxicology, molecular therapeutics, plant genetic engineering, molecular sensors, organic & polymer based nanostructures, thin films, solar & fuel cells, low dimensional materials, medical physics, radiation physics, computational proteomics/ genomics, computational biology, bioinformatics, mathematical modeling, theoretical and computational fluid dynamics, organic chemistry, synthetic organic chemistry, environmental & health economics, information society, new media, computer education, technology in education and applied linguistics.

Almost all the research areas in which the faculty members, scholars and students of Bharathiar University are concentrating are associated with computational needs including data storage, analysis, simulations, modeling, software development, visualization of complex data and rapid mathematical calculations. Hence, there is a need for high performance computing cluster that has the capacity to handle and analyze massive amounts of data at high speed. The powerful computer systems, massive data storage and archives, visualization engines and special-purpose hardware provided by high performance computing cluster would support in cutting-edge research. Tasks that can take months using computers can be done in days or even

minutes. Computer-based simulations can be used in modeling of increasingly complex systems such as global environment, share market, social media or a living cell, which often incorporate many length and time scales and require the integration of scientific expertise over traditional disciple boundaries. Massive acquisition of experimental data such as genome, protein sequences, big data, or data from sensors which continuously monitor the environment requires the storing and sophisticated, computer-based analyses of the enormous datasets. Some experiments which are complex to be carried out experimentally can be used to design and carry out experiments in a virtual environment. The increasing capacity of high performance computing resources enables scientists to carry out their studies with dramatically increased accuracy. It also allows total new problems with unprecedented complexity to be adopted into research agendas. The computational approach can increasingly differentiate between the qualitatively correct and incorrect predictions for complicated systems. The high performance computing cluster (HPCC) would support the increasingly diverse research computing needs on campus and its affiliating colleges including data storage, analysis, simulations, modeling, software development and visualization of complex data.

1.1 Hardware Overview

This section provides the hardware configuration details of the High Performance Computational Facility.

1. Master Node – 1 No

S.No	Feature	Description
1	Form Factor	2U Rack mountable
2	Processor	2 x Intel Xeon E5-2683 v4. 16 Core and 40MB Cache
3	Memory	Should be configured with 128 GB RAM Memory should be scalable to double the capacity configured
4	Storage	NLSAS for a storage of 32TB (i.e 4 x 8 TB)
5	RAID Controller	RAID Controller with min 2 GB Cache and should support RAID 0,1,5,6,10

2 Compute Nodes –10 Nos GPU

S.No	Feature	Description
1	Form Factor	1U Rack mountable
2	Processor	2 x Intel Xeon E5-2683 v4. 16 Core and 40MB Cache
3	Memory	Should be configured with 128 GB RAM Memory should be scalable to double the capacity configured
4	Internal HDD	2 x 2 TB SATA HDD drives
5	GPU	1 x NVIDIA P100 12GB Modules

3 Compute Nodes –12 Nos CPU

S.No	Feature	Description
1	Form Factor	1U Rack mountable
2	Processor	2 x Intel Xeon E5-2683 v4. 16 Core and 40MB Cache
3	Memory	Should be configured with 128 GB RAM Memory should be scalable to double the capacity configured
4	Internal HDD	2 x 2 TB SATA HDD drives

1.2 Software Overview

Operating System: Latest Optimized Open source Linux OS with support for High Performance Computing

Cluster Management Software: Open source ROCKS optimized for HPC

Compilers: Open source ForTran 77/90/95, Parallel ForTran for main system, ANSI C, C++ with corresponding Parallel language for the parallel computing, Perl, Python, Parallel compilers for MPI & Open MP, symbolic debugging tools with source line display and source language program formatter where appropriate.

Libraries: Open Source Standard mathematical, statistical, and numerical function libraries, Standard scientific functions libraries, Parallel Libraries for MPI & Open MP, MKL, Latest X11 with MOTIF; Open source libraries LAPACK, BLAS, FFTW etc

1.3 HPC Applications

Application
GROMACS, R, Gaussian09, Molpro, VASP, Quantum Espresso, MySQL, SCILAB, Python

Chapter 2: Accessing High Performance Computational Facility

Access to the supercomputers is available through encrypted connections, such as ssh. The ssh program allows you to open a text console session on a remote computer. Thus ssh is essentially an encrypted version of telnet. When a connection with a computer is made, user name and password must be given which can be done using a graphical ssh program under windows, such as PuTTY.

2.1 Login from Windows

For Windows operating system, you need to additionally use a ssh utility such as SSH Secure Shell Client. You can download the free version of the SSH Secure Shell Cleint utility from the https://www.wm.edu/offices/it/services/software/licensedsoftware/webeditingsftp/sshsecureshell/index.php site.

Installation Notes

1. Download SSH Secure Shell Client from the above website.
2. Download the exe file of SSH Secure Shell Client (sshsecureshellclient-3.2.9.exe). Save it to an easily accessible place (your Windows desktop is a good choice). It is a single executable file that needs to be run to use SSH Secure Shell Client.
3. Start the installation by double-clicking on sshsecureshellclient-3.2.9.exe in Windows Explorer.
4. When the installation is complete, double click on the Desktop Icon to start the program. NOTE: The installation places 2 SSH icons on your desktop (or location where you saved it). Double click the yellow folder titled SSH Secure File Transfer.
5. Once you reach the SSH window click Quick Connect as given in Figure 1.

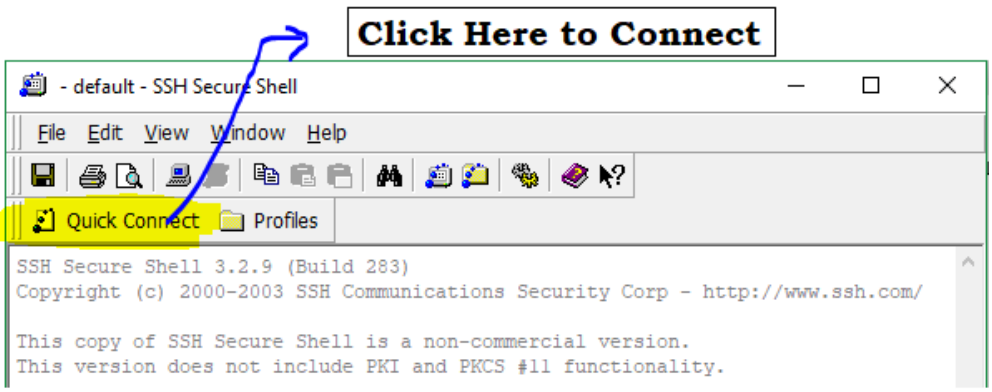

Figure 1: Connecting to host using SSH Secure Shell Client

6. Enter the host name (host name is 172.16.144.35), your username (in order to obtain your username and password please send a mail to hpcf@buc.edu.in) and port number (port number is 22) as given in figure 2.
7. Authentication Method is Password.
8. Click Connect.

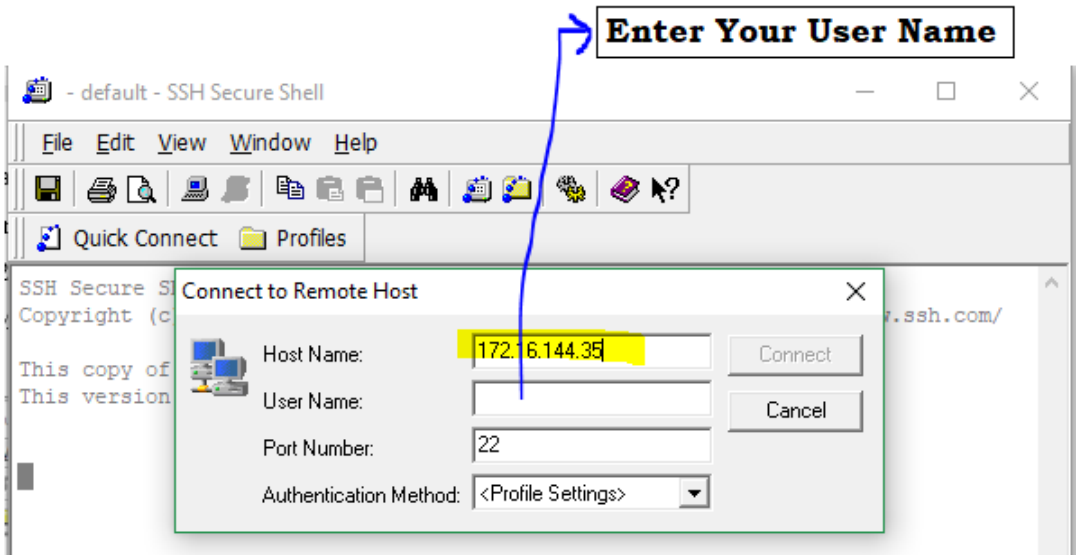

Figure 2: Connect to remote Host using Host Name, User Name, port number and Authentication Method

9. When prompted with pop up windows, click OK, then enter password as shown in figure 3.

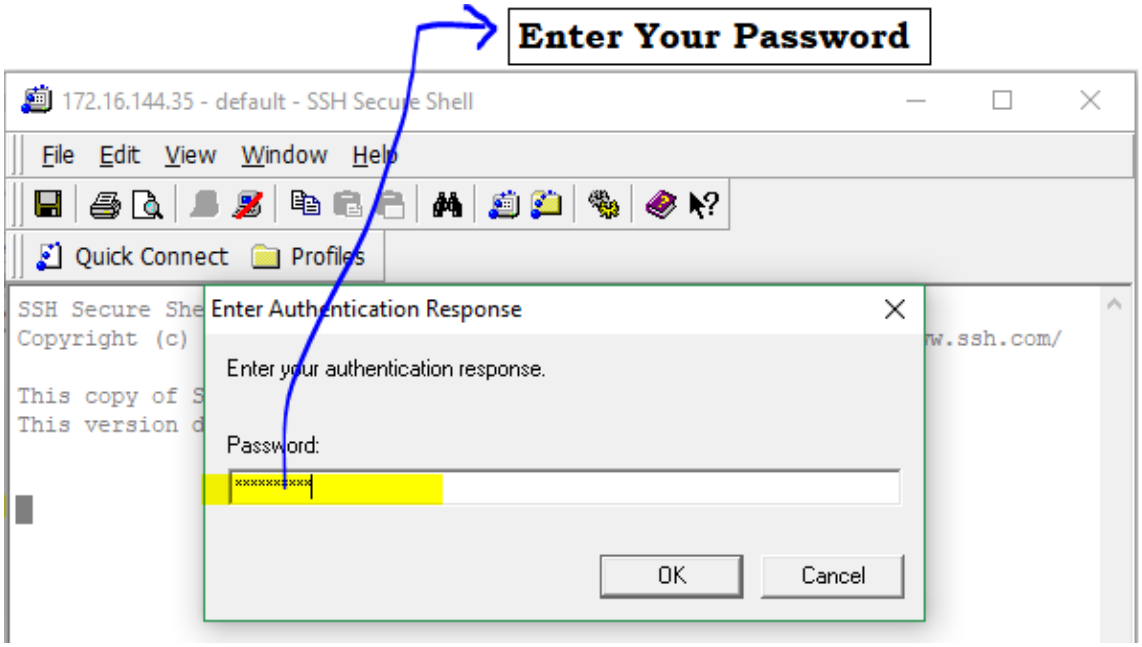

Figure 3: Authentication Response

10. On successful login the command prompt of SSH secure shell client appears with a message ‘connected to 172.16.144.35’ as shown in figure 4.

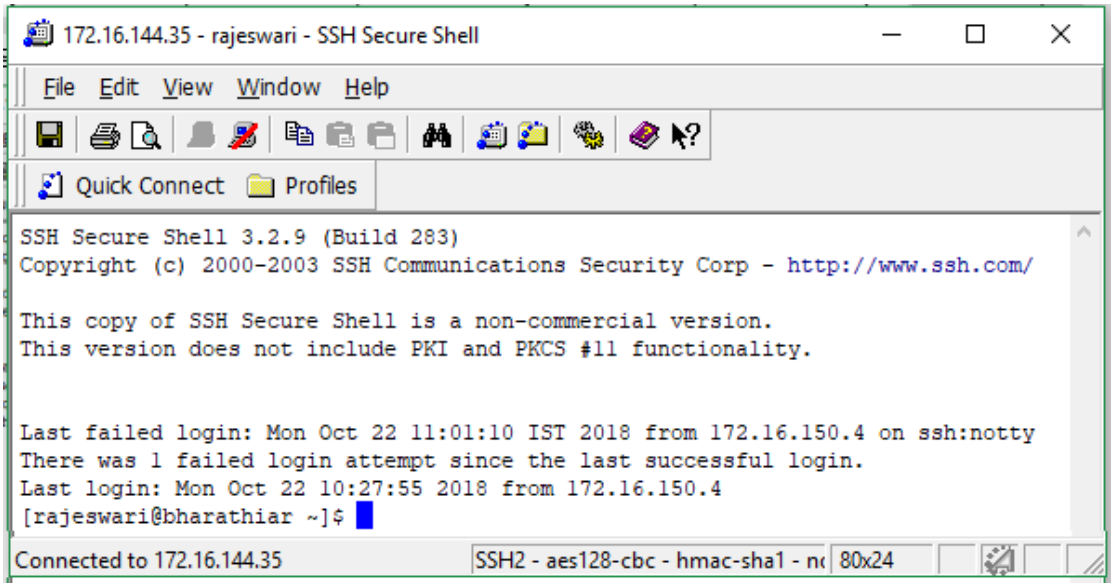

Figure 4: Command Prompt after successful login

2.3 Job submission in HPC

While it is possible to submit programs directly to HPCF it is generally preferable to create a job script. The job script is just like any other script that contains commands that the user would like to run. The difference is that instead of a user running the script, it is submitted to the queue scheduling system of HPCF which then runs the script.

Consider a very simple job script shown in figure 5.

```
#!/bin/sh
# This is a very simple example job script
/Users/usr1/my_program
```

Figure 5: simple job script ‘sample.sh’

The ‘sample.sh’ job script can be submitted to the job scheduling system of HPCF using ‘qsub’ command as shown in figure 6.

```
qsub sample.sh
```

Figure 6: usage of ‘qsub’ command

After submitting a job, or a set of jobs, with qsub, we can check on the job(s) with the command qstat. The qstat command returns the status of jobs in the queue. The job status abbreviations, returned by qstat, correspond to

qw	pending (waiting in queue)
r	running
t	in transfer (typically from qw to r)
Eqw	error and waiting in queue (for ever)
d	marked for deletion

2.4 Accessing Python

Python is a popular programming language. It was created in 1991 by Guido van Rossum. It is used for web development (server-side), software development, mathematics and system scripting. Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc). Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick. Python can be treated in a procedural way, an object-orientated way or a functional way.

Executing python scripts in a serial manner

Python files can be created using editors such as vi or nano as shown in figure 7.

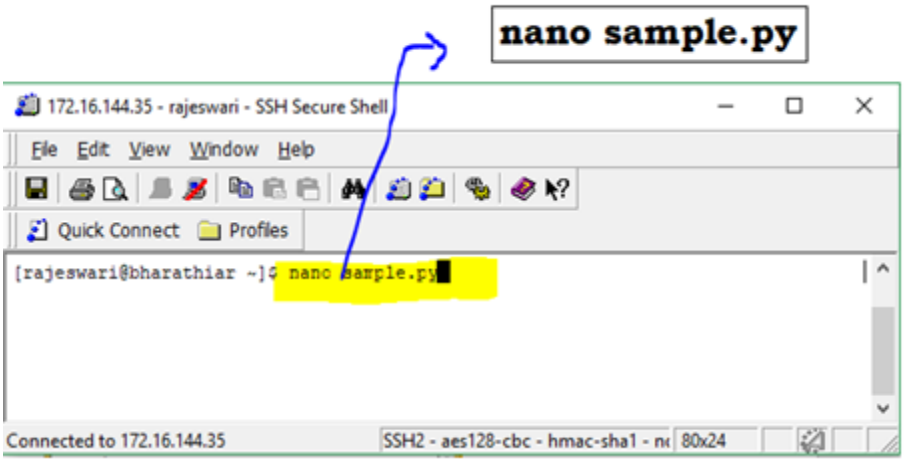

Figure 7: creating ‘sample.py’ python file using nano editor

The created file can be executed in a serial manner using the python command as shown in figure 8.

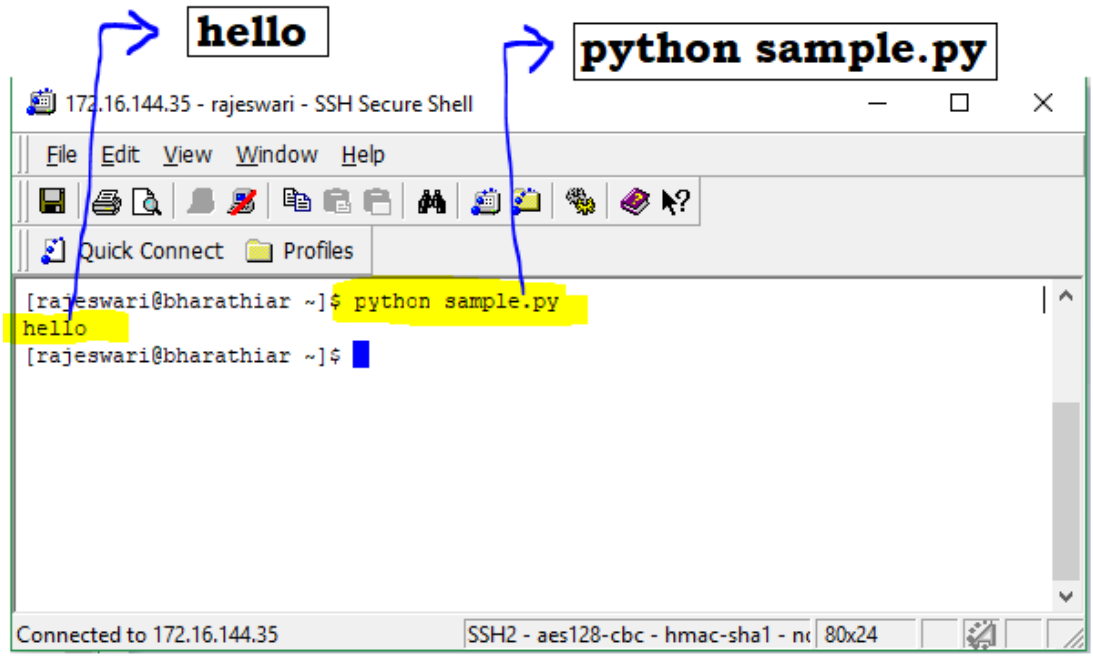

Figure 8: Executing a python file sample.py

Executing Python scripts in a parallel manner

Python has multiprocessing module but that only works within one node. Message Passing Interface (MPI) module has to be used to achieve parallelism over many nodes. MPI is a widely used library for parallel programming. In MPI multiple processors are used with independent memory running in parallel. Since memory is not shared, data is exchanged through calls to MPI routines. Each process runs same code, but can identify itself in the process set and execute code differently.

A sample file 'hellopy.py' which uses MPI is shown in figure 9. The file 'hellopy.py' can be executed using mpirun command as shown in figure 10.

```
from mpi4py import MPI
comm = MPI.COMM_WORLD
size = comm.Get_size()
rank = comm.Get_rank()
print "hello world from process ", rank, " of", size
```

Figure 9: hellopy.py

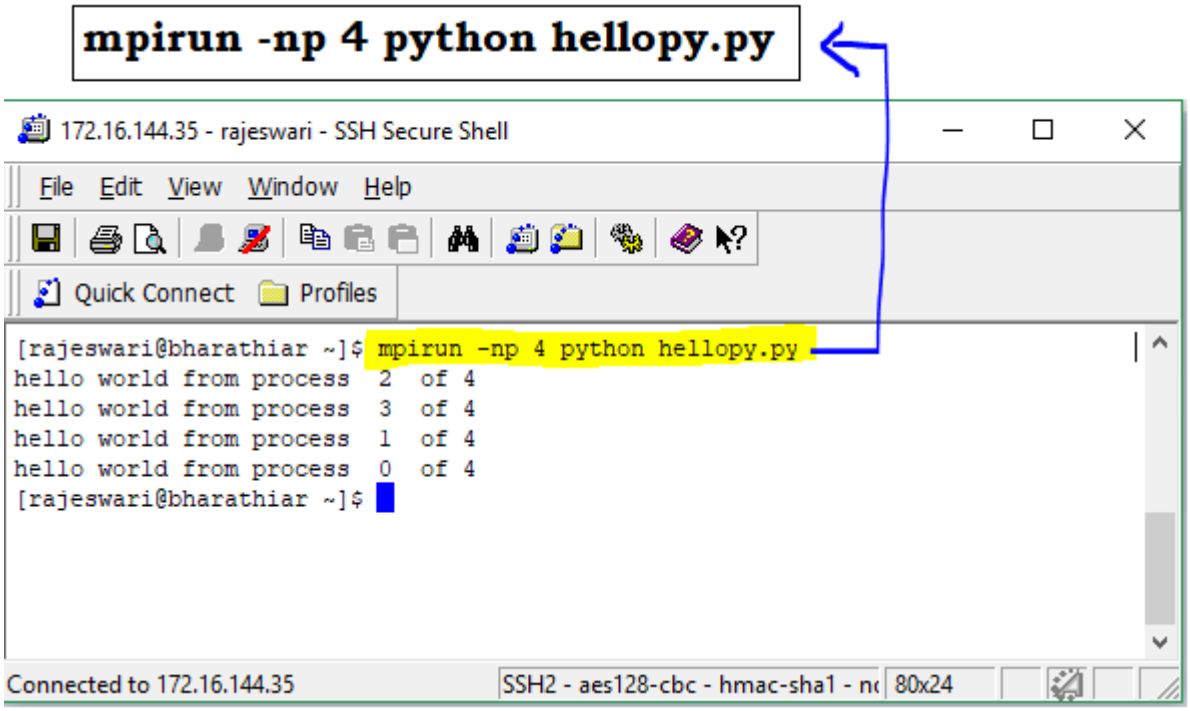

Figure 10: usage of mpirun command

2.5 Accessing Gaussian09

User can use nano editor to create input files with the extension '.com' as shown in figure 11. The number of nodes and number of processors per node must be specified in the Gaussian input file.

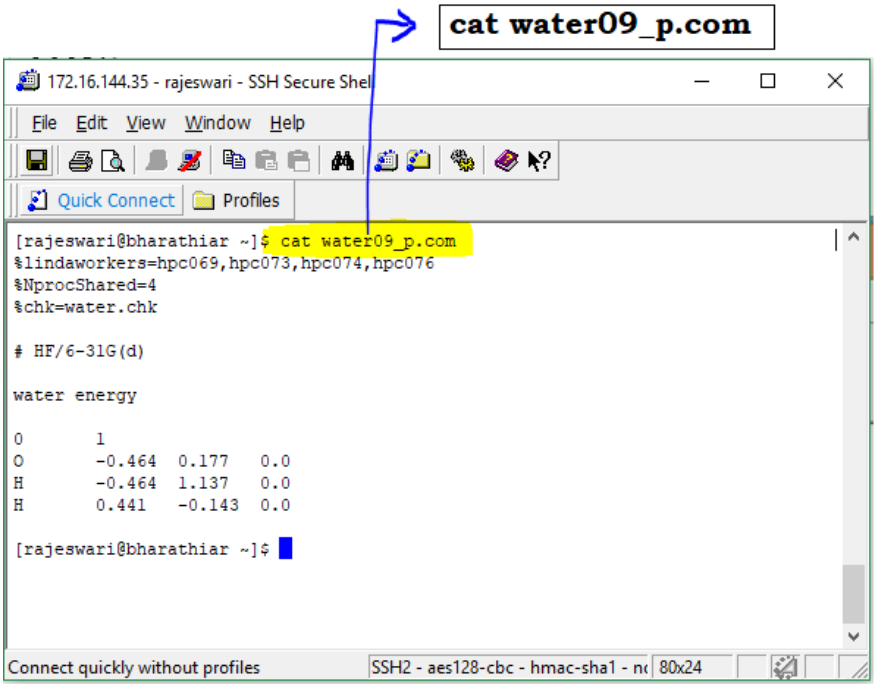

Figure 11: sample input file for Gaussian

To run a Gaussian g09 parallel batch job on the cluster, user needs to create a PBS script for it. The sample script shown in figure 12 includes a request for four nodes with four processors on each compute node.

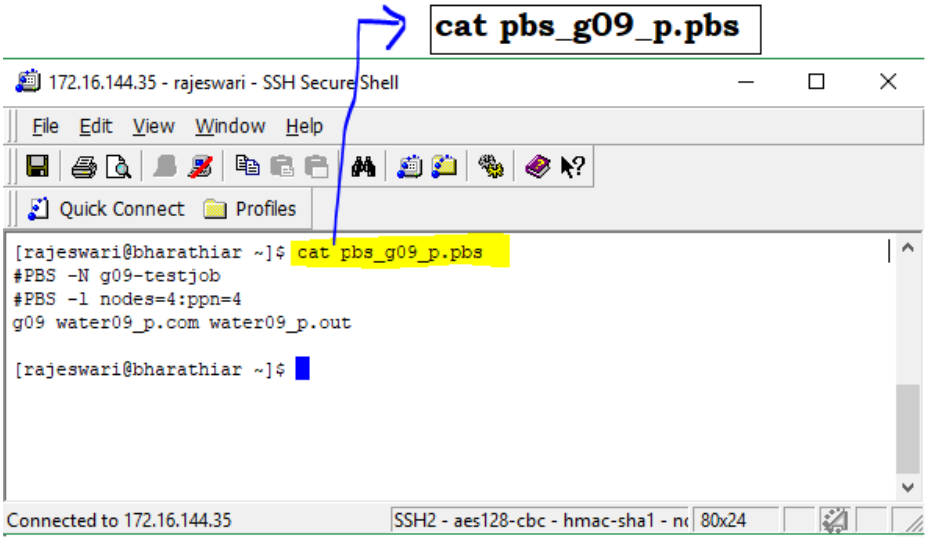

Figure 12: sample pbs script file pbs_g09_p.pbs

Users can submit their jobs by using qsub command followed by pbs script name. User can also check the status of the submitted job using qstat command as shown in figure 14.

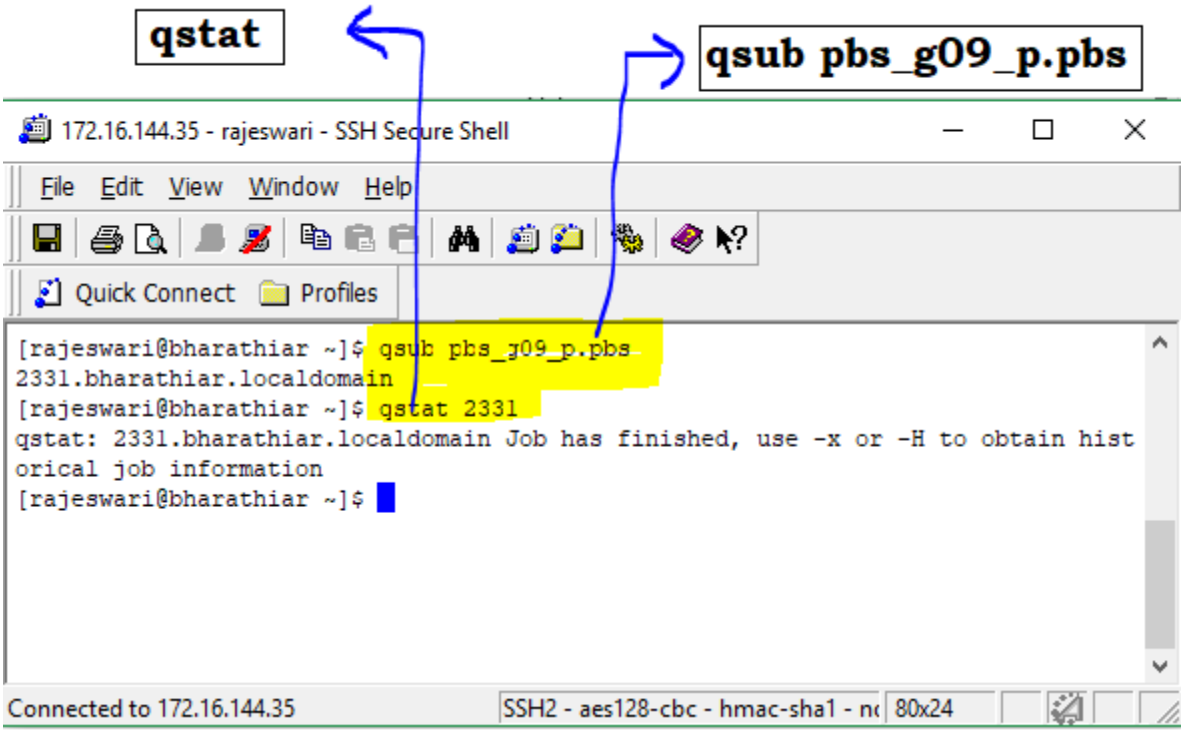

Figure 14: checking the status of submitted job using qstat command

Chapter 3: Basic Linux commands

S. No	Command	Description
1	**-h** **or** **–help**	Sometimes you may know the functionality of a command very well, but cannot recollect all the available options for a specific command. Use -h option of the command to review all available options of the command.
2	**info**	When you are not able to find the required information from the Unix man page, try the info documents using the Unix info command as shown below. $ info UNIXCOMMAND $ **info sed**
3	**who**	The who command simply return user name, date, time and host information. It does not print what users are doing
4	**whoami**	The whoami command prints the name of the current user
5	**ls**	The ls command displays list of files in human readable format.
6	**cp**	The cp command copies file from source to destination preserving same mode.
7	**cd**	The cd command (change directory) takes us to the destination directory.
8	**cat**	cat command can be used the display the contents of a single or multiple files
9	**pwd**	pwd command return the present working directory
10	**mkdir**	create a folder or a directory.
11	**rmdir**	delete a folder or a directory.
12	**touch**	The touch command is used to create a file. It can be anything, from an empty txt file to an empty zip file. For example, “touch new.txt”.
13	**cp**	Use the **cp** command to copy files through the command line. cp directory1 directory2
14	**mv**	Use the **mv** command to move files through the command line. mv directory1 directory2
15	**locate**	The **locate** command is used to locate a file in a Linux system locate filename
16	**nano, vi**	**nano** and **vi** are already installed text editors in the Linux command line.
17	**sudo**	A widely used command in the Linux command line, sudo stands for "SuperUser Do". So, if you want any command to be done with administrative or root privileges, you can use the sudo command.
18	**df**	Use the df command to see the available disk space in each of the partitions in your system.
19	**tar**	Use tar to work with tarballs (or files compressed in a tarball archive) in the Linux command line.
20	**zip, unzip**	Use zip to compress files into a zip archive, and unzip to extract files from a zip archive.
21	**uname**	Use uname to show the information about the system your Linux

		distro is running. Using the command “uname -a” prints most of the information about the system. This prints the kernel release date, version, processor type, etc.
22	**apt-get**	Use apt to work with packages in the Linux command line. Use apt-get to install packages. This requires root privileges
23	**ping**	Use ping to check your connection to a server
24	**clear**	You can use the clear command to clear the terminal if it gets filled up with too many commands.
25	**TAB Key**	TAB can be used to fill up in terminal. For example, You just need to type “cd Doc” and then TAB and the terminal fills the rest up and makes it “cd Documents”.
26	**Ctrl+c** **Ctrl+z**	Ctrl+C can be used to stop any command in terminal safely. If it doesn't stop with that, then Ctrl+Z can be used to force stop it.
27	**exit**	You can exit from the terminal by using the exit command.
28	**Halt** **reboot**	You can power off or reboot the computer by using the command sudo halt and sudo reboot.