CSC4005: Distributed and Parallel Computing

how to run your program on the cluster

Liu Haolin Email: 115010192@link.cuhk.edu.cn

1 Login to the master virtual machine

Windows On windows, simply use MobaXterm and configure a ssh gateway for the ssh section. The following figure illustrates how you should configure the setting.

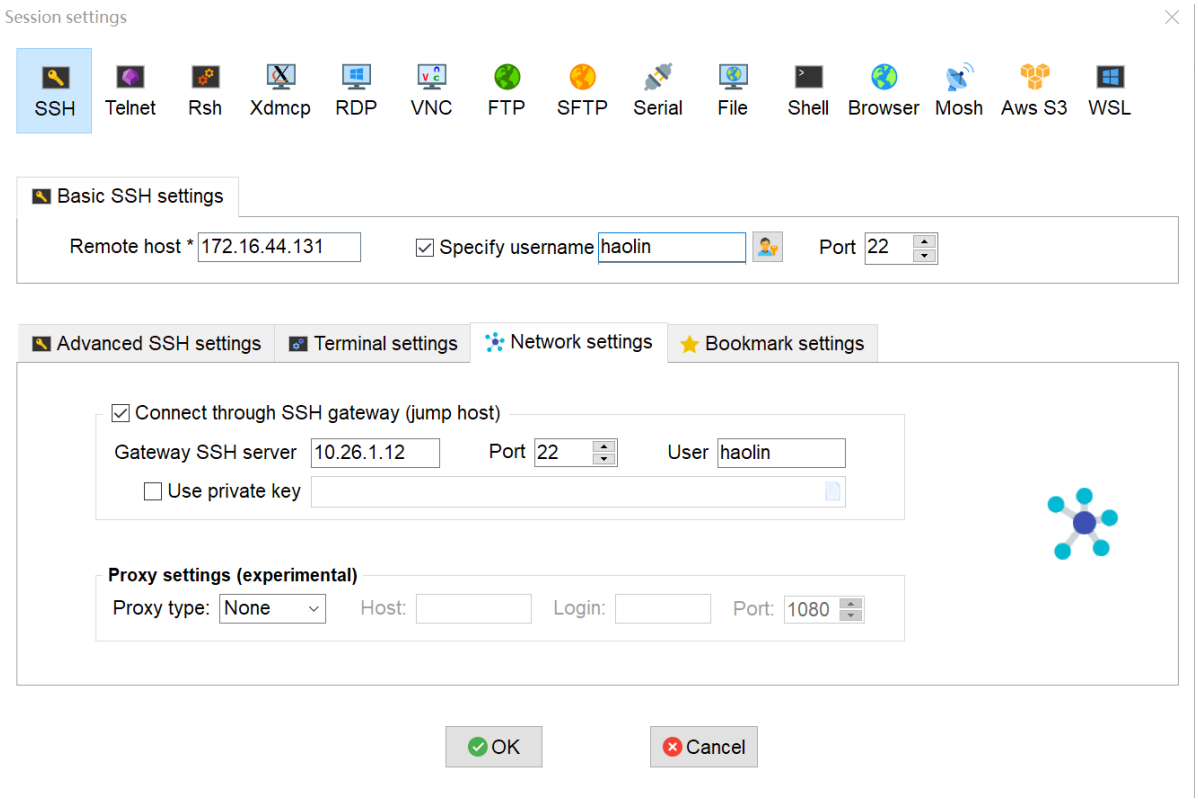

Figure 1: ssh configuration for MobaXterm, you should replace the user name with your student ID. The password is still studentID+123. 172.16.44.131 is the IP address of the virtual machine, which can only be accessed from the host 10.26.1.12.

Mac Open a terminal, firstly login to the host machine by typing "ssh -X username@10.26.1.12". After you login to the host, type "ssh -X username@172.16.44.131" to login to the virtual machine. The '-X' flag means allowing X-forwarding, which should be used starting from assignment 2. From Assignment 2, you should download a software called XQuartz, and use it to login the server in order to access the X11 GUI.

2 Write your code in a specified directory

The directory */code* is a shared file system for all 8 machines, your executable should be inside this directory so that all 8 machines can execute your files. All students should store all your executable files under */code/$username*, for example, if your student id is 115010192, you should save your executable files under */code/*115010192.

3 How to protect your codes

In order to protect your codes from being seen by the others students. You should put your code under */home/$username*, which can only be accessed by your own account. For convenient usage, you can save your code in */home/$username*, and save your executable file to */code/$username* using the '-o' flag during the compilation.

4 Write a script to submit your job

For a MPI executable program, you should run it by typing 'mpiexec -n 8 -f */home/mpi_config* ./*executable_file*'. However, a script is required to submit this command to a job queue. A template script is shown in Figure 2, **and make sure you have to copy the contents exactly except for the last line and make sure you have to use -f** */home/mpi_config*.

```
#!/bin/bash
#PBS -l nodes=1:ppn=5,mem=1g,walltime=00:02:00
#PBS -q batch
#PBS -m abe
#PBS -V

for i in {1..16}
do
timeout 60 mpiexec -n 5 -f /home/mpi_config /code/haolin/test.our -w 6400 -h 6400 #the command that you want to execu
te
done
```

Figure 2: A template sciprt for submitting a job. **updated!:the walltime in the second line is changed, make sure to copy it otherwise your program will not run, now a single script can only last for 5 minutes.**

You can replace the command in the last line in order to run any commands you like. Rememeber to add 'timeout 60' before every command, it limits the program running time for 60 seconds, which prevent you for occupying the computational resource for a long time. Any codes for your assignments should not exceed 60 seconds, you should specify a proper problem size for your assignment (e.g a proper number of elements in your input array).
You should simply create a script file by 'touch script.pbs', then use 'vim script.pbs' to type the content into the script. Then, use 'qsub script.pbs' to submit your job, and you could use 'qstat' to check the status of your job. The output of the 'qstat' command should be in Figure 3.

```
Job id                    Name             User            Time Use S Queue
------------------------- ---------------- --------------- -------- - -----
64.master                 script.pbs       haolin          00:00:45 C batch
65.master                 script.pbs       haolin                 0 R batch
66.master                 script.pbs       haolin                 0 Q batch
67.master                 script.pbs       haolin                 0 Q batch
68.master                 script.pbs       haolin                 0 Q batch
```

Figure 3: Output of the qstat command.

There is a 'S' column in this figure, 'C' stands for completion, 'R' stands for runing, 'Q' stands for wating in the queue.
The output of your program would be stored in a output file in your current directory, use 'ls' command to check the file name of the output file, then use 'chmod 777 filename' to modify the permission of the file. Then, use 'cat filename' to check the output of your program.
updated!: Use 'qdel' $<$ *queue_id* $>$ **to kill your job if you do not need it anymore.**