User documentation for EECS cluster

Some notes before you start using the cluster

1. Please spend some getting familiar with Linux if you haven't use it before. Try to remember all frequently used commands.

A good introduction one Linux command line.
http://cli.learncodethehardway.org/book/

2. There is several computing nodes in the cluster, our server is *compute-4-1*. Only people in our group can access our server. **I highly recommend: when you need to do some large scale experiment(like those for writing a paper), you should send an email to everyone in our group. We can avoid some confliction which will affect the performance of your algorithm.** For those small scale task, just run them on other nodes.
3. If you have any questions about the cluster (like you can not access the cluster or submit a task), please send Randall (rsvancara@wsu.edu) an email and ask for his help. If you have other questions, you can contact me any time and I'll try my best to help you solve them.
4. If you think there is something wrong with this document or you think some important thing I do not mention. Feel free to add everything to this document.

Access the cluster:

$ ssh qsong@aeolus.wsu.edu
$scp -l qsong aeolus.wsu.edu
After entering the password you can login to the eecs cluster. Our server is also managed by this cluster system.
Note: when you login into the cluster, you are actually in a login node(with very poor performance). Do not execute any tasks here. We need to use *qsub* to submit our task to some computing node.

Automatic login via SSH:
Step1 generate keys in your laptop.
$ ssh-keygen -t rsa

Press Enter whatever prompts.

```
Admins-MacBook-Pro:~ plin1$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/plin1/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /Users/plin1/.ssh/id_rsa.
Your public key has been saved in /Users/plin1/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:9f/tJDvSMcwTmVbFDdEkb/TzUArdVrq3+mkre3jMd8k plin1@Admins-MacBook-Pro.local
The key's randomart image is:
+---[RSA 2048]----+
|            ..+BX|
|             ..BO|
|          .   +*+|
|         . .  =+o|
|        S   .+..o|
|             .*..|
|             .B=o|
|            .o+E=|
|             +O*B|
+----[SHA256]-----+
Admins-MacBook-Pro:~ plin1$
```

Now id_rsa.pub is your public key file.

Step2 copy the public key to the server.
$ scp .ssh/id_rsa.pub aeolus.wsu.edu:~/tmp.txt
Input the password when prompted. Then login aeolus. There should be a file namely tmp.txt in your folder.

Step3 append the public key to the authorized_keys in the server.

$ cat tmp.txt >> .ssh/authorized_keys

Now you can login aeolus without password prompted.

```
Admins-MacBook-Pro:~ plin1$ ssh aeolus.wsu.edu
Last login: Tue Nov 24 15:19:08 2015 from wl-nat113.it.wsu.edu
Rocks 6.0 (Mamba)
****************************************
Welcome to Aeolus

****************************************
-bash-4.1$ # No password prompt. Yay!
-bash-4.1$
```

Directly accessing a computing node:

We can directly access a computing node to monitor the running status, like memory and CPU consumption. Our server is compute-4-1, so you can use:

$ssh compute-4-1

to access the server. Also you can access compute-2-X, these are other available machines in the cluster(I remember from 2-1 to 2-9). You can use *qstat* to find which machine your task is submitted to.

Note: **do dot run your task directly on this server use command line, remember use qsub to submit your task.**

Writing a script:

You know how to run your task locally, for example, you can run jar file using *java -jar test.jar.* Now you have to write a script, here is an example:

```
#!/bin/bash

#PBS -N test                                  //name of your task
#PBS -l nodes=1:ppn=64,walltime=1:00:00       //how many nodes you can use, also the max running time
                                                task will be killed after this time if it do not finish running
                                                (It killed a lot of time. I didn't know this)
#PBS -e /home/your_name/test/error.txt        //error message output path
#PBS -o /home/your_name/test/out.txt          //output path
#PBS -M songqi1990@gmail.com                  //notification email, you will receive two emails (one when
                                                submitting the task and one finishing the task)
#PBS -m abe

                                              //Here write the command you will use for your algorithm
module load jdk/1.8.0_51                      //First, load the module you need
cd /home/qsong/program/GraMi                  //Second, remember you must first go to the folder which
                                                contains your execution file
java -jar SPMine.jar                          //The command you use to run your task
```

Note: 1. You can use this script as a template, but remember remove all those comments. I'll give a script with no comments in appendix 1. 2. Using this script your task will be submit to any idle computing node. If you want to run it on our server, please add *#PBS -q doppa* in the script. 3. You can check the document I send before about all available modules you can use.

Submit a task and track it:

Now you have a script *test.pbs*.

Submit the task using:

$qsub test.pbs
The the system will give you a 6-digit task number, for example 183967
If you want to see the execution time of this task, you can use
$qstat 183967
If you want to look at the execution details, you can use
$qstat -f 183967
This will give you the detailed running state, like which computing node it is submitted.
If you want to stop this task, you can use:
$qsel 183967
If the execution finished, you can find the output in the out.txt file you set as the output path. It will not print the result in the terminal as you run locally. If there is some error in your code, you can check error.txt for details.
Note: There is some other commands for submitting the task, I only list the most frequent one here. You can check http://www.dartmouth.edu/~rc/HPC/man/qsub.html for more details if you need other functions.

Directories:
Now the default directory is /home/yourname. As the computing node is connected with the storage node using ethernet, you should use another to run the task which contains a lot of disk operations.
You can build a folder under */fastscratch*, like */fastscrath/qsong*. Then put all your files here and run the task(modify the path in the script to here).
Note: /fastsctrath is a public folder, everyone can look at and modify your file. Thus I do not recommend you to put every files here, still use your home directory as the main storage. Use this directory only when you have the specific requirement.

Appendix 1. A script example:

```
#!/bin/bash
#PBS -V
#PBS -N SPMine
#PBS -q doppa
#PBS -l nodes=1:ppn=24,walltime=48:00:00
#PBS -e /home/qsong/program/SPMine/error.txt
#PBS -o /home/qsong/program/SPMine/out.txt
#PBS -M songqi1990@gmail.com
#PBS -m abe

module load jdk/1.8.0_51
cd /home/qsong/program/SPMine
time java  -Xms81920m -Xmx81920m -jar LHSPMine.jar /home/qsong/data/neo4j/yago.db 6 0.00007 0.5 64
```

Important:

our server has at most 24 nodes. so if we write "nodes=1:ppn=64" in our script then it may cause some problem for your task. or in my case it just queued and didn't run.

Cluster Programming Guidlines

When you want to run a long time program, make sure that you have a good logger to understand where it is right now. Also, in the end of the program write sth to understand that it's finished properly.

If your program write the output into the file, make sure that you flushed progressively. Because maybe your program produced some input but it didn't flushed them into the file. If anyway it's broken you'll lose lots of your time.