Tutorial

Table of Contents

Useful terms

- *walltime*

The amount of real time a job has been running for

- *queue time*

The amount of time a job has been in the queue.

- *cpu time or processor time*

The number of core hours a job has used (i.e. the number of hours x the number of cores where the core has been doing work)

- *node*

A physical, self-contained, computer. A cluster is made up of a large number of individual nodes

- *processor*

An individual processor chip. A node may have more than one processor, and a processor generally has more than one core. All of Lovelace's nodes have 2 processors

- *core*

An individual processor. Each node in a cluster has several cores within it. Lovelace has 40 physical cores in each node

1 Logging on using SSH

To log into the cluster you will need a secure shell ("ssh") client.

1.1 Microsoft Windows

On Windows PuTTY is the recommended client and it is already available on staff-desktop or can be downloaded free from the PuTTY website [https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html].

You may be warned when logging on for your first time that PuTTY has not seen 'lovelace' before. Click Yes to continue.

1.2 Mac OSX or Linux

On Mac OSX or Linux you can use the *ssh* command from the command line, e.g. `ssh -X username@lovelace.lboro.ac.uk`

1.3 Other Operating system

Due to the risk of compromised downloads of SSH tools for systems such as Android no recommendations can be given. Instead you are recommended to look at ITS support advice at https://www.lboro.ac.uk/services/it/.

1.4 Do you have an account?

Since the cluster is a specialist system, currently accounts are only created after successful completion of a short application process. To begin the process, you need to complete the application from which is available on the HPC website at the application webpage [http://hpc-support.lboro.ac.uk/applying-for-time.html]. Once submitted, the forms are collated and sent to the HPC Stakeholder Subcommittee, which is made up of representatives from around the university, for approval. If your application is approved, IT Services will give you access to Lovelace and let you know the "account settings" for your allocated time. IT Services can approve applications for up to 20,000 core-hours 'seed time' without referring them to the HPC Stakeholder Subcommittee.

1.5 Have a go

- Run putty by launching it from the start menu. It is available in the IT Services Windows labs from the list of applications as `PuTTY`. On OSX or Linux launch a terminal.

- On Windows, type in "lovelace" as the host name to connect to. There should be no need to change any other settings from their defaults. Then click the "Open" button

and it will redirect you to a command prompt. On OSX/Linux, type `ssh -X` `username@lovelace.lboro.ac.uk` and press enter to login

- The first time you connect you will get a host key warning. You will need to click "yes", or on OSX/Linux type "yes", to connect.

- You will then be able to login with your university username and password. On OSX/Linux you will only need to type your password at this stage. (**N.B.** nothing will be displayed as you type your password. Do not worry, this is normal behaviour on Unix-like systems).

- You will now be at a command prompt. This is where you will be able to type commands to submit jobs as well as check and manipulate the queue later on in the tutorial. Try typing in `qstat` followed by the "return" (or "enter") key to list the currently queued jobs and their state.

- When you have finished working you can leave the system by typing `exit` into the command prompt. On Windows, the window will automatically close when you exit. On OSX/Linux you will be returned to your local command prompt

2 Transferring files from/to the cluster

Your home directory can be accessed from campus networks (or for external connections via the VPN) by the CIFS protocol as **\\hpc-storage\home\\{*yourgroup*}\\{*yourusername*}** (where you should replace {*yourgroup*} and {*yourusername*} as appropriate – type "pwd" on the command line to see where your files are on Lovelace). For Macintosh users, from the Finder select "Go" → "Connect to Server" and then enter smb://hpc-storage/home/{yourgroup}/{yourusername}

Alternatively, on Windows systems files can be transferred graphically between your local computer and the remote cluster by using WinSCP program that is available on university desktops. On Linux or Mac OSX systems `scp` command can be used to copy files forth and back, e.g. `scp local_file.txt username@lovelace.lboro.ac.uk:~/tutorial/`.

2.1 Have a go

- Bring up the Search box by clicking on the magnifying glass icon in the task bar.

- Type in \\hpc-storage\home\{yourgroup}\{yourusername}

- That should give you a window which shows your home directory on Lovelace.

- Next, click on this link, or type the address into a web browser, to download the tutorial files: http://hpc-support.lboro.ac.uk/hpc-tutorial.zip

- Click on "Save" when the download dialogue appears. You will need to enter your university username and password to access the download

- Save the file to your documents folder by selecting it from the drop down list then clicking the "Save" button

- Close the download dialogue when the download completes.

- Locate the file you just saved through "My Computer" on the Start Menu. Right click on the zip file and select "Extract All" to start the extraction wizard

- Click on "Extract" to extract the files. Ensure the "Show Extracted Files" tick box is ticked, then click next. The folder where the files have been extracted will appear

- The folder where the files have been extracted will appear. Drag the folder "tutorial" on to the folder window for your home directory on Lovelace.

3 Modules

The Loughborough HPC clusters use a system called "modules" to enable & disable particular software and to switch between multiple versions. When a module is enabled it is said to have been "loaded" and "unloaded" when it is disabled. It is controlled through the use of the `module` command. When loading a module the system will always try to load the default version, unless a specific version is specified.

A List of common actions and the corresponding commands are shown below:

- List available modules: `module avail`

- List currently loaded modules: `module list`

- Load a module: `module load module-name`

- Unload a module: `module unload module-name`

When you specify the module name for the `module load` and `module unload` commands you can specify the version as well (using the same format as the output of module avail). If you choose not to specify the version, module will pick the **default** one that is marked with capital letter "D".

3.1 Have a go

- Login to the cluster

- List the available modules by running `module avail`. Note there are several versions of intel compiler.

- Load the Intel compiler module by running `module load intel/compiler`.

- List the currently loaded modules using `module list` and note that the module `intel/compiler/64/2019/19.0.2` marked with "D" is loaded

- Unload the Intel compiler module with `module unload intel/compiler`

- List the currently loaded modules again

- Load the Intel compiler version 2018 module by specifying the exact version (**hint**: your argument should match the format of the output of the `module avail` command).

- List the currently loaded modules and check that version 2018 has been loaded.

- Unload the Intel compiler 2018 module

4 Finding help

In a Unix-like environment, such as the Linux operating system installed on our HPC cluster, help can often be found using the `man` command. The command `man` comes from abbreviating the word "manual". To find help simply type `man <command>` where "<command>" is the name of the command you need help using. The help displayed is often called the "man page" of a command.

For some programs, such as the Intel compilers, the man pages are only available after the appropriate module has been loaded. Some commercial software does not have any man pages, for information on using such software you will have to refer to the documentation from your supplier.

4.1 Have a go

- Bring up the man page for the command `qsub`

- Scroll the page using the space key to scroll down a page at a time, or the up and down arrow keys to scroll a line at a time

- When you have finished press `q` to quit

- Try to bring up the man page for the command `icc` (the Intel C compiler) - note the system cannot find it

- Load the Intel compiler module

- Now try again to bring up the man page for `icc`. Note that it now appears

- Unload the Intel compiler module

5 Submitting jobs

The HPC clusters at Loughborough are batch-processing systems. At the core of this is a queuing system called "Torque". Jobs are submitted into the queue through the `qsub` command. To submit a job two things are needed: a program or script to be run on the cluster and a submission script

5.1 Submission scripts

A submission script is a "normal" shell script. Generally, lines beginning with a hash mark [#] are comments. The first line, which begins "#!" tells the system what program to use to interpret the submit script and is required. The comments beginning with the letters "PBS" are special comments to signal to the job schedular how to run the job and what resources are required. The email options do no need to be included if you do not wish to receive email when your jobs are run or finish. The parameters are more fully described in the table below.

Parameter	Description	Example
`-q`	Queue to submit the job to. This should be "compute" for most jobs on Lovelace.	`-q compute`
`-N`	The name of the job. This appears in the queue listings and cannot contain a space.	`-N test_job`
`-m`	Specifies when to email the job status. It can be any combination of: b – send an email when the job starts a – send an email if the job is aborted e – send an email when the job exits	`-m bae`
`-M`	The email address to send notifications to.	`-M my.name@lboro.ac.uk`

	Specifies the resources required:	
-l	"walltime=hh:mm:ss" – specifies the maximum amount of time the job may run for. There is a hard limit of 100 hours. If not specified the default is 1 hour. "nodes=nodes:ppn=cores" – specifies the number of cores required per node and the number of nodes needed. If not specified the default is 1 processor on 1 node.	`-l walltime=1:00:00` `-l nodes=2:ppn=40`
-A	Specifies the project code which the job will debit. You will be told the appropriate code by email after an application for time is approved.	`-A training2020`

An example submission script, which would run a script called "`bin/test_script`" from the current user's home directory on one core with a maximum run time of 10 minutes, after loading the Intel Fortran compiler module is:

```
#!/bin/bash
#PBS -q compute
#PBS -N test_job
#PBS -m bae
#PBS -M my.name@lboro.ac.uk
#PBS -l walltime=00:10:00
#PBS -l nodes=1:ppn=40
#PBS -A training2020
cd $PBS_O_WORKDIR
module load intel/compiler/64/2018/18.0.5
${HOME}/bin/test_script
```

`${PBS_O_WORKDIR}` is the absolute path of current working directory, while `${HOME}` is a special variable in bash (the shell this script is using) which will always expand to the home directory of the current user[1].

N.B. The environment in which the submission script is run is identical to the one which you get when you first login. This means, for example, that you **MUST** load any required modules inside the submission script.

_____

1 Again, do not worry if this does not make sense to you - how it works is not that important, you just need to know that it does work.

For the best performance, in most cases, you will want to minimise the number of nodes over which your cores are spread (i.e. "nodes=1:ppn=12" will produce significantly better performance than "nodes=12:ppn=1" even though they both use the same number of cores overall).

It is very important that you check you have typed your project code correctly as this is used to for accounting. If you mistype the project code or try to use a project code with insufficient core-hour credits available your job will not start.

5.2 Running your job

Once you have your submission script (creating submission scripts is described in detail below) you can submit it by running `qsub <script>` where `<script>` is the name of the script you have created. For every job you run, you will be able to examine any output generated by the submission script (including the programs run in the script) by looking at two files which will be called "<job name>.o<job id>" and "<job name>.e<job id> (e.g. "test_job.o1234" and "test_job.e1234"). These contain the output written to STDOUT and STDERR respectively where STDOUT is used for normal output and STDERR for errors[2]. These files can be examined by typing `less <filename>` where `<filename>` is the name of the file you wish to examine. The `less` program allows you to navigate the output in the same way as you previously navigated the man page system, scrolling up and down with the arrow keys and pressing q to quit.

When a job has finished you will receive an email, if you specified the option to receive an email when the jobs ends and put the correct email address in the submission script. If you examine the email you will see it tells you "exit_status" of the job. For most programs on a Unix-like system, such as the cluster, any value other than 0 usually indicates some sort of error occurred. If you encounter this when running jobs on the cluster the program documentation should tell you exactly what the error number means.

5.3 Have a go

- Create a submission script (e.g. run.sh) in notepad++ (`Start → All Programs→ Accessories → Notepad++`) to run the "hello" example you uploaded previously (**hint**: the program to run will be `${HOME}/tutorial/hello` and you do not need to load any modules)

- Save the submission script to your documents folder

- Upload the submission script by dragging it over to your home directory on the cluster

---

2 You do not need to understand the difference between these two outputs, but you need to be aware that the two files combined give you the complete output of the job.

- Login to the cluster

- Submit the job to the queue using the `qsub` command (`qsub run.sh`). Note that `qsub` tells you the job id of the new job it has just created for you

- If an error is printed about the format of the file then you may need to convert it from Windows text format by running "dos2unix run.sh".

- Wait for your job to complete. You can check if it is still queued by running `showq`

- Look at the output in the two files created by the queuing system for your job with the `less` command

- If your submission script specified email notification, check your email and note the exit status of the program was "1"

6 Managing Jobs

Once you have submitted a job to the queuing system you will probably want to check that it has been queued correctly and where in the queue it is. This can be done with the `showq` command. A listing of the full details of a job can be shown with the command `checkjob <job id>`. It is important to note that, in the interest of maximising resource utilisation, jobs may not run in the order listed but the queuing system will endeavour to make sure that large jobs are not starved of resources for too long.

You might also want to remove a job you have submitted in error. This can be done with the `qdel` command which is called with the number of the job you wish to remove (e.g. `qdel 12345`).

6.1 Have a go

- List the jobs currently in the queue by running `showq`

- List the full details for the top job in the queue by running `qstat -f <job id>` where `<job id>` is the job identifier for the top job.

- Create a submission script for the "bubble" example you uploaded earlier. The bubble program will only work within one node, but it can use multiple cores[3]. You will need to tell the program how many cores it can use by adding `export OMP_NUM_THREADS=N` to your submission script, where N is the number of cores

---

3 The bubble code will not scale (improve its performace) beyond 2 cores, but that is not important for the purposes of the exercise.

to be used and should be replaced by a number, e.g. `export OMP_NUM_THREADS=20`

- Submit the "bubble" job

- Look at the job queue to check the job has been queued.

- Wait until the job starts running. (i.e. keep checking the queue to see if your job has gone from "idle" to "active", waiting a couple of seconds between each check. If there are a lot of jobs queued at the time you may not be able to complete this and the next step.)

- Wait 30 seconds then check the full listing of the job to see the CPU time used compared to the walltime (if the CPU time is not being shown, wait a few seconds for the system to update and try again). Note that the job has used more CPU time than walltime which indicates it is utilising more than 1 core

- Use the `qdel` command to remove the job from the queue

- Disconnect from the cluster

7 Conclusion

You have now read about and used all of the essential commands to submit jobs to the HPC cluster, transfer files to and from the cluster and manage your jobs once submitted. For information on using specific commercial packages you will need to refer to the user documentation for the specific package in question. If you wish to write your own code you will need to consult a good book on parallel programming and MPI [Message Passing Interface] to make best use of the cluster and high-speed interconnect.

More support information is available on the HPC area of the IT Services website at http://hpc-support.lboro.ac.uk/