COSC 420 - High-Performance Computing
Lab 5

Dr. Joe Anderson

Due: 22 November

1 Objectives

In this lab you will focus on the following objectives:

1. Gain experience setting up computing environments and clusters from bare hardware
2. Gain experience managing and administrating GNU/Linux environments.
3. Develop familiarity with parallel computing tools `MPICH` and `OpenMPI`

2 Tasks

1. Working in groups of three or four, you will assemble a high-performance computing cluster from the extra hardware available in the lab.
2. Select 2-3 of the spare machines (done during regular class time) to be clustered together, and designate one of the black carts for use by your team.
3. Download a `.iso` file for the GNU/Linux distribution of your choice (e.g. Fedora, CentOS, OpenSUSE, Ubuntu, etc.) – doing some searching beforehand to make sure MPI and MPICH are well-supported – and write it to a flash drive, making the device bootable. There are a number of ways to do this, but most need privileged access; you'll have to use your own machine, or a USB that the instructor and/or lab administrator can provide.
4. A nice general guide (for Ubuntu-based setups) can be found at https://mpitutorial.com/tutorials/running-an-mpi-cluster-within-a-lan/ This document will loosely follow the procedure found on that webpage.
5. Install some standard scientific programming tools (you may have to search a bit in the distribution's repository listings):
   (a) C/C++ compilers: `gcc` and `g++`
   (b) MPI Library: `mpich`
   (c) Remote access tools (needed for MPI communication): `openssh` and `openssh-server`
   (d) File sharing tool NSF: `nfs-kernel-server` and `nfs-common`
   (e) Text editors of choice, optionally `texlive` for LaTeXcompiling
6. Once you are able to run a simple MPI program on your "master" node, repeat the same setup on the other machines.
7. Make sure all machines are connected via an ethernet switch and cat5 cables.

8. You will have to perform most of the following steps using privileged access, usually with the `sudo` command, which runs subsequent commands, e.g. `sudo vim /etc/hosts`, with full privileges.
   (a) **BE VERY CAREFUL WHEN USING sudo!!!**
   (b) If you are reckless, you may inadvertently delete or more important files, resulting in a corrupt system
9. Assign each machine a manual IP address: something like `10.0.0.X` where `X` is different for each machine. Optionally, configure the `hostname`, often found in the `/etc/hostname` file for a more logical name, e.g. `mycluster-node1`, `mycluster-master`, etc.
10. Configure the `/etc/hosts` file on each of the machine as follows:
    (a) The general format is `<Address> <Logical Name>`
    (b) Example for the master node:

```
127.0.0.1 localhost

# Comment, Begin MPI Section
# Master needs address of all the nodes
10.0.0.1  mycluster-master
10.0.0.2  mycluster-node1
10.0.0.3  mycluster-node2
```

    (c) Example for the worker node:

```
127.0.0.1 localhost

# Comment, Begin MPI Section
# Worker Node 2 needs only itself and master
10.0.0.1  mycluster-master
10.0.0.3  mycluster-node2
```

11. Make sure all machines have a user account with the same username, e.g. `mpiuser`
    (a) You can use the command `sudo adduser mpiuser` to create this user with a home directory.
12. Set up a folder to hold the shared executables and data:
    (a) If the folder is in the `mpiuser` home folder, make a file called `/etc/exports` with the contents:

```
/home/mpiuser/cloud *(rw,sync,no_root_squash,no_subtree_check)
```

    (b) Then run `exportfs -a`
    (c) Use your system tools to restart the `NSF` server with `sudo systemctl restart nfs-server`
    (d) On the clients, you will mount the remote directory (make sure the `cloud` directory exists and is empty):

```
sudo mount -t nfs master:/home/mpiuser/cloud ~/cloud
```

    (e) You can use `df -h` to make sure you see an entry for the mounted directory on each client.
    (f) You can make the moutn happen automatically by modifying the `/etc/fstab` (which configures the filesystem). To do this, you will add a line to that file that looks something like

```
#MPI Cluster Directory Mount
master:/home/mpiuser/cloud /home/mpiuser/cloud nfs
```

13. Configure the `ssh` server:
    (a) Use the command `ssh-keygen -t rsa` to generate a keypair
    (b) Copy your key to the other machines `ssh-copy-id <client address>`
    (c) Now execute `eval 'ssh-agent'` and `ssh-add ~/.ssh/id_dsa` to enable logging in without a password (it uses the keys to authenticate – and is actually just as secure!)
    (d) You should be able to type `ssh <machine address>` and get a prompt, logged in to the other nodes. If this does not work, you may have problems with MPI communicating between the different nodes.
14. To run an MPI program, *you do not need to use SLURM or other job managers* (but of course, you can look into it and try it).
    (a) To test your cluster, write a simple MPI program the reports the processor name and rank, prints it to the standard output, and terminates.
    (b) The binary for this program will need to reside in the shared directory (`~/cloud` in the instructions above)
    (c) To run the binary on only the master node, you can use `mpirun -n <N> ./<executable>`
    (d) To run the binary across multiple machines, you will add an option to specify which hosts:

```
mpirun -n <N> -hosts <host1>,<host2>,<host3> ./<executable>
```

    (e) To make this easier, you can put the hosts in to a text file and use the `--hostfile <hosts file name>` option to read the hosts from the file.

3 Submission

The instructor will come around and verify that you can run a clustered job on your system!