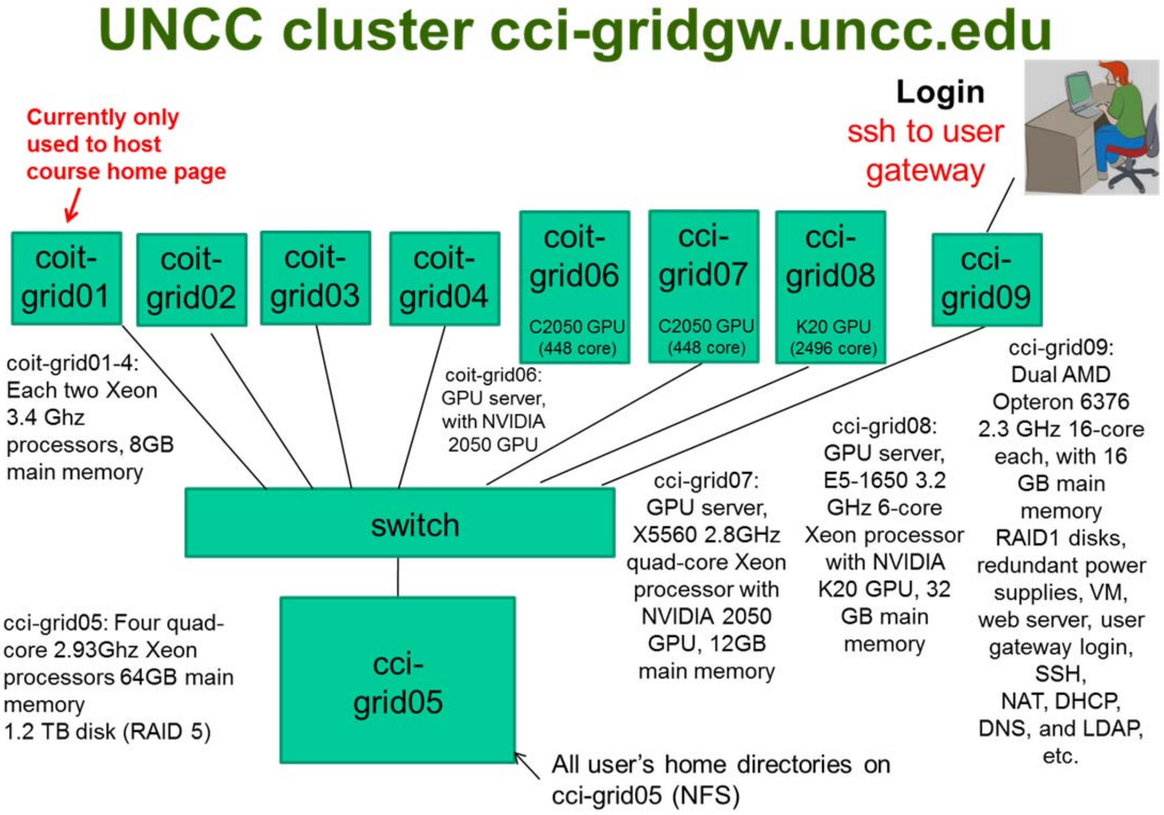
Using UNCC Parallel Programming Cluster

B. Wilkinson: Modification date August 14, 2014

Overview

The current state of the cluster is shown below:


UNCC cluster cci-gridgw.uncc.edu
Login
ssh to user gateway
Currently only used to host course home page
coit-grid01
coit-grid02
coit-grid03
coit-grid04
coit-grid06
C2050 GPU (448 core)
cci-grid07
C2050 GPU (448 core)
cci-grid08
K20 GPU (2496 core)
cci-grid09
coit-grid01-4: Each two Xeon 3.4 Ghz processors, 8GB main memory
coit-grid06: GPU server, with NVIDIA 2050 GPU
cci-grid07: GPU server, X5560 2.8GHz quad-core Xeon processor with NVIDIA 2050 GPU, 12GB main memory
cci-grid08: GPU server, E5-1650 3.2 GHz 6-core Xeon processor with NVIDIA K20 GPU, 32 GB main memory
cci-grid09: Dual AMD Opteron 6376 2.3 GHz 16-core each, with 16 GB main memory RAID1 disks, redundant power supplies, VM, web server, user gateway login, SSH, NAT, DHCP, DNS, and LDAP, etc.
switch
cci-grid05
cci-grid05: Four quad-core 2.93Ghz Xeon processors 64GB main memory 1.2 TB disk (RAID 5)
All user's home directories on cci-grid05 (NFS)

Note: Some names have been changed from coit- to cci-

Note this cluster does not use a job scheduler. Your programs can be executed directly for the command line and you will have direct interactive access to your programs. However, you will not get control of the command line until the program finishes (or you issue a control-C). If necessary, you can run your jobs in the background with the & option on the command line.

FOR THE SAKE OF OTHERS, please make sure you do not leave any long running processes.

VERY VERY IMPORTANT: Do not ssh *from* internal nodes 05, 07 or 08 to anywhere, only from cci-gridgw (09) as currently ssh *from* internal nodes 05, 07 or 08 causes really significant problems including loosing all your files. ONLY run MPI from **cci-gridgw.uncc.edu**. If you need to use only an internal node, specify the node in a machines file.

Backup — There is no automatic file backup on this cluster although a RAID-5 system is in place for home directories. *It is highly recommended that keep a back-up copy of your files on another computer.*

Logging on

Users ssh to **cci-gridgw.uncc.edu** (user gateway on **cci-grid09**), e.g. on Linux command line:

```
ssh –l username cci-gridgw.uncc.edu
```

On a Windows machine, Putty and WinSCP can be used.

When you log in for the first time, accept all default settings (no key passphase, password-less ssh between nodes):

```
login as: xxx
xxx@cci-gridgw.uncc.edu's password:
Creating directory '/nfs-home/xxx'.

We need to set up some things so you can SSH between
nodes securely from your account.

Please hit enter a bunch, accepting all defaults!

Generating public/private rsa key pair.
Enter file in which to save the key (/nfs-home/xxx/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /nfs-home/xxx/.ssh/id_rsa.
Your public key has been saved in /nfs-home/xxx/.ssh/id_rsa.pub.
The key fingerprint is: …

All done!

You should now be able to run the following commands
to connect to the other three nodes:

ssh cci-grid05
ssh cci-grid07
ssh cci-grid08

Enjoy!

[xxx@cci-gridgw ~]$
```

Home directory — The full path is `/nsf-home/<username>/...` .

Executing programs that exist in current directory — As is usual on Linux systems, the current directory is not in the path and it is necessary to provide `./` or the full path otherwise, you will get a "File not found" error.

X11 libraries: Currently installed on **cci-gridgw.uncc.edu** and **cci-grid05**. Compile with –lX11. See separate notes on creating graphical output for more details.

MPI

MPICH2 installed. Compute nodes installed on 05, 07, 08. Run MPI from **cci-gridgw.uncc.edu** only.

Compile `hello.c` with `mpicc`.

```
mpicc –o hello hello.c
```

Execute `hello` with the Hydra process manager (NOT `mpiexec`):

```
mpiexec.hydra -n numprocesses ./hello
```

With this command, the program will run just on **cci-gridgw.uncc.edu** (grid09)

Using multiple computers

The usual way MPI identifies the computers (machines) that it can use for executing an MPI program is by listing them in a file, and using the **-machinefile** flag with **mpiexec** (**mpiexec.hydra** on the cluster). Each MPI process executes on one of the machines in the list. By default, MPI cycles through the list of machines giving processes to machines in a round robin fashion. (One can also specify the number of processes on a particular machine by adding that number after the machine name.)

Because of the way the UNCC cluster is set up, internal compute nodes communicate using local names. Create a file called say ***machines*** containing the list of machines, *using the local names*, e.g.

```
cci-grid05
cci-grid07
cci-grid08
cci-gridgw.uncc.edu
```

To execute the **hello.c** program on the computers specified in the machines file with 8 processes, the command is:

```
mpiexec.hydra -machinefile machines -n 8 ./hello
```

OpenMP

You are allowed to ssh to internal nodes to execute OpenMP programs.

Compile an OpenMP program `omp_hello.c` with the regular gcc compiler using the command:

```
cc -fopenmp -o omp_hello omp_hello.c
```

Execute as a regular executable:

```
./omp_hello.c
```

Combined (hybrid) OpenMP/MPI programs

Compile hybrid program `omp-mpi_hello.c` with:

```
mpicc -fopenmp -o omp-mpi_hello omp-mpi_hello.c
```

Note this command invokes the regular cc compiler.

Execute as an MPI program on **cci-gridgw.uncc.edu** using:

```
mpiexec.hydra -machinefile machines -n numprocesses ./omp-mpi_hello
```