# Compiling and running MPI programs on UINC-Charlotte coit-grid cluster

Author: B. Wilkinson. Modification date: Sept 12, 2012

An implementation of MPI called MPICH is installed on the coit-grid cluster. As with other implementations of MPI, in MPICH, there are two commands (scripts) that will be used mainly:

- **mpicc** to compile MPI programs, and
- **mpiexec** to execute a MPI program.

The cluster does not use a job scheduler. Your programs can be executed directly for the command line using mpiexec with program output returning to you. 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.

## Connecting to coit-grid cluster

Connection to the cluster is by ssh connection using Putty (or another ssh client). **coit-grid01.uncc.edu - coit-grid04.uncc.edu**, and **coit-grid07.uncc.edu** can reached directly from on campus and off-campus. Typically one uses **coit-grid01.uncc.edu** although other servers can be used if available.

## Checking MPI commands

First check the implementation and version of these commands with:

```
which mpicc
which mpiexec
```

The full paths should be returned.

## Compilation:

The command to compile the program **hello.c** to create the executable **hello** is:

```
mpicc -o hello hello.c
```

which uses the gcc compiler to links in the libraries and create an executable hello, and hence all the usual flags that can be used with gcc can be used with mpicc.

## Execution:

The command to execute the program hello with 4 processes is:

```
mpiexec -n 4 ./hello
```

where `./` denotes the path to the current directory (assume hello is in the current directory). As is usual on Linux systems, it is necessary to provide `./` or the full path otherwise, you will get a "File not found" error. The full path is `/nsf-home/<username>/...` .

In this example, the program runs only on the system you are logged onto. So far, the program will execute with four processes just on **coit-grid01.unc.edu**.

### Using multiple computers

In the previous exercise, only one computer was used, coit-grid01.unc.edu, with multiple processes time-shared on the single processor. In the following, multiple computers will be used.

### Determining the available servers in the coit-grid cluster

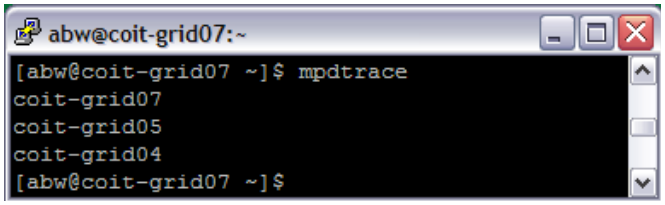
At various times, not all servers are available. For MPI programs to operate between servers, the mpi mpd daemons must be running on each server to form a ring. Execute the command:

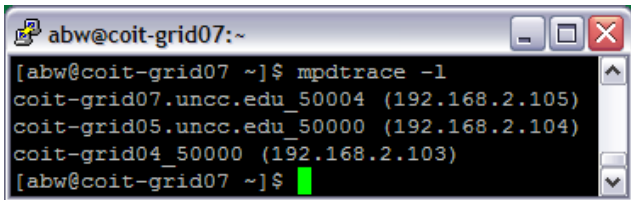
```
mpdtrace
```

or

```
mpdtrace -l
```

which will list those servers where the mpd is running, for example:




Make a note of the servers and use these servers in the following.

### Setting up the list of machines to use

The computers (machines) that MPI can use for executing a program is specified in a file called a machines file, which is then used with –machine option in the mpiexec command:

```
mpiexec -machinefile machines -n 4 ./prog1
```

would run **`prog1`** with four processes using the machines listed in the file called machines. Each process would execute on one of the machines in the list. MPI would cycle through the list of machines giving processes to machines. (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 cluster is set up with servers having dual internal and external Ethernet connects, each server has local name. **coit-grid01.uncc.edu** is called **grid01** and so on. The machines file must use the local names (or the local IP addresses). For example, a machines file with the contents:

```
grid01:4
grid02:4
grid03:4
grid04:4
```

would specify **coit-grid01.uncc.edu**, **coit-grid02.uncc.edu**, **coit-grid03.uncc.edu**, and **coit-grid04.uncc.edu** with four processes on each server**. (**Each of these computers consists of two hyperthreaded Intel processors. Each can naturally support up to four processes at the same time.)

For more information, see the MPICH documentation,
http://www.mcs.anl.gov/research/projects/mpich2/