Using MPI

MPI ("**M**essage **P**assing **I**nterface") is a library of functions and definitions that allow processes on multiple processors to communicate with each other. The MPI API has been standardized, although there are several implementations of MPI; The version we have installed is called MPICH.

MPI allows a user to create a single executable that can be distributed and executed simultaneously on a group of machines (sometimes called a "cluster"). The API provides functions that allow a node to uniquely identify itself within the cluster, and to send and receive messages from other nodes. It also provides routines used to manage the system.

The CS Department does not have a dedicated cluster for MPI, but instead uses a group of general purpose Linux-based machines. Since these machines normally require a normal login process, but MPI needs to access the processors it uses without logging in, some initial setup is necessary in order to use MPI.

Initial MPI setup

The following section decribes the setup necessary to use MPI on the CS Department computers. This setup should only need to be done once. We will be using Linux machines physically located in JEB 211A. Their names are `cs-embedxx.cs.uidaho.edu`, where xx ranges from 11 to 19 (18 is missing). All the machines are identical - however, the first machine - `cs-embed11` - contains the files that are discussed here, and it is assumed in the commands that follow that `cs-embed11` is being used during this setup.

1. Log into `cs-embed11.cs.uidaho.edu` as a normal user. If you are off campus, you will first need to log into `wormulon.cs.uidaho.edu`, since it is the only one of our systems that is outside the firewall, then `ssh` into `cs-embed11`.
2. Several files need to be copied from the directory `/var/tmp` on `cs-embed11` to your home directory, and some permissions need to be changed (using the `chmod` command; 600 $\Rightarrow$ read, write by owner only). You can use the following commands. Note that some of the file names contain '.' (periods) in them - be sure to include the periods in the commands. Also, those files that start with '.' are "hidden" files in Linux, so they won't show up in a file listing unless you include a ''-a' option (ie, `ls -a`):

   ```
   cp /var/tmp/.mpd.conf $HOME
   chmod 600 $HOME/.mpd.conf
   cp /var/tmp/mpd.hosts $HOME
   chmod 600 $HOME/mpd.hosts
   cp /var/tmp/known_hosts $HOME/.ssh/known_hosts
   chmod 600 $HOME/.ssh/known_hosts
   ```

   Optionally, you can also copy the example file `cpi.c` over to your home directory:

   ```
   cp /var/tmp/cpi.c $HOME
   ```

3. Using an editor, open `.mpd.conf` and replace '`your_password`' with your password, or another phrase (the "secret word")..

You should now be set up to run MPI programs!

Running an MPI Program

Each time you wish to run MPI, you will need to log in and do the following:

1. Start the mpi system (this may take a short but noticeable amount of time, typically a few seconds):

   ```
   mpdboot -n 7
   ```

2. Compile and run your program. To aid in this process, our MPI implementation comes with a couple of scripts that serve as "wrappers" to the regular gcc/g++ compilers, called `mpicc` and `mpic++` - these add the necessary steps required to compile MPI programs.

   To test it out, try running the test program `cpi.c` . Do the following:

   ```
   mpicc -o cpi  cpi.c
   mpiexec -np 7 ./cpi
   ```

   You should get no errors during compilation. The output should look similar to the following:

   ```
   cs-embed11:~$ mpiexec -n 7 ./cpi
   Process 0 on cs-embed11
   Process 1 on cs-embed12
   Process 4 on cs-embed15
   Process 2 on cs-embed13
   Process 5 on cs-embed16
   Process 3 on cs-embed14
   Process 6 on cs-embed17
   pi is approximately 3.1416009869231249, Error is 0.0000083333333318
   wall clock time = 0.305170
   cs-embed11:~$
   ```

   Note that each message comes from a different machine, and they are not in any particular order.

3. When finished, clean up by typing:

   ```
   mpdallexit
   ```