Parallel Programming

Installing OpenMPI

Author B. Wilkinson - Modification date May 29, 2015

1. Using the Ubuntu Package Repository

If you have Ubuntu as the Linux distribution (using VirtualBox or as the native host OS), you can use the Ubuntu Package Repository to install OpenMPI:

(a) Check package repository is up to date

Issue the command:

sudo apt-get update

(b) Check gcc compiler and libraries

Issue the command:

sudo apt install build-essential

to make sure the gcc compiler and C libraries are installed correctly.[1]

(c) Installing OpenMPI

Issue the command:

sudo apt-get install openmpi-bin openmpi-doc libopenmpi-dev

to install OpenMPI, documentation and required the OpenMPI libraries.

2. Alternative -- Installation from source (All Linux distributions)

An alternative to installing packages using **apt-get install** is to download the source files and compile them and install them manually. This should get the most recent version and will be necessary for OS's that do not package repositories or the software or particular version is not in the repository. It also gives you more control on where to locate the files on your system. However it is more effort and time consuming. Although usually there are scripts provided to compile and install the software, you will generally also need set up paths in environment variables.

Note: You need to ensure you have a C compiler such as gcc and command line tools. For Max OS X, if necessary, install Xcode and Xcode Command Line Tools first.

[1] A more specific command is **sudo apt-get install gcc g++**

(a) Download OpenMPI

To download from the Internet, you have to use the browser from within OS (Firefox in Ubuntu). With that browser, go to:

http://www.open-mpi.org/software/

and download the most recent OpenMPI source file, (**openmpi-1.8.1.tar.gz** as of the date of this document) to a convenient location, assumed to be the **Downloads** directory here.

(b) Uncompress OpenMPI tar file

Open a terminal and type:

cd ~/Downloads

tar zxvf openmpi-1.8.1.tar.gz

(c) Configure installation file

Switch to the OpenMPI directory and execute the configure script found in the OpenMPi directory by typing:

cd openmpi-1.8.1

./configure --prefix=/usr/local

This does a lot of checking and configuration work so expect to wait a long time! With the **--prefix** option given, OpenMPI binaries are installed in the directory **/usr/local/bin** and shared libraries in **/usr/local/lib**. If you want a different installation location, replace **/usr/local** with your desired directory. Where you choose to install is important to recognize as you will need to set the paths later accordingly.

(d) Compile and Install

Once the configuration is complete, type:

make all

sudo make install

to compile and complete the installation process. Expect to wait a long time!

(e) Set up paths to binaries and libraries

Below the installation directory is **/usr/local**. If it is different, alter the commands accordingly.

Path to binaries. First check that **/usr/local** is not already in the PATH environment variable, with **echo $PATH**. If not, add the line:

export PATH=/usr/local/bin:$PATH

to the end of the **~./profile** file. (Use an editor such as nano.)

Path to libraries. Add the line:

export LD_LIBRARY_PATH=/usr/local/lib

to the end of the **~./profile** file. (This will create **LD_LIBRARY_PATH**, which is generally necessary, as well as set it.)

The **~/.profile** script is executed when the machine is started. To execute it now, type

source ~/.profile

Check both paths with **echo**.

Test MPI Installation

First check **mpicc** and **mpiexec** can be found with the commands:

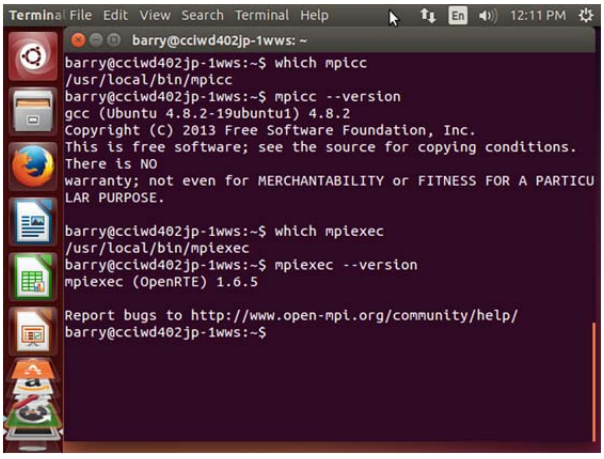
which mpicc
which mpiexec

If you get an error message saying file not found, probably the **PATH** environment variable has not been set properly. Go back and check.

Check their versions with the commands:

mpicc --version
mpiexec –version

You should get a message such as:



If you get an error message saying a shared library such as **libopen-pal.so.6** cannot be found or does not exist, probably the **LD_LIBRARY_PATH** environment variable has not been set probably. Go back and check.

Testing MPI environment with a sample MPI program

It is suggested that you create compile and run a sample MPI program such as:

```
#include <stdio.h>
#include <string.h>
#include <stddef.h>
#include <stdlib.h>
#include "mpi.h"

main(int argc, char **argv ) {
  char message[256];
  int i,rank, size, tag=99;
  char machine_name[256];
  MPI_Status status;

  MPI_Init(&argc, &argv);
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);

  gethostname(machine_name, 255);

  if(rank == 0) {
    printf ("Hello world from master process %d running on %s\n",rank,machine_name);
    for (i = 1; i < size; i++) {
      MPI_Recv(message, 256, MPI_CHAR, i, tag, MPI_COMM_WORLD, &status);
      printf("Message from process = %d : %s\n", i, message);
    }
  } else {
    sprintf(message, "Hello world from process %d running on %s",rank,machine_name);
    MPI_Send(message, 256, MPI_CHAR, 0, tag, MPI_COMM_WORLD);
  }

  MPI_Finalize();
  return(0);
}
```

To organize things, create a directory say called **MPI** and **cd** to that directory.
Create the sample program using an editor such as **gedit** (Ubuntu) or **nano** and call it say **hello.c**.

Compile:

Issue the command: **mpicc –o hello hello.c**

to create the executable **hello**. (With the given program, you may get some C syntax warning messages, which can be ignored.)

Execute:

Issue the command: **mpiexec –n 4 ./hello**

to execute the program using four processes. Output should similar to (with you own computer name):