**Exercise: MPI Hello World**

In this exercise, you’ll be running your first MPI code, specifically an MPI version of the classic Hello World program.

Here are the steps for this exercise:

1. Log in to OSCER’s Linux cluster supercomputer (schooner.oscer.ou.edu).
2. Conﬁrm that you’re in your home directory:

**pwd**

/home/yourusername

1. Check that you have a SIPE subdirectory inside your home directory:

**ls**

SIPE

If this directory does not exist, create by using the following command:

**mkdir SIPE**

1. Go into your SIPE subdirectory:

**cd SIPE**

1. Conﬁrm that you’re in your SIPE subdirectory:

**pwd**

/home/yourusername/SIPE

1. See what files or subdirectories (if any) are in the current working directory:

**ls**

1. Copy the subdirectory named HelloWorld from Henry’s SIPE directory into your SIPE directory:

**cp -r ~hneeman/SIPE/HelloWorld ~/SIPE/**

This command means:

“Copy the subdirectory named HelloWorld inside the directory named SIPE under the home directory of user hneeman into my directory SIPE under my home directory.”

1. Go into your HelloWorld subdirectory:

**cd HelloWorld**

1. Conﬁrm that you’re in your HelloWorld subdirectory:

**pwd**

/home/yourusername/SIPE/HelloWorld

1. See what files or subdirectories (if any) are in the current working directory:

**ls**

1. Choose which language you want to use (C or Fortran90), and cd into the appropriate directory:

**cd C/**

OR:

**cd Fortran90/**

1. Conﬁrm that you’re in your C or Fortran90 subdirectory:

**pwd**

/home/yourusername/SIPE/HelloWorld/C

OR the output of the pwd command might be:

/home/yourusername/SIPE/HelloWorld/Fortran90

1. See what files or subdirectories (if any) are in the current working directory:

**ls**

MPI OpenMP Serial

1. Go into your MPI subdirectory:

**cd MPI**

1. Confirm that you’re in your MPI subdirectory:

**pwd**

/home/yourusername/SIPE/HelloWorld/C/MPI

OR the output of the pwd command might be:

/home/yourusername/SIPE/HelloWorld/Fortran90/MPI

1. See what files or subdirectories (if any) are in the current working directory:

**ls**

1. Edit the batch script hello\_world\_mpi.sbatch to use your username and e-mail address.
2. If you haven’t already examined hello\_world\_mpi.c(or hello\_world\_mpi.f90), do so now.
3. **IMPORTANT IMPORTANTIMPORTANTIMPORTANTIMPORTANTIMPORTANT**

Compile using the *shell script* make\_cmd:

**make\_cmd**

**NOTE**: A *shell script* is a file containing a sequence of Unix commands, which are executed like a program.

If that command fails, try this:

**./make\_cmd**

That is, put a dot (period) and a slash before make\_cmd, with no blank spaces.

1. Submit the batch script file hello\_world\_mpi.sbatch to the batch scheduler:

**sbatch hello\_world\_mpi.sbatch**

**NOTICE** the less than symbol<which is **EXTREMELY IMPORTANT**.

You should get back output something like this:

Submitted batch job #######

Where ####### is replaced by the batch job ID for the batch job that you’ve just submitted, and you may have a different queue name.

1. Check the status of your batch job:

**squeue -u yourusername**

You’ll get one of the following outputs, either:No unfinished job found

(if you get this right after the squeue command, try it several more times, because sometimes there’s a pause just before the batch job starts showing up, as below),

OR something like this:

JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)

####### normal sipehello\_w+ yourusername PD 0:00 1 (None)

Where ####### is replaced by a batch job ID number, and yourusername is replaced by your user name, and where PD is short for “pending,” meaning that your job is waiting to start,

OR something like this:

JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)

####### normal sipehello\_w+ yourusername R0:55 1 c###

Where R is short for “running”and c### is replaced by the name of a compute node.

You may need to check the status of your batch job repeatedly, using the squeue command, until it runs to completion. **This may take several minutes (occasionally much longer).**

You’ll know that the batch job has finished when it no longer appears in the list of your batch jobs.:

No unfinished job found

1. If you find that your batch job waits in the batch queue for more than a few minutes, you can edit your batch script to use the batch queue (partition) named debug.

Specifically, in your batch script, replace this:

#SBATCH –-partition=normal

with this:

#SBATCH –-partition=debug

1. Once your batch job has finished running, ﬁnd the *standard output* and *standard error* files from your job:

**ls -ltr**

Using this command, you should see files named

hello\_world\_mpi\_#######\_stdout.txt

and

hello\_world\_mpi\_#######\_stderr.txt

(where ####### is replaced by the batch job ID).

These files should contain the output of hello\_world\_mpi. Ideally, the stderr file should have length zero, but we sometimes see some minor warning messages.

1. Look at the contents of the standard output file:

**cat hello\_world\_mpi\_#######\_stdout.txt**

(where ####### is replaced by the batch job ID).

You may want to look at the stderr file as well:

**cat hello\_world\_mpi\_#######\_stderr.txt**

1. If this run had **ANY** problems, then send e-mail to:

[support@oscer.ou.edu](mailto:support@oscer.ou.edu)

which reaches all OSCER staff (including Henry), and attach the following files:

make\_cmd

makefile

hello\_world\_mpi.c

hello\_world\_mpi.sbatch

hello\_world\_mpi\_#######\_stdout.txt

hello\_world\_mpi\_#######\_stderr.txt

1. Submit the batch job again, using steps 20-24, above. Compare the output of the first run to the output of this second run. What do you notice about the hello world messages?
2. Edit your batch script to change the number of MPI processes.

Specifically, edit the following two batch directives so that they both have the same number, which should be between 1 and 20:

#SBATCH --ntasks=20

#SBATCH --ntasks-per-node=20

#SBATCH -n 16

#SBATCH -R "span[ptile=16]"

For example, you might change them to:

#SBATCH --ntasks=10

#SBATCH --ntasks-per-node=10

#SBATCH -n 12

#SBATCH -R "span[ptile=12]"

**NOTE**: **DON’T CHOOSE VALUES BELOW 1 OR ABOVE 20**, and make sure that the values for these two batch directives **EXACTLY MATCH**.

1. Submit the batch job again, using steps 20-24, above. Compare the output of the first and second runs to the output of this third run. What do you notice about the hello world messages?
2. Repeat steps 26-27 with at least two more values between 1 and 20, comparing the outputs for the various runs.
3. Repeat steps 26-27, but with values of n greater than 20. For **ANY** value of n greater than 20, set:

#SBATCH --ntasks-per-node=20

#SBATCH -R "span[ptile=16]"

This batch directive means, “put as many as 20 MPI processes on each compute node.”