

MPI on Cirrus and ARCHER

Reusing this material



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

http://creativecommons.org/licenses/by-nc-sa/4.0/deed.en_US

This means you are free to copy and redistribute the material and adapt and build on the material under the following terms: You must give appropriate credit, provide a link to the license and indicate if changes were made. If you adapt or build on the material you must distribute your work under the same license as the original.

Note that this presentation contains images owned by others. Please seek their permission before reusing these images.

Access

- CP-Lab machines directly accessible in lab room 1028
- external access:
 - gateway: `ssh -Y user@ph-cplab.ph.ed.ac.uk`
 - then: `ssh -Y cplabXYZ` (pick your favourite machine, e.g. 066)
 - gateway machine has very limited software available
- ARCHER: `ssh -Y user@login.archer.ac.uk`
- Cirrus: `ssh -Y user@cirrus-msc.epcc.ed.ac.uk`
 - you must use this dedicated MSc login node
- You can access systems using ssh from anywhere
 - Trivial for Linux
 - Mac: enable the X server (xquartz) to display any graphics
 - Windows: need to install an X server program, e.g. xming (which is free!)

Useful files and templates

- Take a copy of `MPP-templates.tar`
 - see the course web pages
- unpack: `tar xvf MPP-templates.tar`
- Crib sheets for MPI programs available on Learn under “problems sheets”

Compiling MPI Programs on CP-Lab

- Must load MPI first: `module load mpi/mpich-x86_64`
- Fortran programmers use `mpif90`
- C programmers use `mpicc`
- There is nothing magic about these MPI compilers!
 - simply wrappers which automatically include various libraries etc
 - compilation done by standard (here GNU) compilers: gcc and gfortran
- You can use the supplied Makefiles for convenience
 - `make -f Makefile_c`
 - `make -f Makefile_f90`
- Easiest to make a copy of one of these called “Makefile”
 - also need to change the line “MF=” in the Makefile itself

Running interactively on CP-Lab

- Timings will not be reliable
 - shared with other users, many more processes than processors
 - but **very useful** during development and for debugging
- `mpirun -n 4 ./mpiprogram.exe`
 - runs your code on 4 processes
- NOTE
 - output might be buffered
 - if your program crashes, you may see no output at all
- May need to explicitly flush prints to screen
 - `FLUSH (6)`
 - `fflush(stdout) ;`
- There is no batch system on the CP-Lab machines

Compiling MPI Programs on Cirrus

- Fortran programmers use `mpif90`
- C programmers use `mpicc`
- There is nothing magic about these MPI compilers!
 - simply wrappers which automatically include various libraries etc
 - compilation done by standard (e.g. Intel) compilers
 - `icc` and `ifort`
- You can use the supplied Makefiles for convenience
 - `make -f Makefile_c`
 - `make -f Makefile_f90`
- Easiest to make a copy of one of these called “Makefile”
 - also need to change the line “MF=” in the Makefile itself

Running interactively on Cirrus

- Timings will not be reliable
 - shared with other users, many more processes than processors
 - but **very useful** during development and for debugging
- `mpirun -n 4 ./mpiprogram.exe`
 - runs your code on 4 processes
- NOTE
 - output might be buffered
 - if your program crashes, you may see no output at all
- May need to explicitly flush prints to screen
 - `FLUSH (6)`
 - `fflush(stdout) ;`

Running batch jobs on Cirrus

- Run via a batch system
 - on Cirrus we use the Portable Batch System (PBS)
 - submit a script that then launches your program
- In **MPP-templates/** is a standard batch script: **cirrusmpi.pbs**
 - make a copy of this file with a name that matches your executable, e.g.
 - `user@cirrus$ cp cirrusmpi.pbs hello.pbs`
- To run on 4 processors: **qsub hello.pbs**
 - automatically runs executable called “hello”
 - output will appear in a file called `hello.pbs.oXXXXXX`
 - can follow job progress using `qstat` or `qstat -u $USER`
 - script also times your program using the Unix “time” command
 - full instructions included as comments in the template
 - no need to alter the script - just rename it as appropriate
 - e.g. to run a program “pingpong” make another copy called “pingpong.pbs”

Cirrus idiosyncrasies

- By default, MPI wrappers are not in your path

```
user@cirrus$ mpicc  
-bash: mpicc: command not found
```

- To access correct version: `module load mpt`
-in batch system, job launcher is called `mpiexec_mpt`
- Intel compilers: `module load intel-compilers-16`
-add these to end of your `.bash_profile` file in home directory
-to check you have the right version (similarly for `mpif90`)

```
user@cirrus$ which mpicc  
/opt/sgi/mpt/mpt-2.14/bin/mpicc
```

- `mpif90` automatically picks up the Intel Fortran compiler
- to use Intel C [C++] compilers: `mpicc -cc=icc [-cc=icpc]`

Compiling MPI Programs on ARCHER

- Fortran programmers use `ftn`
- C programmers use `cc`
- There is nothing magic about these MPI compilers!
 - simply wrappers which automatically include various libraries etc
 - compilation done by standard (Cray) compilers
 - `crayftn` and `craycc`
- You can use the supplied Makefiles for convenience
 - `make -f Makefile_c`
 - `make -f Makefile_f90`
- Easiest to make a copy of one of these called “Makefile”
 - also need to change the line “MF=” in the Makefile itself

ARCHER idiosyncracies

- ▶ Not possible to run directly on front-end
- ▶ Can be a substantial delay in batch queues
 - we may sometimes have dedicated queues for the course
 - instant turnaround!
- ▶ Cannot run from the home file system
 - back-end nodes can only see the work file system
- ▶ Recommendation
 - do everything in `/work/`
 - i.e. change directory to `/work/y14/y14/guestXX/`

Running on ARCHER back-end

- Run via a batch system
 - on ARCHER we use the Portable Batch System (PBS)
 - submit a script that then launches your program
- In `MPP-templates/` is a standard batch script: `mpibatch.pbs`
 - make a copy of this file with a name that matches your executable, e.g.
 - `user@archer$ cp archermpi.pbs hello.pbs`
- Submit: `qsub -q short hello.pbs`
 - short queue is for jobs less than 20 minutes
 - you will need to alter `NPROCS` (the argument to “`aprun`”) by hand
 - ... and `select` more than one node for more than 24 processes
 - output will appear in a file called `hello.pbs.oXXXXX`
 - can follow job progress using `qstat` command
 - script also times your program using the Unix “`time`” command
 - full instructions included as comments in the template

C++ Interface

- MPI is not an OO interface
 - however, can be called from C++
- Originally had different function calls, eg:
 - `MPI::Intracomm comm;`
 - `...`
 - `MPI::Init();`
 - `comm = MPI::COMM_WORLD;`
 - `rank = comm.Get_rank();`
 - `size = comm.Get_size();`
- Compiler is called `mpicxx`
 - see `hello.cc` and `Makefile_cc`

C++ interface is
now removed

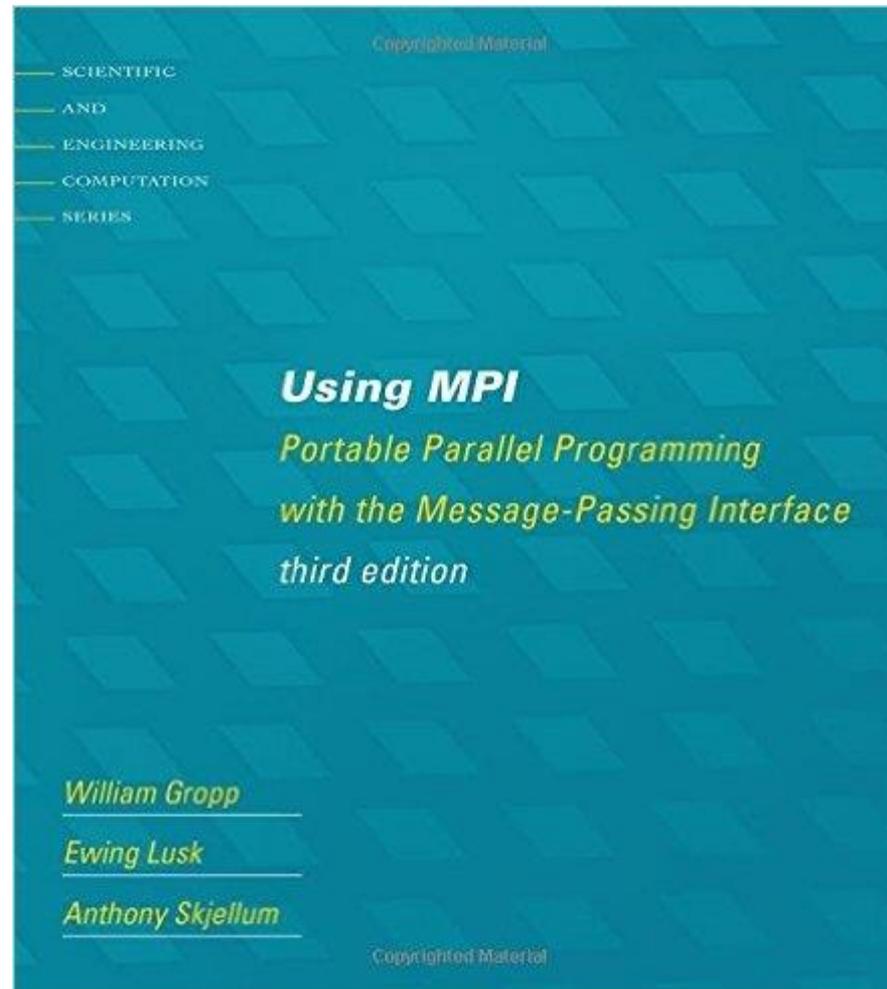
Must therefore
cross-call to C

Documentation

- ▶ MPI Standard available online
 - See: <http://www.mpi-forum.org/docs/>
 - currently version 3.1
- ▶ Available in printed form
 - <http://www.hlrs.de/mpi/mpi31/>
- ▶ Man pages available on CP-Lab and ARCHER
 - must use the C style of naming: `man MPI_Routine_name`, e.g.:
 - `user@computer$ man MPI_Init`



MPI Books



Exercise: Hello World

The minimal MPI program

- See Exercise 1 on the exercise sheet
- Write an MPI program that prints a message to the screen
- Main purpose is to get you compiling and running parallel programs on ness
 - also illustrates the SPMD model and use of basic MPI calls
- We supply some very basic template code
 - see pages 4 and 5 of the notes as well