An introduction to parallel programming using Message Passing with MPI

Birgitte Brydsø

HPC2N, Umeå University

25-28 October 2021

Compiling and running MPI on Kebnekaise at HPC2N

Overview

- Connecting to HPC2N's systems (Kebnekaise) with ThinLinc
- Editors
- The filesystem
- The module environment
- Compiling and linking MPI executables
- Executing MPI jobs on Kebnekaise
  - SLURM
  - Jobscript examples
- Your first MPI program

Compiling and running MPI on Kebnekaise at HPC2N

Connecting to HPC2N's systems with ThinLinc

ThinLinc is a cross-platform remote desktop server developed by Cendio AB. It is especially useful for software with a graphical interface.

- Download the client and install it: `https://www.cendio.com/thinlinc/download`.
- Start the client. Enter the name of the server: **kebnekaise-tl.hpc2n.umu.se**. Enter your HPC2N username.
- (First time only) Go to "Options" $->$ "Security". Check that authentication is set to password.
- (First time only) Go to "Options" $->$ "Screen". Uncheck "Full screen mode".
- Enter your HPC2N password. Click "Connect"
- Click "Continue" when told that the server's host key is not in the registry. Wait for the ThinLinc desktop to open.

Editing your files

- Various editors: vi, vim, nano, emacs ...
- Example, vi/vim:
  - `vi <filename>`
  - Insert before: i
  - Save and exit vi/vim: `Esc :wq`
- **Example, nano:**
  - `nano <filename>`
  - Save and exit nano: `Ctrl-x`
- Example, Emacs:
  - Start with: emacs
  - Open (or create) file: Ctrl-x Ctrl-f
  - Save: Ctrl-x Ctrl-s
  - Exit Emacs: Ctrl-x Ctrl-c

More info here: http://www.hpc2n.umu.se/filesystems/overview

	Project storage	**$HOME (25GB)**	**/scratch**
Recommended for batch jobs	Yes	(No, size)	Yes
Backed up	No	Yes	No
Accessible by batch system	Yes	Yes	Yes (node only)
Performance	High	High	Medium
Default readability	Group only	Owner	Owner
Permissions management	chmod, chgrp, ACL	chmod, chgrp, ACL	N/A for batch jobs
Notes	Storage your group get allocated through the storage projects	Your home-directory	Per node

Compiling and running MPI on Kebnekaise at HPC2N

The Module Environment

Most programs are accessed by first loading them as a 'module'

Modules are:

- used to set up your environment (paths to executables, libraries, etc.) for a particular (set of) software package(s)
- for helping users manage their Unix/Linux shell environment, allowing groups of related environment-variable settings to be made or removed dynamically
- allows having multiple versions of a program or package available by just loading the proper module
- installed in a hierarchial layout; thus some modules are only available after loading a specific compiler and/or MPI version

Compiler toolchains load software-bundles for a complete environment (compiling, using prebuilt software). Includes some/all of: compiler suite, MPI, BLAS, LAPACK, ScaLapack, FFTW, CUDA.

- **foss**: GCC, OpenMPI, OpenBLAS/LAPACK, FFTW, ScaLAPACK
- **intel**: icc, ifort, IntelMPI, IntelMKL

Compiling and running MPI on Kebnekaise at HPC2N

Compiling and linking MPI, example: intel/2021a and foss/2021a

Loading the module (`ml spider intel` and `ml spider foss` for other versions)

Compiler	Rec. toolchain	Loading
Intel	intel/2021a	module load intel/2021a ml intel/2021a
GNU	foss/2021a	module load foss/2021a ml foss/2021a

Unloading all modules: `ml purge`

Compiling

Compiler	Language	Compiling
Intel	C C++ Fortran	mpiicc mpiicpc mpiifort
GNU	C C++ Fortran	mpicc mpicxx/mpiCC mpifort

- MPI C program
  - `mpicc -O3 -march=native -o program program.c`
- MPI F90 program
  - `mpif90 -O3 -march=native -o program program.f90`
- The options in both cases mean:
  1. `-O3` optimization level 3
  2. `-march=native` optimized for our CPUs
  3. `-o program` names the ouput 'program'
- You can use the options for the underlying compiler

- MPI C program
  - `mpiicc -O3 -xHost -o program program.c`
- MPI F90 program
  - `mpiifort -O3 -xHost -o program program.f90`
- The options in both cases mean:
  1. `-O3` optimization level 3
  2. `-xHost` builds for our CPUs
  3. `-o program` names the output 'program'
- You can use the options for the underlying compiler

- Ensure the compiler and MPI modules are loaded (`foss/2021a` or `intel/2021a`)
- MPI jobs must be run through the batch system (SLURM)
  - This will be the case for most (all?) HPC systems
- When submitting jobs to the batch system, you **must** use the course project!
- To run the MPI program, you use `srun -n ./program` inside the batch script.
  - `-n` is the number of MPI tasks. It can be 1 to number of tasks you asked for when submitting the job. If omitted, you will use the number of tasks asked for in the job.

- Large/long/parallel jobs must be run through the batch system
- SLURM is an Open Source job scheduler, which provides three key functions
  - Keeps track of available system resources
  - Enforces local system resource usage and job scheduling policies
  - Manages a job queue, distributing work across resources according to policies
- In order to run a batch job, you need to create and submit a SLURM submit file (also called a batch submit file, a batch script, or a job script).
- Guides and documentation at: http://www.hpc2n.umu.se/support

Compiling and running MPI on Kebnekaise at HPC2N

Useful commands to the Batch System (SLURM)

- Submit job: `sbatch <jobscript.sh>` (successful submission returns a job-id number)

```
b-an01 [~/store/bbrydsoe/MPI]$ sbatch mpi_hello.sh
Submitted batch job 16078379
```

- As default, output/errors are found in `slurm-<job-id>.out`
- Get list of all jobs: `squeue`
- Get list of only your jobs: `squeue -u <username>`

```
b-an01 [~/store/bbrydsoe/MPI]$ squeue -u bbrydsoe
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          16078452    single mpi_hell bbrydsoe CF       0:00      1 b-cn0208
          16078453    single mpi_hell bbrydsoe CF       0:00      1 b-cn0208
          16078449    single mpi_hell bbrydsoe CG       0:06      1 b-cn0206
          16078450    single mpi_hell bbrydsoe  R       0:03      1 b-cn0802
          16078451    single mpi_hell bbrydsoe  R       0:03      1 b-cn0802
```

- Adding the flag `--start` to `squeue` gives the estimated job start time. This can change depending on other people's jobs.
- Check on a specific job: `scontrol show job <job id>`
- Delete a specific job: `scancel <job id>`
- Delete all your jobs: `scancel -u <username>`

Compiling and running MPI on Kebnekaise at HPC2N

SLURM batch script for a standard MPI job

```
#!/bin/bash
#SBATCH -A SNIC2021-22-733 #Project id
#SBATCH -J my-mpi-job #Name of job
#SBATCH --time=00:05:00 #Jobtime (HH:MM:SS) Max:  168H
#SBATCH -o mpijob_%j.out #Naming the output file
#SBATCH -e mpijob_%j.err #Naming the error file
#SBATCH -n 14 #Number of tasks.

ml purge < /dev/null 2>&1
ml foss/2021a

#srun -8 ./my_mpi_program # only use 8 of the tasks
srun ./my_mpi_program
```

Compiling and running MPI on Kebnekaise at HPC2N

Comments: SLURM batch script for a standard MPI job

- Use `srun` for running MPI jobs
- Sometimes you get better performance with core-binding. This can be achieved by adding the SLURM option `--cpu-bind`
- If you need more than the max. number of cores in one node (28 for kebnekaise), you may want to set the number of tasks per node, depending on your job (eg. `#SBATCH --ntasks-per-node=<m>`. This is the (minimum) number of tasks allocated per node.

Compiling and running MPI on Kebnekaise at HPC2N

Python/MPU4PY

- To run Python and MPI for Python, first load a module
  - `SciPy-bundle/2021.05` is compatible with `foss/2021a`, so we will be using this
  - Loading (after first loading `foss/2021a`): `ml SciPy-bundle/2021.05`
- You can then use it in an MPI program (hello.py) like this

```
from mpi4py import MPI

comm = MPI.COMM_WORLD
rank = comm.Get_rank()
size = comm.Get_size()

print('Hello from process {} out of
{}'.format(rank, size))
```

- Running (inside submit file): srun -n <tasks> python hello.py

Compiling and running MPI on Kebnekaise at HPC2N

Various useful info

- A project has been set up for the workshop: `SNIC2021-22-733`
- Use it by adding this to your submit file:
  `#SBATCH -A SNIC2021-22-733`
- The project is ONLY valid during the course and a few weeks after.
- There is a reservation. To use, add this to the submit file:
  MONDAY
  `#SBATCH --reservation=snic2021-22-733-day1`
  TUESDAY
  `#SBATCH --reservation=snic2021-22-733-day2`
  WEDNESDAY
  `#SBATCH --reservation=snic2021-22-733-day3`
  THURSDAY
  `#SBATCH --reservation=snic2021-22-733-day4`
- The reservation is ONLY valid for the specific day of the course.