Running Parallel Applications on Cedar Supercomputer

Getting started

Login:

```
$ ssh <username>@cedar.computecanada.ca<ENTER>
```

Interactive node request:

```
$ salloc --time=1:0:0 --ntasks=16 --mem-per-cpu=1GB --account=def-mludin<ENTER>
```

Download code:

```
$ wget http://shodor.org/~mludin/BW_Capstone/running_code_cluster2.tar<ENTER>
```

Extract the tar Files:

```
$ tar -xvvf running_code_cluster2.tar<ENTER>
```

Change folders:

```
$ cd running_code_cluster2/<ENTER>

$ ls -l
```

Introduction to Compute Canada

Provides Advanced Research Computing services and infrastructure for Canadian researchers and their associated collaborators from different industries and academic.

Any Faculty member in Canada can register with CC.

Registered faculty can sponsor students, research staff, and collaborators

Research projects from across Canada as well as international collaborations from around the world uses CC resources

To Register: https://ccdb.computecanada.ca

For More information: www.computecanada.ca

Quick Overview of CC Hardware (`$ numactl --hardware`)

Cedar has a total of 94,528 CPU cores for computation, and 1352 GPU devices.

# Nodes	# Cores	Available memory	CPU	Storage	GPU
768	48	187G or 192000M	2 x Intel Platinum 8260 Cascade Lake @ 2.4Ghz	2 x 480G SSD	-
640	48	187G or 192000M	2 x Intel Platinum 8160F Skylake @ 2.1Ghz	2 x 480G SSD	-
576	32	125G or 128000M	2 x Intel E5-2683 v4 Broadwell @ 2.1Ghz	2 x 480G SSD	-
192	32	187G or 192000M	2 x Intel Silver 4216 Cascade Lake @ 2.1GHz	1 x 480G SSD	4 x NVIDIA V100 Volta (32G HBM2 memory)
128	32	250G or 257000M	2 x Intel E5-2683 v4 Broadwell @ 2.1Ghz	2 x 480G SSD	-
114	24	125G or 128000M	2 x Intel E5-2650 v4 Broadwell @ 2.2GHz	1 x 800G SSD	4 x NVIDIA P100 Pascal (12G HBM2 memory)
32	24	250G or 257000M	2 x Intel E5-2650 v4 Broadwell @ 2.2GHz	1 x 800G SSD	4 x NVIDIA P100 Pascal (16G HBM2 memory)
24	32	502G or 515000M	2 x Intel E5-2683 v4 Broadwell @ 2.1Ghz	2 x 480G SSD	-
24	32	1510G or 1547000M	2 x Intel E5-2683 v4 Broadwell @ 2.1Ghz	2 x 480G SSD	-
4	32	3022G or 3095000M	4 x Intel E7-4809 v4 Broadwell @ 2.1Ghz	2 x 480G SSD	-

https://docs.computecanada.ca/wiki/Cedar#Node_characteristics

Software Environment on Cedar

Operating system and job scheduler:

CentOS 7 Linux, SLURM job scheduler

Programming Languages and Compilers

C, C++, Fortran, Python, Java, Matlab

Parallel programming support

MPI, OpenMP, OpenACC, CUDA, OpenCL

There are different versions and flavours of compilers and development environment supported on both clusters.

Contact support@computecanada.ca if you would like a software package installed for you or if you want to compile and install it in your own space.

Compute Canada Clusters' Filesystem

Compute Canada provides different storage options for diverse user needs such as:

- High-speed temporary storage
- Long-term storage
- You can copy files to Home Space, but only able to run parallel applications under scratch and project space

Filesystem	Default Quota	Lustre-based?	Backed up?	Purged?	Available by Default?	Mounted on Compute Nodes?
Home Space (NFS) /home	50 GB and 500K files per user[1]	Yes	Yes	No	Yes	Yes
Scratch Space Short-Term Parallel /scratch	20 TB and 1M files per user	Yes	No	Files older than 60 days are purged.[2]	Yes	Yes
Project Space Long-Term Parallel /project	1 TB and 500K files per group[3]	Yes	Yes	No	Yes	Yes
Nearline Space /nearline	2 TB and 5000 files per group	No	N/A	No	Yes	No

https://docs.computecanada.ca/wiki/Storage_and_file_management

Using Module

Modules uses *lmod* software which ensures that only compatible set of configuration and software packages or versions are loaded at any one time.

- List all available modules
  ```
  $ module avail
  ```
- List currently loaded modules
  ```
  $ module list
  ```
- Get more information about the currently loaded version of the module
  ```
  $ module spider StdEnv/2016.4
  ```
- List all versions of specific software
  ```
  $ module spider StdEnv
  ```
- Search for all possible modules
  ```
  $ module keyword key1 key1
  $ module keyword intel mpi
  ```
- To Load a module
  ```
  $ module load pgi/19.4
  ```
- To swap between two versions of a software package
  ```
  $ module swap StdEnv/2016.4
  StdEnv/2018.4
  ```

Using SLURM Scheduler

Submitting Jobs

Submitting a job is accomplished using a command named **salloc**. This command allows us to submit two different kinds of jobs: **interactive** jobs and **batch** jobs. Interactive jobs are more common for compiling, testing and running R applications.

Interactive Jobs:

```
$ salloc --time=<hours>:<minutes>:<seconds> --nodes=<# of compute nodes> --ntasks=<# of processes>\
--cpus-per-task=<# cpus/process/thread> --mem-per-cpu=<amount of RAM per core>\
--account=<def-username><ENTER>
```

```
$ salloc --time=0:20:0 --nodes=2 --ntasks=64 --cpus-per-task=1 --mem-per-cpu=1GB
--account=def-mludin<ENTER>
```

OUTPUT:

```
[mludin@cedar1 bw_capstone]$ salloc --time=0:20:0 --nodes=2 --ntasks=64 --cpus-per-task=1
--mem-per-cpu=1GB --account=def-mludin
salloc: Granted job allocation 45535851
salloc: Waiting for resource configuration
salloc: Nodes cdr[768,774] are ready for job
```

Examples:

[mpi_example.c]

```
$ less mpi_example.c
```

How to compile:

```
$ make mpi_example
```

How to run:

```
$ srun -n 4 ./mpi_example.exe
```

[mpi_reduce.c]

```
$ less omp_pi_area.c
```

How to compile:

```
$ make omp_pi_area
```

How to run:

```
$ export OMP_NUM_THREADS=8
$ srun -n 8 ./omp_pi_area.exe
```

[mpi_pi_area.c]

```
$ less mpi_pi_area.c
```

How to compile:

```
$ make mpi_pi_area
```

How to run:

```
$ srun -n 8 ./mpi_pi_area.exe
```

[acc_laplace.c]

```
$ less acc_laplace.c
```

How to compile:

```
$ make acc_laplace
```

How to run:

```
$ srun ./acc_laplace.exe
```

```
#!/bin/bash
#SBATCH --account=def-someuser
#SBATCH --job-name=acc_laplace
#SBATCH --gres=gpu:p100:1
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --cpus-per-task=1
#SBATCH --mem-per-cpu=1024M
#SBATCH --time=00:00:05
export OMP_NUM_THREADS=$SLURM_CPUS_PER_TASK
echo "Hostname is: `hostname`"
echo "Nvidia-smi info is:"
nvidia-smi
echo "Current working directory is: `pwd`"
srun ./acc_laplace.exe   # mpirun or mpiexec also works
```

References / Further Readings

- Compute Canada Documentation Wiki Page
- Running Jobs
- Using Modules
- Using Globus on Cedar
- How to SSH to Cedar
- Open MPI Organization/Community
- Top500 List of supercomputers in the world
- MPI Library Man Pages
- MPI Standards
- MPI Tutorials
- More MPI Tutorials