CIS 431/531
Intro to Parallel Computing

Talapas 101

System Access

Talapas

The Talapas supercomputer is a High Performance Computing (HPC) cluster with approximately 9,500 cores, 90 TB memory, 120 GPUs, 2 PB storage.

Utilizing HPC clusters

- Login node
  - Gateway to accessing the system via ssh
  - This is where you can develop code and initiate your experiments
- Compute node
  - For actual computation
  - Accessible by using a) **an interactive sessions,** or b) **submitting a "job"**
  - **Do not develop code on a compute node**

Hardware

- GPU nodes (Murdock): (24) 48 core nodes with (54) A100 GPUs
- CPU nodes : (24) 128 core nodes with 512 GB RAM
- Large memory nodes: (2) 1TB RAM, (4) 2TB RAM, (2) 4TB RAM nodes
- New infrastructure nodes

Partitions

Nodes are divided into partitions, depending on their role

- compute
- computelong

} Jobs that require only CPUs

- gpu
- gpulong

} Jobs that require GPUs

- interactive
- interactivegpu

} Jobs that require interactive use of a node

- memory
- memorylong

} Jobs that require large amounts of memory

Use `sinfo` to see a list of available partitions and their state

Information

How-to Guide

e.g. how to submit a OpenMP job, GPU job, MPI job, etc.

https://hpcrcf.atlassian.net/wiki/spaces/TW/pages/2755756444/How-to+articles

Executing Code

lbnl.srun

```
#!/bin/bash
#SBATCH --account=cis431_531                ### your 'charge' account
#SBATCH --partition=compute                  ### queue to submit to
#SBATCH --job-name=my_test                   ### job name
#SBATCH --output=output/test_%A.out          ### file in which to store job stdout
#SBATCH --error=output/test_%A.err           ### file in which to store job stderr
#SBATCH --time=15                            ### wall-clock time limit, in minutes
#SBATCH --mem=64000M                         ### memory limit per node (K | M | G | T)
#SBATCH --nodes=1                            ### number of nodes to use
#SBATCH --ntasks-per-node=1                  ### number of MPI tasks to launch per node
#SBATCH --cpus-per-task=14                   ### number of CPUs for each task


module load mkl
export KMP_AFFINITY=granularity=fine,compact,1
export OMP_NUM_THREADS=56

./build/Linux-x86_64/bin/splatt cpd --stream=5- -f=0.999 --reg=frob,1e-3,5 -r=10 -t=28 ../hpctensor/lbnl-network.tns
```

Module (software)

```
[jeec@talapas-ln1 ~]$ module list

Currently Loaded Modules:
  1) intel/17   2) openmpi/2.1   3) mkl   4) cuda/9.2


[jeec@talapas-ln1 ~]$ module available mkl

----------------- /packages/modulefiles/Compiler/intel/17 ----------------
   mkl (L)

  Where:
   L:  Module is loaded

Use "module spider" to find all possible modules.
Use "module keyword key1 key2 ..." to search for all possible modules
matching any of the "keys".
```

Partition

```
[jeec@talapas-ln1 job_scripts]$sinfo -s
PARTITION       AVAIL  TIMELIMIT   NODES(A/I/O/T) NODELIST
compute            up 1-00:00:00       23/1/0/24 n[0173-0196]
computelong        up 14-00:00:0       16/0/0/16 n[0177-0192]
gpu                up 1-00:00:00        7/9/8/24 n[0149-0172]
gpulong            up 14-00:00:0        6/1/5/12 n[0152-0157,0164-0169]
interactive        up    4:00:00         0/4/0/4 n[0197-0199,0302]
interactivegpu     up    4:00:00         1/0/0/1 n0161
memory             up 1-00:00:00         2/6/0/8 n[0141-0148]
memorylong         up 14-00:00:0         2/2/0/4 n[0142,0144,0146,0148]
preempt            up 7-00:00:00     48/44/8/100 n[0141-0189,0191-0196,0314-0358
]
```

Job Name

```
[jeec@talapas-ln1 mpi_test]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          13333831      long BayesTra  dreuter  R 20-21:41:50      1 n073
          13333834      long BayesTra  dreuter  R 20-21:41:36      1 n073
          13333842      long BayesTra  dreuter  R 20-21:41:13      1 n073
          13333862      long BayesTra  dreuter  R 20-21:40:03      1 n074
          13333865      long BayesTra  dreuter  R 20-21:39:54      1 n074
          13333870      long BayesTra  dreuter  R 20-21:39:42      1 n074
          13333877      long BayesTra  dreuter  R 20-21:39:12      1 n058
          13333881      long BayesTra  dreuter  R 20-21:39:00      1 n058
          13333887      long BayesTra  dreuter  R 20-21:38:39      1 n058
          13333942      long BayesTra  dreuter  R 20-21:34:16      1 n058
          13333943      long BayesTra  dreuter  R 20-21:34:12      1 n058
          13333944      long BayesTra  dreuter  R 20-21:34:06      1 n075
          13333948      long BayesTra  dreuter  R 20-21:33:51      1 n075
          13333951      long BayesTra  dreuter  R 20-21:33:45      1 n075
          13333954      long BayesTra  dreuter  R 20-21:33:39      1 n075
          13333972      long BayesTra  dreuter  R 20-21:32:44      1 n059
          13333977      long BayesTra  dreuter  R 20-21:32:25      1 n059
          13333979      long BayesTra  dreuter  R 20-21:32:19      1 n059
          13334008      long BayesTra  dreuter  R 20-21:30:46      1 n076
          13335103      long BayesTra  dreuter  R 20-20:58:35      1 n061
          13335111      long BayesTra  dreuter  R 20-20:58:15      1 n061
          13335112      long BayesTra  dreuter  R 20-20:58:10      1 n061
          13335142      long BayesTra  dreuter  R 20-20:56:28      1 n016
          13335153      long BayesTra  dreuter  R 20-20:55:46      1 n063
          13335156      long BayesTra  dreuter  R 20-20:55:33      1 n063
```

Output

```
#SBATCH --output=output/mpi_test_%A.out ### file in which to store job stdout
#SBATCH --error=output/mpi_test_%A.err  ### file in which to store job stderr
```

```
-rw-r--r-- 1 jeec talapas    0 Jul 21 14:05 mpi_test_9619097.err
-rw-r--r-- 1 jeec talapas  936 Jul 21 14:05 mpi_test_9619097.out
```

You can customize the output file name (e.g., append node ID or task ID it was executed on).

Time and Memory

```
#SBATCH --time=15                    ### wall-clock time limit, in minutes
#SBATCH --mem=64000M                 ### memory limit per job
```

Time

- Job ends when your application finishes
- When it crashes, it may continue running
- Good to set a time limit, so you don't waste your money if it crashes and runs to maximum allowed time

Memory

- Typically used only when you have some special memory needs
- If not specified, default memory for the node is used
- --mem is typically used for single node job (it's more common to allocate memory per task for multi-node jobs)

Computing Resources

```
#SBATCH --nodes=1                       ### number of nodes to use
#SBATCH --ntasks-per-node=1             ### number of tasks to launch per node
#SBATCH --cpus-per-task=14              ### number of CPUs for each task
```

Number of **nodes**, number of MPI tasks per node, and number of **cores** for each task

Here, cpus mean **physical** CPU cores (not hyperthreads)

Actually Executing Your Code

```
module load mkl
export KMP_AFFINITY=granularity=fine,compact,1
export OMP_NUM_THREADS=56

./build/Linux-x86_64/bin/splatt cpd --stream=5- -f=0.999 --reg=frob,1e-3,5 -r=10 -t=28
../hpctensor/lbnl-network.tns
```

SLURM

Simple Linux Utility for Resource Management

Basically a job scheduler

```
[jeec@talapas-ln1 mpi_test]$ sbatch lbnl.srun
```

Remember

When calculating performance numbers, take average or median for statistically meaningful numbers

- For this class, please use average (i.e., **arithmetic mean**) of **20** tests/runs
- You can run the code 20 times within one batch file, or run the batch files 20 times

DO NOT WASTE RESOURCES!!

- The department is paying for this and there is a maximum budget

  If you follow the rules, you should be fine

  a) Develop code on local machine or login node
  b) Set time limits on tests on compute nodes

Manual

https://hpcrcf.atlassian.net/wiki/spaces/TW/pages/2755758386/Release+Notes+for+the+new+2023+Talapas