Ada User Guide

From HPC

Contents

Overview

Ada cluster consists of forty-two Boston SYS-7048GR-TR nodes equipped with dual Intel Xeon E5-2640 v4 processors, providing 40 virtual cores per node, 128 GB of 2400MT/s DDR4 ECC RAM and four Nvidia GeForce GTX 1080 Ti GPUs, providing 14336 CUDA cores, and 44 GB of GDDR5X VRAM and twenty Tyrone SYS-7048GR-TR nodes equipped with dual Intel E5-2640 v4 processors providing 40 virtual cores per node, 128 GB of 2400MT/s DDR4 ECC RAM and four Nvidia GeForce GTX 2080 Ti GPUs providing 17408 cores, and 44 GB of GDDR6 VRAM. The nodes are connected to each other via a Gigabit Ethernet network. All compute nodes have a 1.8 TB local scratch and a 960 GB local SSD scratch. The compute nodes are running Ubuntu 16.04 LTS. SLURM (https://slurm.schedmd.com/) software is used as job scheduler and resource manager. The aggregate theoretical peak performance of Ada is 47.62 TFLOPS (CPU) + 2984 TFLOPS (FP32 GPU).

Applying for an account

An Ada account is available to IIIT faculty, research staff and research students. To apply for a new account, please send an email to hpc@iiit.ac.in (mailto:hpc@iiit.ac.in) with the following information.

- Name
- Roll Number / Employee ID
- Research Center
- Faculty Advisor
- Preferred Login ID

For a CVIT associated account, fill out this form (https://goo.gl/forms/XnOVvteVFbmT2OdQ2). The form requires that you sign-in using your G-suite account, through college ID.

Accessing Ada

Logging in

You can log in to Ada by SSH from IIIT LAN. For accessing Ada from off-campus, IIIT VPN (https://vpn.iiit.ac.in/) must be used.

```
$ ssh -X user_name@ada.iiit.ac.in
```

Moving files to and from Ada

To move a directory from local machine to Ada:

```
$ scp -r local_directory user_name@ada.iiit.ac.in:
(or)
$ rsync -avz local_directory user_name@ada.iiit.ac.in:
```

To move a directory from Ada to local machine:

```
$ scp -r user_name@ada.iiit.ac.in:remote_directory .
(or)
$ rsync -avz user_name@ada.iiit.ac.in:remote_directory .
```

Both **scp** and **rsync** transfers files locally or over network. If the transfer is interrupted, **rsync** has the ability to continue from where it left off when invoked again.

Partitions, Account, and QoS

A partition can be considered as a collection of nodes. There are two partitions in the cluster. The **short** partition has two nodes and is for compiling / debugging codes. The **long** partition is for serial / parallel jobs that need to run for longer than 6 hours.

Partition	Nodes	DefMemPerCPU	MaxMemPerCPU	Gres	Maxtime	Priority
short	gnodes[01-02]	1024 MB	3000 MB	gpu:4	6:00:00	100
long	gnodes[03-40]	1024 MB	3000 MB	gpu:4	Infinite	100

A SLURM account is like a bank account and all users belong to at least one account. The allocated resources to a job are charged to the job's specified account. All users have access to **research** account. The accounts **cvit**, **nlp**, and **ccnsb** are accessible only to users in projects/centres that have contributed hardware to the cluster.

Account	Access	GrpCPUs	GrpTRES=gres/gpu	GrpJobs	GrpSubmitJobs	Allowed QoS
research	ALL	1640	164	820	1640	medium
cvit	CVIT	600	60	300	600	normal
mll	MLL	40	4	20	40	normal
nlp	NLP	80	8	40	80	normal
ccnsb	CCNSB	80	8	40	80	normal
cesp	CESP	40	4	20	40	normal

It is recommended to specify a Quality of Service (QOS) to each job submitted to SLURM. The default QOS for research account is **medium** and it is **normal** for cvit, mll, nlp, cesp and ccnsb accounts.

QoS	MaxCPUsPerUser	MaxTRESPerUser	MaxJobsPerUser	MaxSubmitJobsPerUser	MaxWall	MaxTresPerJob	Priority
low	10	gres/gpu=1	1	4	4-00:00:00	gres/gpu=1	0
medium	40	gres/gpu=4	4	8	4-00:00:00	gres/gpu=4	10
normal	Account limits	Account limits	Account limits	Account limits	Infinite	gres/gpu=4	0

The following command can be used to list the allowed Accounts and QoSes.

```
$ sacctmgr show assoc user=$USER format=Account,QOS,DefaultQOS
```

cvit account

To submit jobs to CVIT account, users should specify SLURM job directive **-A $USER**.

e.g.

```
$ sinteractive -c 2 -g 1 -A $USER
```

The following command will show the associations for an account.

```
$ sacctmgr show assoc account=$USER
```

sinteractive is non-standard and restrictive in terms of options and documentation. We recommend **sbatch** and **srun** for the same [1] (https://slurm.schedmd.com/sbatch.html) .

Sometimes we reserve nodes and give reservation names for dedicated access. Use a reservation argument to gain access. As of now, the reservation-name = **cvit-trial**

```
$ srun --reservation <reservation-name> ...
for e.g.
$ srun --reservation cvit-trial --pty --partition=long -A $USER --gres=gpu:2 --mem=10G -n 10 bash -l
```

sub account

A new SLURM account has been created on Ada to utilize the idle nodes/cores/GPUs. The details are as follows:

```
SLURM account: sub  (#SBATCH -A sub)
QOS:     sub        (#SBATCH --qos=sub)
Wall time: 6 hours  (#SBATCH --time=6:00:00)
MaxCPUs: 40
Max GPUs: 4
```

The jobs submitted using this account will have low priority and will run only when there are no high priority (QOS: normal,medium) pending jobs. t

File Systems

The cluster provides four types of file storage to users. They are referred here as /home, /share1, /scratch and /ssd_scratch.

The /home located at /home/$USER is a NFS storage and has a disk quota of 25 GB. This space can be used to store your source code and to build your executables. The data store on /home is backed up every day. The /share1 is a RAID6 storage available on master node and has a quota of 100 GB (CVIT users have a group quota of 6 TB). The space can be used for long-term storage and for transferring large data files to and from compute node /scratch. The /scratch is for storing temporary files created during job run time. The /ssd_scratch is for storing temporary files that required very fast disk I/O access. The files older than 10 days are purged from /scratch and /ssd_scratch.

Space	Purpose	Visibility	Backup	Quota	Total Size	File Deletion Policy
/home	Sofware installation space, storing codes and small files	Master and compute nodes	Yes	25 GB	9.8 TB	None
/share1	Long-term storage	Master node only	No	100 GB / 6 TB	20 TB	None
/share2	Long-term storage	Master node only	No		13 TB	None
/scratch	Temporary storage for large files	Local disk attached to each compute node	No	None	2.0 TB	10 days[#]
/ssd_scratch	Temporary storage for jobs that require fast I/O	Local disk attached to each compute node	No	None	960 GB	10 days[#]

[#] File deletion based on creation time (ctime).

Environment Modules

Environment module allows users to set shell environmental variables needed for a software.

To view the list of current loaded modules:

```
[parithi@ada ~]$ module list

Currently Loaded Modulefiles:
  1) openmpi/2.1.1   2) namd/2.12
```

To list the installed modules:

```
[root@ada ~]# module avail

---------------------------------------------------- /opt/Modules/versions -----------------------------------------------------
3.2.10

------------------------------------------- /opt/Modules/3.2.10/modulefiles -------------------------------------------
amber/16                          gflags/2.2.2                      namd/2.13
ceres-solver/1.14.0-165-gd7f428e  glog/0.3.5                        null
cmake/3.15.2                      glog/0.4.0                        openblas/0.3.6
colmap/3.6-dev.2-15-g6b6e825      gromacs/2016.3                    opencv/3.3.0
cuda/10.0                         gromacs/2016.3-plumed             openmpi/2.1.1
cuda/9.0                          gromacs/2019                      openmpi/3.1.0
cuda/9.1                          gromacs/2019.3-plumed             openmpi/4.0.0
cudnn/7.1-cuda-9.1                lammps/7Aug19                     openmpi/4.0.1-cuda10
cudnn/7.3-cuda-10.0               lammps/7Aug19-v2                  pcl/1.8.1
cudnn/7.6.4-cuda-9.0              leptonica/1.78.0                  plumed/2.5.2
cudnn/7.6-cuda-10.0               matlab/R2019b                     python/3.6.8
cudnn/7-cuda-10.0                 mkl/2019.3.199                    python/3.7.4
cudnn/7-cuda-9.0                  mkl/2019.4.243                    singularity/2.5.2
dot                               module-git                        tbb/2018u1-debug
eigen/3.3.7                       module-info                       tbb/2018u1-release
ffmpeg/4.0.1                      modules                           tesseract/4.1.0
ffmpeg/4.2.1                      mpich/3.3.1                       use.own
freesurfer/6.0.0                  mrtrix/3.0                        VTK/8.2.0
gflags/2.2.1                      namd/2.12
```

To load a module:

```
[parithi@ada ~]$ module load openmpi/2.1.1
```

To remove/unload a module

```
[parithi@ada ~]$ module unload openmpi/2.1.1
```

To display the changes made by a module to the user shell environment:

```
[parithi@ada ~]$ module disp cuda/8.0
-------------------------------------------------------------------
/opt/Modules/3.2.10/modulefiles/cuda/8.0:

module-whatis    adds CUDA-8.0 to your environment variable
append-path      PATH /usr/local/cuda-8.0/bin
append-path      LD_LIBRARY_PATH /usr/local/cuda-8.0/lib64
-------------------------------------------------------------------
```

List of Available Modules

Software	Module	Remarks
CUDA 9.0	cuda/9.0	
CUDA 9.1	cuda/9.1	
CUDA 10.0	cuda/10.0	
cuDNN 7	cudnn/7-cuda-9.0	CUDA 9.0
cuDNN 7.6.4	cudnn/7.6.4-cuda-9.0	CUDA 9.0
cuDNN 7.1	cudnn/7.1-cuda-9.1	CUDA 9.1
cuDNN 7	cudnn/7-cuda-10.0	CUDA 10.0
cuDNN 7.3	cudnn/7.3-cuda-10.0	CUDA 10.0
cuDNN 7.6	cudnn/7.6-cuda-10.0	CUDA 10.0
OpenBLAS	openblas/0.3.6	Haswell
OpenCV	opencv/3.3.0	Python 2.7
OpenMPI	openmpi/2.1.1	
OpenMPI	openmpi/3.1.0	
OpenMPI	openmpi/4.0.0	
OpenMPI	openmpi/4.0.1-cuda10	CUDA aware
NAMD 2.13	namd/2.13	
PLUMED 2.5.2	plumed/2.5.2	
GROMACS 2019	gromacs/2019	
GROMACS 2019.3	gromacs/2019.3-plumed	plumed patched
AMBER 16	amber/16	Compiled with GCC-4.7
gflags	gflags/2.2.1	
gflags	gflags/2.2.2	
glog	glog/0.3.5	
glog	glog/0.4.0	
MATLAB	matlab/R2019b	
MPICH	mpich-3.2	Compiled with GCC-5
Intel TBB	tbb/2018u1-release	Compiled with macro TBB_USE_DEBUG=0
Intel TBB	tbb/2018u1-debug	Compiled with macro TBB_USE_DEBUG=1
FFMPEG	ffmpeg/3.4	cuda 9.0, cuvid, nvenc, nonfree, libnpp
FFMPEG	ffmpeg/4.01	

SLURM Commands

Description	Command	Example
Submit a batch job	sbatch (https://slurm.schedmd.com/sbatch.html)	`sbatch job_script.sh`
Submit an interactive job	sinteractive	`sinteractive -c 2 -g 1`
Cancel a job	scancel (https://slurm.schedmd.com/scancel.html)	`scancel job_id`
List all current jobs for an user	squeue (https://slurm.schedmd.com/squeue.html)	`squeue -u username`
Statistics of a completed job	sacct (https://slurm.schedmd.com/sacct.html)	`sacct -j job_id --format=user,jobid,jobname,partition,state,time,start,end,elapsed,allocgres,ncpus,nodelist`
Pause a job	scontrol (https://slurm.schedmd.com/scontrol.html)	`scontrol hold job_id`
Resume a job	scontrol (https://slurm.schedmd.com/scontrol.html)	`scontrol resume job_id`
Modify attributes of a submitted job	scontrol (https://slurm.schedmd.com/scontrol.html)	`scontrol update jobid=job_id TimeLimit=4-00:00:00`
Display a job's characteristics	scontrol (https://slurm.schedmd.com/scontrol.html)	`scontrol show job job_id`
Display information about nodes and partitions	sinfo (https://slurm.schedmd.com/sinfo.html)	`sinfo -a`

Job Submission

Interactive Jobs

Cores = 10, partition = long, Account = research, GPU = 1

```
parithi@ada ~]$ sinteractive -c 10 -p long -A research -g 1
salloc: Granted job allocation 141
parithi@gnode03:/home/parithi$
```

Batch Jobs

Sample script: NAMD

```
#!/bin/bash
#SBATCH -A research
#SBATCH --qos=medium
#SBATCH -n 20
#SBATCH --gres=gpu:2
#SBATCH --mem-per-cpu=2048
#SBATCH --time=1-00:00:00
#SBATCH --mail-type=END

module add namd/2.12

charmrun +p$SLURM_NPROCS namd2 +idlepoll +devices $CUDA_VISIBLE_DEVICES apoa1.namd > output-gpu1.out
```

Sample script: Python

```
#!/bin/bash
#SBATCH -A $USER
#SBATCH -n 40
#SBATCH --gres=gpu:4
#SBATCH --mem-per-cpu=2048
#SBATCH --time=1-00:00:00
#SBATCH --mail-type=END

module add cuda/8.0
module add cudnn/7-cuda-8.0

python .....
```

The variable CUDA_VISIBLE_DEVICES holds the ids of assigned GPUs

Tips and tricks

Using Jupyter notebook / TensorBoard on compute nodes

Please check the following steps to use Jupyter notebook/TensorBoard on ADA/Abacus.

1. Start jupyter notebook/ TensorBoard with any port
2. Run the following command on your compute node

   ```
   ssh -N -f -R <port1>:localhost:<port2> <user_name>@<local_machine_ip>
   ```
   port1: Any port that you wish to use on local machine (laptop / pc)
   port2: The port on which you are running your jupyter notebook/TensorBoard (step 1)

1. Now you can access your jupyter notebook or TensorBoard on "localhost:port1" from local machine

This is link to sample script click here (https://drive.google.com/open?id=0B80uSIXktkRJdGVZalZrNURndGc)

Note

- To avoid generating the unique authentication token for jupyter notebook every time, set password using following steps :

  ```
  jupyter notebook --generate-config
  jupyter notebook password
  ```

- SSH server is required on the local machine for this method. For windows, you can use following server.
  - bitvise (https://www.bitvise.com/ssh-serve)
- You may also try to add reverse SSH tunnel code in .ssh/config file, Link (https://unix.stackexchange.com/questions/162093/reverse-ssh-tunnel-in-config)