TSUBAME3.0 User's Guide

TSUBAME Computing Services, Global Scientific Information and Computing Center

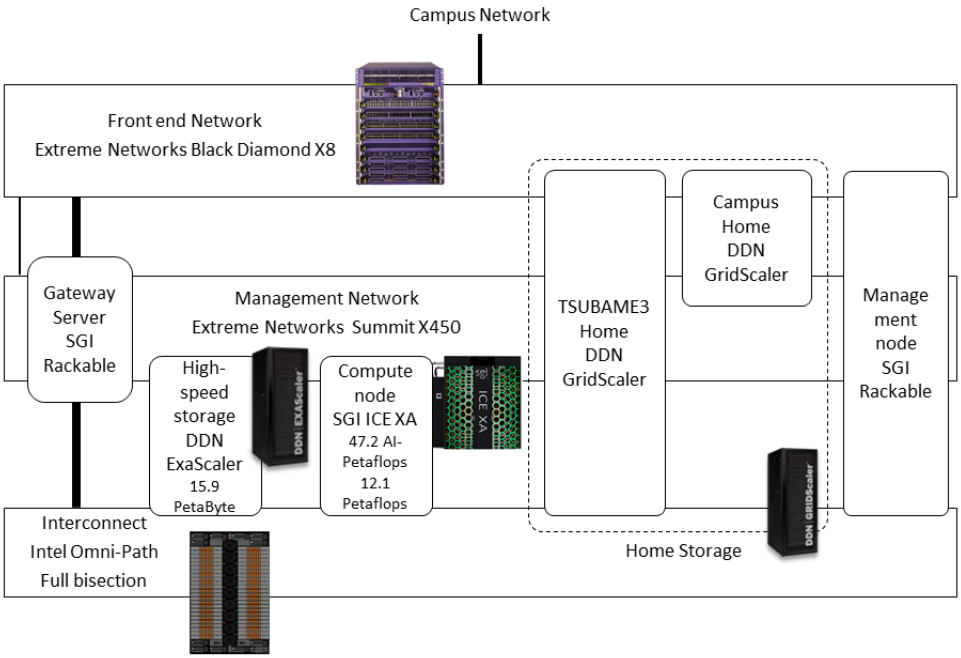
2023-10-18

Table of contents

1. Introduction to TSUBAME3.0

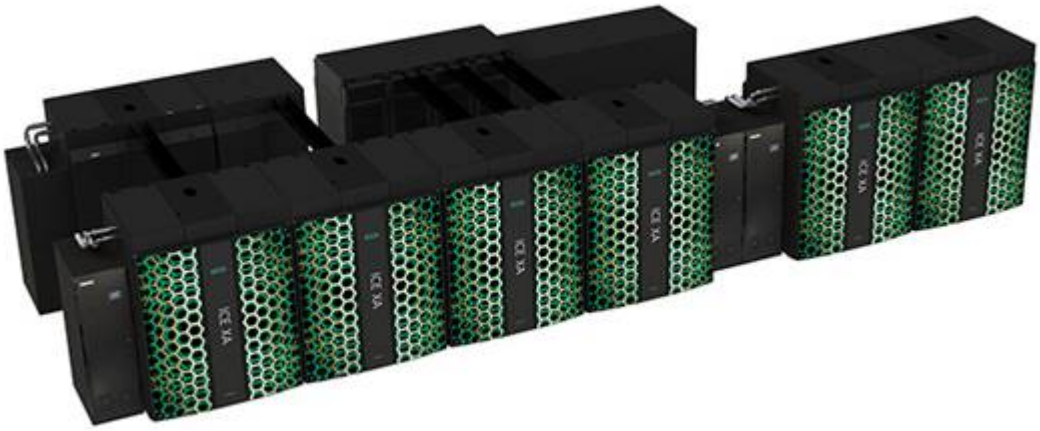
1.1. System architecture

This system is a shared computer that can be used from various research and development departments at Tokyo Institute of Technology.
Each compute node and storage system are connected to the high-speed network by Omni-Path and are now connected to the Internet at a speed of 20 Gbps, and in the future, they will be connected to the Internet at a speed of 100 Gbps via SINET5 (as of May 2019). The system architecture of TSUBAME 3.0 is shown below.



1.2. Compute node configuration

The computing node of this system is a blade type large scale cluster system consisting of SGI ICE XA 540 nodes.
One compute node is equipped with two Intel Xeon E5-2680 v4 (2.4 GHz, 14 core), and the total number of cores is 15,120 cores.
The main memory capacity is 256 GiB per compute node, total memory capacity is 135 TiB.
Each compute node has 4 ports of Intel Omni-Path interface and constitutes a fat tree topology by Omni-Path switch.



The basic specifications of TSUBAME 3.0 machine are as follows.

Unit name	Compute Node x 540
Configuration (per node)	
CPU	Intel Xeon E5-2680 v4 2.4GHz x 2CPU
cores/thread	14cores / 28threads x 2CPU
Memory	256GiB
GPU	NVIDIA TESLA P100 for NVlink-Optimized Servers x 4
SSD	2TB
Interconnect	Intel Omni-Path HFI 100Gbps x 4

1.3. Software configuration

The operating system (OS) of this system has the following environment.

- SUSE Linux Enterprise Server 12 SP2

Regarding the application software that can be used in this system, please refer to ISV Application, Freeware.

1.4. Storage configuration

This system has high speed / large capacity storage for storing various simulation results.
On the compute node, the Lustre file system is used as the high-speed storage area, and the home directory is shared by GPFS + cNFS.

In addition, 2 TB SSD is installed as a local scratch area in each compute node. A list of each file system that can be used in this system is shown below.

Usage	Mount point	Capacity	Filesystem
Home directory Shared space for applications	/home /apps	40TB	GPFS+cNFS
Massively parallel I/O spaces 1	/gs/hs0	4.8PB	Lustre
Massively parallel I/O spaces 2	/gs/hs1	4.8PB	Lustre
Massively parallel I/O spaces 3	/gs/hs2	4.8PB	Lustre
Local scratch	/scr	1.9TB/nde	xfs（SSD)

2. Get Started

2.1. Get an account

In order to use this system, it is necessary to register a user account.
Please refer Getting Accounts Page for details, as the procedure depends on your affiliation and program to apply.

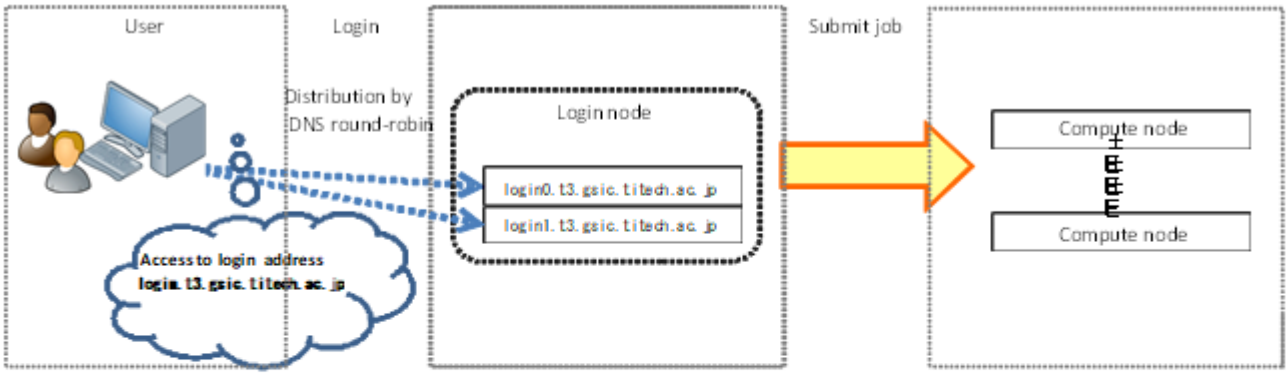
2.2. Login

You need to upload the SSH public key to access the login node.
Please refer to TSUBAME portal User's Guide "SSH public key registration" for the operation of public key registration.
Once registration of the SSH public key is completed, you can log in to the login node.
When you connect to the login node, you will be redirected to one of the login nodes automatically by DNS round-robin.

Warning

As login nodes are shared with many users, please do not execute heavy operations there.

The usage image is shown below.



Connect to the login node with SSH. And you can transfer files using SFTP.

```
login.t3.gsic.titech.ac.jp
```

To connect to a specific login node, log in with the following hostname (FQDN).

```
login0.t3.gsic.titech.ac.jp
login1.t3.gsic.titech.ac.jp
```

The following example shows a connection method with X transfer option enabled from Linux / Mac / Windows (Cygwin).
example) UserName: `gsic_user` and Private key: `~/.ssh/t3-key`

```
$ ssh gsic_user@login.t3.gsic.titech.ac.jp -i ~/.ssh/t3-key -YC
```

Info

If you have the key pair in the standard path and with the standard file name, the `-i` option is not required.

In the first connection, the following message may be sent depending on the client's setting. In that case, enter `yes`.

```
The authenticity of host 'login0.t3.gsic.titech.ac.jp (131.112.3.21)' can't be established.
ECDSA key fingerprint is SHA256:RImxLoC4tBjIYQljwIImCKshjef4w7Pshjef4wtBj
Are you sure you want to continue connecting (yes/no)?
```

2.2.1. Restrictions for heavy work in login nodes

As login nodes (login, login0, login1) are shared with many users at the same time, please do not execute programs which dominate CPU time. For parallel or long-time computation, please use compute nodes using qsub and qrsh commands, Followings are examples of judgment criteria. When the system administrator noticed your program is preventing others, it will be terminated even if it is permitted or not prohibited here,

Permitted operations

- File transfer, compression, decompression (e.g., scp, sftp, rsync, tar)
- Program compilation (If you consume lots of resources by parallel compilation etc., use compute nodes)

Prohibited operations

- Calculation using ISV applications, freeware, or programs made by yourself
- Execution of programs that exceeds 10 minutes (except for file transfer)
- Execution of parallel programs (including python and MPI)
- Execution of programs that consumes lots of memory
- Execution of lots of processes simultaneously (e.g., parallel compilation)
- Execution of server programs or auto-restart programs (e.g., VSCode Server, Jupyter Notebook)
- Other operations which use lots of CPU resources

The login nodes have 4 GB memory limit per process. The system administrator will terminate programs with excessive loads without prior notice. If you want to execute such operations, or you feel login nodes are too heavy, please use compute nodes as interactive jobs via job scheduler.

2.3. Password Management

The user account of this system is managed by the LDAP, and authentication in the system is done by SSH key authentication.
For this reason, you do not need a password to use the compute nodes, but you will need a password to access the Windows/Mac terminal in the university, storage system.
If you need to change the password, please change from the TSUBAME3.0 portal.
The rules of available passwords are described on the TSUBAME3.0 portal password setting page.

2.4. Changing default login shell

At the time of user registration, the login shell of each user account is /bin/bash.
If you want to change the default login shell, please use the `chsh` command.
The available login shells are bash, csh, ksh, tcsh and zsh.
You can check the available login shell with the `chsh` command without arguments.

```
$ chsh
Usage: chsh shell(/bin/bash /bin/csh /bin/sh /bin/ksh /bin/tcsh /bin/zsh).
```

The following is an example of changing the default login shell to tcsh.

```
$ chsh /bin/tcsh
Please input Web Portal Password(not SSH Passphrase)
```

```
Enter LDAP Password: xxxxxx   <- Please enter your password.
Changing shell succeded!!
```

2.5. How to check TSUBAME points

To check TSUBAME points with this command, follow the steps below.

```
$ t3-user-info  group point -g  TESTGROUP
gid     group_name                        deposit      balance
------------------------------------------------------------
xxxx  TESTGROUP                             5000     800000000
```

You can check the situation where current deposit point is 5000, and the remaining TSUBAME point of the specified TESTGROUP group is 800000000.

3. Storage system

In this system, in addition to the home directory, you can also use file systems such as the Lustre file system of the high-speed storage area, the SSD of the local scratch area, and the shared scratch area, BeeGFS On Demand, which creates the SSD for each job.

3.1. Home Directory

Up 25 GB of the home directory space is available per user. The following is an example of checking the HOME directory capacity of TESTUSER.

```
$ t3-user-info disk home
  uid name         b_size(GB) b_quota(GB)    i_files    i_quota
------------------------------------------------------------
 2011 TESTUSER              7          25     101446    2000000
```

Of the 25 GB quota limit, 7 GB is used, regarding the inode limit, we can check the situation that we are using approximately 100,000 out of the 2 million quota limit.
Please note that new writing cannot be performed if the quota limit is exceeded.
Deleting the file so that it falls below the quota limit makes it possible to write again.
Even if you delete files, it rarely occurs that it keeps remaining in the quota limit. In that case, please wait for a day at maximum, the quota is recalculated and it returns to normal value.

3.2. High-speed storage area

The high-speed storage area consists of the Lustre file system and you can be used by purchasing it as a group disk.
Please refer to "TSUBAME portal User's Guide" for the method of purchasing group disk.
The following is an example of checking the capacity of the group disk of TESTGROUP.

```
$ t3-user-info disk group -g TESTGROUP
                                          /gs/hs0                                 /gs/hs1                                 /gs/hs2
      gid group_name                 size(TB) quota(TB)   file(M)  quota(M)  size(TB) quota(TB)   file(M)  quota(M)  size(TB) quota(TB)   file(M)  quota(M)
  ------------------------------------------------------------------------------------------------------------------------------------------------------
      xxxx TESTGROUP                    0.00         0      0.00         0     59.78       100      7.50       200      0.00         0      0.00         0
```

In the specified TESTGROUP group, only /gs/hs1 is purchased, about 60 TB of the 100 TB quota limit is used,
Regarding the inode limit, we can check the situation of using 7.5 million out of the 200 million quota limit.

3.3. Storage service (CIFS)

In TSUBAME 3.0, users can access the high-speed storage area from the Windows / Mac terminal in the university using the CIFS protocol.
You can access it with the following address.

```
\\gshs.t3.gsic.titech.ac.jp
```

It will be accessible with TSUBAME3.0 account and the password set in the portal.
When you access from Windows, please specify the TSUBAME domain as below.

UserName	TSUBAME\ *(TSUBAME3.0 Account Name)*
Password	*(TSUBAME3.0 Account Password)*

The folder name corresponds to /gs/hs0, /gs/hs1, /gs/hs2 of the compute node, and it is T3_HS0, T3_HS1, T3_HS2.
Please access the folder purchased as a group disk.

4. Software Environment

4.1. Change User Environment

In this system, you can switch the compiler and application use environment by using the module command.

4.1.1. List the Available Modules

You can check available modules with “module avail" or “module ava".

```
$ module avail
```

Available modules are described in Application software.

4.1.2. Display the named module information

One can display the short information by issuing the command “module whatis MODULE".

```
$ module whatis intel/17.0.4.196
intel/17.0.4.196     : Intel Compiler version 17.0.4.196 (parallel_studio_xe_2017) and MKL
```

4.1.3. Load the named module

One can load the named module by issuing the command “module load MODULE"

```
$ module load intel/17.0.4.196
```

Please use the same module that you used at compile time for the module to be loaded in the job script.

4.1.4. List all the currently loaded modules

One can list the modules currently loaded by issuing the command “module list".

```
$ module list
Currently Loaded Modulefiles:
  1) intel/17.0.4.196   2) cuda/8.0.61
```

4.1.5. Unoad the named module

One can unload the named module by issuing the command “module unload MODULE".

```
$ module list
Currently Loaded Modulefiles:
  1) intel/17.0.4.196   2) cuda/8.0.61
$ module unload cuda
$ module list
Currently Loaded Modulefiles:
  1) intel/17.0.4.196
```

4.1.6. Remove all modules

One can remove all modules by issuing the command "module purge".

```
$ module list
Currently Loaded Modulefiles:
  1) intel/17.0.4.196   2) cuda/8.0.61
$ module purge
```

```
$ module list
No Modulefiles Currently Loaded.
```

4.2. Usage in job script

When executing the module command in the job script, it is necessary to initialize the module command in the job script as follows.

[sh, bash]

```
. /etc/profile.d/modules.sh
module load intel/17.0.4.196
```

[csh, tcsh]

```
source /etc/profile.d/modules.csh
module load intel/17.0.4.196
```

4.3. Intel Compiler

In this system, you can use Intel compiler, PGI compiler and GNU compiler as compiler. The Intel compiler commands are as follows.

Command	Language	Syntax
ifort	Fortran 77/90/95	`$ ifort [option] source_file`
icc	C	C
icpc	C++	C++

To use it, please load "intel" with the module command.
If you specify the --help option, a list of compiler options is displayed.

4.3.1. Compiler options

The compiler options are shown below.

Option	Description
`-O0`	Disables all optimizations. Using for debugging,etc.
`-O1`	Affects code size and locality. Disables specific optimizations.
`-O2`	Default optimizations. Same as -O. Enables optimizations for speed, including global code scheduling, software pipelining, predication,
`-O3`	Aggressive optimizations for maximum speed (, but does not guarantee higher performance). Optimization including data prefetching, scalar replacement, loop transformations.
`-xCORE-AVX2`	The generated executable will not run on non-Intel processors and it will not run on Intel processors that do not support Intel AVX2 instructions.
`-xSSE4.2`	The generated executable will not run on non-Intel processors and it will not run on Intel processors that do not support Intel SSE4.2 instructions.
`-xSSSE3`	The generated executable will not run on non-Intel processors and it will not run on Intel processors that do not support Intel SSE3 instructions.
`-qopt-report=n`	Generates optimizations report and directs to stderr. n=0 : disable optimization report output n=1 : minimum report output n=2 : medium output (DEFAULT) n=3 : maximum report output
`-fp-model precise`	Tells the compiler to strictly adhere to value-safe optimizations when implementing floating-point calculations. It disables optimizations that can change the result of floating-point calculations. These semantics ensure the accuracy of floating-point computations, but they may slow performance.
`-g`	Produces symbolic debug information in object file (implies -O0 when another optimization option is not explicitly set)
`-traceback`	Tells the compiler to generate extra information in the object file to provide source file traceback information when a severe error occurs at runtime. Specifying -traceback will increase the size of the executable program, but has no impact on runtime execution speeds.

4.3.2. Recommended optimization options

The recommended optimization options for compilation of this system are shown below.

Option	Description
`-O3`	Aggressive optimizations for maximum speed (, but does not guarantee higher performance). Optimization including data prefetching, scalar replacement, loop transformations.
`-xCORE-AVX2`	The generated executable will not run on non-Intel processors and it will not run on Intel processors that do not support Intel AVX2 instructions.

If the performance of the program deteriorates by using the above option, lower the optimization level to -O2 or change the vectorization option. If the results do not match, try the floating point option as well.

4.3.3. Intel 64 architecture memory model

Tells the compiler to use a specific memory model to generate code and store data.

Memory model	Description
small (`-mcmodel=small`)	Tells the compiler to restrict code and data to the first 2GB of address space. All accesses of code and data can be done with Instruction Pointer (IP)-relative addressing.
medium (`-mcmodel=medium`)	Tells the compiler to restrict code to the first 2GB; it places no memory restriction on data. Accesses of code can be done with IP-relative addressing, but accesses of data must be done with absolute addressing.
large (`-mcmodel=large`)	Places no memory restriction on code or data. All accesses of code and data must be done with absolute addressing.

When you specify option -mcmodel=medium or -mcmodel=large, it sets option -shared-intel. This ensures that the correct dynamic versions of the Intel run-time libraries are used.
If you specify option -static-intel while -mcmodel=medium or -mcmodel=large is set, an error will be displayed.

```
<some lib.a library>(some .o): In Function <function>:
  : relocation truncated to fit: R_X86_64_PC32 <some symbol>
…
  : relocation truncated to fit: R_X86_64_PC32 <some symbol>
```

When you specify option `-mcmodel=medium` or `-mcmodel=large`, it sets option `-shared-intel`. This ensures that the correct dynamic versions of the Intel run-time libraries are used.
If you specify option `-static-intel` while `-mcmodel=medium` or `-mcmodel=large` is set, an error will be displayed.

4.4. PGI compiler

PGI compiler commands are shown below.

Command	Language	Syntax
pgfortran	Fortran 77/90/95	`$ pgfortran [option] source_file`
pgcc	C	`$ pgcc [option] source_file`
pgc++	C++	`$ pgc++ [option] source_file`

There are two versions of PGI compiler, one is LLVM version and another one is no LLVM version.
To use LLVM version, invoke below.

```
module load pgi
```

To use no LLVM version, invoke below.

```
module load pgi-nollvm pgi
```

For details of each command, please refer to `$ man pgcc` etc.

4.5. Parallelization

4.5.1. Thread parallel (OpenMP, Automatic parallelization)

The command format when using OpenMP, automatic parallelization is shown below.

Language	Command
OpenMP	
Fortran 77/90/95	`$ ifort -qopenmp [option] source_file`
C	`$ icc -qopenmp [option] source_file`
C++	`$ icpc -qopenmp [option] source_file`
Automatic Parallelization	
Fortran 77/90/95	`$ ifort -parallel [option] source_file`
C	`$ icc -parallel [option] source_file`
C++	`$ icpc -parallel [option] source_file`

`-qopt-report-phase=openmp` : Reports loops, regions, sections, and tasks successfully parallelized.

`-qopt-report-phase=par` : Reports which loops were parallelized.

4.5.2. Process parallel (MPI)

The command format when MPI is used is shown below. When using, please read each MPI with the module command.

MPI Library	Launguage	Command
Intel MPI	Fortran 77/90/95	`$ mpiifort [option] source_file`
	C	`$ mpiicc [option] source_file`
	C++	`$ mpiicpc [option] source_file`
Open MPI	Fortran 77/90/95	`$ mpifort [option] source_file`
	C	`$ mpicc [option] source_file`
	C++	`$ mpicxx [option] source_file`
SGI MPT	Fortran 77/90/95	`$ mpif90 [option] source_file`
	C	`$ mpicc [option] source_file`
	C++	`$ mpicxx [option] source_file`

4.6. GPU Environment

TSUBAEM3.0 provid environmtn of Intel CPUs in conjunction with GPU (NVIDIA TESLA P100).

4.6.1. Interactive execution and debug

As login nodes (login, login0, login1) do not have GPU, you can not run GPU codes, only comple and link work. In addition to that, heavy work in login node is restricted.

You can run GPU codes with interactive and debug on compute nodes by batch system. Please refer Interactive job for more details.

4.6.2. Supprted application for GPU

Current GPU compatible applications are as follows. (As of 2017.12.18)

- ABAQUS 2017 --- Please refer to ABAQUS usage guide (separate volume).
- NASTRAN 2017.1 --- Please refer to NASTRAN usage guide (separate volume).
- ANSYS 18 --- Please refer to ANSYS usage guide (separate volume).
- AMBER 16 --- Please refer to AMBER usage guide (separate volume).
- Maple 2016 --- Please refer to Maple usage guide (separate volume).
- Mathematica 11.2 --- Please refer to Mathematica usage guide (separate volume).
- MATLAB --- Please refer to MATLAB usage guide (separate volume).
- Forge --- Please refer to Forge usage guide (separate volume).
- PGI Compiler --- Please refer to PGI usage guide (separate volume).

Even for other applications, we will provide it sequentially.

4.6.3. MPI Environment with CUDA

MPI environment compatible with CUDA is available.

OpenMPI + gcc Environment

```
# load CUDA and Open MPI Environment (gcc is default setting)
module load cuda openmpi
```

OpenMPI + pgi environment

```
# Load CUDA and PGI Environment(First load the compiler environment)
module load cuda pgi
# Load Open MPI Environment(The OpenMPI environment according to the compiler is set up)
module load openmpi
```

Info

specific verion `openmpi/2.1.2-pgi2019` described before is no more necesarry at present.

In the PGI bundle version , there is no linkage from the batch system, so you need to specify the host list at run.
An example job script is shown below.

OpenMPI + Intel Environment

```
# Load CUDA and Intel Environment(First load the compiler environment)
module load cuda intel
# Load Open MPIEnvinronment (The OpenMPI environment according to the compiler is set up)
module load openmpi
```

4.6.4. NVIDIA GPUDirect

Currently, there are four functions (GPUDIRECT SHARED GPU SYSMEM, GPUDIRECT P2P, GPUDIRECT RDMA, GPUDIRECT ASYNC) as NVIDIA GPUDirect (GPUDIRECT FAMILY). (As of 2017.12.18)
Of these, TSUBAME 3.0 supports GPUDIRECT SHARED GPU SYSMEM, GPUDIRECT P2P, GPUDIRECT RDMA.

- GPUDIRECT SHARED GPU SYSMEM (Version1)
  It is a function that can directly specify the address of CUDA pinned memory and device memory in the send / receive buffer of MPI. When the device memory address is specified, the data is actually transferred via the buffer on the host memory.
- GPUDIRECT P2P (Version2)
  It is a function of direct data transfer (P2P) between GPU via PCI - Express and NVLink. In TSUBAME 3.0, four GPUs is installed per node, but one CPU is connected to two GPUs via PLX switch. Between four GPUs, high speed NVLink is connected.
- GPUDIRECT RDMA (Version3)
  It is a function to realize high-speed data transfer between GPUs of different nodes by directly transferring data (RDMA) between the GPU and the interconnect (Intel Omni-Path in TSUBAME 3.0) without going through the host memory.
- GPUDIRECT ASYNC
  It is asynchronous communication between the GPU and the interconnect without going through the host memory.Currently, Intel Omni-Path of TSUBAME 3.0 does not support it.

Reference: http://on-demand.gputechconf.com/gtc/2017/presentation/s7128-davide-rossetti-how-to-enable.pdf

For GPUDirect, please also refer to the following URL.

- https://developer.nvidia.com/gpudirect
- http://docs.nvidia.com/cuda/gpudirect-rdma

4.6.5. GPUDirect RDMA

Calling cudaSetDevice() before calling MPI_Init() is mandatory to use GPUDirect RDMA on OPA1.9. https://www.intel.com/content/dam/support/us/en/documents/network-and-i-o/fabric-products/Intel_PSM2_PG_H76473_v12_0 .pdf p.15

> CUDA support is limited to using a single GPU per process.
> You set up the CUDA runtime and pre-select a GPU card (through the use of cudaSetDevice() or a similar CUDA API) pr ior to calling psm2_init() or MPI_Init(), if using MPI.
> While systems with a single GPU may not have this requirement, systems with multiple GPU may see non-deterministic results without proper initialization.
> Therefore, it is strongly recommended that you initialize the CUDA runtime before the psm2_init() or MPI_Init() cal l.

So modify your code with the above, or use openmpi/3.1.4-opa10.10-t3 module file, that does the modification in the openmpi.
It is available by `module load cuda openmpi/3.1.4-opa10.10-t3`.

The following shows how to execute GPUDirect RDMA with OpenMPI. Below, it is an execution example with two nodes, MPI x 2.

```
$ module load cuda openmpi/3.1.4-opa10.10-t3
$ mpirun -np 2 -npernode 1 -x PSM2_CUDA=1 -x PSM2_GPUDIRECT=1 -x LD_LIBRARY_PATH -x PATH [program]
```

- PSM2_CUDA --- Enables CUDA support in PSM2 of Omni-Path
- PSM2_GPUDIRECT --- Enable NVIDIA GPUDirect RDMA in PSM2

4.6.6. GPU COMPUTE MODE

Only when using resource type f_node batch job, you can change GPU compute mode.
To change GPU compute mode, specify f_node in the job script and specify `#$ -v GPU_COMPUTE_MODE=<MODE>` for additional.
The following three modes are available.

Mode	Description
0	DEFAULT mode Multiple contexts are allowed per device.
1	EXCLUSIVE_PROCESS mode Only one context is allowed per device, usable from multiple threads at a time.
2	PROHIBITED mode No contexts are allowed per device (no compute apps).

Here is a sample job script.

```
#!/bin/sh
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=1:00:00
#$ -N gpumode
#$ -v GPU_COMPUTE_MODE=1
/usr/bin/nvidia-smi
```

When using interactive job, it can be used as follows.

```
$ qrsh -g [TSUBAME group] -l f_node=1 -l h_rt=0:10:00 -pty yes -v TERM -v GPU_COMPUTE_MODE=1 /bin/bash
```

5. Job Scheduler

On this system, UNIVA Grid Engine manages the running and scheduling of jobs.

5.1. kind of compute nodes

5.1.1. baremetal environments

5.1.1.1. Available resource type

In this system, a job is executed using a logically divided computing node called "resource type."
When submitting a job, specify how many resource types to use (ex: `-l f_node = 2`). A list of available resource types is shown below.

Type	Resource Type Name	Physical CPU cores	Memory(GB)	GPUs
F	f_node	28	235	4
H	h_node	14	120	2
Q	q_node	7	60	1
C1	s_core	1	7.5	0
C4	q_core	4	30	0
G1	s_gpu	2	15	1

"Physical CPU Cores", "Memory (GB)", "GPUs" are the available resources per resource type.

- Resource type combinations are not available.
- Maximum run time is 24 hours.
- TSUBAME 3 has various limit values as follows.
- Number of concurrently executable jobs per person
- The total number of slots that can be simultaneously executed per person etc.
  A list of current limit values can be confirmed with the following URL.
  https://www.t3.gsic.titech.ac.jp/en/resource-limit Please note that it may change at any time according to resource usage.

5.1.2. Container environments

In this system, in order to absorb the system dependency of the application that is difficult to operate on host OS due to the software dependency, we provide the system container using Docker and the application container using Singularity.
This chapter describes how to use system container jobs using Docker. Please refer to the freeware chapter for Singularity.

5.1.2.1. Available resource types

The following resource types can be used for container usage jobs.
When used in a batch script, it has an .mpi suffix at the end.

Type	Using nodes Resource type Name	Using containers Batch Job(Multi containers)	Using containers Interactive Job(Single container)
F	f_node	t3_d_f_node.mpi	t3_d_f_node
H	h_node	t3_d_h_node.mpi	t3_d_h_node
Q	q_node	t3_d_q_node.mpi	t3_d_q_node
C1	s_core	(t3_d_s_core)	t3_d_s_core
C4	q_core	t3_d_q_core.mpi	t3_d_q_core
G1	s_gpu	t3_d_s_gpu.mpi	t3_d_s_gpu

The resource type t3_d_s_core allows communication to the Internet but does not support inter-container communication.
Therefore, please specify another container resource when performing MPI or multi-container communication.

Info

If only one container is needed to start by specifying `-t 1-1`, please use the resource type without `.mpi`. It does not work with the resource type with `.mpi` and with `-t 1-1`.

The following shows the qsub command options when using nodes and containers.

	Using nodes	Using containers
Set image		-ac d=[Container image]
Set resource type	-l [Resource type Name] =[Number]	-jc [Container resource type] -t 1-[Number]
Set walltime	-l h_rt=[Maximum run time]	-adds l_hard h_rt [Maximum run time]

For container jobs, the -t option is the number of containers.
For example, with 4 containers, specify as -t 1-4. MPI node files are stored as files in `$SGE_JOB_SPOOL_DIR`. Please use the host file of each MPI at the time of execution.

MPI	Hostfile Name
Intel MPI	impi_hostfile
OpenMPI	ompi_hostfile
MPICH	mpich_hostfile

You can use only the images provided by the system, please refer System Software page for list of available images.

5.2. Job submission

To execute the job in this system, log in to the login node and execute the `qsub` command.

5.2.1. Job submission flow

In order to submit a job, create and submit a job script. The submission command is `qsub`.

1. Create a job script
2. Submit a job using qsub
3. Status check using qstat
4. Cancel a job using qdel
5. Check job result

The qsub command confirms billing information (TSUBAME 3 points) and accepts jobs.

5.2.2. Creating job script

Here is a job script format:

```
#!/bin/sh
#$ -cwd
#$ -l [Resource type Name] =[Number]
#$ -l h_rt=[Maximum run time]
#$ -p [Priority]
[Initialize module environment]
[Load the relevant modules needed for the job]
[Your program]
```

Warning

shebang(`#!/bin/sh` line) must be located at the first of the job script.

- [Initialize module environment]
  By executing the following, initialize the module environment.

```
. /etc/profile.d/modules.sh
```

- [Load the relevant modules needed for the job]
  [Load the relevant modules needed for the job with the module command.
  For example, load the intel compiler:

```
module load intel
```

- [Your program] Execute your program. For example, if your binary is named "a.out":

```
./a.out
```

In a shell script, you can set the qsub options in lines that begin with #$.
There is an alternative way to pass them with the qsub command.

You should always specify "Resource type" and "Maximum run time."
The option used by qsub is following.

Option	Description
-l [Resource type Name] =[Number] (Required)	Specify the resource type.
-l h_rt=[Maximum run time] (Required)	specify the maximum run time (hours, minutes and seconds) You can specify it like HH: MM: SS or MM: SS or SS.
-N	name of the job (Script file name if not specified)
-o	name of the standard output file
-e	name of the standard error output file
-m	Will send email when job ends or aborts. The conditions for the -m argument include: a: mail is sent when the job is aborted. b: mail is sent when the job begins. e: mail is sent when the job ends. It is also possible to combine like abe. When a large number of jobs with mail option are submitted, a large amount of mail is also sent, heavy load is applied to the mail server, and it may be detected as an attack and the mail from Tokyo Tech may be blocked. If you need to execute such jobs, please remove the mail option or review the script so that it can be executed with one job.
-M	Email address to send email to
-p (Premium Options)	Specify the job execution priority. If -4 or -3 is specified, a charge factor higher than -5 is applied. The setting values -5, -4, -3 correspond to the priority 0, 1, 2 of the charging rule. -5: Standard execution priority. (Default) -4: The execution priority is higher than -5 and lower than -3. -3: Highest execution priority. Note that all priority number to specify is negative value. Do not forget preceding minus sign.
-t	Submits a Array Job specified with start-end[:step]
-hold_jid	Defines the job dependency list of the submitted job. The job is executed after the specified dependent job is finished.
-ar	Specify the reserved AR ID when using the reserved node.

5.2.3. Job script examples

5.2.3.1. serial job/GPU job

The following is an example of a job script created when executing a single job (job not parallelized) or GPU job.
For GPU job, please replace `-l s_core=1` with `-l s_gpu=1` and load necessary modules such as CUDA environment.

```
#!/bin/sh
## Run in current working directory
#$ -cwd
## Resource type F: qty 1
#$ -l s_core=1
## maximum run time
#$ -l h_rt=1:00:00
#$ -N serial
## Initialize module command
. /etc/profile.d/modules.sh
# Load CUDA environment
module load cuda
## Load Intel compiler environment
module load intel
./a.out
```

5.2.3.2. SMP job

An example of a job script created when executing an SMP parallel job is shown below.
Hyper-threading is enabled for compute nodes. Please explicitly specify the number of threads to use.

```
#!/bin/sh
#$-cwd
## Resource type F: qty 1
#$ -l f_node=1
#$ -l h_rt=1:00:00
#$ -N openmp
. /etc/profile.d/modules.sh
module load cuda
module load intel
## 28 threads per node
export OMP_NUM_THREADS=28
./a.out
```

5.2.3.3. MPI job

An example of a job script created when executing an MPI parallel job is shown below.
Please specify an MPI environment according to the MPI library used by you for MPI jobs as follows.
For OpenMPI, to pass library environment variables to all nodes, you need to use `-x LD_LIBRARY_PATH`.

Intel MPI

```
#!/bin/sh
#$-cwd
## Resource type F: qty 4
#$ -l f_node=4
#$ -l h_rt=1:00:00
#$ -N flatmpi
. /etc/profile.d/modules.sh
module load cuda
module load intel
## Load Intel MPI environment
module load intel-mpi
## 8 process per node, all MPI process is 32
mpiexec.hydra -ppn 8 -n 32 ./a.out
```

OpenMPI

```
#!/bin/sh
#$-cwd
## Resource type F: qty 4
#$ -l f_node=4
#$ -l h_rt=1:00:00
#$ -N flatmpi
. /etc/profile.d/modules.sh
module load cuda
module load intel
## Load Open MPI environment
module load openmpi
```

```
## 8 process per node, all MPI process is 32
mpirun -npernode 8 -n 32 -x LD_LIBRARY_PATH ./a.out
```

SGI MPT

```
#!/bin/sh
#$-cwd
## Resource type F: qty 4
#$ -l f_node=4
#$ -l h_rt=1:00:00
#$ -N flatmpi
. /etc/profile.d/modules.sh
module load cuda
module load intel
## Load SGI MPT environment
module load mpt
## 8 process per node, all MPI process is 32
mpiexec_mpt -ppn 8 -n 32 ./a.out
```

The file of the node list assigned to the submitted job can be referred from `$PE_HOSTFILE`.

```
$ echo $PE_HOSTFILE
/var/spool/uge/r6i0n4/active_jobs/4564.1/pe_hostfile
$ cat /var/spool/uge/r6i0n4/active_jobs/4564.1/pe_hostfile
r6i0n4 28 all.q@r6i0n4 <NULL>
r6i3n5 28 all.q@r6i3n5 <NULL>
```

5.2.3.4. Hybrid parallel

An example of a job script created when executing a process/thread parallel (hybrid) job is shown below.
Please specify an MPI environment according to the MPI library used by you for MPI jobs as follows.
For OpenMPI, to pass library environment variables to all nodes, you need to use =-x LD_LIBRARY_PATH`.

Intel MPI

```
#!/bin/sh
#$-cwd
## Resource type F: qty 4
#$ -l f_node=4
#$ -l h_rt=1:00:00
#$ -N hybrid
. /etc/profile.d/modules.sh
module load cuda
module load intel
module load intel-mpi
## 28 threads per node
export OMP_NUM_THREADS=28
## 1 MPI process per node, all MPI process is 4
mpiexec.hydra -ppn 1 -n 4./a.out
```

OpenMPI

```
#!/bin/sh
#$-cwd
## Resource type F: qty 4
#$ -l f_node=4
#$ -l h_rt=1:00:00
#$ -N hybrid
. /etc/profile.d/modules.sh
module load cuda
module load intel
module load openmpi
## 28 threads per node
export OMP_NUM_THREADS=28
## 1 MPI process per node, all MPI process is 4
mpirun -npernode 1 -n 4 -x LD_LIBRARY_PATH ./a.out
```

5.2.3.5. Container job

Here is a container job script format:

```
#!/bin/sh
#$ -cwd
#$ -ac [Container image]
#$ -jc [Container resource type]
#$ -t 1-[Number]
#$ -adds l_hard h_rt=[Maximum run time]
[Initialize module environment]
[Load the relevant modules needed for the job]
[Your program]
```

Please note that the method of specifying the resource type and walltime is different from the case of normal use.

The following is an example of a job script created when executing a container job.
The usage of the GPU and the usage of MPI parallel jobs are the same as the normal usage.

```
#!/bin/sh
#$ -cwd
## set container image SLES12SP2
#$ -ac d=sles12sp2-latest
## Resource type Q
#$ -jc t3_d_q_node.mpi
## container numver: qty 4
#$ -t 1-4
## maximum run time
#$ -adds l_hard h_rt 0:10:00

. /etc/profile.d/modules.sh
module load cuda
module load intel
module load openmpi
mpirun -npernode 6 -n 24 -hostfile $SGE_JOB_SPOOL_DIR/ompi_hostfile -x LD_LIBRARY_PATH ./a.out
```

Info

If only one container is needed to start by specifying `-t 1-1`, please use the resource type without `.mpi`. It does not work with the resource type with `.mpi` and with `-t 1-1`.

5.2.4. Job submission

Job is queued and executed by specifying the job submission script in the qsub command.
You can submit a job using qsub as follows.

```
qsub -g [TSUBAME group] SCRIPTFILE
```

Option	Description
-g	Specify the TSUBAME group name. Please add as qsub command option, not in script.

5.2.4.1. Trial run

Info

This feature is available only for TSUBAME account holders. It is designed mainly for Tokyo Tech users who can sign up by themselves.

TSUBAME provides the "trial run" feature, in which users can execute jobs without consuming points, for those who are anxious whether TSUBAME applies to their research or not.
To use this feature, submit jobs without specifying a group via `-g` option. In this case, the job is limited to 2 nodes, 10 minutes of running time, and priority -5 (worst).

Warning

The trial run feature is only for testing whether your program works or not. Do not use it for actual execution for your research and measurement.
It does not mean that you can execute jobs freely without charge if the job size meets the limitation written in above.

TSUBAME3 has the function of Trial run, that is for checking program operation without consuming points.
In the case of a trial run, the following restrictions apply to the amount of resources.

Maximum number of the resource type specified(*1)	2
Maximum usage time	10 min.
number of concurrent runs	1
resource type	no limitation

(*1): When using Container job, 1

For Trial run, it is necessary to run a job without specifying a TSUBAME group.
Note that the points are consumed when you submit a job with the TSUBAME group.

5.2.5. Job status

The qstat command is a job status display command

```
$qstat [option]
```

The options used by qstat are following.

Option	Description
-r	Displays job resource information.
-j [job-ID]	Display additional information about the job.

Here is the result of qstat command.

```
$ qstat
job-IDprior  nameuser   statesubmit/start at  queuejclass  slotsja-task-ID
```

Item	Description
Job-ID	Job-ID number
prior	Priority of job
name	Name of the job
user	ID of the user who submitted job
state	'state' of the job r running qw waiting in the queue h on hold d deleting t a transition like during job-start s suspended S suspended by the queue T has reached the limit of the tail E error Rq Rescheduled and then waiting for run Rr Rescheduled and then running
submit/start at	Submit or start time and date of the job
queue	Queue name
jclass	job class name
slots	The number of slot the job is taking.
ja-task-ID	Array job task-id

5.2.6. Job delete

To delete your job, use the `qdel` command.

```
$ qdel [job-ID]
```

Here is the result of qdel command.

```
$ qstat
job-IDprior  nameuser   statesubmit/start at  queuejclass  slotsja-task-ID
-----------------------------------------------------------------------------
307 0.55500 sample.sh testuser r 02/12/2015 17:48:10  all.q@r8i6n1A.default32

$ qdel 307
testuser has registered the job 307 for deletion

$ qstat
job-IDprior  nameuser   statesubmit/start at  queuejclass  slotsja-task-ID
-----------------------------------------------------------------------------
```

5.2.7. Job results

The standard output is stored in the file "SCRIPTFILE.o[job-ID]" in the job execution directory.
The standard error output is "SCRIPTFILE.e[job-ID]".

5.2.8. Array Job

There is an array job as a function to parameterize and execute the operation contained in the job script repeatedly.

Info

Because each task in an array job is scheduled as a separate job, there is a schedule latency proportional to the number of tasks.
If each task is short or has a large number of tasks, it is strongly recommended that you reduce the number of tasks by combining multiple tasks.
Example: Combine 10000 tasks into 100 tasks, each processing 100 tasks.

Each job executed in the array job is called a task and managed by the task ID.

```
#  In job script
#$ -t 2-10:2
```

In the above example (2-10:2), start number 2, end number 10, and step size 2 (one skip index) are specified, and it has five tasks 2, 4, 6, 8, 10.
Each task number is set to the environment variable $ SGE_TASK_ID.
By using this environment variable in the job script, you will be able to do parameter studies.
The standard output is stored in the file "SCRIPTFILE.o[job-ID].[task-ID]" in the job execution directory.
The standard error output is "SCRIPTFILE.e[job-ID].[task-ID]".
If you want to delete a specific task, use the qdel -t option as follows.

```
$ qdel [job-ID] -t [task-id]
```

5.3. Reserve compute nodes

It is possible to execute jobs exceeding 24 hours and/or 72 nodes by reserving computation nodes.

- Make a reservation from TSUBAME portal
- Check reservation status, cancel a reservation from TSUBAME portal
- Submit a job using qsub for reserved node
- Cancel a job using qdel
- Check job result
- Check the reservation status and AR ID from the command line

Please refer to TSUBAME Portal User's Guide "Reserving compute nodes" on reservation from the portal, confirmation of reservation status and cancellation of the reservation.

When reservation time is reached, you will be able to execute jobs with the reservation group account.
The following example shows job submission with an AR ID that is a reservation ID.
(note) Resource types available are f_node, h_node and q_node. q_core, s_core, s_gpu cannot be used.

- with qsub

```
$ qsub -g [TSUBAME group] -ar [AR ID]  SCRIPTFILE
```

- with qrsh

```
qrsh -g [TSUBAME group] -l [resource type]=[number of resources] -l h_rt=[time limit] -ar [AR ID]
```

After submitting the job, you can check the status of the job with the qstat command, and delete the job with the qdel command.
The format of the job script is the same as that of the non-reserved job.

`t3-user-info compute ar` can be used to check the reservation status and AR ID from the command line.

```
xxxxx@login0:~> t3-user-info compute ar
ar_id   uid user_name         gid group_name               state     start_date          end_date        time_hour n
ode_count      point return_point
-------------------------------------------------------------------------------------------------
------------------------------
 1320  2005 A2901247         2015 tga-red000                 r   2018-01-29 12:00:00 2018-01-29 13:00:00          1
    1   18000       0
 1321  2005 A2901247         2015 tga-red000                 r   2018-01-29 13:00:00 2018-01-29 14:00:00          1
    1   18000       0
 1322  2005 A2901247         2015 tga-red000                 w   2018-01-29 14:00:00 2018-02-02 14:00:00         96
    1  1728000    1728000
 1323  2005 A2901247         2015 tga-red000                 r   2018-01-29 14:00:00 2018-02-02 14:00:00         96          1    1728000      1728000
 1324  2005 A2901247         2015 tga-red000                 r   2018-01-29 15:00:00 2018-01-29 16:00:00          1         17     306000            0
 1341  2005 A2901247         2015 tga-red000                 w   2018-02-25 12:00:00 2018-02-25 13:00:00          1         18     162000       162000
 3112  2004 A2901239         2349 tgz-training               r   2018-04-24 12:00:00 2018-04-24 18:00:00          6         20     540000            0
 3113  2004 A2901239         2349 tgz-training               r   2018-04-25 12:00:00 2018-04-25 18:00:00          6         20     540000            0
 3116  2005 A2901247         2015 tga-red000                 r   2018-04-18 17:00:00 2018-04-25 16:00:00        167          1    3006000            0
 3122  2005 A2901247         2014 tga-blue000                r   2018-04-25 08:00:00 2018-05-02 08:00:00        168          5   15120000            0
 3123  2005 A2901247         2014 tga-blue000                r   2018-05-02 08:00:00 2018-05-09 08:00:00        168          5    3780000            0
 3301  2005 A2901247         2015 tga-red000                 r   2018-08-30 14:00:00 2018-08-31 18:00:00         28          1     504000            0
 3302  2005 A2901247         2009 tga-green000               r   2018-08-30 14:00:00 2018-08-31 18:00:00         28          1     504000            0
 3304  2005 A2901247         2014 tga-blue000                r   2018-09-03 10:00:00 2018-09-04 10:00:00         24          1     432000            0
 3470  2005 A2901247         2014 tga-blue000                w   2018-11-11 22:00:00 2018-11-11 23:00:00          1          1       4500         4500
 4148  2004 A2901239         2007 tga-hpe_group00            w   2019-04-12 17:00:00 2019-04-12 18:00:00          1          1       4500         4500
 4149  2005 A2901247         2015 tga-red000                 w   2019-04-12 17:00:00 2019-04-13 17:00:00         24          1     108000       108000
 4150  2004 A2901239         2007 tga-hpe_group00            w   2019-04-12 17:00:00 2019-04-12 18:00:00          1          1       4500         4500
-------------------------------------------------------------------------------------------------------------------------------------------
total :                                                                                                          818         97   28507500      3739500
```

To check the availability of the current month's reservations from the command line, use `t3-user-info compute ars`.

5.4. Interactive job

To execute an interactive job, use the qrsh command, and specify the resource type and running time.
After job submission with qrsh, when the job is dispatched, the command prompt will be returned.
The usage of the interactive job is as follows.

```
#!bash
$ qrsh -g [TSUBAME group] -l [resource type name]=[numbers] -l h_rt=[max running time]
Directory: /home/N/username
(Job start time)
username@rXiXnX:~> [Commands to run]
username@rXiXnX:~> exit
```

If group designation is not specified, the job will be treated as a trial run.
In the trial run, the number of the resource is limited to 2, and execution time is limited to 10 minutes, and priority is fixed to -5.
The following example is for resource type F, 1node, and maximum run time is 10minutes.

```
#!bash
$ qrsh -g [TSUBAME group] -l f_node=1 -l h_rt=0:10:00
Directory: /home/N/username
(Job start time)
username@rXiXnX:~> [Commands to run]
username@rXiXnX:~> exit
```

To exit the interactive job, type exit at the prompt.

The following shows how to use containers in interactive jobs.
Specifying multiple containers is not permitted in interactive jobs.

```
$ qrsh -g [TSUBAME group] -jc [container resource type] -add l_hard h_rt [max running time] ?ac [image name]
```

The following is a sample that has been set resource type Q 1 container, and maximum run time is 10minutes.

```
$  qrsh -g tga-hpe_group00 -jc t3_d_q_node -adds l_hard h_rt 0:10:00 -ac d=sles12sp2-latest
Directory: /home/9/hpe_user009
Mon Jun  4 13:35:47 JST 2018
```

5.4.1. X forwarding

1. Enable X forwarding and connect to login node with ssh.
2. Execute qrsh command with X11 forwarding like the following example.

Info

With the scheduler update implemented in April 2020, you no longer need to specify `-pty yes -display "$DISPLAY" -v TERM /bin/bash` when executing qrsh.

In the following example, one node of resource type s_core and 2 hours of execution time are specified.

```
# Execution of qrsh command
$ qrsh -g [TSUBAME group] -l s_core=1 -l h_rt=2:00:00
username@rXiXnX:~> module load [Application module to load]
username@rXiXnX:~> [Command to run X11 application]
username@rXiXnX:~> exit
```

The following is an example of the interactive job with t3_d_s_core container resource type and X forwarding.

```
$ qrsh -g [TSUBAME group] -jc t3_d_s_core -adds l_hard h_rt 0:10:00 -ac d=sles12sp2-latest
```

5.4.2. Connection to the network applications

If your application requires We browser manipulation on the interactive job by the container, SSH port forwarding makes it possible with the web browser on your PC.

(1) Obtain the hostname connected by the interactive node with qrsh

```
$ qrsh -g tga-hpe_group00 -jc t3_d_q_node -adds l_hard h_rt 0:10:00 -ac d=sles12sp2-latest
$ hostname
r7i7n7-cnode00
$ [Execute the program it requires Web browser]
```

After launching the interactive job by qrsh, obtain the hostname of the machine.
r7i7n7-cnode00 is the hostname in the above example.
The console operation is finished but please keep the job session until the end of your application work.

(2) Connect to login node with enabling SSH port forwarding from the console which is the source of the console.(it is not the login node nor the interactive job)

```
ssh -l username -L 8888:r7i7n7-cnode00:<network port of the appliction to connect your PC> login.t3.gsic.titech.ac.jp
```

The connection application network port is different based on the application. For more details, please refer to each manual of the application, or, check the startup message of the application.

Tips

Depending on the console software to SSH to TSUBAME3, SSH port fowrward setup procedure could be different. Please refer to the manual of each SSH console, or to the FAQ.

5.4.3. Interactive queue

The interactive queue is prepared to make immediate execution of visualization and interactive jobs easier, even if TSUBAME is too crowded to allocate nodes for normal jobs, by sharing the same resources with multiple users.

The following is how to submit the jobs to the interactive queue.

Info

Tokyo Tech users including access card holders (users beloging to tgz-edu group or having Axxx login name) can submit jobs without group specification and for free.

```
iqrsh -g [TSUBAME group]-l h_rt=<time>
```

Please note that CPU/GPU overcommit is allowed on interactive queue.
About the resource limits of the interactive queue, please refer to here.

5.5. SSH login to the compute node

You can log in using ssh directly to the computing nodes allocated to your job with the resource type f_node.
You can check the available nodes with the following command.

```
t3-test00@login0:~> qstat -j 1463
==============================================================
job_number:                 1463
jclass:                     NONE
exec_file:                  job_scripts/1463
submission_time:            07/29/2017 14:15:26.580
owner:                      t3-test00
uid:                        1804
group:                      tsubame-users0
gid:                        1800
supplementary group:        tsubame-users0, t3-test-group00
sge_o_home:                 /home/4/t3-test00
sge_o_log_name:             t3-test00
sge_o_path:           /apps/t3/sles12sp2/uge/latest/bin/lx-amd64:/apps/t3/sles12sp2/uge/latest/bin/lx-amd64:/home/4/t3-test00/bin:/usr/local/bin:/usr/bin:/
bin:/usr/bin/X11:/usr/games
sge_o_shell:                /bin/bash
sge_o_workdir:              /home/4/t3-test00/koshino
sge_o_host:                 login0
account:                    2 0 0 0 0 0 600 0 0 1804 1800
cwd:                        /home/4/t3-test00
hard resource_list:         h_rt=600,f_node=1,gpu=4
mail_list:                  t3-test00@login0
notify:                     FALSE
job_name:                   flatmpi
priority:                   0
jobshare:                   0
env_list:
RGST_PARAM_01=0,RGST_PARAM_02=1804,RGST_PARAM_03=1800,RGST_PARAM_04=2,RGST_PARAM_05=0,RGST_PARAM_06=0,RGST_PARAM_07=0,RGST_PARAM_08=0,RGST_PARAM_09=0,RGST_PARAM
script_file:                flatmpi.sh
parallel environment:  mpi_f_node range: 56
department:                 defaultdepartment
binding:                    NONE
mbind:                      NONE
submit_cmd:                 qsub flatmpi.sh
start_time            1:    07/29/2017 14:15:26.684
job_state             1:    r
exec_host_list        1:    r8i6n3:28, r8i6n4:28        <--  Available nodes : r8i6n3、r8i6n4
granted_req.          1:    f_node=1, gpu=4
usage                 1:    wallclock=00:00:00, cpu=00:00:00, mem=0.00000 GBs, io=0.00000 GB, iow=0.000 s, ioops=0, vmem=N/A, maxvmem=N/A
binding               1:    r8i6n3=0,0:0,1:0,2:0,3:0,4:0,5:0,6:0,7:0,8:0,9:0,10:0,11:0,12:0,13:1,0:1,1:1,2:1,3:1,4:1,5:1,6:1,7:1,8:1,9:1,10:1,11:1,12:1,13,
r8i6n4=0,0:0,1:0,2:0,3:0,4:0,5:0,6:0,7:0,8:0,9:0,10:0,11:0,12:0,13:1,0:1,1:1,2:1,3:1,4:1,5:1,6:1,7:1,8:1,9:1,10:1,11:1,12:1,13
resource map          1:    f_node=r8i6n3=(0), f_node=r8i6n4=(0), gpu=r8i6n3=(0 1 2 3), gpu=r8i6n4=(0 1 2 3)
scheduling info:            (Collecting of scheduler job information is turned off)
```

You can log in using ssh directly to the containers allocated to your job.
You can check the available containers with the following command.

```
hpe_user009@nfs1:~> qstat -j 476
==============================================================
job_number:                 476
jclass:                     t3_d_s_gpu.mpi
exec_file:                  job_scripts/476
submission_time:            06/04/2018 13:41:36.715
owner:                      hpe_user009
uid:                        2779
group:                      tga-hpe_group00
```

```
gid:                        2007
supplementary group:        tsubame-users, tgz-edu, tga-hpe_group00, tga-hpe-2017081600
sge_o_home:                 /home/9/hpe_user009
sge_o_log_name:             hpe_user009
sge_o_path:                 /apps/t3/sles12sp2/uge/latest/bin/lx-amd64:/home/9/hpe_user009/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games
sge_o_shell:                /bin/bash
sge_o_workdir:              /home/9/hpe_user009/koshino
sge_o_host:                 nfs1
account:                    0 0 0 1 0 0 600 0 0 2779 2007 1,2,3,4 0
cwd:                        /home/9/hpe_user009/koshino
merge:                      y
hard resource_list:         hostipv4=1,docker=true,s_gpu=1,h_rt=600
soft resource_list:         docker_images=suse/sles12sp2:latest
mail_list:                  hpe_user009@nfs1
notify:                     FALSE
job_name:                   flatmpi
priority:                   -5
hard_queue_list:            docker.q
env_list:
SGE_ARRAY_MPI=true,RGST_PARAM_01=0,RGST_PARAM_02=2779,RGST_PARAM_03=2007,RGST_PARAM_04=0,RGST_PARAM_05=0,RGST_PARAM_06=0,RGST_PARAM_07=1,RGST_PARAM_0
script_file:               mpi.sh
parallel environment:       mpi_f_node range: 2
pe allocation rule:         2
department:                 defaultdepartment
job-array tasks:            1-4:1
task_concurrency:           all
docker_run_options:         --hostname=${hostipv4(0)},-v /home:/home,-v /scr:/scr,-v /dev/shm:/dev/shm,-v /etc/hosts:/etc/hosts,-v /var/lib/sss/pipes:/var/
lib/sss/pipes,-v /apps
binding:                    NONE
mbind:                      NONE
submit_cmd:                 qsub -A tga-hpe_group00 mpi.sh
start_time            1:    06/04/2018 13:41:36.766
start_time            2:    06/04/2018 13:41:36.772
start_time            3:    06/04/2018 13:41:36.775
start_time            4:    06/04/2018 13:41:36.780
job_state             1:    r
job_state             2:    r
job_state             3:    r
job_state             4:    r
exec_host_list        1:    r7i7n7:2
exec_host_list        2:    r7i7n7:2
exec_host_list        3:    r7i7n7:2
exec_host_list        4:    r7i7n7:2
granted_req.          1:    hostipv4=1, s_gpu=1
granted_req.          2:    hostipv4=1, s_gpu=1
granted_req.          3:    hostipv4=1, s_gpu=1
granted_req.          4:    hostipv4=1, s_gpu=1
usage                 1:    wallclock=00:00:00, cpu=00:00:00, mem=0.00000 GBs, io=0.00000 GB, iow=0.000 s, ioops=0, vmem=N/A, maxvmem=N/A
usage                 2:    wallclock=00:00:00, cpu=00:00:00, mem=0.00000 GBs, io=0.00000 GB, iow=0.000 s, ioops=0, vmem=N/A, maxvmem=N/A
usage                 3:    wallclock=00:00:00, cpu=00:00:00, mem=0.00000 GBs, io=0.00000 GB, iow=0.000 s, ioops=0, vmem=N/A, maxvmem=N/A
usage                 4:    wallclock=00:00:00, cpu=00:00:00, mem=0.00000 GBs, io=0.00000 GB, iow=0.000 s, ioops=0, vmem=N/A, maxvmem=N/A
binding               1:    r7i7n7=0,0:0,1
binding               2:    r7i7n7=0,7:0,8
binding               3:    r7i7n7=1,0:1,1
binding               4:    r7i7n7=1,7:1,8
resource map          1:    hostipv4=r7i7n7=(r7i7n7-cnode00), s_gpu=r7i7n7=(0)
resource map          2:    hostipv4=r7i7n7=(r7i7n7-cnode01), s_gpu=r7i7n7=(1)
resource map          3:    hostipv4=r7i7n7=(r7i7n7-cnode02), s_gpu=r7i7n7=(2)
resource map          4:    hostipv4=r7i7n7=(r7i7n7-cnode03), s_gpu=r7i7n7=(3)
^ Available containers:  r7i7n7-cnode00, r7i7n7-cnode01, r7i7n7-cnode02, r7i7n7-cnode03
scheduling info:            (Collecting of scheduler job information is turned off)
```

Info

When connecting to a compute node via ssh, the default GID of the processes after ssh is tsubame-users (2000), so the processes of your running job are not visible nor attachable by debuggers such as gdb except for trial execution cases.
To make it visible, do the following with the group name of the executed job after ssh.

`newgrp <group name>`

or

`sg <group name>`

5.6. Storage use on Compute Nodes

5.6.1. Local scratch area

Each node has SSD as local scratch disk space available to your job as `$TMPDIR` and `$T3TMPDIR`.
The local scratch area is an individual area of each compute node and is not shared.
To use it, you need to stage in and out from the job script to the local host.
The following example is a script to copy on one node. It does not correspond to multiple nodes.
Since `$TMPDIR` is deleted after each MPI termination, use `$T3TMPDIR` when using multiple MPI in one job.

```
#!/bin/sh
# copy input files
cp -rp $HOME/datasets $TMPDIR/
# execution
./a.out $TMPDIR/datasets $TMPDIR/results
# copy output files
cp -rp $TMPDIR/results $HOME/results
```

5.6.2. Shared scratch area

Only when using resource type f_node batch job, you can use BeeGFS On Demand (BeeOND), which creates SSD of reserved multiple computing nodes on demand as a shared file system. To enable BeeOND, specify f_node in the job script and specify " `#$ - v USE_BEEOND=1` for additional.
You can use it by referring to /beeond on the compute node. Here is a sample job script.

```
#!/bin/sh
#$ -cwd
#$ -l f_node=4
#$ -l h_rt=1:00:00
#$ -N flatmpi
#$ -v USE_BEEOND=1
. /etc/profile.d/modules.sh
module load cuda
module load intel
module load intel-mpi
mpiexec.hydra -ppn 8 -n 32 ./a.out
```

When using an interactive job, it can be used as follows. It takes a little time to mount the disk as compared with not using it.

```
$ qrsh -g [TSUBAME group] -l f_node=2 -l h_rt=0:10:00 -pty yes -v TERM -v USE_BEEOND=1 /bin/bash
```

The BeeOND shared scratch area is created at the timing secured by the job. You need to stage in and out from within the job script to /beeond.

```
#!/bin/sh
# copy input files
cp -rp $HOME/datasets /beeond/
# execution
./a.out $TMPDIR/datasets /beeond/results
# copy output files
cp -rp /beeond/results $HOME/results
```

6. ISV application

Under the license agreement, users who can use the ISV application are limited.
Users other than "1. Student/Staff ID" who belong to Tokyo Tech can only use the following ISV applications.

- Gaussian/Gauss View
- AMBER(Only users affiliated with academic institutions)
- Intel Compiler
- PGI Compiler
- Arm Forge

The list of installed ISV applications is as follows.

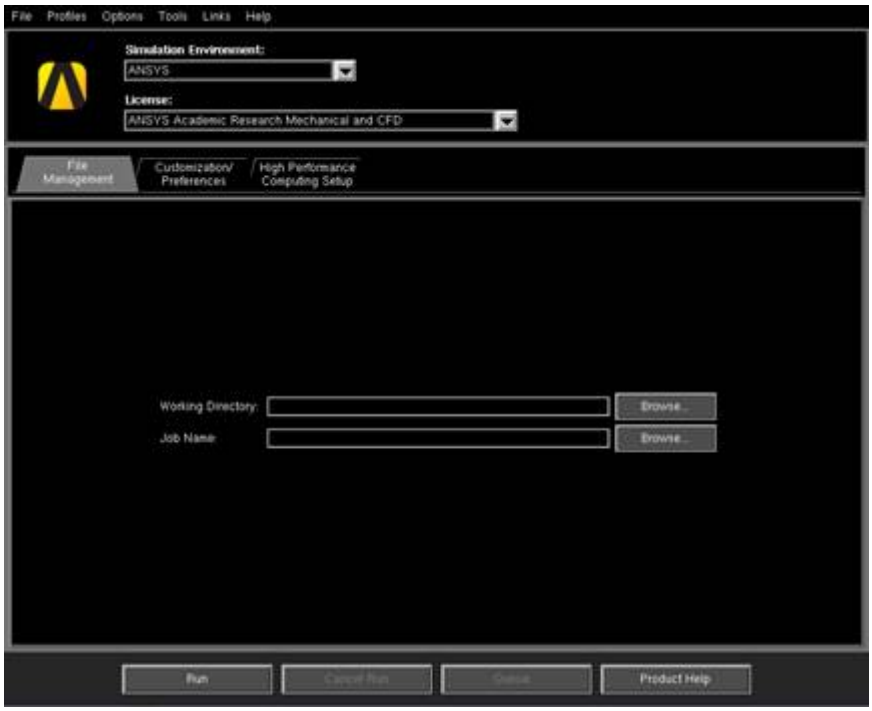
Software name	Description
ANSYS	Finite element software
Fluent	Finite volume software
ABAQUS	Finite element software
ABACUS CAE	Finite element software
Marc & Mentant / Dytran	Finite element software
Nastran	Finite element software
Patran	Finite element software Pre-Post tool
Gaussian	Computational chemistry Software
GaussView	Computational chemistry Software Pre-Post tool
AMBER	Computational chemistry Software
Materials Studio	Computational chemistry Software
Discovery Studio	Computational chemistry Software
Mathematica	Mathematical symbolic computation program
Maple	Mathematica l symbolic computation program
AVS/Express	Visualization software
AVS/Express PCE	Visualization software
LS-DYNA	Finite element software
LS-PrePost	Finite element software Pre-Post tool
COMSOL	Finite element software
Schrodinger	Computational chemistry Software
MATLAB	Mathematical software
Arm Forge	Debugger
Intel Compiler	Compiler
PGI Compiler	Compiler

6.1. ANSYS

You could run interactive use like in these examples.

GUI

```
$ module load ansys
$ launcher
```



CLI

```
$ module load ansys
$ mapdl
```

The following command could be used instead of the mapdl command.

```
When ANSYS 18.2 is loaded. The name of the command is different among the versions.
$ ansys182
```

Type exit to exit.

You could also specify the input file to run it with batch mode.

```
Example 1:
$ mapdl [options] < inputfile > outputfile
Example 2:
$ mapdl [options] -i inputfile -o outputfile
```

You could submit a batch job like in this example.

```
#### in case, sample.sh
$ qsub sample.sh
```

The following is a sample job script for MPI

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l f_node=2
#$ -l h_rt=0:10:0
```

```
. /etc/profile.d/modules.sh
module load ansys

mapdl -b -dis -np 56 < inputfile > outputfile
```

A sample script for GPU

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l f_node=1
#$ -l h_rt=0:10:0

. /etc/profile.d/modules.sh
module load ansys

mapdl -b -dis -np 28 -acc nvidia -na 4 < inputfile > outputfile
```

When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S ansyslmd -c 27001@lice0:27001@remote:27001@t3ldap1
```

6.2. Fluent

You can start with the following commands:

GUI

```
$ module load ansys
$ fluent
```

CLI

```
$ module load ansys
$ fluent -g
```

Type exit to exit.

You could run interactive use like in this example.

When the input file name is fluentbench, you and run with 3D version:

```
$fluent 3d -g -i fluentbench.jou
```

You could submit a batch job like in this example.

```
## in case, sample.sh
$ qsub sample.sh
```

The following is a sample job script: MPI parallel (f_node)

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l f_node=2
#$ -l h_rt=0:10:0

. /etc/profile.d/modules.sh
module load ansys

JOURNAL=journalfile
OUTPUT=outputfile
VERSION=3d

fluent -mpi=intel -g ${VERSION} -cnf=${PE_HOSTFILE} -i ${JOURNAL} > ${OUTPUT} 2>&1
```

The following is a sample job script: MPI parallel (h_node)

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l h_node=1
#$ -l h_rt=0:30:0

. /etc/profile.d/modules.sh
module load ansys

JOURNAL=journalfile
OUTPUT=outputfile
VERSION=3d

fluent -ncheck -mpi=intel -g ${VERSION} -cnf=${PE_HOSTFILE} -i ${JOURNAL} > ${OUTPUT} 2>&1
```

Since it is not possible to set across resources using f_node, set `#$ -l {resource name}=1` (for example, `#$ -l h_node=1` for h_node) and include the "-ncheck" option in the command.

When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S ansyslmd -c 27001@lice0:27001@remote:27001@t3ldap1
```

6.3. ABAQUS

You could run interactive use like in this example.

```
$ module load abaqus
$ abaqus job=inputfile [options]
```

You could submit a batch job like in this example.

```
#### in case, sample.sh
$ qsub sample.sh
```

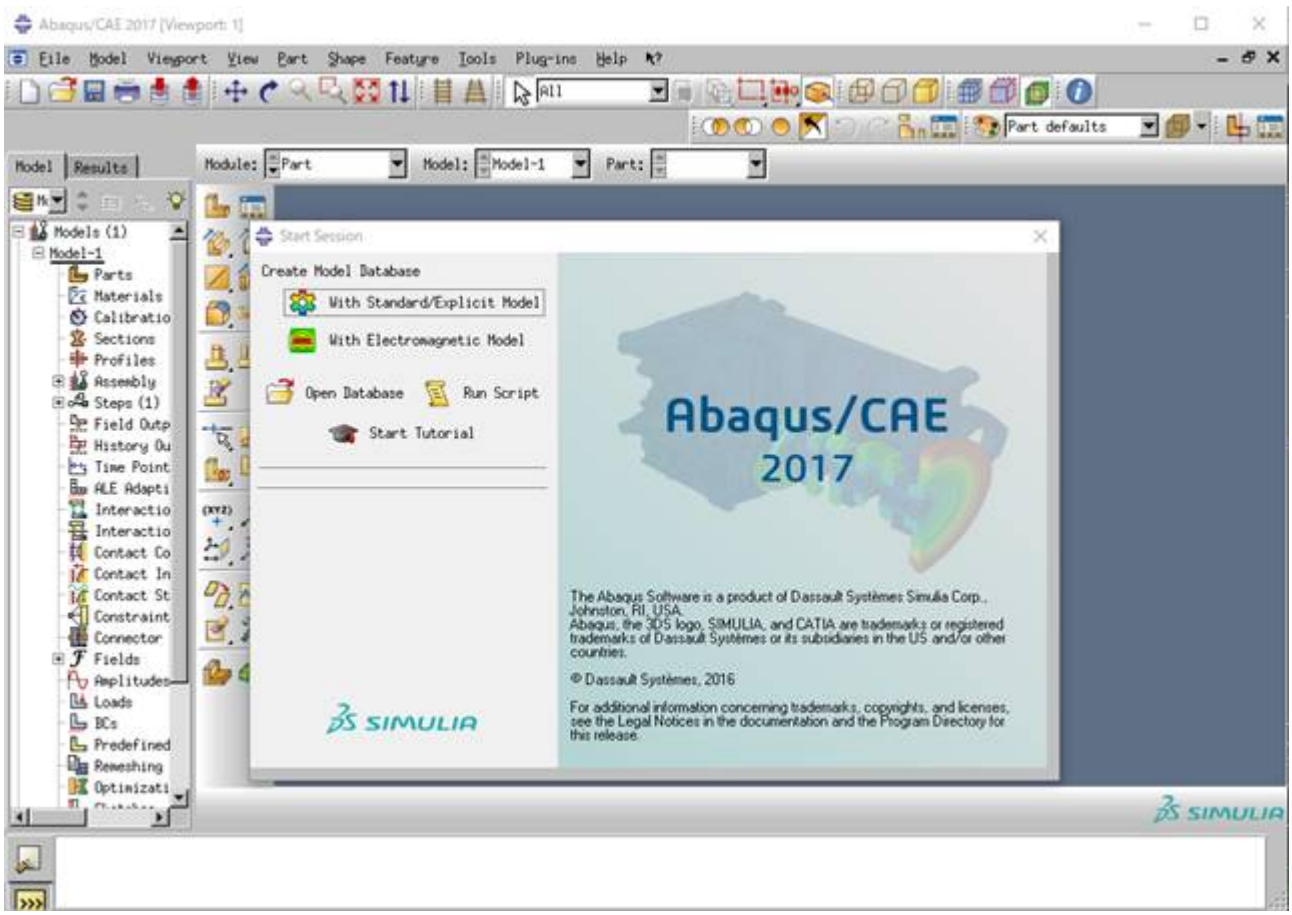
The following is a sample job script: MPI parallel

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l q_core=1
#$ -l h_rt=0:10:0

. /etc/profile.d/modules.sh
module load abaqus

## ABAQUS settings.
INPUT=s2a
ABAQUS_VER=2017
ABAQUS_CMD=abq${ABAQUS_VER}
SCRATCH=${base_dir}/scratch
NCPUS=2

cd ${base_dir}

${ABAQUS_CMD} interactive \
job=${INPUT} \
cpus=${NCPUS} \
scratch=${SCRATCH} \
mp_mode=mpi > ${INPUT}.`date '+%Y%m%d%H%M%S'`log 2>&1
```

6.4. ABAQUS CAE

You can start with the following commands:

```
$ module load abaqus
$ abaqus cae
```



Click File> Exit on the menu bar to exit.

6.5. Marc & Mentat / Dytran

6.5.1. Overview

For an overview of each product, please refer to the website of MSC Software Corporation.

- Marc: http://www.mscsoftware.com/ja/product/marc
- Dytran: http://www.mscsoftware.com/ja/product/dytran

6.5.2. Documentations

Please refer following documentations.

- Marc & Mentat Docs（mscsoftware.com）
- Dytran Docs（mscsoftware.com）

6.5.3. Marc

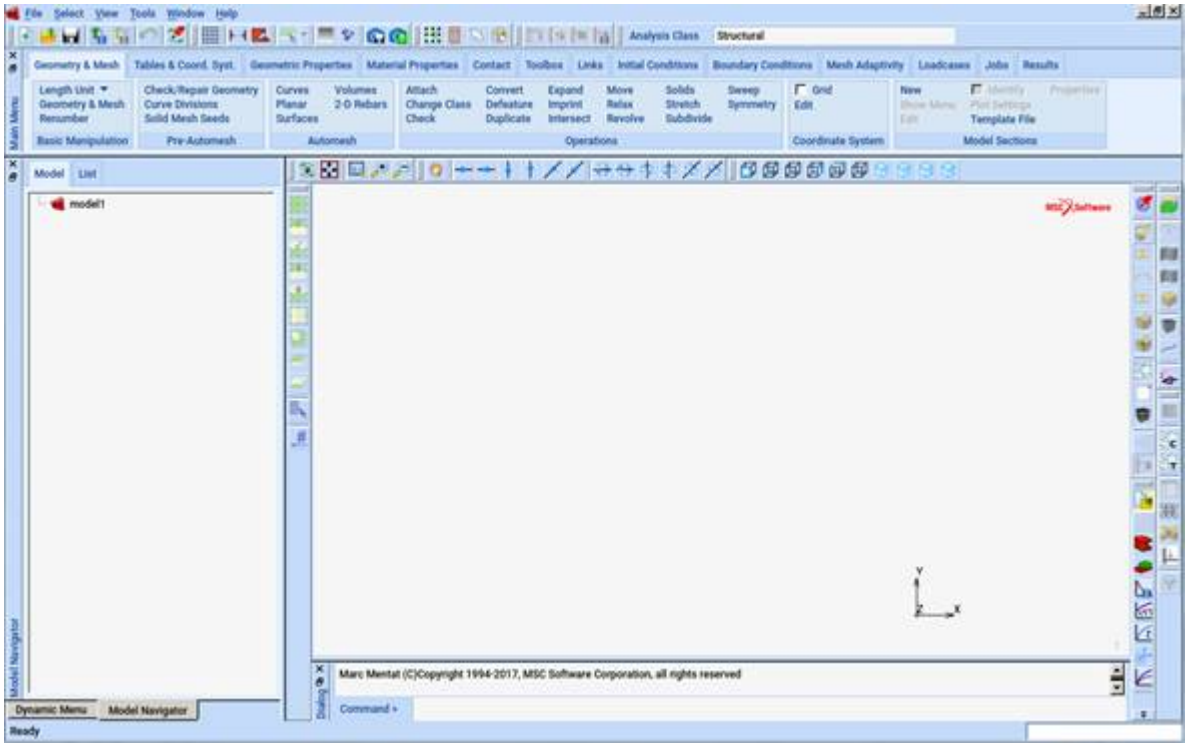
You could run interactive use like in this example.

```
$ module load intel intel-mpi cuda marc_mentat/2017
#### in case, sample file (e2x1.dat)
$ cp /apps/t3/sles12sp2/isv/msc/marc/marc2017/demo/ e2x1.dat ./
$ marc -jid e2x1
```

6.5.4. Mentat

You can start with the following commands:

```
$ cd <work directory>
$ module load intel intel-mpi cuda marc_mentat/2017
$ mentat
```



Click File> Exit on the menu bar to exit.

When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S MSC -c 27004@lice0:27004@remote:27004@t3ldap1
```

6.6. Nastran

You can start with the following commands:

```
$ module load nastran/2017.1
## In case, sample file (um24.dat)
$ cp /apps/t3/sles12sp2/isv/msc/MSC_Nastran/20171/msc20171/nast/demo/um24.dat ./
$ nast20171 um24
```

You could submit a batch job like in this example.

```
## In case, sample (parallel.sh)
$ qsub parallel.sh
```

The following is a sample job script:

```
#!/bin/bash
#$ -cwd
#$ -N nastran_parallel_test_job
#$ -l q_core=1
#$ -l h_rt=0:10:00
#$ -V

export NSLOTS=4
```

```
. /etc/profile.d/modules.sh
module load cuda openmpi nastran/2017.1


mpirun -np $NSLOTS \
nast20171 parallel=$NSLOTS um24
```

When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S MSC -c 27004@lice0:27004@remote:27004@t3ldap1
```

6.7. Patran

You can start with the following commands:

```
$ module load patran/2017.0.2
$ pat2017
```

Click File> Exit on the menu bar to exit.

When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S MSC -c 27004@lice0:27004@remote:27004@t3ldap1
```

6.8. Gaussian

You can start with the following commands: You can run interactive use like in this example.
Using the module for GPUs (GAUSS_CDEF and GAUSS_GDEF environmental variables will be set):

```
$ module load gaussian16/revision_gpu
$ g16 inputfile

Specify Gaussian's revision to revision. The example below is the case of Gaussian 16 Rev. B01.
$ module load gaussian16/B01_gpu
```

Using the non-GPU module (GAUSS_CDEF and GAUSS_GDEF not be set in the module):

```
$ module load gaussian16/revision
$ g16 inputfile
```

Using Linda:

```
$ module load gaussian16_linda
$ g16 inputfile
```

You could submit a batch job like in this example.

```
#### in case, sample.sh
$ qsub sample.sh
```

The following is a set of sample scripts for calculating the geometry optimization and vibration analysis (IR + Raman intensity) of glycine:

glycine.sh

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
```

```
#$ -l h_rt=0:10:0
#$ -V

. /etc/profile.d/modules.sh
module load gaussian16

g16 glycine.gjf
```

glycine.gjf

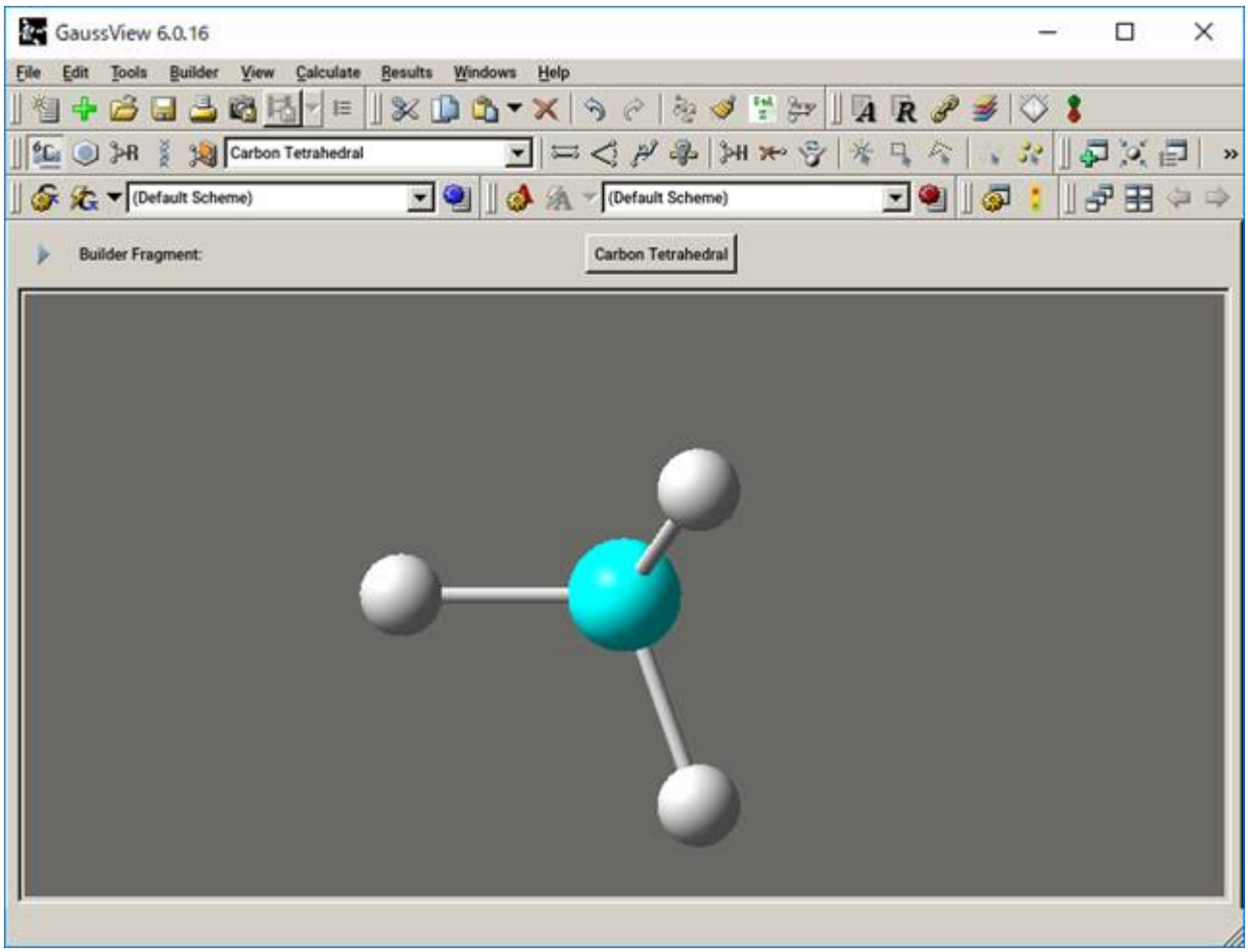
```
%chk=glycine.chk
%cpu=0-27    <- No need to describe when GAUSS_CDEF and GAUSS_CDEF are set.
%gpucpu=0-3=0,1,2,3    <- No need to describe when GAUSS_CDEF and GAUSS_CDEF are set or when you will not use GPUs.
%mem=120GB
#P opt=(calcfc,tight,rfo) freq=(raman)

glycine Test Job

0 2
 N                  0   -2.15739574   -1.69517043   -0.01896033 H
 H                  0   -1.15783574   -1.72483643   -0.01896033 H
 C                  0   -2.84434974   -0.41935843   -0.01896033 H
 C                  0   -1.83982674    0.72406557   -0.01896033 H
 H                  0   -3.46918274   -0.34255543   -0.90878333 H
 H                  0   -3.46918274   -0.34255543    0.87086267 H
 O                  0   -0.63259574    0.49377357   -0.01896033 H
 O                  0   -2.22368674    1.89158057   -0.01896033 H
 H                  0   -2.68286796   -2.54598119   -0.01896033 H

1 2 1.0 3 1.0 9 1.0
2
3 4 1.0 5 1.0 6 1.0
4 7 1.5 8 1.5
5
6
7
8
9
```

You can calculate by placing the above glycine.sh and glycine.gjf on the same directory and executing the following command.
After calculation, glycinetest.log, glycinetest.chk will be generated.
See GaussViewor verifying the analysis result.

6.9. GaussView

You can start with the following commands:

```
$ module load gaussian16 gaussview
$ gview.exe
```

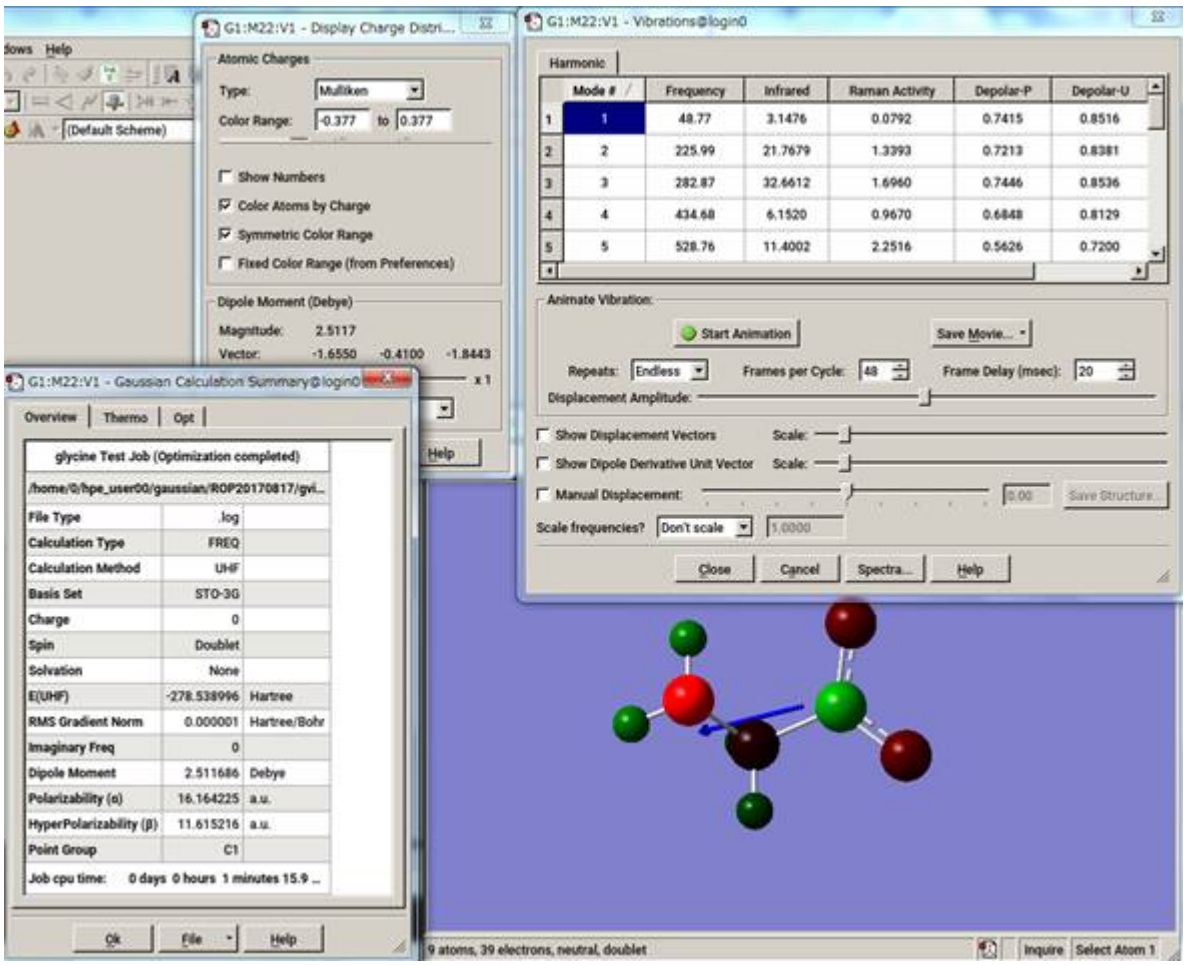


Click File> Exit on the menu bar to exit.

Example: glycine.log

```
$ module load gaussian16 gaussview
$ gview.exe glycine.log
```

The result of the analysis can be confirmed from [Result].
You can check calculation overview, charge information and vibration analysis from [Summary], [Charge Distribution] and [Vibration], respectively.
Since vibration analysis was performed in this example, the state of vibration can be confirmed from the [Start Animation] in the Vibration dialog.



6.10. AMBER

(1) You could run interactive use like in this example: CPU serial

```
$ module load amber/16
$ sander [-O|A] -i mdin -o mdout -p prmtop -c inpcrd -r restrt
```

(2) You could run interactive use like in this example: CPU parallel (sander.MPI)

```
$ module load amber/16
$ mpirun -np -[Number of processes] sander.MPI [-O|A] -i mdin -o mdout -p prmtop -c inpcrd -r restrt
```

(3) You could run interactive use like in this example: GPU serial (pmemd.cuda)

```
$ module load amber/16_cuda
$ pmemd.cuda [-O] -i mdin -o mdout -p prmtop -c inpcrd -r restrt
```

(4) You could run interactive use like in this example: GPU parallel (pmemd.cuda.MPI)

```
$ module load amber/16_cuda
$ mpirun -np -[Number of processes] pmemd.cuda.MPI [-O] -i mdin -o mdout -p prmtop -c inpcrd -r restrt
```

(5) You could submit a batch job like in this example.

```
## in case, parallel.sh
$ qsub parallel.sh
```

The following is a sample job script: CPU parallel

```
#!/bin/bash
#$ -cwd
#$ -l f_node=2
#$ -l h_rt=0:10:00
#$ -V
export NSLOTS=56

in=./mdin
out=./mdout_para
inpcrd=./inpcrd
top=./top

cat <<eof > $in
 Relaxtion of trip cage using
&cntrl
  imin=1,maxcyc=5000,irest=0, ntx=1,
  nstlim=10, dt=0.001,
  ntc=1, ntf=1, ioutfm=1
  ntt=9, tautp=0.5,
  tempi=298.0, temp0=298.0,
  ntpr=1, ntwx=20,
  ntb=0, igb=8,
  nkija=3, gamma_ln=0.01,
  cut=999.0,rgbmax=999.0,
  idistr=0
/
eof

. /etc/profile.d/modules.sh
module load amber/16

mpirun -np $NSLOTS \
sander.MPI -O -i $in -c $inpcrd -p $top -o $out < /dev/null

/bin/rm -f $in restrt
```

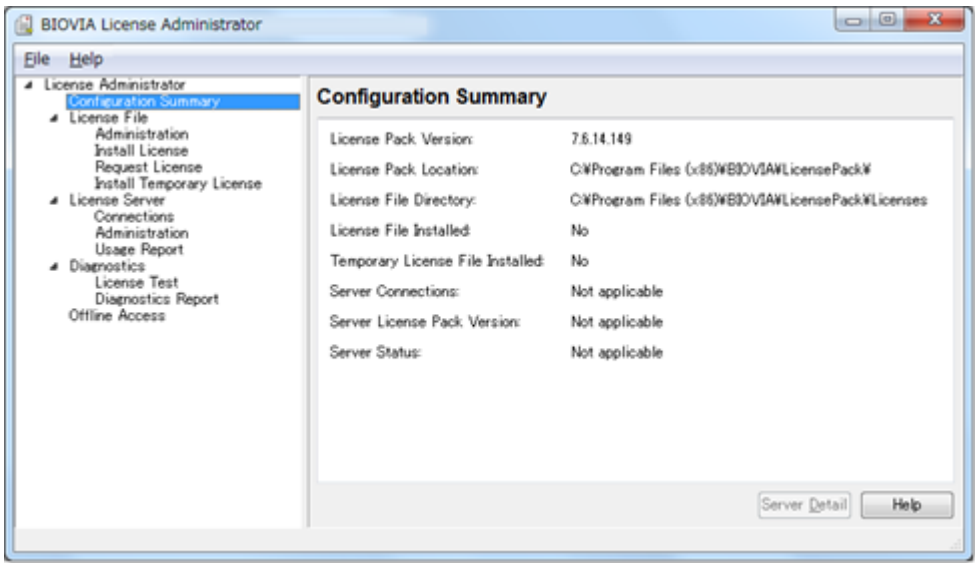
The following is a sample job script: GPU parallel

```
#!/bin/bash
#$ -cwd
#$ -l f_node=2
#$ -l h_rt=0:10:0
#$ -V

export NSLOTS=56

in=./mdin
out=./mdout
inpcrd=./inpcrd
top=./top

cat <<eof > $in
FIX (active) full dynamics ( constraint dynamics: constant volume)
&cntrl
   ntx = 7,       irest = 1,
   ntpr = 100,     ntwx = 0,     ntwr = 0,
   ntf = 2,       ntc = 2,       tol = 0.000001,
   cut = 8.0,
   nstlim = 500,  dt = 0.00150,
   nscm = 250,
   ntt = 0,
   lastist = 4000000,
   lastrst = 6000000,
/
eof

. /etc/profile.d/modules.sh
module load amber/16_cuda

mpirun -np $NSLOTS \
pmemd.cuda.MPI -O -i $in -c $inpcrd -p $top -o $out < /dev/null

/bin/rm -f $in restrt
```

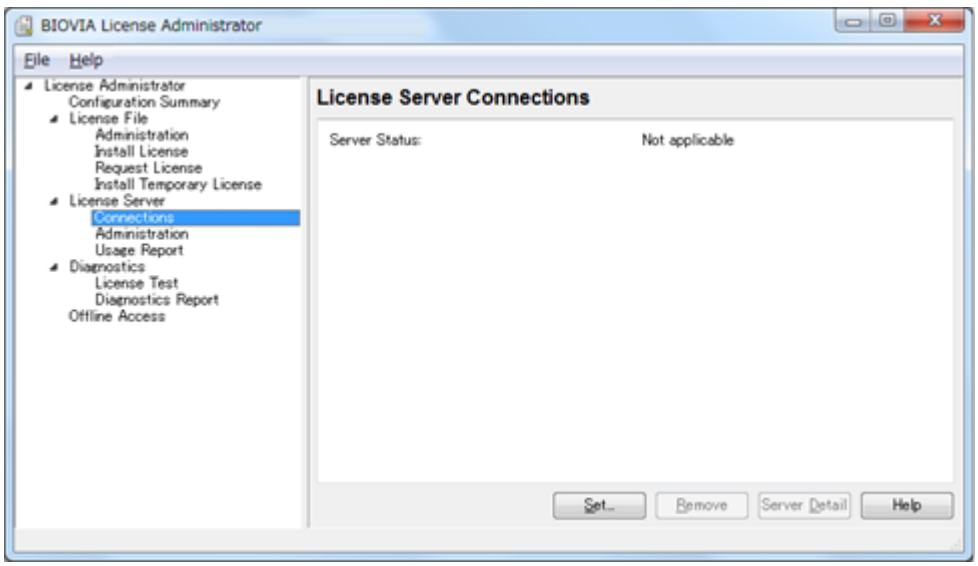
6.11. Materials Studio

6.11.1. License connection setting

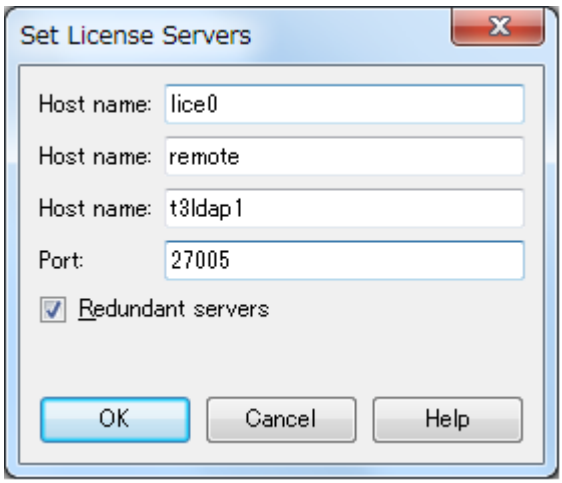
Execute All Programs > BIOVIA > Licensing > License Administrator 7.6.14 from the Windows [Start menu] with system administrator privileges.



Click [Connections] -[Set] , and open "Set License Server" dialog.



Select Redundant Server and type each host name and a port number.

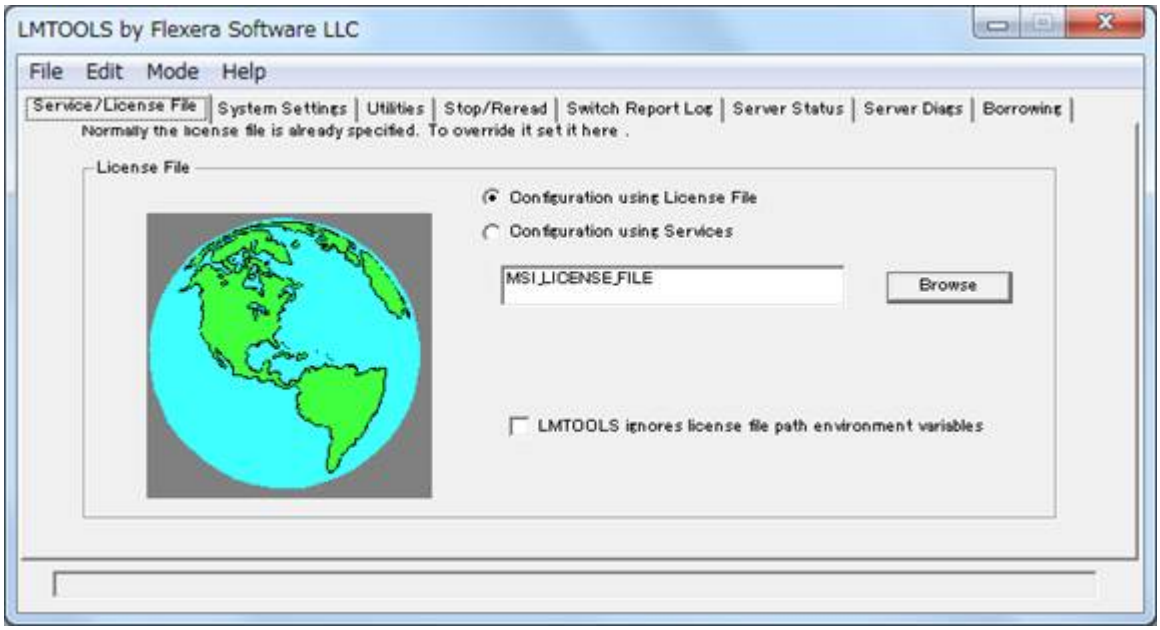


If server status is displayed as "Connected", setting is completed.
(note) You need to establish a connection with two or more license servers.

6.11.2. License Usage Status

6.11.2.1. On Windows

Execute All Programs > BIOVIA > Licensing > License Administrator 7.6.14 > Utilities (FLEXlm LMTOOLs) from the Windows [Start menu] . Open [Service/License File] tab and slect [Configulation using License File] .
Make sure that MSI_LICENSE_FILE is displayed.



Open [Server Status] tab, click [Perform Status Enqurity] and you can see usage status of the license.
If you want to display only specific licenses, enter the license name that you want to display in [Individual Feature] and execute [Perform Status Enqurity].

6.11.2.2. On login node

When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S msi -c 27005@lice0,27005@remote,27005@t3ldap1
```

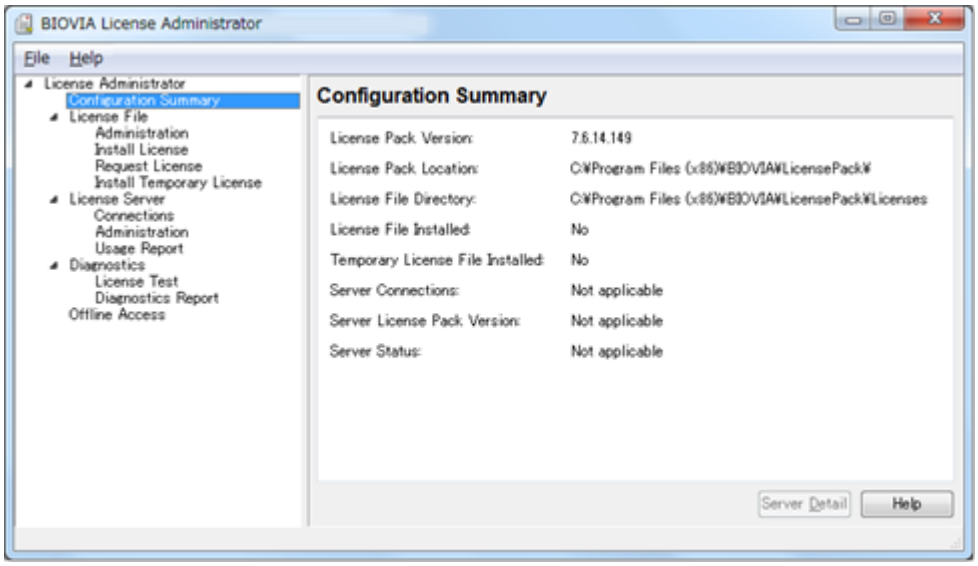
6.11.3. Start up Materials Studio

Click BIOVIA > Materials Studio 2017 R2 from the Windows [Start menu].

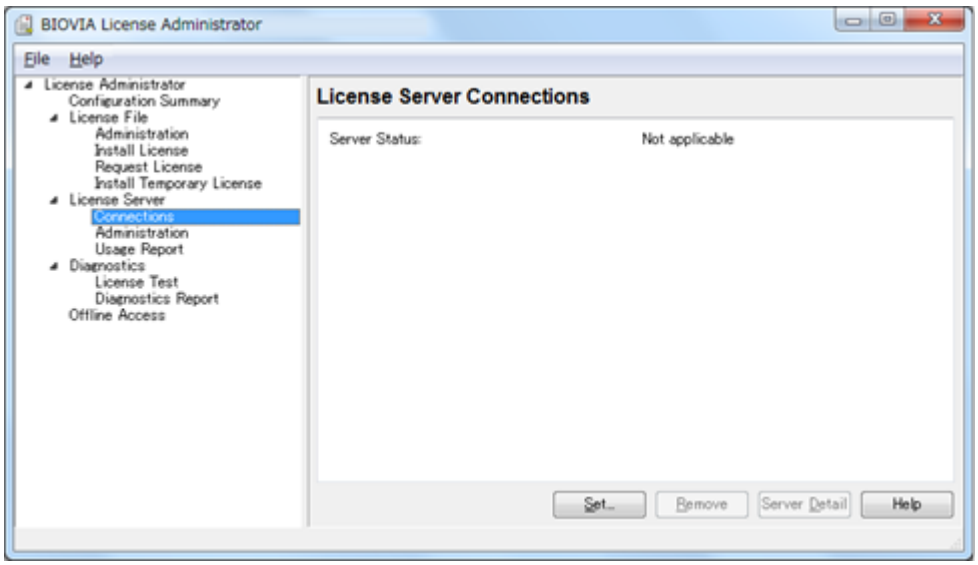
6.12. Discovery Studio

6.12.1. License connection setting

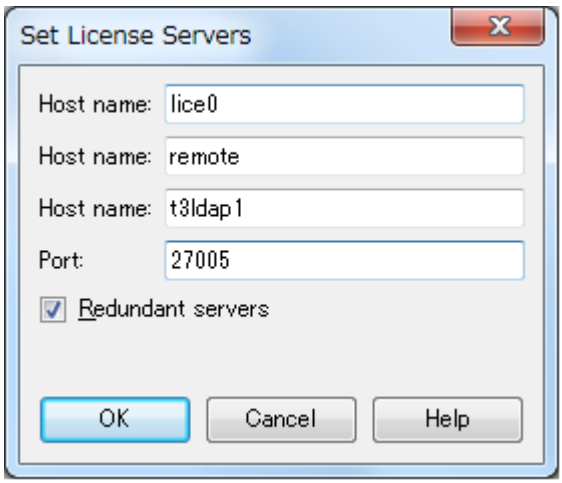
Execute All Programs > BIOVIA > Licensing > License Administrator 7.6.14 from the Windows [Start menu] with system administrator privileges.



Click [Connections] -[Set] , and open "Set License Server" dialog.



Select Redundant Server and type each host name and a port number.

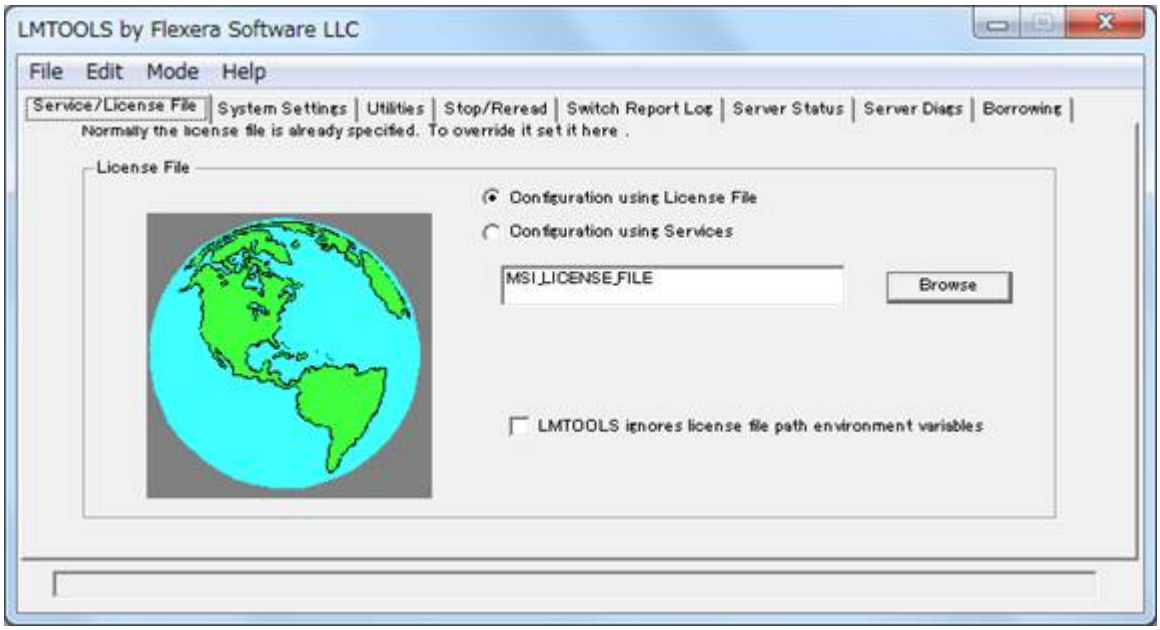


If server status is displayed as "Connected," the setting is completed.
(note) You need to establish a connection with two or more license servers.

6.12.2. License Usage Status

6.12.2.1. On Windows

Execute All Programs > BIOVIA > Licensing > License Administrator 7.6.14 > Utilities (FLEXlm LMTOOLs) from the Windows [Start menu] .
Open [Service/License File] tab and slect [Configulation using License File] .
Make sure that MSI_LICENSE_FILE is displayed.



Open [Server Status] tab, click [Perform Status Enqurity] and you can see usage status of the license.
If you want to display only specific licenses, enter the license name that you want to see in [Individual Feature] and execute [Perform Status Enqurity].

6.12.2.2. On login node

When you execute the following command, usage status is displayed.

```
$ lmutil lmstat -S msi -c 27005@lice0,27005@remote,27005@t3ldap1
```

6.12.3. Start up Discovery Studio

Click BIOVIA > Discovery Studio 2017 R2 64-bit Client from the Windows [Start menu] .

6.13. Mathematica

You can start with the following commands:

CLI

```
$ module load mathematica
$ math
Mathematica 11.1.1 Kernel for Linux x86 (64-bit)
Copyright 1988-2017 Wolfram Research, Inc.

In[1]:=
```

Type Quit to exit.

GUI

```
$ module load mathematica
$ Mathematica
```

To exit the Wolfram System, you typically choose the "Exit" menu item in the notebook interface.

6.14. Maple

You can start with the following commands:

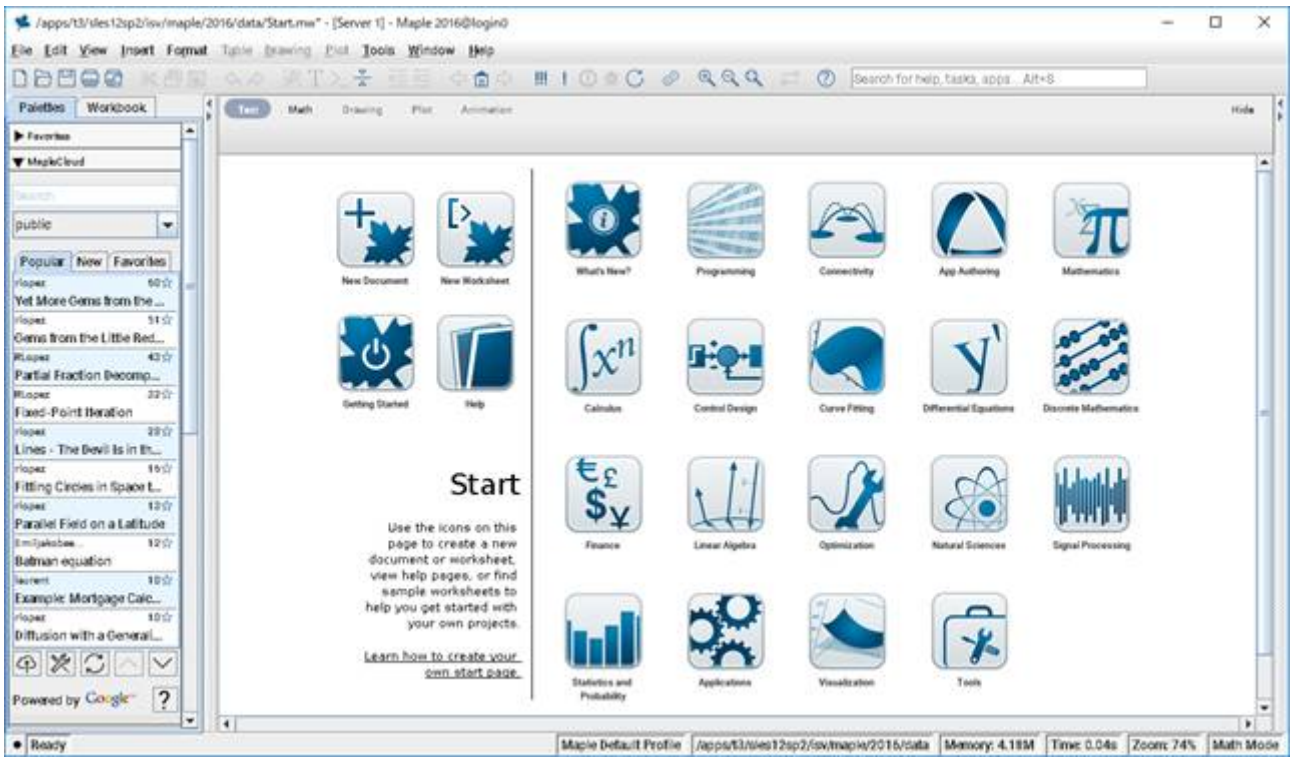
CLI

```
$ module load maple/2016.2
$ maple
    |\^/|     Maple 2016 (X86 64 LINUX)
._|\|   |/|_. Copyright (c) Maplesoft, a division of Waterloo Maple Inc. 2018
 \  MAPLE  /  All rights reserved. Maple is a trademark of
 <____ ____>  Waterloo Maple Inc.
      |       Type ? for help.
>
```

Type Quit to exit.

GUI

```
$ module load maple/2016.2
$ xmaple
```



Click File> Exit on the menu bar to exit.

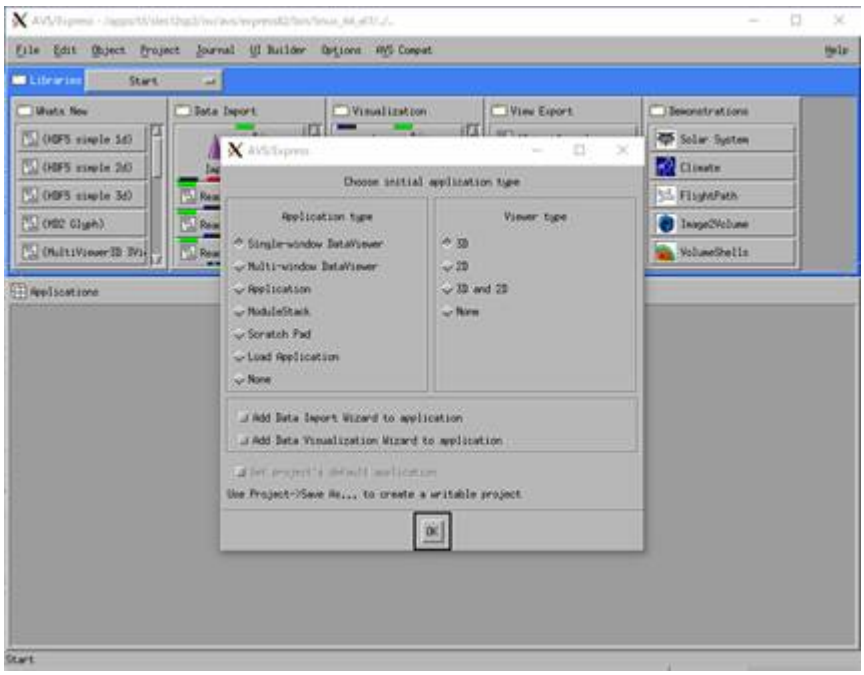
When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S maplelmg -c 27007@lice0:27007@remote:27007@t3ldap1
```

6.15. AVS/Express

You can start with the following commands:

```
$ module load avs/8.4
$ xp
```



The option "nohw" is needed to start without hardware acceleration.
Click File> Exit on the menu bar to exit.

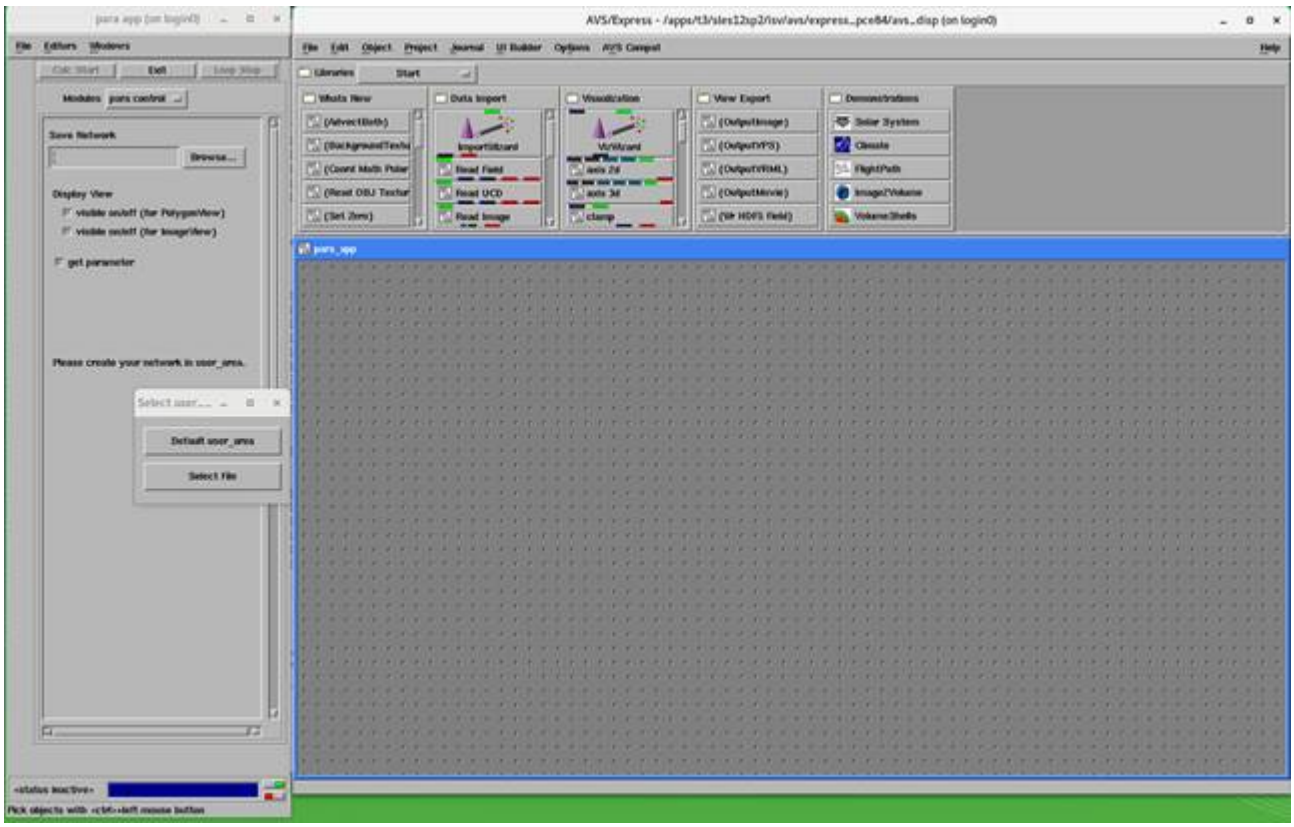
When you execute the following command, license Usage Status is displayed.

```
$ w3m http://lice0:33333/STATUS
```

6.16. AVS/Express PCE

You can start with the following commands:

```
$ module load avs/8.4
$ para_start
```



Click File> Exit on the menu bar to exit.

When you execute the following command, license Usage Status is displayed.

```
$ w3m http://lice0:33333/STATUS
```

6.17. LS-DYNA

6.17.1. Overview LS-DYNA

LS-DYNA is a general-purpose finite element program capable of simulating complex real-world problems. It is used by the automobile, aerospace, construction, military, manufacturing, and bioengineering industries.

6.17.2. Executing LS-DYNA

You can use the following sample scripts to submit jobs.
Please replace input files and versions appropriately.

[SMP in sigle precision]

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l h_node=1
#$ -l h_rt=0:10:0

. /etc/profile.d/modules.sh
module load cuda/8.0.44
module load lsdyna/R9.1.0

export base_dir=/home/4/t3-test00/isv/lsdyna
cd $base_dir/smp_s

export exe=smpdynas

#export LSTC_LICENSE=network
```

```
#export LSTC_MEMORY=auto

export NCPUS=4
export OMP_NUM_THREADS=${NCPUS}
export INPUT=$base_dir/sample/airbag_deploy.k

${exe} i=${INPUT} ncpus=${NCPUS}
```

[SMP in double precision]

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l h_node=1
#$ -l h_rt=0:10:0

. /etc/profile.d/modules.sh
module load cuda/8.0.44
module load lsdyna/R9.1.0

export base_dir=/home/4/t3-test00/isv/lsdyna
cd $base_dir/smp_d

export exe=smpdynad

#export LSTC_LICENSE=network
#export LSTC_MEMORY=auto

export NCPUS=4
export OMP_NUM_THREADS=${NCPUS}
export INPUT=$base_dir/sample/airbag_deploy.k

${exe} i=${INPUT} ncpus=${NCPUS}
```

[MPP in sigle precision]

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l h_node=1
#$ -l h_rt=0:10:0

. /etc/profile.d/modules.sh
module load cuda/8.0.44
module load lsdyna/R9.1.0 mpt/2.16

export base_dir=/home/4/t3-test00/isv/lsdyna
cd $base_dir/mpp_s

export exe=mppdynas_avx2
export dbo=l2as_avx2

#export LSTC_LICENSE=network
#export LSTC_MEMORY=auto

export NCPUS=4
export OMP_NUM_THREADS=1
export INPUT=$base_dir/sample/airbag_deploy.k

export MPI_BUFS_PER_PROC=512
export MPI_REMSH=ssh

mpiexec_mpt -v -np 4 dplace -s1 ${exe} i=${INPUT} ncpus=${NCPUS}
${dbo} binout*
```

Info

Instead of standalone LS-DYNA with lsdyna module, you can choose LS-DYNA included in ANSYS (ansys module).
Please refer to the following example job scirpt for necessary configurations.

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l h_node=1
#$ -l h_rt=5:00:0

. /etc/profile.d/modules.sh

module load ansys intel-mpi

export dynadir=/apps/t3/sles12sp2/isv/ansys_inc/v231/ansys/bin/linx64/
export exe=$dynadir/lsdyna_sp_mpp.e
export dbo=$dynadir/lsl2a_sp.e

export LSTC_LICENSE_SERVER='(27008@lice0 27008@remote 27008@t3ldap1)'
export NCPUS=4
export INPUT=$base_dir/sample/airbag_deploy.k

mpiexec -np ${NCPUS} ${exe} i=${INPUT}

${dbo} binout*
```

[MPP in doble precision]

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l h_node=1
#$ -l h_rt=0:10:0

. /etc/profile.d/modules.sh
module load cuda/8.0.44
module load lsdyna/R9.1.0 mpt/2.16

export base_dir=/home/4/t3-test00/isv/lsdyna
cd $base_dir/mpp_d

export exe=mppdynad_avx2
export dbo=l2ad_avx2

#export LSTC_LICENSE=network
#export LSTC_MEMORY=auto

export NCPUS=4
export OMP_NUM_THREADS=1
export INPUT=$base_dir/sample/airbag_deploy.k

export MPI_BUFS_PER_PROC=512
export MPI_REMSH=ssh

mpiexec_mpt -v -np 4 dplace -s1 ${exe} i=${INPUT} ncpus=${NCPUS}

${dbo} binout*
```

Info

Instead of standalone LS-DYNA with lsdyna module, you can choose LS-DYNA included in ANSYS (ansys module).
Please refer to the following example job scirpt for necessary configurations.

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l h_node=1
#$ -l h_rt=5:00:0

. /etc/profile.d/modules.sh

module load ansys intel-mpi

export dynadir=/apps/t3/sles12sp2/isv/ansys_inc/v231/ansys/bin/linx64/
export exe=$dynadir/lsdyna_dp_mpp.e
export dbo=$dynadir/lsl2a_dp.e

export LSTC_LICENSE_SERVER='(27008@lice0 27008@remote 27008@t3ldap1)'
export NCPUS=4
export INPUT=$base_dir/sample/airbag_deploy.k

mpiexec -np ${NCPUS} ${exe} i=${INPUT}

${dbo} binout*
```

Please change the script according to the user's environment.
The input file is specified as INPUT=inputfile in the shell script.

When you execute the following command, license Usage Status is displayed.

```
$ lstc_qrun
```

6.18. LS-PrePost

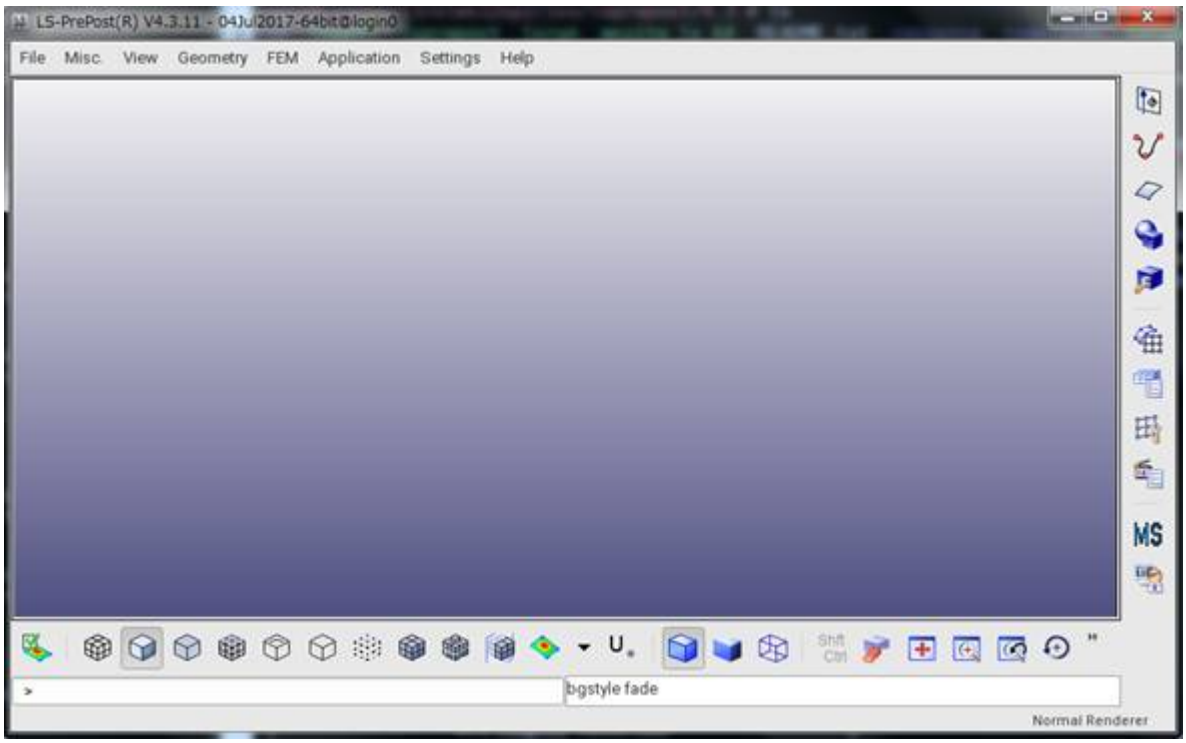
6.18.1. Overview LS-PrePost

LS-PrePost is an advanced pre and post-processor that is delivered free with LS-DYNA. The user interface is designed to be both efficient and intuitive. LS-PrePost runs on Windows, Linux, and Unix utilizing OpenGL graphics to achieve fast rendering and XY plotting.

6.18.2. Executing LS-PrePost

You can start with the following commands:

```
$ module load lsprepost/4.3
$ lsprepost


 _____________________________________________
|                                             |
 |     Livermore Software Technology Corporation          |
|                                             |
 |                L S - P R E P O S T                     |
|                                             |
 |    Advanced Pre- and Post-Processor for LS-DYNA         |
|                                             |
 |         LS-PrePost(R) V4.3.11 - 04Jul2017               |
|                                             |
 |            LSTC Copyright (C) 1999-2014               |
 |                All Rights Reserved                     |
 |_____________________________________________|

 OpenGL version 3.0 Mesa 11.2.1
```

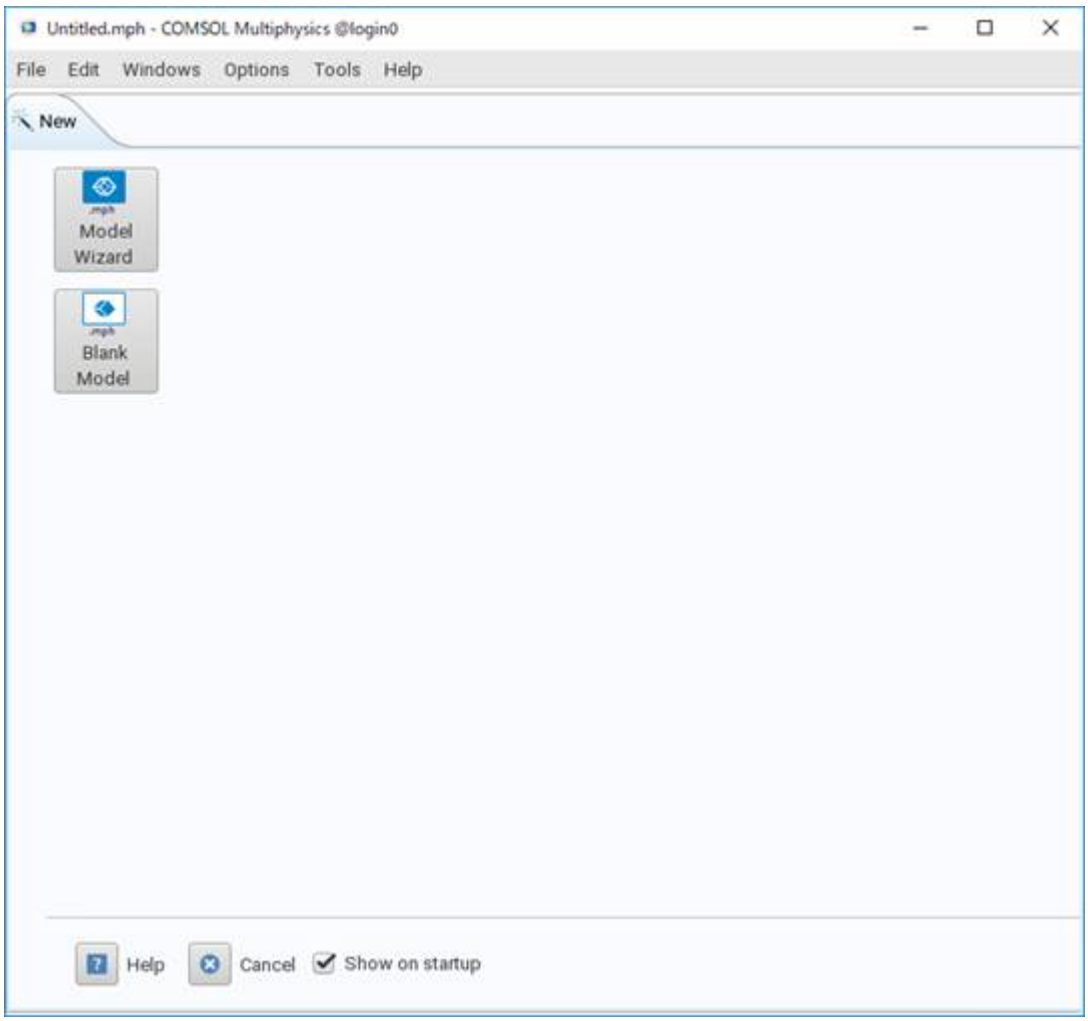


Click File> Exit on the menu bar to exit.

6.19. COMSOL

You can start with the following commands:

```
$ module load comsol
$ comsol
```



Click File> Exit on the menu bar to exit.

When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat -S LMCOMSOL -c 27009@lice0:27009@remote:27009@t3ldap1
```

6.20. Schrodinger

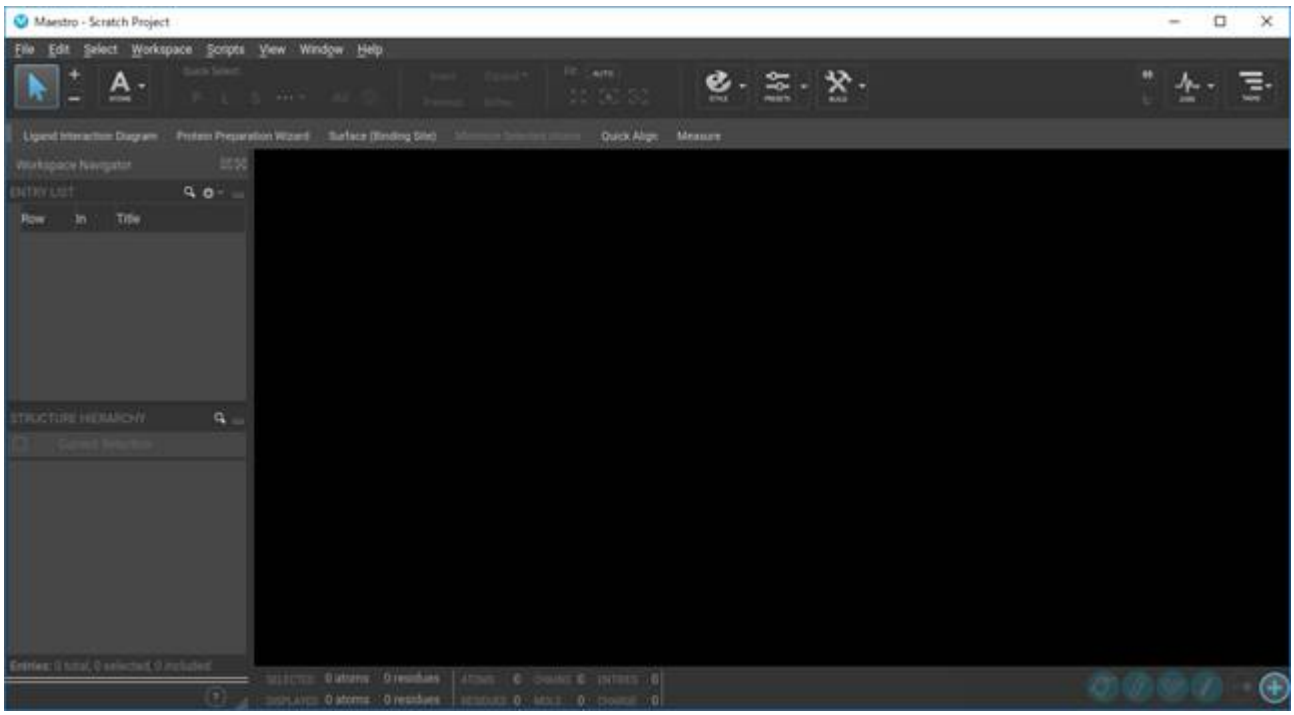
You can start with the following commands:

CLI

```
$ module load schrodinger/Feb-17
$ ligprep -ismi <input file> -omae <output file>
```

GUI

```
$ module load schrodinger/Feb-17
$ maestro
```



Click File> Exit on the menu bar to exit.

When you execute the following command, license Usage Status is displayed.

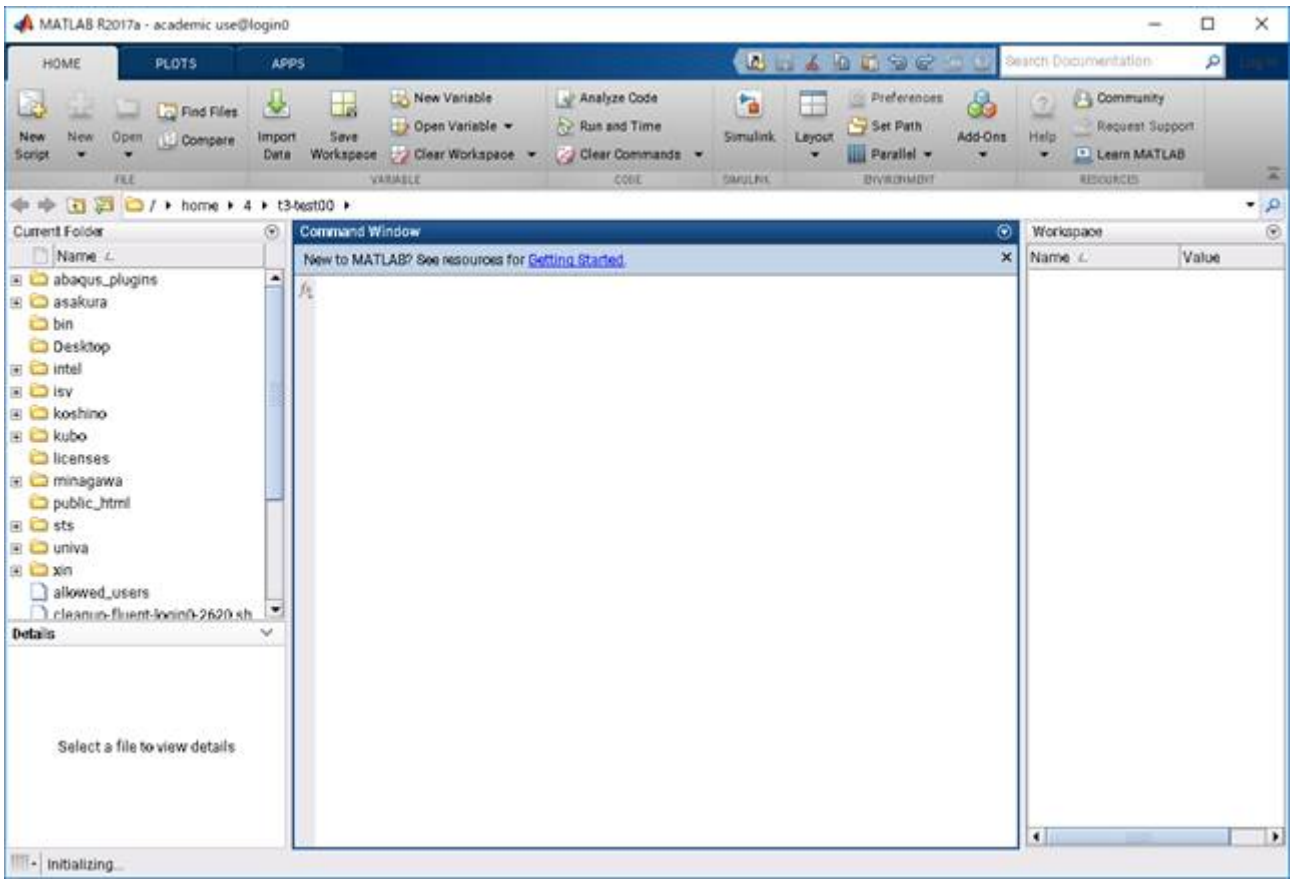
```
$ lmutil lmstat -S SCHROD -c 27010@lice0:27010@remote:27010@t3ldap1
```

6.21. MATLAB

You can start with the following commands:

GUI

```
$ module load matlab
$ matlab
```



CLI

```
$ module load matlab
$ matlab -nodisplay
```

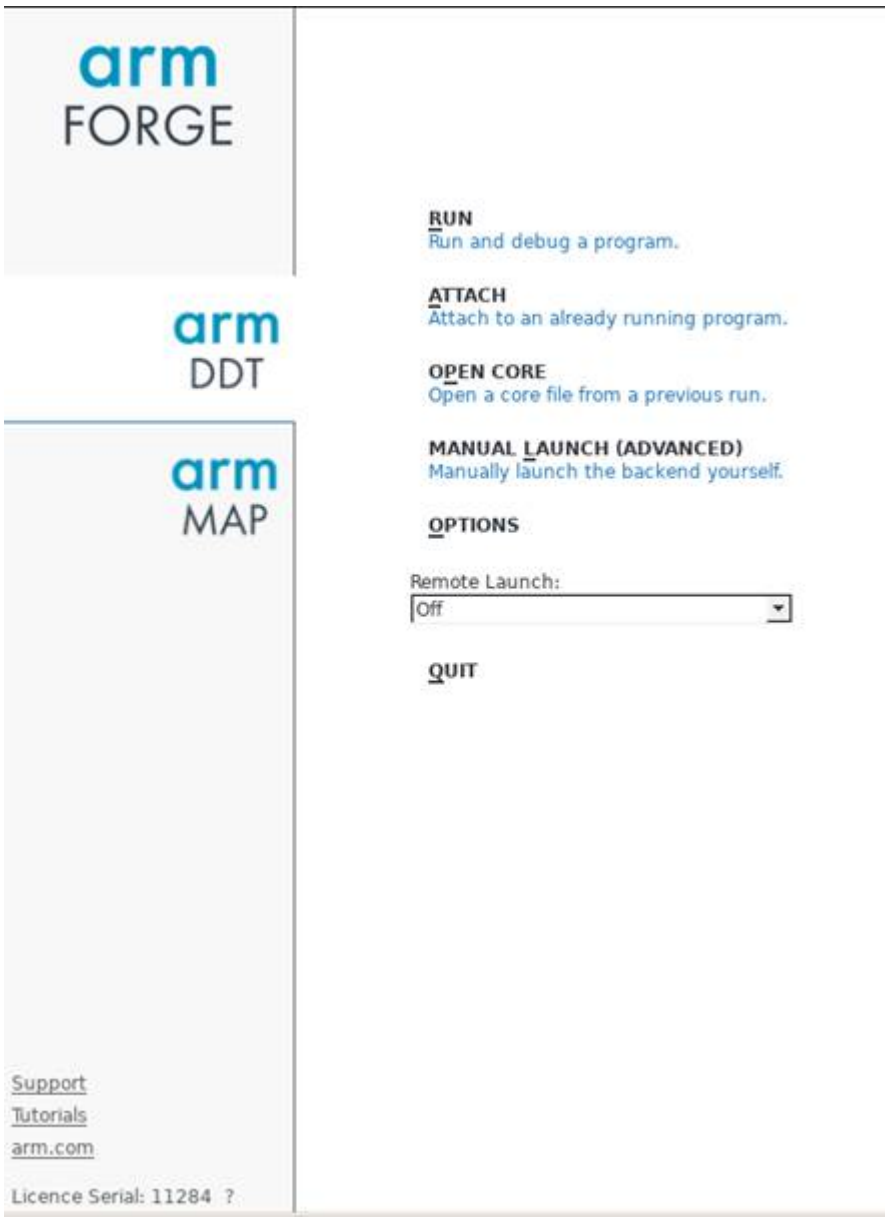
When you execute the following command, license Usage Status is displayed.

```
$ lmutil lmstat-S MLM -c 27014@lice0:27014@remote:27014@t3ldap1
```

6.22. Arm Forge

You can start with the following commands:

```
$ module load forge
$ forge
```



Click File> Exit on the menu bar to exit.

7. Freeware

The list of the installed freeware is as follows:

Software name	Description
GAMESS	Computational chemistry Software
Tinker	Computational chemistry Software
GROMACS	Computational chemistry Software
LAMMPS	Computational chemistry Software
NAMMD	Computational chemistry Software
CP2K	Computational chemistry Software
QUANTUM ESPRESSO	Computational chemistry Software
OpenFOAM	Computational Software
CuDNN	GPU library
NCCL	GPU library
Caffe	DeepLearning Framework
Chainer	DeepLearning Framework
TensorFlow	DeepLearning Framework
DeePMD-kit	DeepLearning Framework for MD
R	statistics Interpreter
clang	compiler
Apache Hadoop	Distributed data processing tool
POV-Ray	Visualization software
ParaView	Visualization software
VisIt	Visualization software
turbovnc	Remote GUI
gnuplot	Data visualization
Tgif	Graphics tool
GIMP	Image display and manipulation
ImageMagick	Image display and manipulation
TeX Live	TeX distribution
Java SDK	Development environment
PETSc	Scientific Computation Library
FFTW	FFT library
DMTCP	Checkpoint tool
Singularity	Linux container for HPC

7.1. Computational chemistry Software

7.1.1. GAMESS

The following is a sample job script.

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N gamess

. /etc/profile.d/modules.sh
module load intel intel-mpi gamess

cat $PE_HOSTFILE | awk '{print $1}' > $TMPDIR/machines
cd $GAMESS_DIR
./rungms exam08 mpi 4 4
```

For more details, please refer the following site: https://www.msg.ameslab.gov/gamess/index.html

7.1.2. Tinker

The following is a sample job script.

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N tinker

. /etc/profile.d/modules.sh
module load intel tinker

cp -rp $TINKER_DIR/example $TMPDIR
cd $TMPDIR/example
dynamic waterbox.xyz -k waterbox.key 100 1 1 2 300
cp -rp $TMPDIR/example $HOME
```

For more details, please refer the following site: https://dasher.wustl.edu/tinker/

7.1.3. GROMACS

The following is a sample job script.

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N gromacs

. /etc/profile.d/modules.sh

module load cuda/11.2.146 intel-mpi python/3.11.2 gcc/10.2.0 gromacs
cp -rp $GROMACS_DIR/examples/water_GMX50_bare.tar.gz $TMPDIR
cd $TMPDIR
tar xf water_GMX50_bare.tar.gz
cd water-cut1.0_GMX50_bare/3072
gmx_mpi grompp -f pme.mdp
OMP_NUM_THREADS=2 mpiexec.hydra -np 4 gmx_mpi mdrun
cp -rp $TMPDIR/water-cut1.0_GMX50_bare $HOME
```

For more details, please refer the following site: http://www.gromacs.org/

7.1.4. LAMMPS

The following is a sample job script.

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N lammps

. /etc/profile.d/modules.sh

module load intel cuda openmpi/3.1.4-opa10.10-t3 ffmpeg python/3.11.2 lammps
cp -rp $LAMMPS_DIR/examples/VISCOSITY $TMPDIR
cd $TMPDIR/VISCOSITY
mpirun -x PATH -x LD_LIBRARY_PATH -x PSM2_CUDA=1 -np 4 lmp -pk gpu 0 -in in.gk.2d
cp -rp $TMPDIR/VISCOSITY $HOME
```

For more details, please refer the following site: http://lammps.sandia.gov/doc/Section_intro.html

7.1.5. NAMD

The following is a sample job script.

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N namd

. /etc/profile.d/modules.sh
module load cuda intel namd

cp -rp $NAMD_DIR/examples/stmv.tar.gz $TMPDIR
cd $TMPDIR
tar xf stmv.tar.gz
cd stmv
namd3 +idlepoll +p4 +devices 0,1,2,3 stmv.namd
cp -rp $TMPDIR/stmv $HOME
```

Info

Command name is `namd2` on older versions, please replace `namd3` with `namd2` when you use them.

For more details, please refer the following site: https://www.ks.uiuc.edu/Research/namd/3.0/ug/

7.1.6. CP2K

The following is a sample job script.

```
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N cp2k

. /etc/profile.d/modules.sh
module load cuda gcc openmpi/3.1.4-opa10.10-t3 cp2k
cp -rp $CP2K_DIR/benchmarks/QS $TMPDIR
cd $TMPDIR/QS
export OMP_NUM_THREADS=1
mpirun -x PATH -x LD_LIBRARY_PATH -x PSM2_CUDA=1 -np 4 cp2k.psmp -i H2O-32.inp -o H2O-32.out
cp -rp $TMPDIR/QS $HOME
```

For more details, please refer the following site: https://www.cp2k.org/

7.1.7. QUANTUM ESPRESSO

The following is a sample job script.

```
#!/bin/sh
#$ -cwd
#$ -l h_rt=00:10:00
#$ -l f_node=1
```

```
#$ -N q-e
. /etc/profile.d/modules.sh

module purge
module load cuda/10.2.89 pgi openmpi/3.1.4-opa10.10-t3 quantumespresso

cp -p $QUANTUMESPRESSO_DIR/test-suite/pw_scf/scf.in .
cp -p $QUANTUMESPRESSO_DIR/example/Si.pz-vbc.UPF .

mpirun -x ESPRESSO_PSEUDO=$PWD -x PATH -x LD_LIBRARY_PATH -x PSM2_CUDA=1 -x PSM2_GPUDIRECT=1 -np 4 pw.x < scf.in
```

For more details, please refer the following site: https://www.quantum-espresso.org/

7.2. CFD software

7.2.1. OpenFOAM

There are two versions of OpenFOAM, Foundation version is named "openfoam" and ESI version is named "openfoam-esi".
The following is a sample job script.

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N openform

. /etc/profile.d/modules.sh
module load cuda openmpi openfoam

mkdir -p $TMPDIR/$FOAM_RUN
cd $TMPDIR/$FOAM_RUN
cp -rp $FOAM_TUTORIALS .
cd tutorials/incompressible/icoFoam/cavity/cavity
blockMesh
icoFoam
paraFoam
```

If you want to use ESI version, please replace `module load cuda openmpi openfoam` with `module load cuda openmpi openfoam-esi`.

For more details, please refer the following site:

https://openfoam.org/resources/
http://www.openfoam.com/documentation/

7.3. Numerical GPU libraries

7.3.1. cuBLAS

cuBLAS is BLAS(Basic Linear Algebra Subprograms) library for GPU.

usage

```
$ module load cuda
$ nvcc -gencode arch=compute_60,code=sm_60 -o sample sample.cu -lcublas
```

If you need to call cuBLAS in the usual C program, -I, -L and -l options are required in the compilation.

```
$ module load cuda
$ gcc -o blas blas.c -I${CUDA_HOME}/include -L${CUDA_HOME}/lib64 -lcublas
```

7.3.2. cuSPARSE

cuSPARSE is sparse matrix computation library for nvidia GPU.

usage

```
$ module load cuda
$ nvcc -gencode arch=compute_60,code=sm_60 sample.cu -lcusparse -o sample
```

If you need to call cuSPARSE in the usual C program, -I, -L and -l options are required in the compilation.

```
$ module load cuda
$ g++ sample.c -lcusparse_static -I${CUDA_HOME}/include -L${CUDA_HOME}/lib64  -lculibos -lcudart_static -lpthread -ldl -o sample
```

7.3.3. cuFFT

cuFFT is parallel FFT(Fast Fourier Transformation) library for nvidia GPU.

usage

```
$ module load cuda
$ nvcc -gencode arch=compute_60,code=sm_60 -o sample sample.cu -lcufft
```

If you need to call cufft in the usual C program, -I, -L and -l options are required in the compilation.

```
$ module load cuda
$ gcc -o blas blas.c -I${CUDA_HOME}/include -L${CUDA_HOME}/lib64 -lcufft
```

7.4. Machine learning, big data analysis software

7.4.1. CuDNN

You can load with the following commands:

```
$ module load cudacudnn
```

7.4.2. NCCL

You can load with the following commands:

```
$ module load cudanccl
```

7.4.3. Caffe

You can load and use interactively with the following commands:

```
$ module load intel cuda nccl cudnn caffe
```

For more details, please refer the following site: http://caffe.berkeleyvision.org/ If you want to use MKL from caffe, you should add `#define USE_MKL` in the code which invokes caffe, to ensure libraries are loaded from `$MKLROOT`.

7.4.4. Chainer

You can load and use interactively with the following commands:

```
$ module load intel cuda nccl cudnn openmpi/2.1.2-opa10.9-t3 chainer
```

For more details, please refer the following site: https://docs.chainer.org/en/stable/

7.4.5. TensorFlow

You could run interactive use like in this example.

- python2.7

```
$ module load python-extension
$ cp -rp $PYTHON_EXTENSION_DIR/examples/tensorflow/examples .
$ cd examples/tutorials/mnist
$ python mnist_deep.py
```

- python3.9.2

```
$ module load python/3.9.2 cuda/11.2.146 cudnn/8.1 nccl/2.8.4 tensorflow
```

https://www.tensorflow.org/

7.4.6. DeePMD-kit

DeePMD-kit is a machine learning framework for MD.
The followings are some exmples of the job script.

7.4.6.1. DeePMD-kit + LAMMPS

7.4.6.1.1. DeePMD-kit + LAMMPS 1 node

The following is an example job script of DeePMD-kit + LAMMPS(1 node, 4GPUs).

```
#!/bin/sh
#$ -l h_rt=6:00:00
#$ -l f_node=1
#$ -cwd

. /etc/profile.d/modules.sh
module purge
module load deepmd-kit/2.1.5 intel ffmpeg lammps/23jun2022_u2
module li 2>&1

# enable DeePMD-kit for lammps/23jun2022_u2
export LAMMPS_PLUGIN_PATH=$DEEPMD_KIT_DIR/lib/deepmd_lmp

# https://tutorials.deepmodeling.com/en/latest/Tutorials/DeePMD-kit/learnDoc/Handson-Tutorial%28v2.0.3%29.html

wget https://dp-public.oss-cn-beijing.aliyuncs.com/community/CH4.tar
tar xf CH4.tar

cd CH4/00.data
python3 <<EOF
import dpdata
import numpy as np
data = dpdata.LabeledSystem('OUTCAR', fmt = 'vasp/outcar')
print('# the data contains %d frames' % len(data))
# random choose 40 index for validation_data
index_validation = np.random.choice(200,size=40,replace=False)
# other indexes are training_data
index_training = list(set(range(200))-set(index_validation))
data_training = data.sub_system(index_training)
data_validation = data.sub_system(index_validation)
# all training data put into directory:"training_data"
data_training.to_deepmd_npy('training_data')
# all validation data put into directory:"validation_data"
data_validation.to_deepmd_npy('validation_data')
print('# the training data contains %d frames' % len(data_training))
print('# the validation data contains %d frames' % len(data_validation))
EOF

export PSM2_DEVICES="shm,self,hfi"

cd ../01.train
dp train input.json
dp freeze -o graph.pb
```

```
dp compress -i graph.pb -o graph-compress.pb
dp test -m graph-compress.pb -s ../00.data/validation_data -n 40 -d results

cd ../02.lmp
ln -s ../01.train/graph-compress.pb
lmp -i in.lammps
```

7.4.6.1.2. DeePMD-kit + LAMMPS 2 nodes

The following is an example job script of DeePMD-kit + LAMMPS(2 nodes, 8GPUs).

```
#!/bin/sh
#$ -l h_rt=12:00:00
#$ -l f_node=2
#$ -cwd

. /etc/profile.d/modules.sh
module purge
module load deepmd-kit/2.1.5 intel ffmpeg lammps/23jun2022_u2
module li 2>&1

# enable DeePMD-kit
export LAMMPS_PLUGIN_PATH=$DEEPMD_KIT_DIR/lib/deepmd_lmp

# https://tutorials.deepmodeling.com/en/latest/Tutorials/DeePMD-kit/learnDoc/Handson-Tutorial%28v2.0.3%29.html

wget https://dp-public.oss-cn-beijing.aliyuncs.com/community/CH4.tar
tar xf CH4.tar

cd CH4/00.data
python3 <<EOF
import dpdata
import numpy as np
data = dpdata.LabeledSystem('OUTCAR', fmt = 'vasp/outcar')
print('# the data contains %d frames' % len(data))
# random choose 40 index for validation_data
index_validation = np.random.choice(200,size=40,replace=False)
# other indexes are training_data
index_training = list(set(range(200))-set(index_validation))
data_training = data.sub_system(index_training)
data_validation = data.sub_system(index_validation)
# all training data put into directory:"training_data"
data_training.to_deepmd_npy('training_data')
# all validation data put into directory:"validation_data"
data_validation.to_deepmd_npy('validation_data')
print('# the training data contains %d frames' % len(data_training))
print('# the validation data contains %d frames' % len(data_validation))
EOF

export PSM2_DEVICES="shm,self,hfi"

cd ../01.train
mpirun -x PATH -x LD_LIBRARY_PATH -x PYTHONPATH -x PSM2_CUDA=1 -x NCCL_BUFFSIZE=1048576 -npernode 4 -np 8 dp train i¥
nput.json
dp freeze -o graph.pb
dp compress -i graph.pb -o graph-compress.pb
dp test -m graph-compress.pb -s ../00.data/validation_data -n 40 -d results

cd ../02.lmp
ln -s ../01.train/graph-compress.pb
mpirun -x PATH -x LD_LIBRARY_PATH -x PYTHONPATH  -x LAMMPS_PLUGIN_PATH -x PSM2_CUDA=1 -npernode 4 -np 8 lmp -i in.la¥
mmps
```

7.4.6.2. DeePMD-kit + GROMACS

The following is an example job script of DeePMD-kit + LAMMPS(1 node, 4GPUs).

```
#!/bin/sh
#$ -l h_rt=8:00:00
#$ -l f_node=1
#$ -cwd

. /etc/profile.d/modules.sh
module purge
module load deepmd-kit/2.1.5 gromacs-deepmd/2020.2
module li 2>&1
```

```
export PSM2_DEVICES="shm,self,hfi"

cp -pr $DEEPMD_KIT_DIR/examples/examples/water .
cd water/se_e2_a

dp train input.json
dp freeze -o graph.pb
dp compress -i graph.pb -o graph-compress.pb
dp test -m graph-compress.pb -s ../data/data_3 -n 40 -d results

cd ../gmx
ln -s ../se_e2_a/graph-compress.pb frozen_model.pb
export GMX_DEEPMD_INPUT_JSON=input.json
gmx_mpi grompp -f md.mdp -c water.gro -p water.top -o md.tpr -maxwarn 3
gmx_mpi mdrun -deffnm md
gmx_mpi rdf -f md.trr -s md.tpr -o md_rdf.xvg -ref "name OW" -sel "name OW"
```

For more details, please refer to the following site:
https://docs.deepmodeling.com/projects/deepmd/en/master/index.html

7.4.7. R

Rmpi for parallel processing and rpud for GPU are installed.
You could run interactive use like in this example.

```
$ module load intel cudaopenmpi r
$ mpirun -stdin all -np 2 R --slave --vanilla < test.R
```

7.4.8. clang

clang is C/C++ compiler whose backend is LLVM.
The following is an exmple to use clang with GPU offloading.

- for C

```
$ module load cuda clang
$ clang -fopenmp -fopenmp-targets=nvptx64-nvidia-cuda --cuda-path=$CUDA_HOME -Xopenmp-target -march=sm_60 test.c
```

- for C++

```
$ module load cuda clang
$ clang++ -stdlib=libc++ -fopenmp -fopenmp-targets=nvptx64-nvidia-cuda --cuda-path=$CUDA_HOME -Xopenmp-target -march=sm_60 test.cxx -lc++abi
```

For more details, please refer to the following site:
https://clang.llvm.org/

7.4.9. Apache Hadoop

You could run interactive use like in this example.

```
$ module load jdk hadoop
$ mkdir input
$ cp -p $HADOOP_HOME/etc/hadoop/*.xml input
$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.8.0.jar grep input output 'dfs[a-z.]+'
$  cat output/part-r-00000
1       dfsadmin
```

You could submit a batch job. The following is a sample job script.

```
#!/bin/bash
#$ -cwd
#$ -l f_node=1
#$ -l h_rt=0:10:0
#$ -N hadoop

. /etc/profile.d/modules.sh
module load jdk hadoop
```

```
cd $TMPDIR
mkdir input
cp -p $HADOOP_HOME/etc/hadoop/*.xml input
hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.8.0.jar grep input output 'dfs[a-z.]+'
cp -rp output $HOME
```

7.5. Visualization software

7.5.1. POV-Ray

You can start with the following commands:

```
$ module load pov-ray
$ povray -benchmark
```

For more details, please refer the following site: http://www.povray.org/

7.5.2. ParaView

You can start with the following commands:

```
$ module load cuda openmpi paraview
$ paraview
```

7.5.2.1. Vizualization with multiple GPUs

It is possible to vizualize by using multiple nodes/GPUs with `paraview/5.10.0` and `paraview/5.10.0-egl`.
Please note that `paraview/5.10.0-egl` does not have `paraview` command, it only includes commandline executables.
The following is an exmple use of 8 GPUs with `f_node=2`.

- wrap.sh

```
#!/bin/sh

num_gpus_per_node=4
mod=$((OMPI_COMM_WORLD_RANK%num_gpus_per_node))

if [ $mod -eq 0 ];then
  export VTK_DEFAULT_EGL_DEVICE_INDEX=0
elif [ $mod -eq 1 ];then
  export VTK_DEFAULT_EGL_DEVICE_INDEX=1
elif [ $mod -eq 2 ];then
  export VTK_DEFAULT_EGL_DEVICE_INDEX=2
elif [ $mod -eq 3 ];then
  export VTK_DEFAULT_EGL_DEVICE_INDEX=3
fi

$*
```

- job.sh

```
#!/bin/sh
#$ -cwd
#$ -V
#$ -l h_rt=8:0:0
#$ -l f_node=2

. /etc/profile.d/modules.sh

module purge
module load cuda openmpi/3.1.4-opa10.10-t3 paraview/5.10.0-egl

mpirun -x PSM2_CUDA=1 -x PATH -x LD_LIBRARY_PATH -npernode 4 -np 8 ./wrap.sh pvserver
```

Note that if `openmpi/3.1.4-opa10.10-t3` with `PSM2_CUDA=1` is not used, visualization part is only excecued on GPU and computation part is not executed.

Please do not forget setting the execution permission to `wrap.sh` (`chmod 755 wrap.sh`).
With the above job script, submit the job.

```
qsub -g <group name> job.sh
```

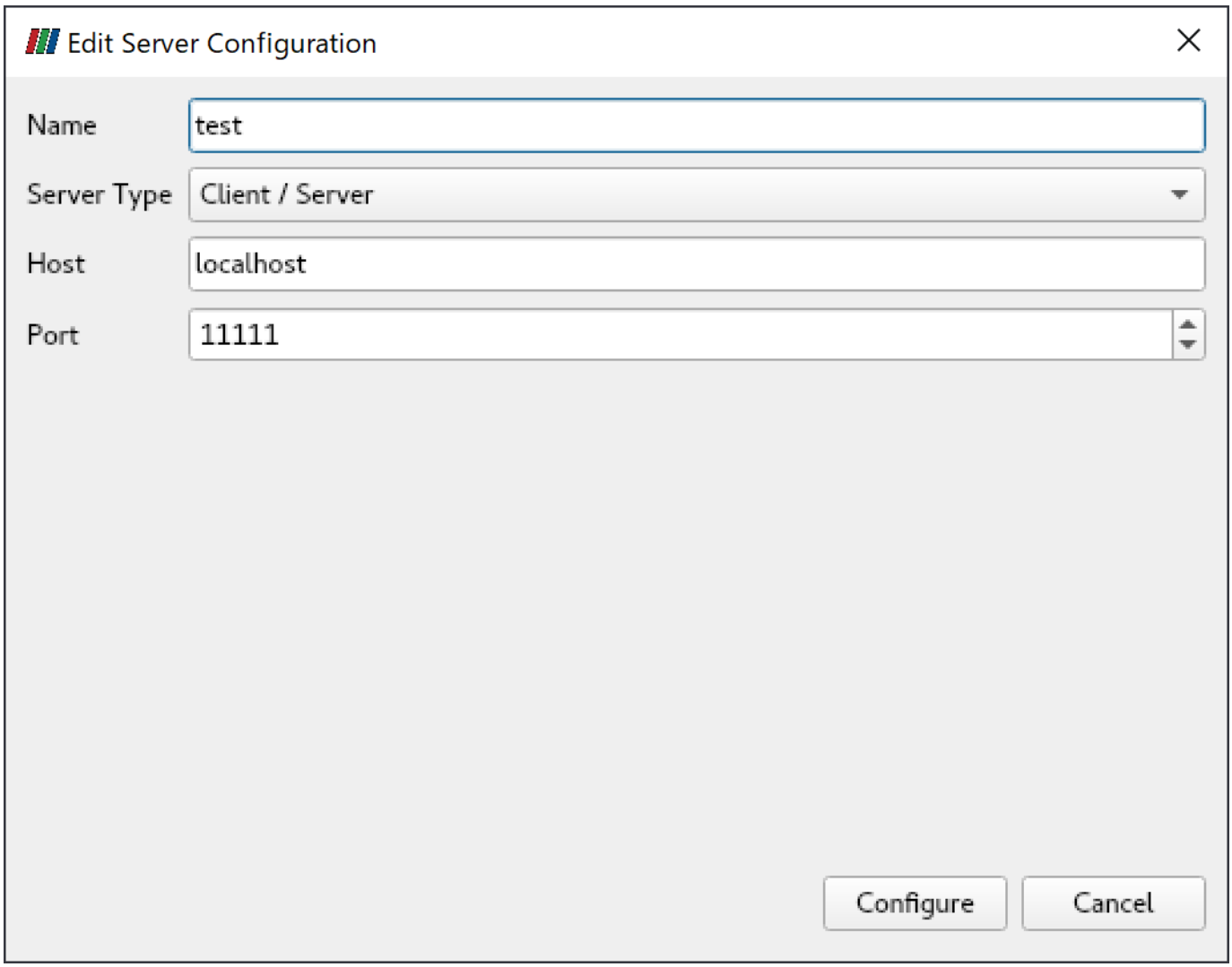
Then, check if the job is running by `qstat` .

```
yyyyyyyy@login0:~> qstat
job-ID     prior   name       user         state submit/start at     queue                          jclass                         slots ja-task-ID
-----------------------------------------------------------------------------------------------------------------------------------------
   xxxxxxx 0.55354 job.sh     yyyyyyyy     r     05/31/2020 09:24:19 all.q@rXiYnZ                                                   56
```

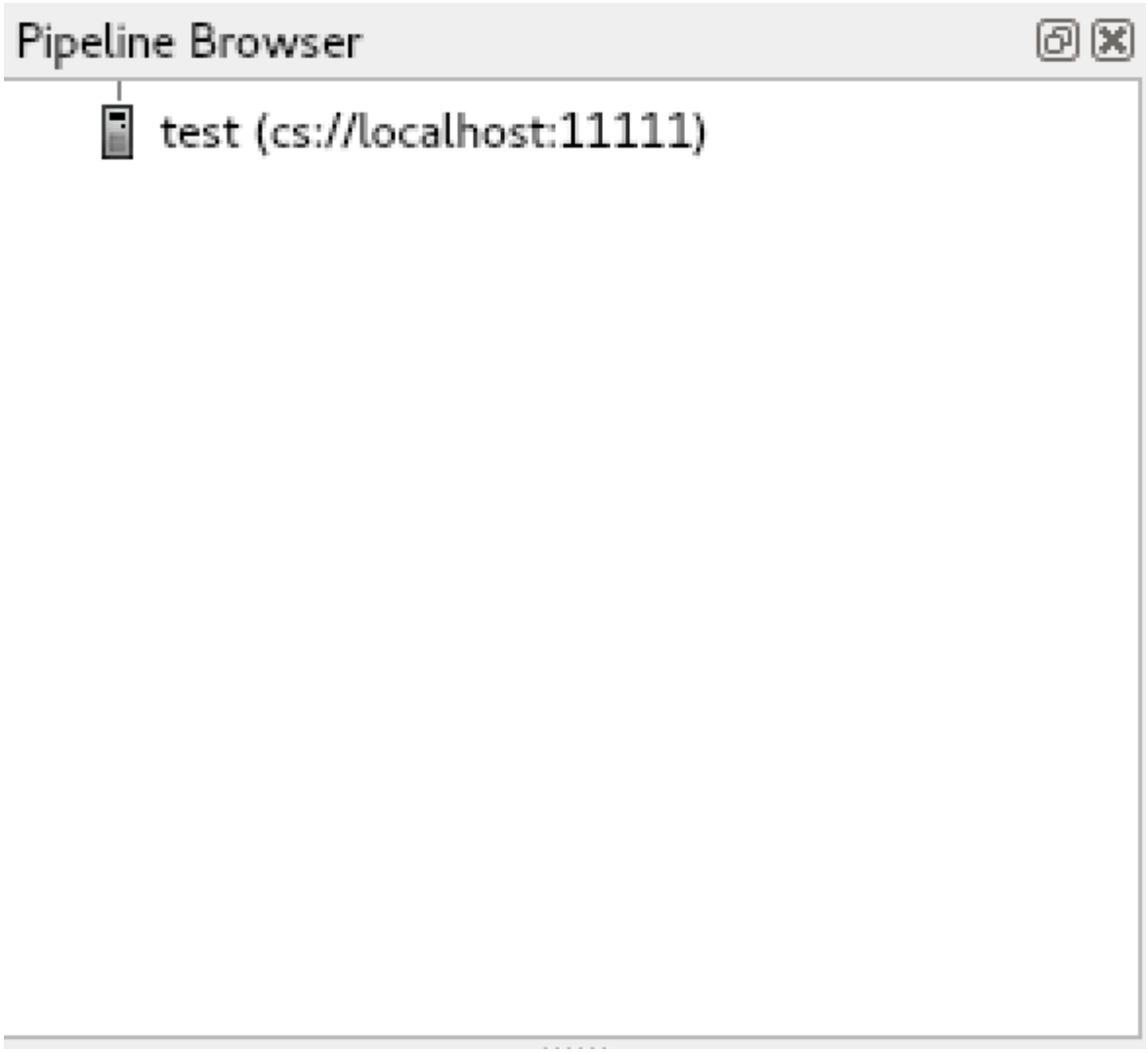
Login to the allocated node by ssh command with X forwarding.

```
yyyyyyyy@login0:~> ssh -CY rXiYnZ
yyyyyyyy@rXiYnZ:~> module load cuda openmpi paraview/5.10.0
paraview
```

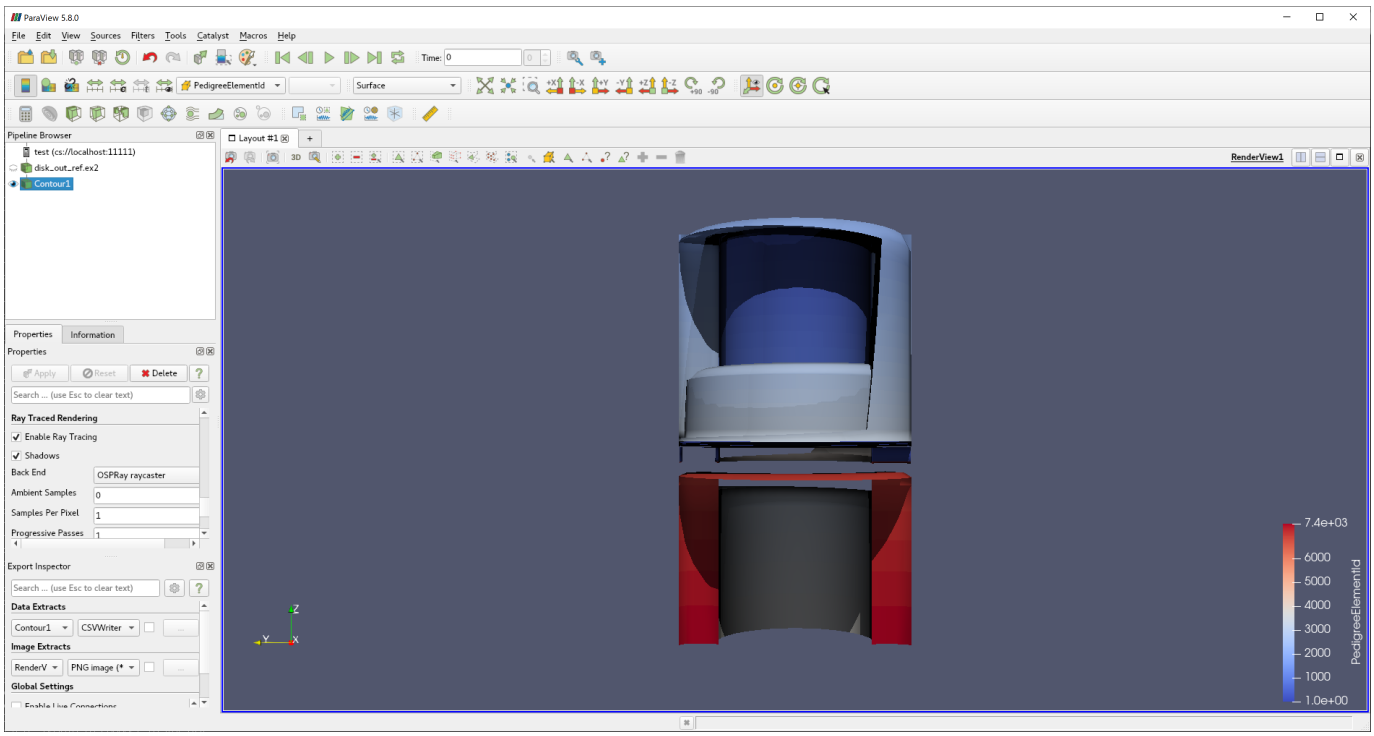
turbovnc is also possible to use.
After starting paraview, click "File"->"Connect", then click "Add Server".
Input "Name" field appropriately("test" as an example), click "Configure".



Click "Connect".
When the connection is established, `test(cs://localhost:11111)` is displayed in "Pipeline Browser".



paraview examples data can be downloaded from here.



For more details, please refer the following site: https://www.paraview.org/

7.5.3. VisIt

You can start with the following commands:

```
$ module load cuda openmpi vtk visit
$ visit
```

For more details, please refer the following site: https://wci.llnl.gov/simulation/computer-codes/visit/

7.6. Other freeware

7.6.1. turbovnc

turobvnc is an open source VNC software.
The following is an example to use turbovnc.
Please try them on the compute node by qrsh.

- allocate a compute node

```
$ qrsh -g <group name> -l <resource type>=<count> -l h_rt=<time>
```

- start vncserver on the node

```
$ module load turbovnc
$ vncserver

You will require a password to access your desktops.

Password:  # <-- set the password
Verify:
Would you like to enter a view-only password (y/n)? n

Desktop 'TurboVNC: rXiYnZ:1 ()' started on display rXiYnZ:1 # <-- remember the VNC display number ":1"

Creating default startup script /home/n/xxxx/.vnc/xstartup.turbovnc
Starting applications specified in /home/n/xxxx/.vnc/xstartup.turbovnc
Log file is /home/n/xxxx/.vnc/rXiYnZ:1.log
```

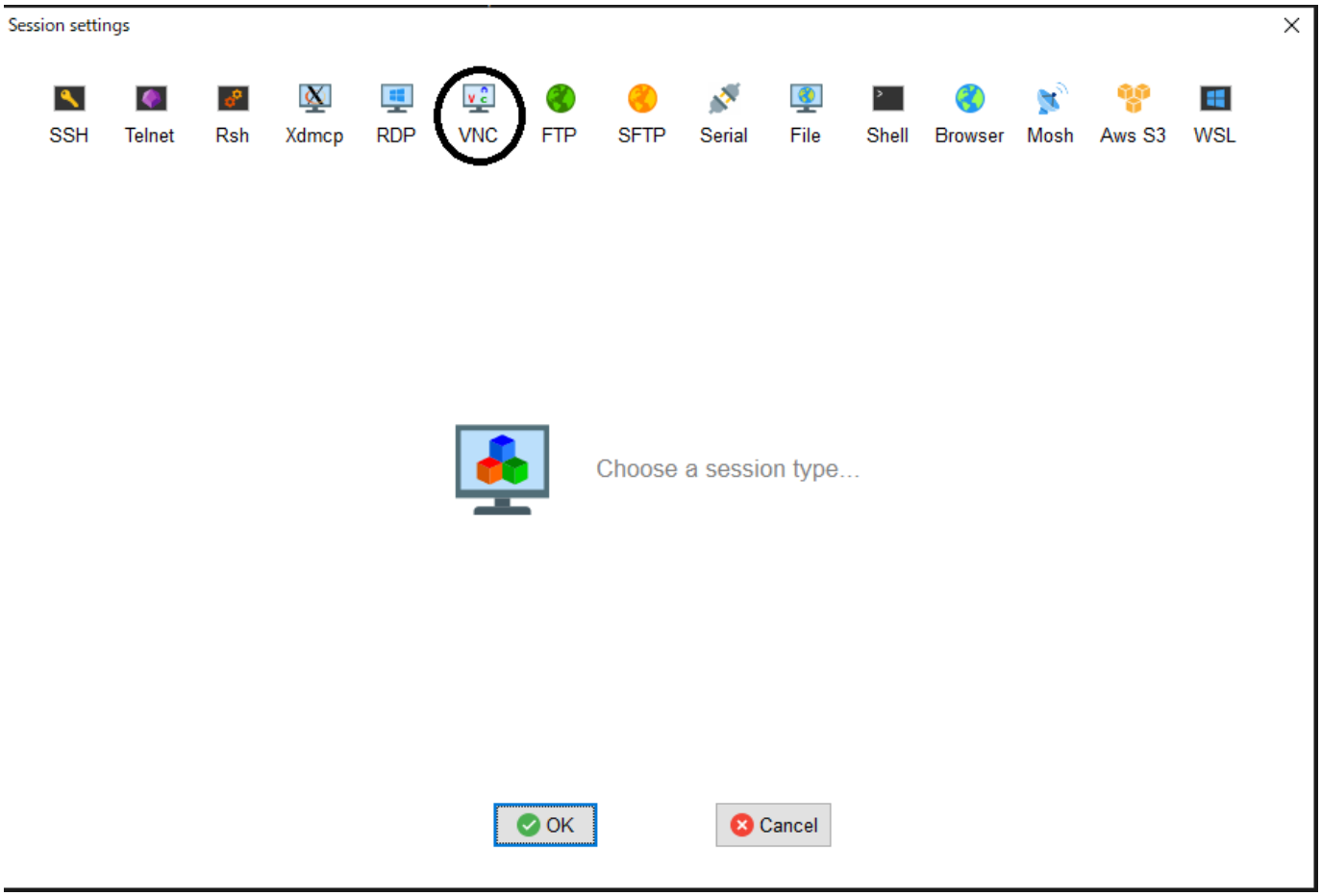
If you want to enlarge the VNC screen size, do `vncserver -geometry <WIDTH>x<HEIGHT>` and set the size.

- Download the installer from https://sourceforge.net/projects/turbovnc/files/ and install turbovnc viewer into your local PC
- From the terminal software you are connecting to TSUBAME, setup port forwarding as local port 5901 to the compute node port 5901.(if the display number is rXiYnZ:n, set the port number 5900+n)
- Start vncviewer from your PC, connect to localhost:5901 and input the password you set.

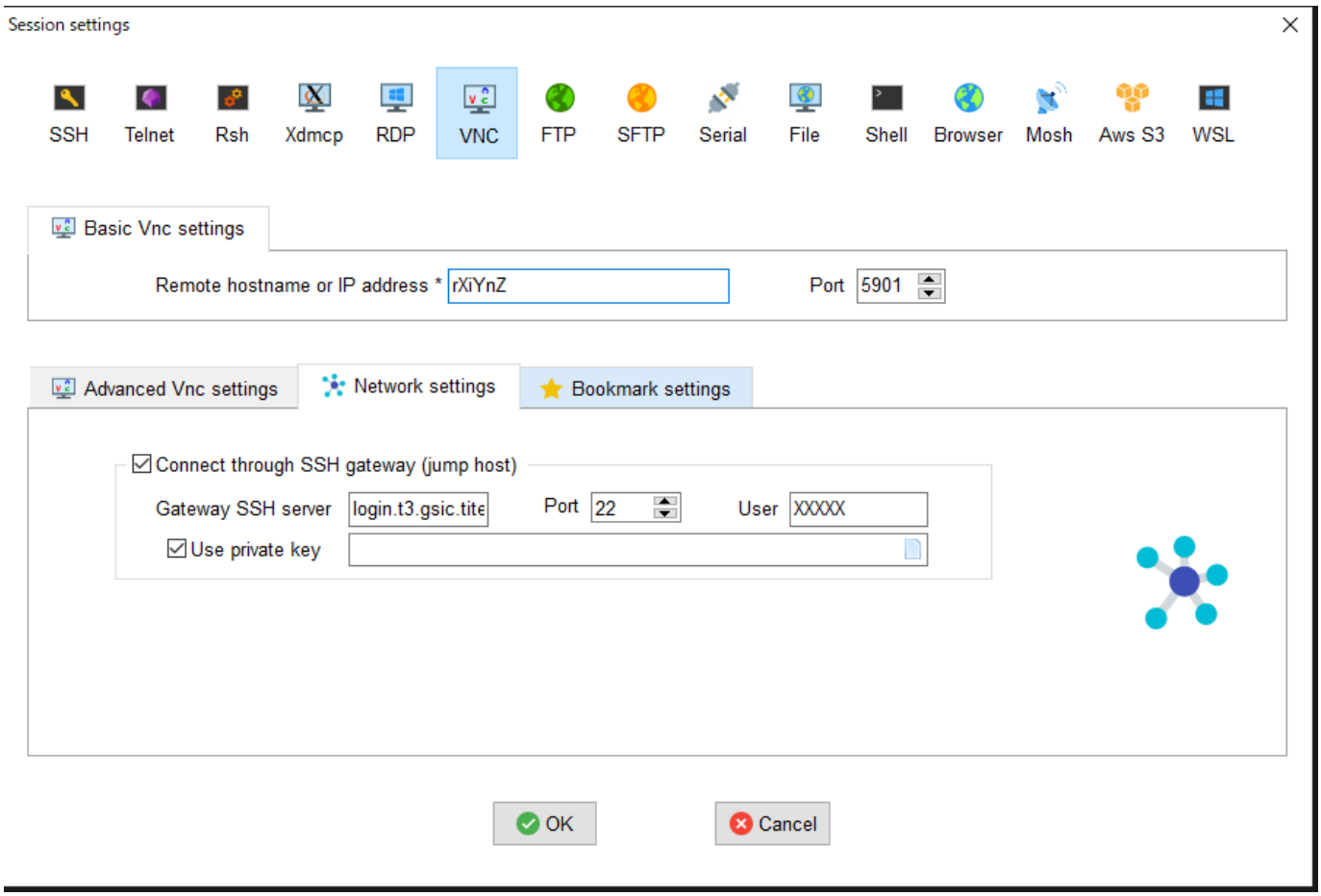
7.6.1.1. use VNC client from MobaXterm

MobaXterm includes VNC client, therefore installing VNC client is not necessary.

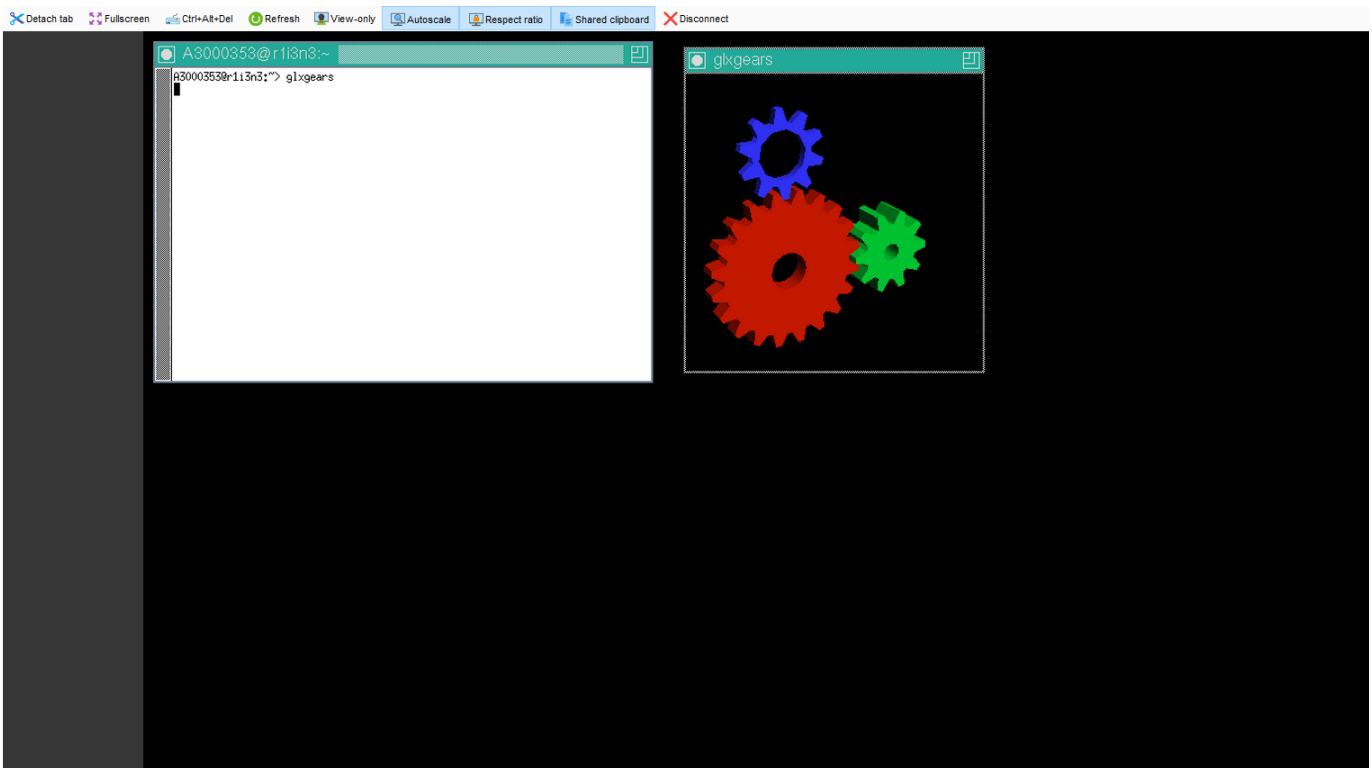
- From MobaXterm, choose「Sessions」->「New session」->「VNC」.



- Type the hostname of the allocated node into「Remote hostname or IP address」on「Basic Vnc settings」, input 5900+n into「Port」, click「Connect through SSH gateway(jump host)」in「Network settings」then type login.t3.gsic.titech.ac.jp into「Gateway SSH server」,「Port」is 22, type your login name in TSUBAME into「User」, check「Use private key」and input the path of your private key.



• Click「OK」then VNC client will be started.



7.6.1.2. turbovnc + VirtualGL

For resource types (s_gpu, q_node, h_node, f_node) that uses one or more GPUs when turbovnc is used, it is possible to visualize using the GPU by VirtualGL.
Some applications fail to draw with X forwarding or normal VNC session(e.g. GpuTest, and UNIGINE), please try VirtualGL with such applications.
For example, the following is an exmpale to use VirtualGL with s_gpu.

```
$ qrsh ... -l s_gpu=1
$ . /etc/profile.d/modules.sh
$ module load turbovnc
$ Xorg -config xorg_vgl.conf :99 & # where :99 is aribitrary display number for VirtualGL
$ vncserver
```

Warning

Please note that the display number for VirtualGL is different from the one of VNC.
If anohter user is using the display number, the following error occurs when executing `Xorg`.

```
user@r7i7n7:~> Xorg -config xorg_vgl.conf :99 &
user@r7i7n7:~> (EE)
Fatal server error:
(EE) Server is already active for display 99
        If this server is no longer running, remove /tmp/.X99-lock
        and start again.
(EE)
(EE)
Please consult the The X.Org Foundation support
         at http://wiki.x.org
 for help.
(EE)
```

In this case, set :99 to :100 to assign a display number that is not used by other users.

- connect VNC client and do the following

```
$ vglrun -d :99 <OpenGL application> # where :99 is the display number for VirtualGL that is set by Xorg above
```

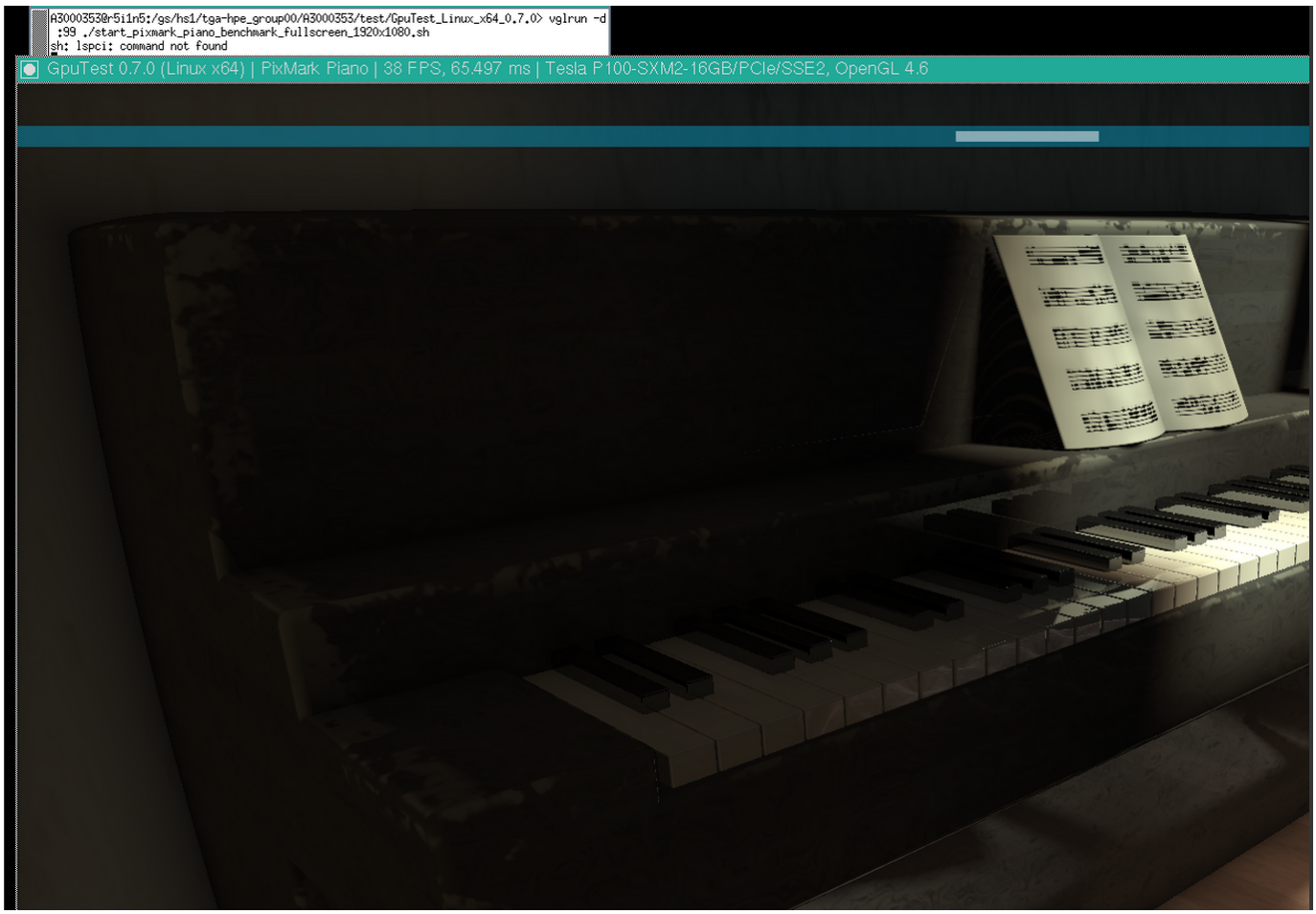
If you allocated multiple GPUs and want to use second or subsequent GPUs, add the screen number to the display number.

```
$ vglrun -d :99.1 <OpenGL application> # where :99 is the display number for VirtualGL that is set by Xorg above, .1 is the screen number
```

In the above example, the third GPU is used if screen number .2 is set, and the forth GPU is used if screen number .3 is set.

- An example use of VirtualGL with GpuTest

```
vglrun -d :99 ./start_pixmark_piano_benchmark_fullscreen_1920x1080.sh
```



7.6.2. gnuplot

In addition to the standard configure option, it is built to correspond to X11, latex, PDFlib-lite, Qt4.
You can start with the following commands:

```
$ module load gnuplot
$ gnuplot
```

7.6.3. Tgif

You can start with the following commands:

```
$ module load tgif
$ tgif
```

(note) Cannot open the Default (Msg) Font '--*courier-medium-r-normal*--14-----*-iso8859-1'.
If the above error occurs and it does not start up, add the following line to ~ / .Xdefaults.

```
Tgif.DefFixedWidthFont:             -*-fixed-medium-r-semicondensed--13-*-*-*-*-*-*-*
Tgif.DefFixedWidthRulerFont:        -*-fixed-medium-r-semicondensed--13-*-*-*-*-*-*-*
Tgif.MenuFont:                    -*-fixed-medium-r-semicondensed--13-*-*-*-*-*-*-*
Tgif.BoldMsgFont:                 -*-fixed-medium-r-semicondensed--13-*-*-*-*-*-*-*
Tgif.MsgFont:                     -*-fixed-medium-r-semicondensed--13-*-*-*-*-*-*-*
```

7.6.4. GIMP

You can start with the following commands:

```
$ module load gimp
$ gimp
```

7.6.5. ImageMagick

In addition to standard configure options, it is built to correspond to X11, HDRI, libwmf, jpeg. You can start with the following commands:

```
$ module load imagemagick
$ convert -size 48x1024 -colorspace RGB 'gradient:#000000-#ffffff' -rotate 90 -gamma 0.5 -gamma 2.0 result.jpg
```

7.6.6. pLaTeX2e

You can start with the following commands:

```
$ module load texlive
$ platex test.tex
$ dvipdfmx test.dvi
```

(note) Please use dvipdfmx to create pdf. Dvipdf does not convert Japanese properly.

7.6.7. Java SDK

You can start with the following commands:

```
$ module load jdk
$ javac Test.java
$ java Test
```

7.6.8. PETSc

Two different installations are provided: supporting real numbers and complex numbers. You can start with the following commands:

```
$ module load intel intel-mpi
$ module load petsc/3.7.6/real           <-- real number
 OR
$ module load petsc/3.7.6/complex       <-- complex number
$ mpiifort test.F -lpetsc
```

7.6.9. FFTW

Different versions are installed: 2.x series and 3.x series. You can start with the following commands:

```
$ module load intel intel-mpi fftw         <-- in case, Intel MPI
OR
$ module load intel cuda openmpi fftw     <-- in case, Open MPI
$ ifort test.f90 -lfftw3
```

7.6.10. DMTCP

An example using DMTCP is as follows.

- Create the checkpoint

```
#!/bin/sh
# Descriptions about other options is omitted
#$ -ckpt user
#$ -c sx
module load dmtcp
export DMTCP_CHECKPOINT_DIR=<store directory>
export DMTCP_COORD_HOST=`hostname`
export DMTCP_CHECKPOINT_INTERVAL=<time>
dmtcp_coordinator --quiet --exit-on-last --daemon 2>&1        # start DMTCP
# Test if the first start or restarted
```

```
dmtcp_launch ./a.out                              # execute program by using DMTCP
$DMTCP_CHECKPOINT_DIR/dmtcp_restart_script.sh     # restart
```

- Restart from the checkpoint

```
#!/bin/sh
# Descriptions about other options is omitted
#$ -ckpt user
#$ -c sx
module load dmtcp
export DMTCP_CHECKPOINT_DIR=<store directory>
export DMTCP_COORD_HOST=`hostname`
export DMTCP_CHECKPOINT_INTERVAL=<time>
$DMTCP_CHECKPOINT_DIR/dmtcp_restart_script.sh      # restart
```

Refer to the site shown below. http://dmtcp.sourceforge.net/

7.6.11. Singularity

please try them with a qrsh session.

- Start a shell session

```
$ module load singularity
$ cp -p $SINGULARITY_DIR/image_samples/centos/centos7.6-opa10.9.sif .
$ singularity shell --nv -B /gs -B /apps -B /scr centos7.6-opa10.9.sif
```

- Execute a command in the container image

```
$ module load singularity
$ cp -p $SINGULARITY_DIR/image_samples/centos/centos7.6-opa10.9.sif .
$ singularity exec --nv -B /gs -B /apps -B /scr centos7.6-opa10.9.sif <command>
```

- Execute a MPI program

```
$ module load singularity cuda openmpi
$ cp -p $SINGULARITY_DIR/image_samples/centos/centos7.6-opa10.9.sif .
$ mpirun -x LD_LIBRARY_PATH -x SINGULARITYENV_LD_LIBRARY_PATH=$LD_LIBRARY_PATH -x SINGULARITYENV_PATH=$PATH -x <environment variables> -npernode <# of
processes/node> -np <# of processes> singularity exec --nv -B /apps -B /gs -B /scr/ centos7.6-opa10.9.sif <MPI binary>
```

From singularity/3.4.2, `fakeroot` option is available to edit the image by user privilege.

Info

The fakeroot feature is introduced into Singularity 3.3.0. However, due to a system-specific problem, the function does not work on Singularity 3.4.1 or prior. To edit the image by using fakeroot option, it is necessary to invoke them on `$T3TMPDIR`.

The following is an expample to install vim into centos image.

```
$ cd $T3TMPDIR
$ module load singularity
$ singularity build -s centos/ docker://centos:latest
INFO:    Starting build...
Getting image source signatures
…
$ singularity shell -f -w centos # -f is the fakeroot option
Singularity> id
uid=0(root) gid=0(root) groups=0(root)
Singularity> unset TMPDIR # a workaround for the error "Cannot create temporary file - mkstemp: No such file or directory"
Singularity> yum install -y vim
Failed to set locale, defaulting to C.UTF-8
CentOS-8 - AppStream                                                            6.6 MB/s | 5.8 MB     00:00
CentOS-8 - Base                                                                 5.0 MB/s | 2.2 MB     00:00
CentOS-8 - Extras
…
Installed:
  gpm-libs-1.20.7-15.el8.x86_64            vim-common-2:8.0.1763-13.el8.x86_64  vim-enhanced-2:8.0.1763-13.el8.x86_64
  vim-filesystem-2:8.0.1763-13.el8.noarch  which-2.21-12.el8.x86_64
```

```
Complete!
Singularity> which vim
/usr/bin/vim
Singularity> exit
$ singularity build -f centos.sif centos/
INFO:    Starting build...
INFO:    Creating SIF file...
INFO:    Build complete: centos.sif
$ singularity shell centos.sif
Singularity> which vim
/usr/bin/vim # <--- vim has been installed
```

- Install CUDA OPA driver libraries into the container image(for installing OPA10.9.0.1.2 CUDA version into centos7.5 image)

note: OPA version of the system might be updated on the system maintenance, so please change the version of OPA if needed.
The version of OPA can be checked as follows.

```
$ rpm -qa |grep opaconfig
opaconfig-10.9.0.1-2.x86_64
```

Download the OPA installer from this link

```
$ module load singularity/3.4.2
$ cp -p IntelOPA-IFS.RHEL75-x86_64.10.9.0.1.2.tgz ~
$ singularity build -s centos7.5/ docker://centos:centos7.5.1804
$ find centos7.5/usr/ -mindepth 1 -maxdepth 1 -perm 555 -print0 |xargs -0 chmod 755 # some files in the image does not have writable permission, so add it
$ singularity shell -f -w centos7.5
Singularity centos:~> tar xf IntelOPA-IFS.RHEL75-x86_64.10.9.0.1.2.tgz
Singularity centos:~> cd IntelOPA-IFS.RHEL75-x86_64.10.9.0.1.2/IntelOPA-OFA_DELTA.RHEL75-x86_64.10.9.0.1.2/RPMS/redhat-ES75/CUDA/
Singularity centos:~> yum install -y numactl-libs hwloc-libs libfabric libibverbs infinipath-psm
Singularity centos:~> rpm --force -ivh libpsm2-*.rpm
Singularity centos:~> exit
$ find centos7.5/usr/bin -perm 000 -print0 |xargs -0 chmod 755 # after yum install, somehow permission 000 file is installed in /usr/bin, so change the
permission
$ singularity build centos7.5.sif centos7.5/
```

Though IFS version is used in the previous example, BASIC version can be also used.
For more details, please visit the following page:

https://sylabs.io/docs/

If you want to use apptainer, please do `module load apptainer`, and replease `singularity` with `apptianer` in the prevous commands.

7.6.12. Alphafold

Alphafold is a protein structure prediction program that uses machine learning.
The following is an example to use Alphafold.

- initial setup(login nodes or compute nodes)

```
module purge
module load cuda/11.0.3 alphafold/2.0.0

cp -pr $ALPHAFOLD_DIR.
cd alphafold
git pull # update to the latest version
# if you want to use a specific version, please add the following.(in this case, the version is v2.0.1)
# git checkout -b v2.0.1 v2.0.1
```

- execution(an example of job script for alphafold/2.0.0)

```
#!/bin/sh
#$ -l h_rt=24:00:00
#$ -l f_node=1
#$ -cwd

. /etc/profile.d/modules.sh
module purge
module load cuda/11.0.3 alphafold/2.0.0
module li

cd alphafold
./run_alphafold.sh -a 0,1,2,3 -d $ALPHAFOLD_DATA_DIR -o dummy_test/ -m model_1 -f ./example/query.fasta -t 2020-05-14
```

- execution(an example of job script for alphafold/2.1.1)

```
#!/bin/sh
#$ -l h_rt=24:00:00
#$ -l f_node=1
#$ -cwd

. /etc/profile.d/modules.sh
module purge
module load cuda/11.0.3 alphafold/2.1.1
module li

cd alphafold
./run_alphafold.sh -a 0,1,2,3 -d $ALPHAFOLD_DATA_DIR -o dummy_test/ -f ./example/query.fasta -t 2020-05-14
```

For `2.2.0` , please replace `module load cuda/11.0.3 alphafold/2.1.1` with `module load cuda/11.0.3 alphafold/2.2.0` in the example of `alphafold/2.1.1` .

Please note that due to the large size of the database files, **please avoid downloading them individually** if at all possible.

For more details of Alphafold, please refer to the following.
https://github.com/deepmind/alphafold

Revision History

Date	Change
2023/10/18	Update the example in "7.1.3. GROMACS"
2023/08/18	Update the example in "7.1.3. GROMACS"
2023/07/10	Update the example in "7.1.4. LAMMPS"
2023/04/14	Update "7.1.5. NAMD" to be suitable for NAMD 3.0b2
2023/04/06	Add an explanation of apptainer in "7.6.11. Singularity"
2023/03/15	Add C++ example of "7.4.8. clang"
2022/12/27	Update the examples of "7.4.6. DeePMD-kit"
2022/12/22	Update "6. ISV application"
2022/12/16	Add "7.4.6.1.2. DeePMD-kit + LAMMPS 2 nodes"
2022/12/13	Add "7.4.6. DeePMD-kit"
2022/12/07	Remove description of load balancer of "2.2. Login"
2022/09/09	Update the example script of "7.1.4. LAMMPS"
2022/08/23	Update the example script of "7.1.7. QUANTUM ESPRESSO"
2022/05/31	Add alphafold/2.2.0 in "7.6.12. Alphafold"
2022/04/12	Update the example script of "7.1.5. NAMD"
2022/04/05	miscellaneous updates for version up
2022/01/26	Removed infromation about non-commercial license from "7.6.12. Alphafold"
2022/01/25	Added supplemental explanation to "5.2. Job submission"
2021/12/13	Add version 2.1.1 into "7.6.12. Alphafold"
2021/11/12	Add "7.1.7. QUANTUM ESPRESSO"
2021/11/04	Update "7.6.12 Alphafold" to be able to use a specific version
2021/09/20	Update "2.5. How to check TSUBAME points"
2021/08/19	Add how to update the repository in "7.6.12. Alphafold"
2021/08/18	Update user restrictions for "5.4.3. Interactive queue"
2021/08/01	Add "7.6.12. Alphafold"
2021/04/06	Various updates for maintenance in Apr 2021 (5.4.3. Interactive queue, etc.)
2021/01/20	Add a note for one container job in "5.2.3.5. Container job"
2020/11/17	Add singularity into the list in "7. Freeware", update example ussages in "7.6.11. Singularity"
2020/11/17	Add MobaXterm usage in "7.6.1. turbovnc" and add an exmple use of VirtualGL
2020/11/06	Add an explanation of Rq and Rr state in "5.2.5. Job status"
2020/10/09	Update the example of fakeroot in "7.6.11. Singularity"
2020/08/05	Add an info about omitting the key pair in "2.2. Login"
2020/06/01	Add an example use of paraview with multiple GPUs in "7.5.2. ParaView"
2020/05/25	Update an example use of cp2k in "7.1.6. CP2K"
2020/05/18	Update an example use of lammps in "7.1.4. LAMMPS"

Date	Change
2020/04/30	Update "5.4.1 X forwarding"
2020/04/22	Add `t3-user-info compute ars` in "5.3. Reserve compute nodes"
2020/04/22	Replace openmpi/2.1.2-opa10.9-t3 with openmpi/3.1.4-opa10.10-t3 in "4.6.5. GPUDirect RDMA"
2020/04/09	Merge https://www.t3.gsic.titech.ac.jp/node/129 into "5.2.4.1. Trial run"
2020/04/07	Add a notice of the size of array job in "5.2.8. Array Job"
2020/04/07	Update memory limitation in "5.1.1.1. Available resource type"
2020/02/28	Add "4.4. PGI compiler"
2020/02/05	Add a notice of shebang in "5.2.2. Creating job script"
2020/01/17	Add how to view own processes when ssh'ed to compute node in "5.5. SSH login to the compute node"
2019/12/16	Add an explanation of Foundation and ESI in "7.2.1. OpenFOAM"
2019/12/06	Update "5.4.3. Interactive queue"
2019/12/03	Add "7.6.1.1. turbovnc + VirtualGL"
2019/11/29	Add "7.4.7. clang"
2019/11/29	Add "7.6.1. turbovnc"
2019/11/12	Add "5.4.3. Interactive queue"
2019/10/21	Update some examples for fakeroot option in "7.6.10. Singularity"
2019/10/04	Add some examples for fakeroot option in "7.6.10. Singularity"
2019/09/27	Update a link in "5.3. Reserve compute nodes"
2019/08/02	Update the configuration of "5. Job scheduler"
2019/07/31	Repalce f_node with s_core in the example of 5.2.3.1. serial job/GPU job
2019/07/30	Move numerical GPU libraries from PGI compiler manual to 7.3
2019/07/17	Delete "5.4. Checkpointing", add "7.5.9. DMTCP"
2019/07/10	Modified "5.3. Interactive job" to include how to terminate job
2019/06/12	Add "5.8.5 Connection to the network applications"
2019/06/10	Update "4.5.3. MPI Environment with CUDA" and "4.5.5. GPUDirect RDMA" for OPA10.9
2019/06/07	Add some explanations of the job script in "5.2.2 Creating job script"
2019/05/31	Add "2.2.1. Restrictions for heavy work in login nodes"
2019/05/30	Changed documenting system Minor changes to expressions
2019/05/20	Modify "5.5 Reservation service"
2019/05/15	Update the link of "7.1.4 LAMMPS" and "7.1.5 NAMD"
2019/03/29	Add "5.8 Container job" Modify "5.6.1 Local scratch area"
2019/01/25	Modify references due to chapter number change
2019/01/17	Modify Chapter name, order, category
2019/01/16	Modify "4.2.2 Creating Job Script" Modify "4.6.1 Home Directory"

Date	Change
2018/12/03	Add "3.5.1.GPU COMPUTE MODE" Modify "4.6.4.Shared scratch area"
2018/10/26	Modify "4.1. Available resource type"
2018/09/19	Modify "5. ISV application" to fit the current environment
2018/09/06	Add "6.5.9 singularity"
2018/08/23	Modify "6.3 Caffe, Chainer and Tensorflow"
2018/07/27	Modify "5.8 Gaussian" and "5.9 GaussView"
2018/05/25	Add "2.5 How to check TSUBAME points" Add note to "4.6.1 Home Directory", "4.6.2 High-speed storage area"
2018/05/09	Remove the list of available modules, add a link to the web page
2018/04/05	Add "2.4 Changing default login shell" Add "3.5 GPU Environment" Add note job priority to "4.2.2 Creating job script"
2018/01/23	Add "4.5 Reservation service"
2018/01/12	Add note to "4.2 Job submission" Add note to "4.3.1 x-forwarding" Add "4.4 Signal Notification/Checkpinting" Add note to "4.5 Storage system"
2017/10/26	Removed "5.12.4 User authentication"
2017/09/25	Add "4.3.1 x-forwarding"
2017/09/14	Add note CIFS volume display to "2.4 Storage service(CIFS)"
2017/09/11	Add login node memory limit to "2.2 Login" Add note CIFS volume display to "2.4 Storage service(CIFS)" Add usage method to "4.4.4 Shared scratch area"
2017/09/06	Second
2017/08/23	First