

TSUBAME4.0 User's Guide

Table of contents

1. Introduction to TSUBAME4.0	4
1.1. System architecture	4
1.2. Compute node configuration	5
1.3. Software configuration	6
1.4. Storage configuration	6
2. Getting Started	7
2.1. Get an account	7
2.2. Login	7
2.3. Password Management	8
2.4. Changing default login shell	8
2.5. How to check TSUBAME points	9
3. Storage system	10
3.1. Home directory	10
3.2. Group disk	10
3.3. 1 High-speed storage area	10
3.4. 2 Large-scale (Big) storage area	10
3.5. 3 Usage status of group disk	10
3.6. CIFS access from inside campus	10
3.7. Local scratch area	11
4. Software Environment	12
4.1. Switch User Environment	12
4.2. Intel Compiler	13
4.3. NVIDIA HPC SDK	15
4.4. AOCC	16
4.5. GPU Environment	16
5. Job Scheduler	18
5.1. Compute nodes	18
5.2. Job submission	19
5.3. Reserving compute nodes	25
5.4. Interactive job	26
5.5. SSH login to compute nodes	27
5.6. Storage use on Compute Nodes	28
6. Commercial applications	29
6.1. ANSYS	30
6.2. ANSYS/Fluent	31

6.3. ANSYS/LS-DYNA	33
6.4. ABAQUS	34
6.5. ABAQUS CAE	35
6.6. Gaussian	35
6.7. GaussView	36
6.8. AMBER	38
6.9. Materials Studio	40
6.10. Discovery Studio	42
6.11. Mathematica	44
6.12. COMSOL	44
6.13. Schrodinger	45
6.14. MATLAB	46
6.15. VASP	47
6.16. Arm Forge	48
7. Freeware	51
7.1. Quantum chemistry/MD related software	53
7.2. CFD related software	55
7.3. Numerical calculation library for GPU	56
7.4. Machine learning and big data analysis related software	57
7.5. Visualization related software	59
7.6. Other freeware	63
Appendix. Comparison of old and new TSUBAME	70
Appendix.1. Architecture	70
Appendix.2. Compute nodes	70
Appendix.3. Software	71
Appendix.3.1 System software	71
Appendix.3.2 Commercial application	72
Appendix.3.3 Freeware	73
Appendix.4. Storage	74
Revision History	75

1. Introduction to TSUBAME4.0

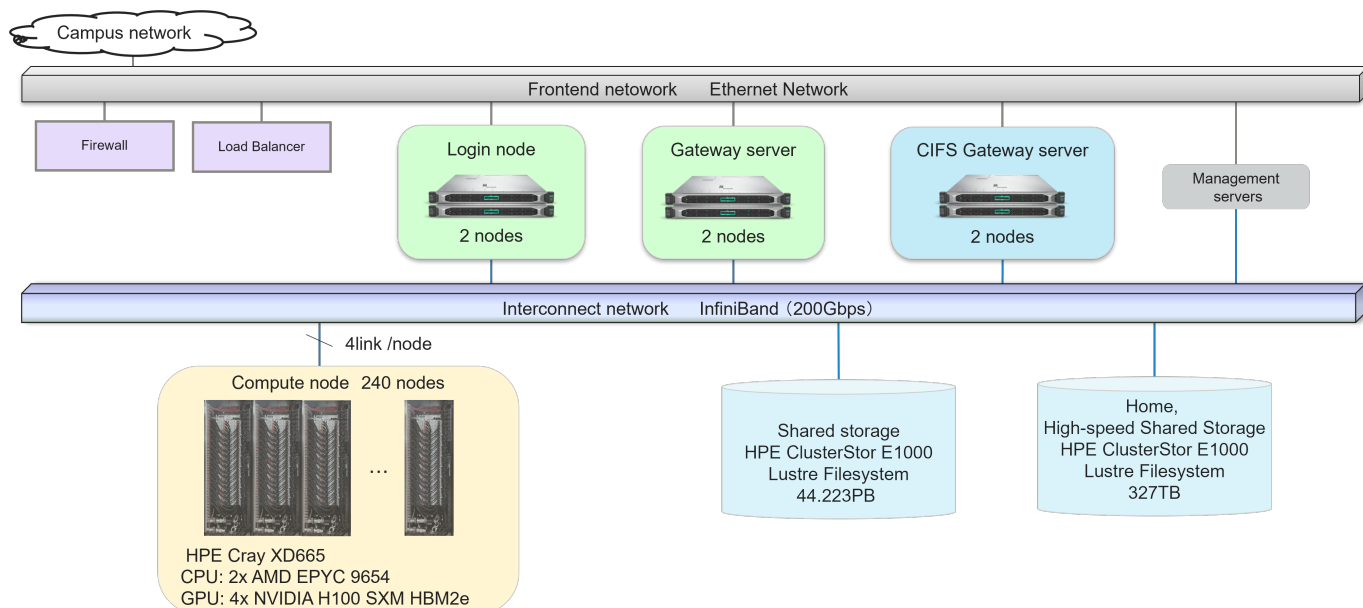
1.1. System architecture

This system is a shared computer that can be used from various research and development departments at Tokyo Institute of Technology.

The system's theoretical peak double precision performance is 66.8PFLOPS, theoretical peak half precision performance is 952PFLOPS, total main memory is 180TiB, total SSD capacity is 327TB and total HDD capacity is 44.2PB.

Each compute node and storage system are connected to the high-speed network by InfiniBand and are connected to the internet at a speed of 100Gbps via SINET6 directly from Suzukakedai campus, Tokyo Institute of Technology.

The system architecture of TSUBAME4.0 is shown below.





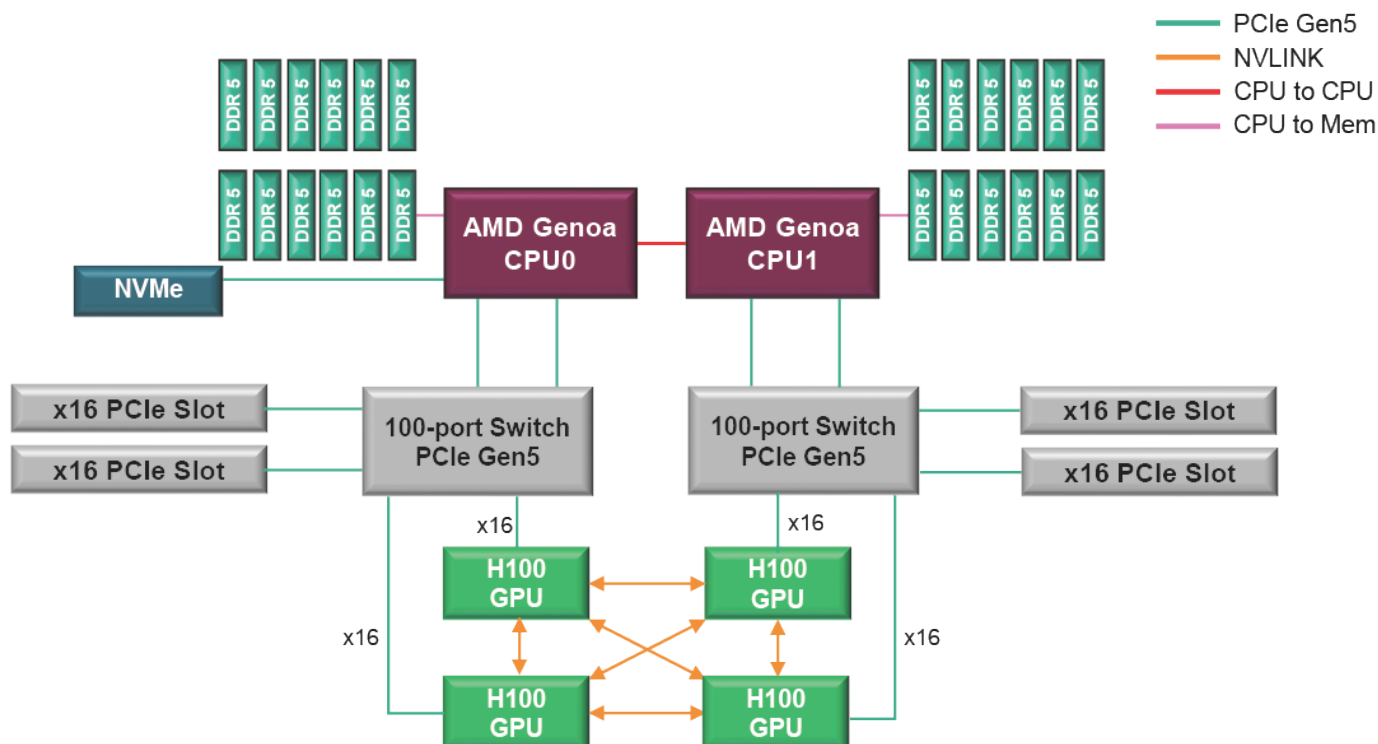
1.2. Compute node configuration

The computing node of this system a large scale cluster system consisting of HPE Cray XD665 240 nodes.

One compute node is equipped with two AMD EPYC 9654 (96 core, 2.4GHz), and the total number of cores is 23,040.

the main memory capacity is 768GiB per compute node, total memory capacity is 180TiB.

Each compute node has 4 port of InfiniBand NDR200 interface and consistutes a fat tree topology by InfiniBand switch.



The specifications of each TSUBAME4.0 node are as follows:

Components	Model/Specification
CPU	AMD EPYC 9654 2.4GHz x 2 Socket
Core/Thread	96 core / 192 thread x 2 Socket
Memory	768GiB (DDR5-4800)
GPU	NVIDIA H100 SXM5 94GB HBM2e x 4
SSD	1.92TB NVMe U.2 SSD
Interconnect	InfiniBand NDR200 200Gbps x 4

1.3. Software configuration

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

- RedHat Enterprise Linux Server 9.3

The OS configuration is dynamically changed according to the service execution configuration.

For application software that is available in this system, please refer to [Commercial applications](#) and [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, the large-scale storage area, Home directory and common application deployments.

In addition, 1.92TB NVMe SSD is installed as a local scratch area in each compute node.

The file systems available in this system are listed below:

Usage	Mount point	Capacity	Filesystem
High-speed storage area	/gs/fs	372TB	Lustre
Home directory (SSD)	/home		
Large-scale storage area	/gs/bs	44.2PB	Lustre
Common application deployment (HDD)	/apps		
Local scratch area (SSD)	/local	Each node 1.92TB	xf

For more details about local scratch area in each resource type, please refer to [Available resource type](#).

2. Getting 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.

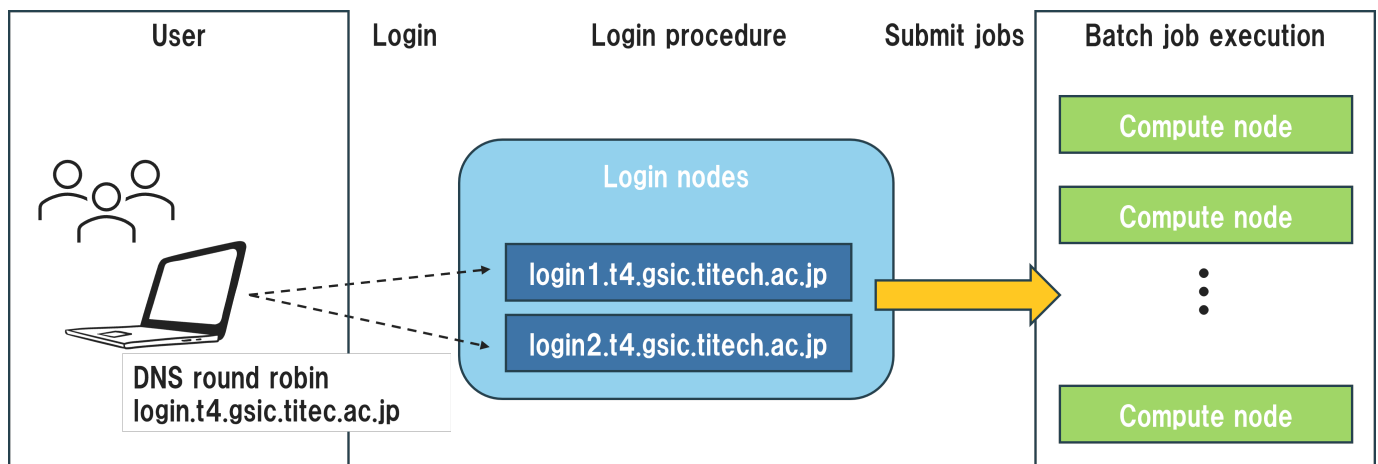
When you finish registration of the SSH public key, you can login to the login node. You will be redirected to one of the login nodes automatically by DNS round-robin.



Warning

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

The use case is shown below.



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

```
login.t4.gsic.titech.ac.jp
```

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

```
login1.t4.gsic.titech.ac.jp
login2.t4.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/t4-key`

```
$ ssh gsic_user@login.t4.gsic.titech.ac.jp -i ~/ssh/t4-key -YC
```



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`.
(fingerprint type depends on your ssh version)

```
The authenticity of host 'login.t4.gsic.titech.ac.jp (131.112.133.51)' can't be established.  
ECDSA key fingerprint is SHA256:CyUFJvJ9ExAE4QEM3BTcKHZ2g7Mw57NnCzbvqIM0ewo.  
ECDSA key fingerprint is MD5:d0:99:50:c7:ae:d4:32:62:a6:b8:5e:c0:bc:11:37:96.  
ED25519 key fingerprint is SHA256:l++Qi9lvnHIRQbQ+638MgWsZdSyZvdAWicwDsYq5KBI.  
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 `qsh` 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 commercial 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 TSUBAME4.0 portal. The rules of available passwords are described on the TSUBAME4.0 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`, `tcsh` and `zsh`.

You can check the available login shell with the `chsh` command without arguments.

```
$ chsh  
Usage: chsh shell (/bin/bash /usr/bin/zsh /usr/bin/tcsh).
```




It may take one hour to complete the change at most.

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

```
$ chsh /usr/bin/tcsh
Please input Web Portal Password(not SSH Passphrase)
Enter LDAP Password: xxxxxx      <-- Enter your password.

Changing shell succeeded!!
```

2.5. How to check TSUBAME points

You can check TSUBAME points with `t4-user-info group point` command as below.

```
$ t4-user-info group point -g TESTGROUP
gid      group_name      deposit      balance
-----
xxxx    TESTGROUP          10           5000
```

In this case, current deposit point is 10, and remaining TSUBAME point of TESTGROUP is 5000.

3. Storage system

In this system, you can use the Lustre file system for the high-speed and the large-scale storage area as well as home directory. And local scratch area on each compute node is also available.

3.1. Home directory

HOME directory is 25GiB available per user.

You can check usage status with `t4-user-info disk home` command.

```
$ t4-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. If you remove some files and free space is available again, you can create new files.

Even if you delete files, sometimes the quota limit status has remains. In this case, it takes a day at most until refreshing the status to recover.

3.2. Group disk

Group disk is used to share files among specific group members. The high-speed or the large-scale storage area are available depend on your purpose.

To purchase group disk, please refer [TSUBAME Portal User's Guide](#).

3.3. 1 High-speed storage area

The high-speed storage area consists of the Lustre file system using SSD and you can be used by purchasing it as a group disk.

3.4. 2 Large-scale (Big) storage area

The large-scale (big) storage area consists of the Lustre file system using HDD and you can be used by purchasing it as a group disk.

3.5. 3 Usage status of group disk

You can check usage status of specific group disk with `t4-user-info disk group -g TESTGROUP` command.

```
$ t4-user-info disk group -g TESTGROUP
gid group_name      /gs/bs      /gs/fs
size(TB) quota(TB) file(M) quota(M) size(TB) quota(TB) file(M) quota(M)
-----
xxxx TESTGROUP      59.78    100    7.50    200    0.00      0    0.00      0
```

In this case, TESTGROUP purchased /gs/bs and used 60TB out of 100TB quota limit, its inode used 7.5M out of 200M.

3.6. CIFS access from inside campus

In TSUBAME4.0, you can access group disk from the Windows / Mac terminal in the university using the CIFS protocol.

```
\\gshs.t4.gsic.titech.ac.jp
```

You need TSUBAME4.0 account. The password is the one you set in the portal.

Note that "@TSUBAME" is needed subsequent your account when you use Windows.

User	(TSUBAME4.0_ACCOUNT)@TSUBAME
Password	(TSUBAME4.0_PASSWORD)

/gs/bs, /gs/hs are displayed as t4_bs and t4_fs respectively. Access to your group disk beneath them.



t4_bs



t4_fs



Home directory is not accessible with CIFS.

3.7. Local scratch area

Local scratch area is located in SSD on each compute node and available for workspace or temporarily area to read/write data rapidly.

The files on local scratch area will be removed after a job is completed. If you need them, you should copy or move them home directory or group disk at the end of job script.

The capacity of local scratch area in each resource type is described [here](#).

4. Software Environment

4.1. Switch User Environment

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

For Environment Modules system, please refer to the man command or [Environment Modules](#).

4.1.1. List available modules

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

```
$ module avail
```

For each version available, please refer to following pages:

[System software](#)

[Supported applications](#)

4.1.2. Module information

If you want to know information about a module, `module whatis MODULE` command.

```
$ module whatis intel/2024.0.2
----- /apps/t4/rhel9/modules/modulefiles/compiler -----
intel/2024.0.2: Intel oneAPI compiler 2024.0 and MKL
```

4.1.3. Load modules

Modules are loaded with `module load MODULE` command.

```
$ module load intel/2024.0.2
```

You need to use same modules in a job script as compiled/build.

4.1.4. List loaded modules

`module list` displays currently loaded modules.

```
$ module list
Currently Loaded Modulefiles:
 1) intel/2024.0.2  2) cuda/12.3.2
```

4.1.5. Unload modules

If you want to unload modules in use, type `module unload MODULE`.

```
$ module list
Currently Loaded Modulefiles:
 1) intel/2024.0.2  2) cuda/12.3.2
$ module unload cuda
$ module list
Currently Loaded Modulefiles:
 1) intel/2024.0.2
```

4.1.6. Refresh module environment

`module purge` can refresh module environment. All modules currently loaded will be unloaded.

```
$ module list
Currently Loaded Modulefiles:
```

```
1) intel/2024.0.2 2) cuda/12.3.2
$ module purge
$ module list
No Modulefiles Currently Loaded.
```

4.2. Intel Compiler

In this system, you can use Intel compiler (oneAPI), AMD compiler (AOCC), NVIDIA compiler (NVIDIA HPC SDK) and GNU compiler.

Intel compiler's commands are listed below:

Command	Language	Syntax	Former command
ifx	Fortran 77/90/95	<code>\$ ifx [OPTION] source_file</code>	ifort
icx	C	<code>\$ icx [OPTION] source_file</code>	icc
icpx	C++	<code>\$ icpx [OPTION] source_file</code>	icpc

When you use Intel compiler, load intel using module command. `--help` option shows various compiler options.



`icc` and `icpc` commands are no longer supported in Intel oneAPI 2024. You should use `icx` and `icpx` instead. `ifort` is also suspected to decommit in near future, we recommend you to use `ifx` instead.



Since Intel oneAPI 2024, the default standard for C++ has changed from C++14 to C++17. Therefore, syntax errors may occur. For more information, see [here](#).

4.2.1. Compiler options

The compiler options are listed below.

Option	Description
<code>-O</code>	Same as <code>O2</code>
<code>-O0</code>	Disables all optimizations. Using for debugging,etc.
<code>-O1</code>	Affects code size and locality. Disables specific optimizations.
<code>-O2</code>	Default optimizations. Same as <code>-O</code> . Enables optimizations for speed, including global code scheduling, software pipelining, predication.
<code>-O3</code>	Aggressive optimizations for maximum speed (, but does not guarantee higher performance). Optimization including data prefetching, scalar replacement, loop transformations.
<code>-xCORE-AVX512</code>	Generates Intel Advanced Vector Extensions 512 (Intel AVX512), Intel AVX2, AVX, SSE4.2, SSE4.1, SSSE3, SSE3, SSE2, SSE instructions for Intel processors. Optimizes for Intel processors with Intel AVX512 instruction set support.
<code>-xCORE-AVX2</code>	Generates Intel Advanced Vector Extensions 2 (Intel AVX2), Intel AVX, SSE4.2, SSE4.1, SSSE3, SSE3, SSE2, and SSE instructions for Intel processors. Optimized for Intel processors supporting the Intel AVX2 instruction set.
<code>-xSSE4.2</code>	Generates Intel SSE4 high-efficiency and high-speed string processing instructions, Intel SSE4 vectorized compiler and media accelerator instructions, and Intel SSSE3, SSE3, SSE2, and SSE instructions for Intel processors. Optimizes for Intel processors supporting the Intel SSE4.2 instruction set.
<code>-xSSSE3</code>	Generates Intel SSSE3, SSE3, SSE2, and SSE instructions for Intel processors, optimized for Intel processors supporting the Intel SSSE3 instruction set.
<code>-qopt-report=n</code>	Generates an optimization report. By default, the report is output to a file with an <code>.optrpt</code> extension, where n is the level of detail from 0 (no report) to 5 (most detailed). The default is 2.
<code>-fp-model precise</code>	Controls the semantics of floating-point operations. Disables optimizations that affect the precision of floating-point data and rounds intermediate results to the precision defined by the source.
<code>-g</code>	The <code>-g</code> option instructs the compiler to generate symbolic debugging information in the object file that increases the size of the object file.
<code>-traceback</code>	This option instructs the compiler to generate supplemental information in the object file so that when a fatal error occurs at run-time, the source file traceback information can be displayed. When a fatal error occurs, correlation information for the source file, routine name, and line number is displayed along with the hex address of the call stack (program counter trace). The map file and the hex address of the stack displayed when an error occurs can be used to determine the cause of the error. This option increases the size of the executable program.

4.2.2. Recommended optimization options

Recommended compile optimization options are listed below. The AMD EPYC 9654 in this system supports the Intel AVX512 instruction set, so the `-xCORE-AVX512` option can be specified. When `-xCORE-AVX512` is specified, the compiler will analyze the source code and generate the optimal AVX512, AVX2, AVX, and SSE instructions. The recommended optimization option is an aggressive and safe option. Optimization may change the order of calculations and may introduce errors in the results.



AVX512, adopted in Intel Xeon (Skylake and later), is now supported by AMD in the 4th generation EPYC of the Zen4 architecture, which is included in this system.

Option	Description
<code>-O3</code>	O2 optimization to enable more powerful loop transformations such as fusion, unroll and jam blocks, and IF statement folding.
<code>-xCORE-AVX512</code>	Generates Intel Advanced Vector Extensions 512 (Intel AVX512), Intel AVX2, AVX, SSE4.2, SSE4.1, SSSE3, SSE3, SSE2, SSE instructions for Intel processors. Optimizes for Intel processors with Intel AVX512 instruction set support.

If the above options worsen the performance of your program, reduce the level of optimization to -O2 or change the vectorization options. Also try the floating point option if the results do not match.

4.2.3. Intel 64 architecture memory model

Create an executable binary using one of the following memory models

Memory model	Description
small (<code>-mmodel=small</code>)	Code and data are limited to the first 2GB of address space so that all access to code and data is instruction pointer (IP) relative addressing. If -mmodel option is not specified, this is the default.
medium (<code>-mmodel=medium</code>)	Code is limited to the first 2GB of address space, but data is not. Code can be accessed via IP relative addressing, but data must be accessed via absolute addressing.
large (<code>-mmodel=large</code>)	Neither code nor data is restricted. Both code and data access uses absolute addressing.

IP relative addressing requires only 32 bits, while absolute addressing requires 64 bits. This affects code size and performance. (IP relative addressing is slightly faster to access.)

When the total amount of common blocks, global data, and static data in a program exceeds 2 GB, the following error message is output at link time

```
<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>
```

In this case, compile/link with `-mmodel=medium` and `-shared-intel`.

If you specify the medium or large memory model, you must also specify the -shared-intel compiler option to ensure that the appropriate dynamic version of Intel's runtime libraries is used.

4.3. NVIDIA HPC SDK

Each command of NVIDIA HPC SDK (formerly PGI compiler) is as follows.

Command	Language	Syntax	Former command
<code>nvfortran</code>	Fortran 77/90/95/2003/2008/2018	<code>\$ nvfortran [OPTION] source_file</code>	<code>pgfortran</code>
<code>nvc</code>	C	<code>\$ nvcc [OPTION] source_file</code>	<code>pgcc</code>
<code>nvc++</code>	C++	<code>\$ nvc++ [OPTION] source_file</code>	<code>pgc++</code>

For more details about each command, see `$ man nvcc`.

To use it, load nvhpc with the module command. If you specify the `--help` option, a list of compiler options will be displayed.

4.4. AOCC

Each command of AMD Optimizing C/C++ and Fortran Compilers (AOCC) is as follows.

Command	Language	Syntax
flang (clang)	Fortran 95/2003/2008	<code>\$ flang [OPTION] source_file</code>
clang	C	<code>\$ clang [OPTION] source_file</code>
clang-cpp (clang)	C++	<code>\$ clang-cpp [OPTION] source_file</code>

To use it, load aocc with the module command. If you specify the `--help` option, a list of compiler options will be displayed.

4.5. GPU Environment

This system provides you GPU (NVIDIA H100 SXM5) environment in conjunction with CPU.

4.5.1. Execution and debug for interactive job

The login nodes (login, login1, login2) do not have GPUs and can only compile and link. Also, execution of high-load programs on login nodes is **limited**.

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

4.5.2. Supported applications for GPU

GPU supported applications are listed below (as of 2024.4.1).

- ABAQUS 2024 --- Refer to ABAQUS Usage guide.
- ANSYS 2024R1 --- Refer to ANSYS Usage guide.
- AMBER 22up05 --- Refer to AMBER Usage guide.
- Mathematica 14 --- Refer to Mathematica Usage guide.
- MATLAB 2024 --- Refer to MATLAB Usage guide.
- Arm Forge --- Refer to Allinea Forge Usage guide.
- NVIDIA HPC SDK --- Refer to NVIDIA HPC Usage guide.

For other applications, we will provide it sequentially.

4.5.3. MPI Environment with CUDA

MPI Environment with CUDA is available.

OpenMPI + gcc environment

```
# load OpenMPI, GCC
$ module load openmpi/5.0.2-gcc
Loading openmpi/5.0.2-gcc
Loading requirement: cuda/12.3.2
```



The required related modules are loaded automatically.

OpenMPI + NVIDIA HPC SDK environment

```
# load OpenMPI, NVIDIA HPC SDK
$ module load openmpi/5.0.2-nvhpc
Loading openmpi/5.0.2-nvhpc
```

OpenMPI + Intel environment

```
# load OpenMPI, Intel
$ module load openmpi/5.0.2-intel
Loading openmpi/5.0.2-intel
Loading requirement: intel/2024.0.2 cuda/12.3.2
```



The required related modules are loaded automatically.

5. Job Scheduler

The system's job scheduling uses Altair Grid Engine, which efficiently schedules single and parallel jobs according to priority and required resources.

5.1. Compute nodes

5.1.1. Baremetal environments

5.1.1.1. Resource types

This system uses resource types in which compute nodes are logically divided to reserve system resources.

When submitting a job, specify how many resource types will be used (e.g., `-l node_f=2`). The available resource types are listed below.

Resource type	Physical CPU cores	Memory (GB)	GPUs	Local scratch area (GB)
node_f	192	768	4	1920
node_h	96	384	2	960
node_q	48	192	1	480
node_o	24	96	1/2	240
gpu_1	8	96	1	240
gpu_h	4	48	1/2	120
cpu_160	160	368	0	96
cpu_80	80	184	0	48
cpu_40	40	92	0	24
cpu_16	16	36.8	0	9.6
cpu_8	8	18.4	0	4.8
cpu_4	4	9.2	0	2.4

- "Physical CPU cores", "Memory (GB)", "GPUs" are the available resources per resource type.
- Same resource type can be used with [Resource type]=[Num]. Different resource types combinations are not available.
- Maximum run time is 24 hours.
- TSUBAME4 has various limits such as "the number of jobs that can be executed at the same time" and "the total number of slots that can be executed". (slots=Physical cores in each resource type x nodes number (same as slots of qstat))

Current limits are shown below:

<https://www.t4.gsic.titech.ac.jp/en/resource-limit>

Note that it may change depends on usage any time.

5.1.2. Container

This system provides an application container using Apptainer (formerly Singularity) to enable applications that are difficult to run on the host OS due to software dependencies.

For Apptainer, please check [Freeware](#).

5.2. Job submission

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

5.2.1. Job submission flow

To submit a job, create and submit a job script. The submission command is `qsub`.

- Create a job script
- Submit a job using `qsub`
- Status check using `qstat`
- Cancel a job using `qdel`
- Check job result

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

5.2.2. Creating job script

Here is a job script format:

```
#!/bin/sh
#$ -cwd
#$ -l [Resource type] =[Number]
#$ -l h_rt=[Maximum run time]
#$ -p [Priority]

[Load relevant modules required for the job]

[Your program]
```

Warning

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

- [Load relevant modules required for the job] Load required related modules that you need. For example, loading Intel compiler is as below:

```
module load intel
```

- [Your program]
Execute your program Running `a.out` binary is like below:

```
./a.out
```

Resource type can be set at command line, or in comment block (`#$`) at the top of job script file.

Make sure to set resource type and maximum run time, which are requirement to submit a job.

The main options for the qsub command are listed below.

Option	Description
-l [Resource type Name] =Number	Specify the resource type.
-l h_rt=[Maximum run time] (Required)	Specify the maximum run time (hours, minutes and seconds). [[HH:]MM:]SS, HH:MM:S, 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 filename of the standard error output file
-j y	Integrate standard error output into a standard 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.
-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.

Note that -V option for passing environment variables in the job submission environment is not available in this system.

5.2.3. Job script examples

5.2.3.1. Single 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 cpu_4=1` with `-l gpu_1=1` and load required modules such as CUDA environment.

```
#!/bin/sh
# run in current working directory
#$ -cwd

#$ -l cpu_4=1
# maximum run time
#$ -l h_rt=1:00:00
#$ -N serial

# load cuda module
module load cuda
# load Intel Compiler
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
# node_f 1 node
#$ -l node_f=1
#$ -l h_rt=1:00:00
#$ -N openmp

module load cuda
module load intel
# 192 threads per node
export OMP_NUM_THREADS=192
./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
# node_f x 4
#$ -l node_f=4
#$ -l h_rt=1:00:00
#$ -N flatmpi

module load cuda
module load intel
# Intel MPI
module load intel-mpi
# 8 process/node, all MPI process is 32
mpirun -np 32 ./a.out
```

OpenMPI

```
#!/bin/sh
#$ -cwd
# node_f x 4
#$ -l node_f=4
#$ -l h_rt=1:00:00
#$ -N flatmpi

# Load OpenMPI: Intel compiler, CUDA are loaded automatically
module load openmpi/5.0.2-intel
# 8 process/node, all MPI process is 32
mpirun -np 32 -x LD_LIBRARY_PATH ./a.out
```

5.2.3.4. Hybrid parallel (Hybrid, MPI+OpenMP)

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
# node_f x 4
#$ -l node_f=4
#$ -l h_rt=1:00:00
#$ -N hybrid

module load cuda
module load intel
module load intel-mpi
# 192 threads per node
export OMP_NUM_THREADS=192
# 1 MPI process per node, all MPI process is 4
mpirun -np 4 ./a.out
```

OpenMPI

```
#!/bin/sh
#$ -cwd
# node_f x 4
#$ -l node_f=4
#$ -l h_rt=1:00:00
#$ -N hybrid

# Open MPI: Intel compiler, CUDA are loaded automatically
module load openmpi/5.0.2-intel
# 192 threads per node
export OMP_NUM_THREADS=192
# 1 MPI process per node, all MPI process is 4
mpirun -npnnode 1 -n 4 -x LD_LIBRARY_PATH ./a.out
```

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.
-q prior	Subscription job. Wait one hour at most until execution.

5.2.4.1. Trial run



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).



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.

The trial run function is provided that users who are not sure whether they can use TSUBAME for their own research can check the operation before purchasing points. Please do not use the trial run to perform calculations that deviate significantly from these purposes or that directly lead to research results.

Please consider using [interactive queue](#) if you wish to perform small calculations for educational purposes in class.

For trial runs, the following restrictions apply to the amount of resources

Maximum number of the resource type specified	2
Maximum run time	10 minutes
number of concurrent runs	1
Resource type	no limitation

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.

5.2.4.2. Subscription job

Submittin a job for compute node subscription requires `-q prior` option. Other options are same as the other jobs.

```
$ qsub -q prior -g [TSUBAME group] SCRIPTFILE
```

For more details about compute node subscription, check [here](#).



Even if a job for the subscription group, note that if `-q prior` is not specified, the job will be processed as a pay-as-you-go job.

5.2.5. Job status

The qstat is a job status display command.

```
$ qstat [option]
```

The options of qstat is listed below.

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

Here is an example of qstat:

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

```
307 0.55500 sample.sh testuser r 02/12/2023 17:48:10 all.q@r8n1A.default32
(snip)
```

Item	Description
Job-ID	Job-ID number
prior	Priority of job
name	Name of the job
user	ID or user who submitted job
state	State of the job r runnig 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 waiting for run Rr rescheduled and 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 (slot=physical cores in each resource type x node number)
ja-task-ID	Array job task-id

5.2.6. Deleting job

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

```
$ qdel [Job-ID]
```

An example of deleting job is shown as below:

```
$ qstat
job-IDprior  nameuser  statesubmit/start at  queuejclass  slotsja-task-ID
-----
307 0.55500 sample.sh testuser r 02/12/2023 17:48:10 all.q@r8n1A.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 result

The standard output of AGE jobs is stored in the file "script file name.o job ID" in the directory where the job was executed. Also, the standard error output is "script file name.eJob ID".

5.2.8. Array job

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

Each job executed in an array job is called a task and is managed by a task ID. A job ID that does not specify a task ID has the entire task ID as its range.



Each task in an array job is scheduled as a separate job, so the schedule wait time is proportional to the number of tasks. If each task is short or the number of tasks is large, it is strongly recommended to reduce the number of tasks by grouping several tasks together. Example: 10000 tasks are combined into 100 tasks that each process 100 tasks.

The task number can be specified as an option of the qsub command or defined in a job script. Submission options are specified as `-t (start number)-(end number):(step size)`. If the step size is 1, it can be omitted. An example is shown below.

```
# describe below in a job script
#$ -t 2-10:2
```

The above example (2-10:2) specifies a start number of 2, an end number of 10, a step size of 2 (one skipped index), and consists of five tasks with task numbers 2, 4, 6, 8, and 10.

The task number for each task is set in an environment variable named \$SGE_TASK_ID, so this environment variable can be used in the job script to enable parameter study. The result file will be output with the job name followed by the task ID.

Also, if you want to remove a specific task ID before/while it is running, use qdel's -t option as follows.

```
$ qdel [Job-ID] -t [Task-ID]
```

5.3. Reserving compute nodes

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

Steps for reservation is as follows.

- 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 the reservation time is reached, the job can be executed in the reservation group account. An example of job submission specifying an AR ID, which is a reservation ID, is shown below.

- Submitting a job to a reserved node with qsub

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

- Submitting an interactive job to a reserved node with qsh

```
$ qsh -g [TSUBAME group] -l [Resource type]=[number] -l h_rt=[maximum run time] -ar [AR ID]
```

The resource types available for reserved execution are node_f, node_h, node_q, and node_o. Other resource types are not available for reservation.

The qstat command is used to check the status of a job after it has been submitted, and the qdel command is used to delete a job.

Also, the format of the script is the same as that of the normal execution.

Use `t4-user-info compute ar` to check the reservation status and AR ID from the command line.

xxxxx@login1:~> t4-user-info compute ar													
ar_id	uid	user_name	gid	group_name	state	start_date		end_date		time_hour	node_count	point	return_point
1320	2005	A2901247	2015	tga-red000	r	2023-01-29	12:00:00	2023-01-29	13:00:00	1	1	18000	0
1321	2005	A2901247	2015	tga-red000	r	2023-01-29	13:00:00	2023-01-29	14:00:00	1	1	18000	0
1322	2005	A2901247	2015	tga-red000	w	2023-01-29	14:00:00	2023-02-02	14:00:00	96	1	1728000	1728000
1323	2005	A2901247	2015	tga-red000	r	2023-01-29	14:00:00	2023-02-02	14:00:00	96	1	1728000	1728000
1324	2005	A2901247	2015	tga-red000	r	2023-01-29	15:00:00	2023-01-29	16:00:00	1	17	306000	0
1341	2005	A2901247	2015	tga-red000	w	2023-02-25	12:00:00	2023-02-25	13:00:00	1	18	162000	162000
3112	2004	A2901239	2349	tgz-training	r	2023-04-24	12:00:00	2023-04-24	18:00:00	6	20	540000	0
3113	2004	A2901239	2349	tgz-training	r	2023-04-25	12:00:00	2023-04-25	18:00:00	6	20	540000	0
3116	2005	A2901247	2015	tga-red000	r	2023-04-18	17:00:00	2023-04-25	16:00:00	167	1	3006000	0
3122	2005	A2901247	2014	tga-blue000	r	2023-04-25	08:00:00	2023-05-02	08:00:00	168	5	15120000	0
3123	2005	A2901247	2014	tga-blue000	r	2023-05-02	08:00:00	2023-05-09	08:00:00	168	5	3780000	0
3301	2005	A2901247	2015	tga-red000	r	2023-08-30	14:00:00	2023-08-31	18:00:00	28	1	504000	0
3302	2005	A2901247	2009	tga-green000	r	2023-08-30	14:00:00	2023-08-31	18:00:00	28	1	504000	0
3304	2005	A2901247	2014	tga-blue000	r	2023-09-03	10:00:00	2023-09-04	10:00:00	24	1	432000	0
3470	2005	A2901247	2014	tga-blue000	w	2023-11-11	22:00:00	2023-11-11	23:00:00	1	1	4500	4500
4148	2004	A2901239	2007	tga-hpe_group00	w	2024-04-12	17:00:00	2024-04-12	18:00:00	1	1	4500	4500
4149	2005	A2901247	2015	tga-red000	w	2024-04-12	17:00:00	2024-04-13	17:00:00	24	1	108000	108000
4150	2004	A2901239	2007	tga-hpe_group00	w	2024-04-12	17:00:00	2024-04-12	18:00:00	1	1	4500	4500
total :										818	97	28507500	3739500

Use `t4-user-info compute ars` to check the availability of appointments for the current month from the command line.

5.4. Interactive job

The system's job scheduler has the ability to run programs and shell scripts interactively. To run an interactive job, use the `qrsh` command and specify the resource type and elapsed time with `-l`. After submitting a job with `qrsh`, a command prompt will be returned when the job is dispatched. The following shows how to use an interactive job.

```
#!/bash
$ qrsh -g [TSUBAME group] -l [Resource type]=[number] -l h_rt=[maximum run time]
Directory: /home/N/username
(job start time)
username@rXnY:~> [commands to run]
username@rXnY:~> exit
```

If the `-g` option group is not specified, up to two resource types, up to 10 minutes of elapsed time, and a "trial run" with a priority of `-5` will be performed.

Example specifying resource type `node_f`, 1 node, elapsed time 10 minutes

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

Enter `exit` can stop the interactive job.

5.4.1. X forwarding using interactive nodes

To perform X forwarding directly from a node connected by `qrsh`, please follow the procedure below.

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

Example)

The example shows a 2-hour job with resource type `cpu_4` and 1 node.

The assigned node is dispatched by the scheduler from the free nodes at the time the command is executed, so you cannot explicitly specify a node.

```
# Execution of qrsh command
$ qrsh -g [TSUBAME group] -l cpu_4=1 -l h_rt=2:00:00
username@rXnY:~> module load [application modules to load]
username@rXnY:~> [command to run X11 application]
username@rXnY:~> exit
```

5.4.2. Connecting to the network applications

If you need to operate an application using a Web browser or other means, you can use SSH port forwarding to access the application from a Web browser at local.

(1) Get hostname of interactive node connected by qrsh

```
$ qrsh -g tga-hpe_group00 -l cpu_4=1 -l h_rt 0:10:00
$ hostname
r1n1
$ [execute the program that requires Web browser]
```

After starting an interactive job with qrsh, get the hostname of the machine. In the above example, `r1n1` is the hostname. There is nothing to do anymore in the console, but keep it until the work by the application is finished.

(2) Connect with SSH port forwarding enabled from the console from which you are connecting to ssh. (not on the login node or interactive job)

```
ssh -i /path/to/key -l username -L 8888::<> login.t4.gsic.titech.ac.jp
```

The network port of the application to be connected is different for each application. For details, please refer to the documentation of each application or the application's startup message.



SSH port forwarding settings depend on the SSH console (or terminal software) used to SSH into TSUBAME4. For details, please check the manual of each SSH console or refer to the FAQ.

(3) Connect to the application with a web browser.

Launch a web browser (Microsoft Edge, Firefox, Safari, etc.) on the console at hand and go to `http://localhost:8888/`.

5.4.3. Interactive queue

An interactive queue is an environment where the same resource is shared among users, making it less likely to fail to secure a node even when congested, and providing a quick start to visualization and interactive programs.

Submitting jobs to the interactive queue is as follows



Only intramural users (tgz-edu) and access card users can run it for free, omitting the group specification.

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

Note that CPU/GPU over-commitment is allowed in the interactive queue.

Limit for interactive queue is described [here](#).

5.5. SSH login to compute nodes

Nodes that have been submitted jobs with resource type `node_f` can be logged in directly via ssh. The secured node can be checked by the following procedure.

```
$ qstat -j 1463
=====
job_number:          1463
(snip)
exec_host_list      1:   r8n3:28, r8n4:28    <-- assigned nodes: r8n3, r8n4
(snip)
```



When you ssh into a compute node, the GID of the sshed process is set to tsubame-users(2000), so you cannot see your job process running immediately after ssh, except for a trial run, and you cannot attach to it with gdb. To make it visible, execute the following with the name of the group that executed the job after ssh.

```
newgrp <groupname>
or
sg <groupname>
```

5.6. Storage use on Compute Nodes

5.6.1. Local scratch area

SSD on each compute node can be used as a local scratch area.

The local scratch area is set to `/local/${JOB_ID}` (or `/local/${JOB_ID}/${SGE_TASK_ID}` for array jobs). It can be referenced by specifying the path of the work area in the job script.

Since the local scratch area is a separate area for each compute node and is not shared, input and output files from within the job script must be staged on the local host.

The following example copies the input data set from the home directory to the local scratch area and returns the results to the home directory when there is only one compute node used. (Multiple nodes are not supported.)

```
#!/bin/sh
# copy input data to the scratch
cp -rp $HOME/datasets /local/${JOB_ID}
# run your program that uses input/output data
./a.out /local/${JOB_ID}/datasets /local/${JOB_ID}/results
# copy the result back to your home directory
cp -rp /local/${JOB_ID}/results $HOME/results
```



`/local/${JOB_ID}` will be deleted when the job completes.

6. Commercial applications

Under the license agreement, users who can use commercial applications is limited.

Users other than "1. Student/Staff ID" who belong to Tokyo Tech can only use the following commercial applications.

- Gaussian/Gauss View
- AMBER (Only users affiliated with academic institutions)
- Intel oneAPI Compiler
- NVIDIA HPC SDK Compiler
- Arm Forge



Each application fee is required for the use of some commercial applications. For more details, please refer to [Fare Overview](#) **Commercial Applications** (Partially charged in TSUBAME4.0).

Commercial applications are listed below:

Software name	Description
ANSYS	Analysis Software
ANSYS/Fluent	Analysis Software
ANSYS/LS-DYNA	Analysis Software
ABAQUS	Analysis Software
ABACUS CAE	Analysis Software
Gaussian	Quantum chemistry calculation program
GaussView	Quantum chemistry calculation program Pre-Post tool
AMBER	Molecular dynamics calculation program
Materials Studio	Chemical Simulation Software
Discovery Studio	Chemical Simulation Software
Mathematica	Mathematical Processing Software
COMSOL	Analysis Software
Schrodinger	Chemical Simulation Software
MATLAB	Numerical calculation software
VASP	Quantum-mechanical molecular dynamics program
Arm Forge	Debugger
Intel oneAPI Compiler	Compiler, Development tool
NVIDIA HPC SDK Compiler	Compiler, Development tool

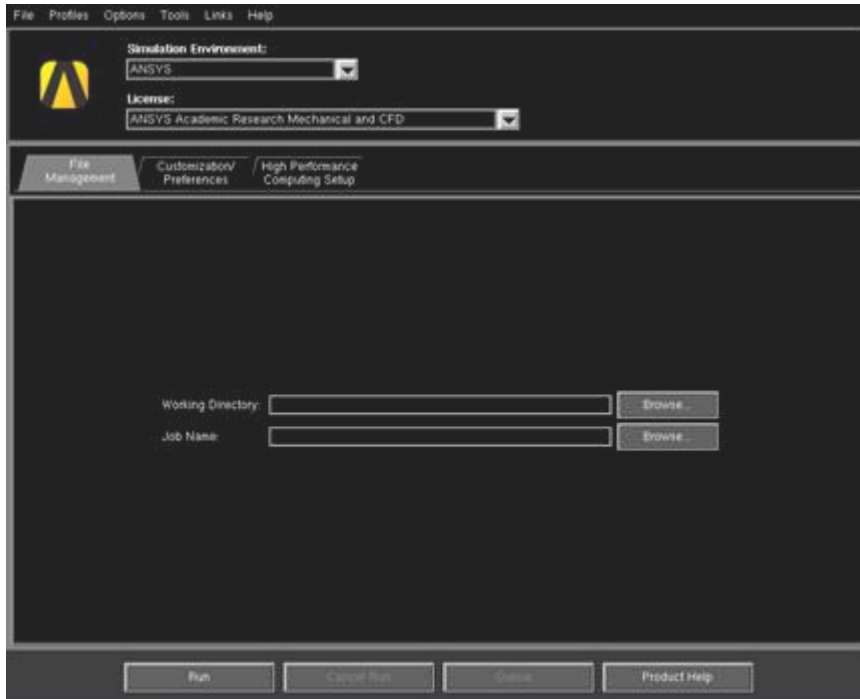


For "module" command, please refer to [Switch User Environment](#).

6.1. ANSYS

The procedure for using GUI is shown below.

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



The procedure for using CLI is shown below.

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

Following command is also available instead of mapdl.

(As of ANSYS24.1. It depends on the version.)

```
$ ansys241
```

Enter `exit`, then the program ends.

Specify input file, you can run it interactively.

```
Example-1
$ mapdl [options] < inputfile > outputfile

Example-2
$ mapdl [options] -i inputfile -o outputfile
```

To use the batch queue system, create a shell script and run it in the CLI as follows

```
$ qsub sample.sh
```

Script example : MPI parallel processing

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

module load ansys

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

Script example : With GPU

```
#!/bin/bash
#$ -cwd
#$ -V
#$ -l node_f=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
```

The license usage of ANSYS can be checked as below:

```
$ lutil lmstat -S ansyslmd -c *****@kvm5:*****@kvm6:*****@ldap2
```



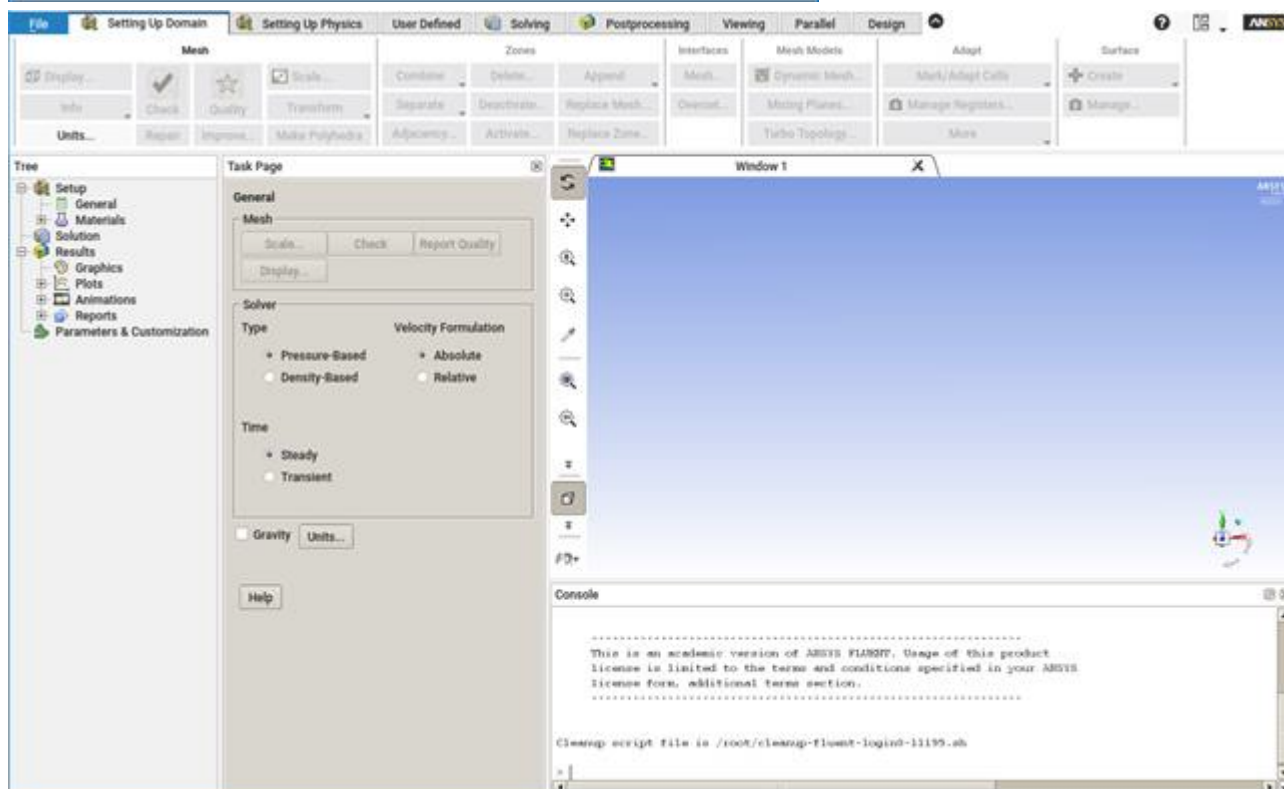
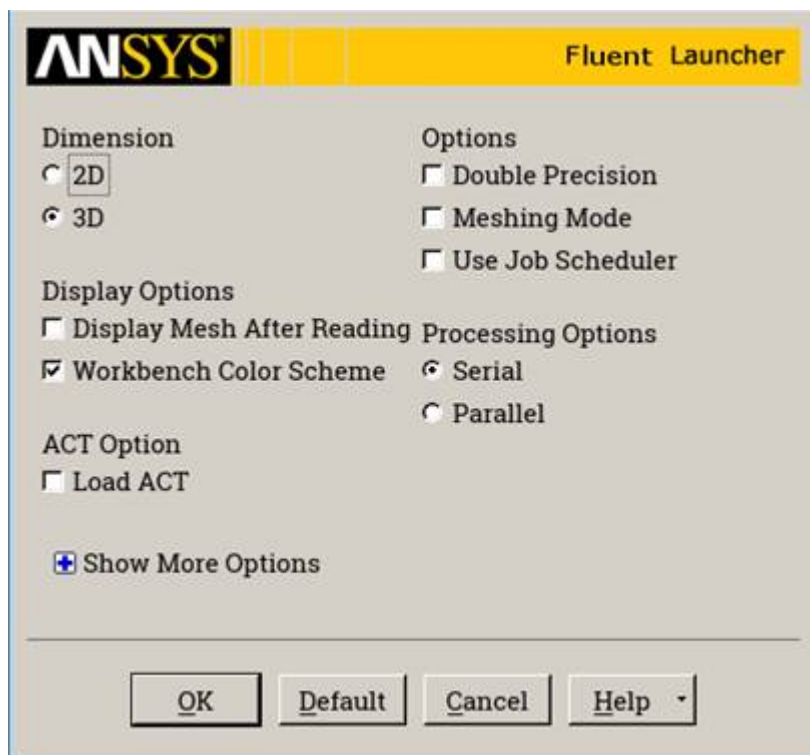
The port number for the license is visible after you purchase the application with [Application activation \(TSUBAME4\)](#).
/apps/t4/rhel9/isv/ansys_inc/t4-license
The port number is changed in mid April every year.

6.2. ANSYS/Fluent

ANSYS/Fluent is a thermal fluid analysis application. The usage procedure is shown below.

GUI startup procedure is shown below.

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



CLI startup procedure is shown below.

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

Entering exit ends the program.

To run interactively using the journal file, execute the command as follows


```
journal file is fluentbench.jou, 3D

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

To use the batch queue system, create a shell script and run it in the CLI as follows

```
$ qsub sample.sh
```

Script example : MPI Parallel processing (node_f)

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

module load ansys

JOURNAL=journalfile
OUTPUT=outputfile
VERSION=3d

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

Script example : MPI Parallel processing (node_h)

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

module load ansys

JOURNAL=journalfile
OUTPUT=outputfile
VERSION=3d

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

Since settings across resources are not possible when using anything other than `node_f`, please use `#$ -l {resource name}=1` (for example, `#$ -l node_h=1` for `node_h`) and include the `-ncheck` option in the command.

The license usage of ANSYS/Fluent can be checked as below:

```
$ lmutil lmstat -S ansyslmd -c *****@kvm5:*****@kvm6:*****@ldap2
```



The port number for the license is visible after you purchase the application with [Application activation \(TSUBAME4\)](#).
/apps/t4/rhel9/lsv/ansys_inc/t4-license
The port number is changed in mid April every year.

6.3. ANSYS/LS-DYNA

6.3.1. ANSYS/LS-DYNA overview

LS-DYNA is a program that analyzes the large deformation behavior of structures with an explicit time history, and is a highly reliable program with a world-class installation record in these fields, demonstrating its power in crash/impact analysis, drop analysis, plastic forming analysis, and penetration/crack/fracture analysis.

6.3.2. Launching ANSYS/LS-DYNA

ANSYS/LS-DYNA is used in batch jobs. An example of a batch script is shown below.

Please read and execute the script according to the version you wish to use.

Script example : MPP single-precision version

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

module load ansys intel-mpi

export dynadir=/apps/t4/rhel9/lsv/ansys_inc/v241/ansys/bin/linux64
export exe=$dynadir/lsdyna_sp_mpp.e
export dbo=$dynadir/lsl2a_sp.e

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

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

${dbo} binout*
```

Script example : MPP double-precision edition

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

module load ansys intel-mpi

export dynadir=/apps/t4/rhel9/lsv/ansys_inc/v241/ansys/bin/linux64
export exe=$dynadir/lsdyna_dp_mpp.e
export dbo=$dynadir/lsl2a_dp.e

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

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

${dbo} binout*
```

Scripts should be modified to suit the user's environment. In the example script above, the input file is specified as INPUT=inputfile in the shell script.

6.4. ABAQUS

The procedure for interactive use is shown below.

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

To use the batch queue system, create a shell script and run it in the CLI as follows

```
$ qsub sample.sh
```

Scrip example : MPI Parallel processing

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

module load abaqus

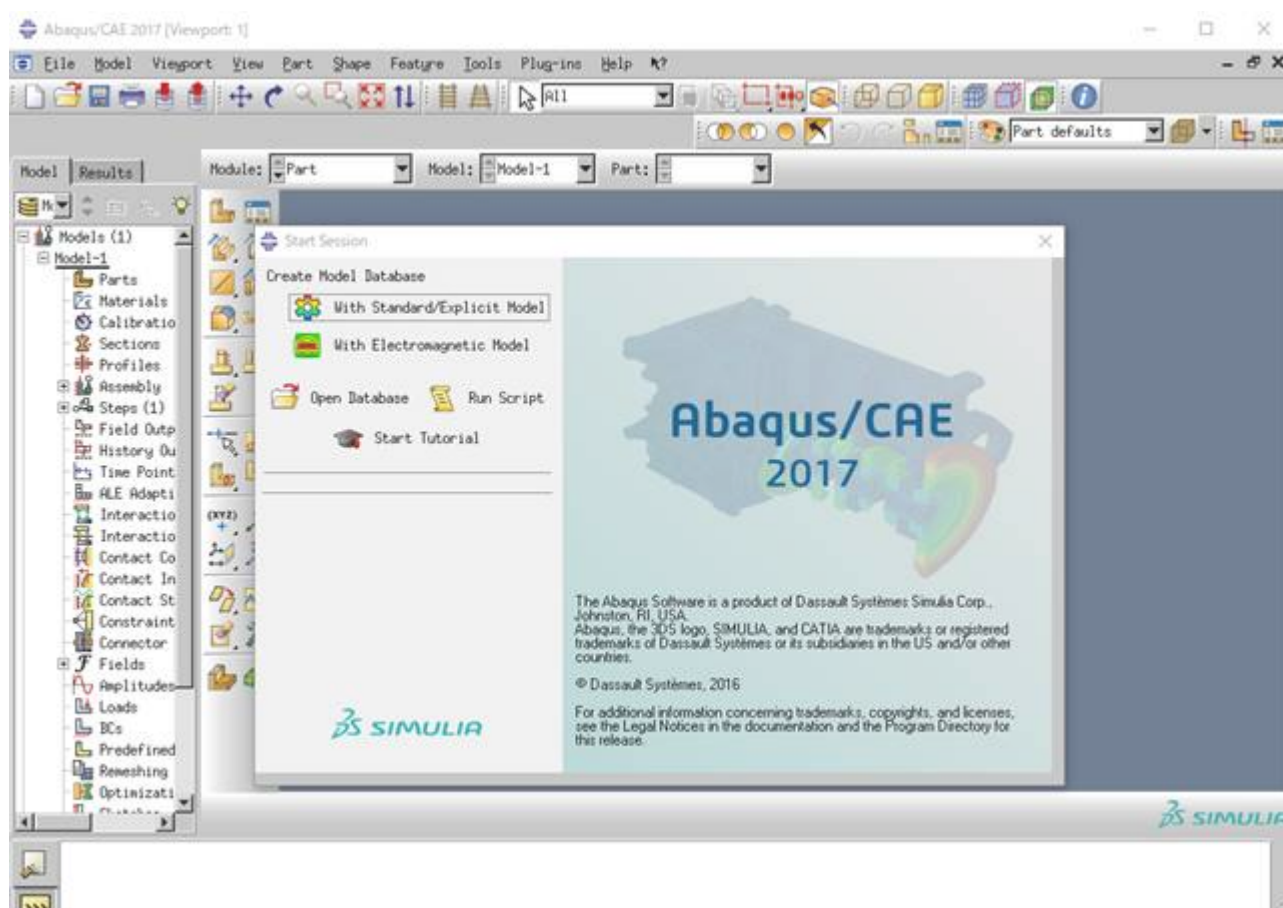
# ABAQUS settings.
INPUT=s2a
ABAQUS_VER=2024
ABAQUS_CMD=abq${ABAQUS_VER}
SCRATCH=${TMPDIR}
NCPUS=4

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

6.5. ABAQUS CAE

The procedure for using ABAQUS CAE is shown below.

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



Click File > Exit on the menu bar to exit.

6.6. Gaussian

The interactive usage procedure is shown below.

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

Specify the revision you are using for revision; for Gaussian16 Rev.C02, it is as follows

```
$ module load gaussian/16C2_cpu
$ g16 inputfile
```

To use the batch queue system, create a shell script and run it in the CLI as follows

```
$ qsub sample.sh
```

Script example : Intra-node parallel processing

Sample scripts for calculating structural optimization and vibrational analysis (IR+Raman intensity) of Glycine.

The calculation can be performed by placing the following files, glycine.sh and glycine.gjf, in the same directory and executing the following command. After calculation, glycine.log and glycine.chk are generated.

The results of the analysis are explained in GaussView.

```
$ qsub glycine.sh
```

glycine.sh

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

module load gaussian

g16 glycine.gjf
```

glycine.gjf

```
chk=glycine.chk
cpu=0-191    <--Not necessary if the module that automatically sets the environment variables GAUSS_CDEF/GAUSS_GDEF is loaded.
mem=700GB

P opt=(calcf,c,tight,rfo) freq=(raman)

glycine Test Job

  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    0.90878333 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

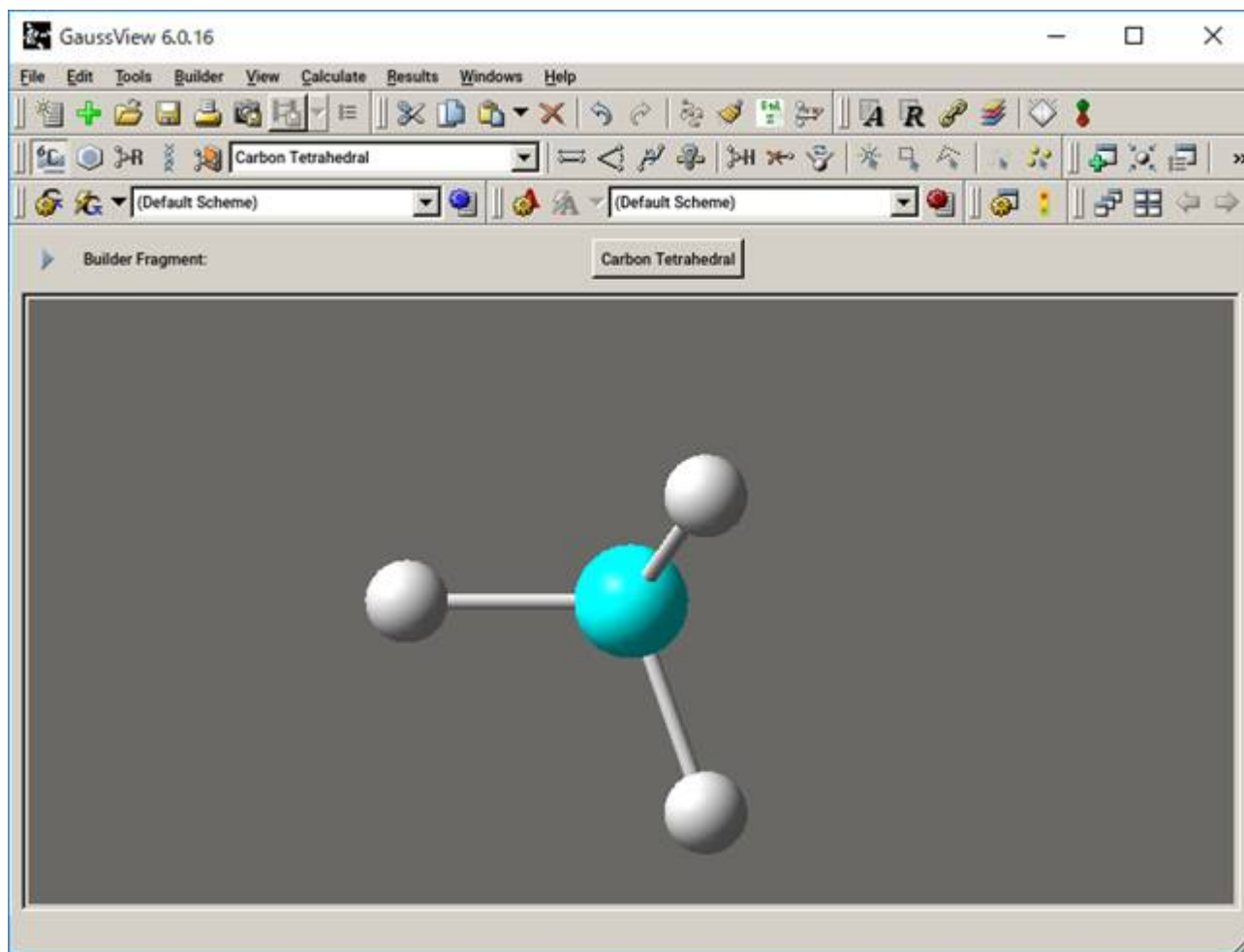
1 2 1.0 3 1.0 9 1.0
2
3 4 1.0 5 1.0 6 1.0
4 7 1.5 8 1.5
5
6
7
8
9
```

6.7. GaussView

GaussView is an application that visualizes Gaussian results.

The procedure for using GaussView is shown below.

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



Click File > Exit from the menu bar to exit.

Example: glycine.log

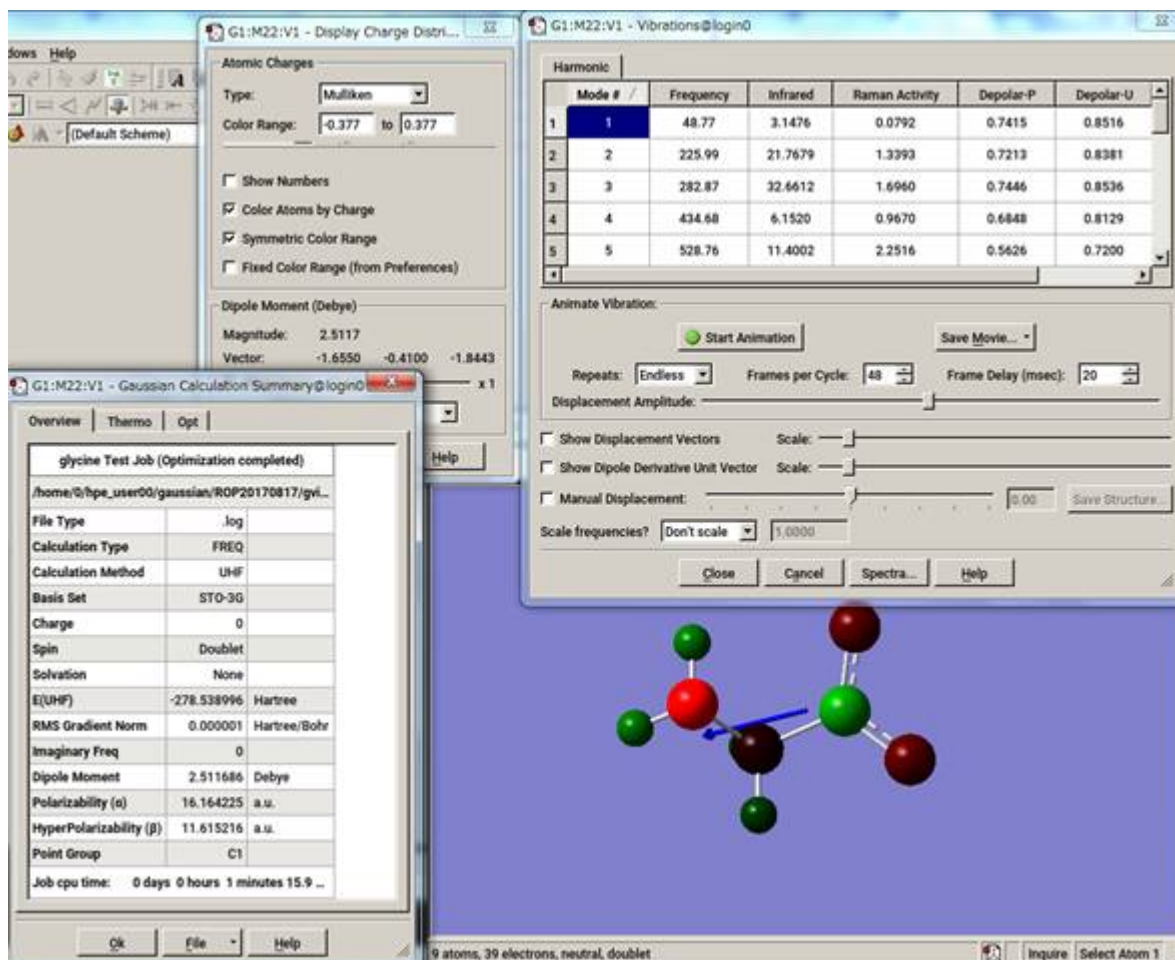
The following explanation uses as an example the analysis of a result file obtained by executing the script shown as a sample in the Gaussian section.

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

You can check the analysis results from Result.

You can see a summary of the calculation in Result>Summary, the charge information in Result>ChageDistribution..., and the results of the vibration analysis in Vibration....

In the sample, vibration analysis is performed, so you can see the vibration from StartAnimation in the Vibration dialog.



6.8. AMBER

AMBER was originally developed for molecular dynamics calculations of proteins and nucleic acids, but recently parameters for sugars have also been developed, making it an increasingly useful tool for chemical and biological research. If you wish to use AMBER in your own research, you should carefully study the manual and related papers to understand the limitations of the models and theories employed by AMBER, as well as the scope of application. Currently, AMBER does not allow unlimited copying of its source code, but it is possible to use it internally at Tokyo Tech, so you can incorporate further developed methods based on it.

Please read the following as appropriate for the version you wish to use.

The following is the procedure for interactive sequential processing.

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

The following is the procedure for using parallel processing (sander.MPI) in an interactive manner.

```
$ module load amber
$ mpirun -np [parallel number] sander.MPI [-O]A -i mdin -o mdout -p prmtop -c inpcrd -r restrt
```

The following is the procedure for using GPU sequential processing (pmemd.cuda) in interactive mode.

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

The following is the procedure for using GPU parallel processing (pmemd.cuda.MPI) in interactive.

```
$ module load amber
$ mpirun -np [parallel number] pmemd.cuda.MPI [-O] -i mdin -o mdout -p prmtop -c inpcrd -r restrt
```

The following is the procedure for using the system in the case of a batch queue system.

```
$ qsub parallel.sh
```

Script example : CPU Parallel processing

```
#!/bin/bash
#$ -cwd
#$ -l node_f=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

module load amber

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

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

Script example : GPU Parallel processing

```
#!/bin/bash
#$ -cwd
#$ -l node_f=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

module load amber

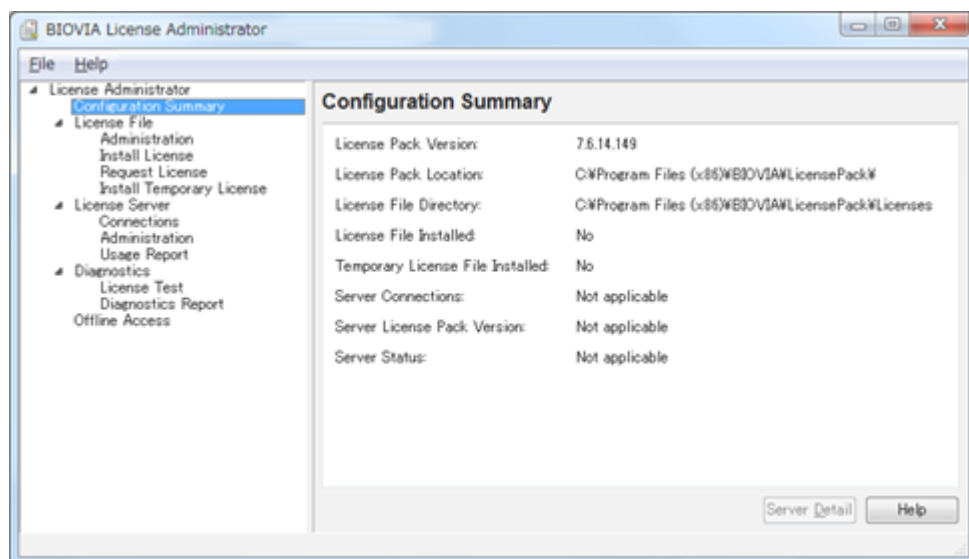
mpirun -np $NSLOTS \
pmemd.cuda.MPI -O -i $in -c $inpcrd -p $top -o $out < /dev/null

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

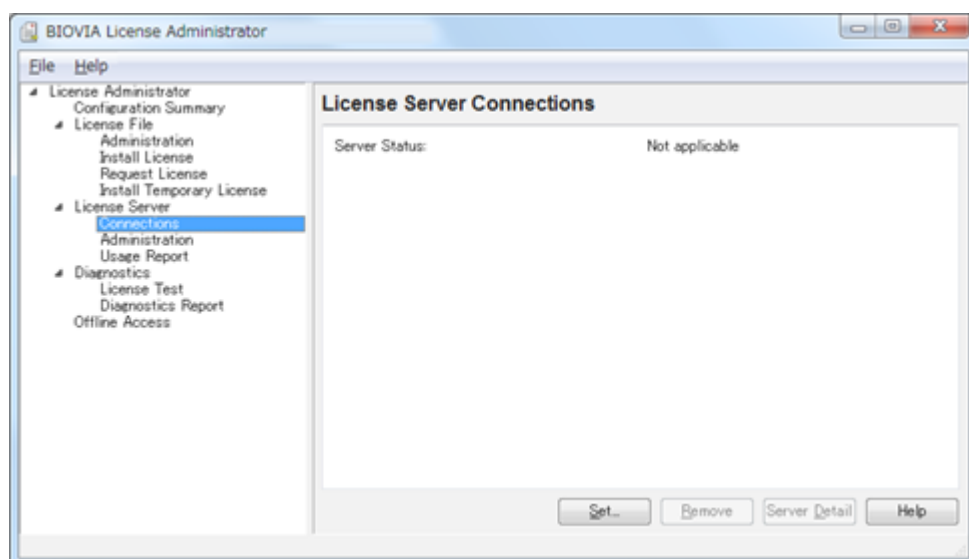
6.9. Materials Studio

6.9.1. License connection setting

BIOVIA > Licensing > License Administrator x.x.xx 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.

Host name	kvm5
Host name	kvm6
Host name	ldap2



The port number for the license is seen after you purchase the application with Application distribution for Labs. They are described in the install manual. The port number is changed in mid April every year.

If the Server Status displays "Connected", the setting is complete.

In order to use Materials Studio, a connection must be established to the license server on two or more hosts.

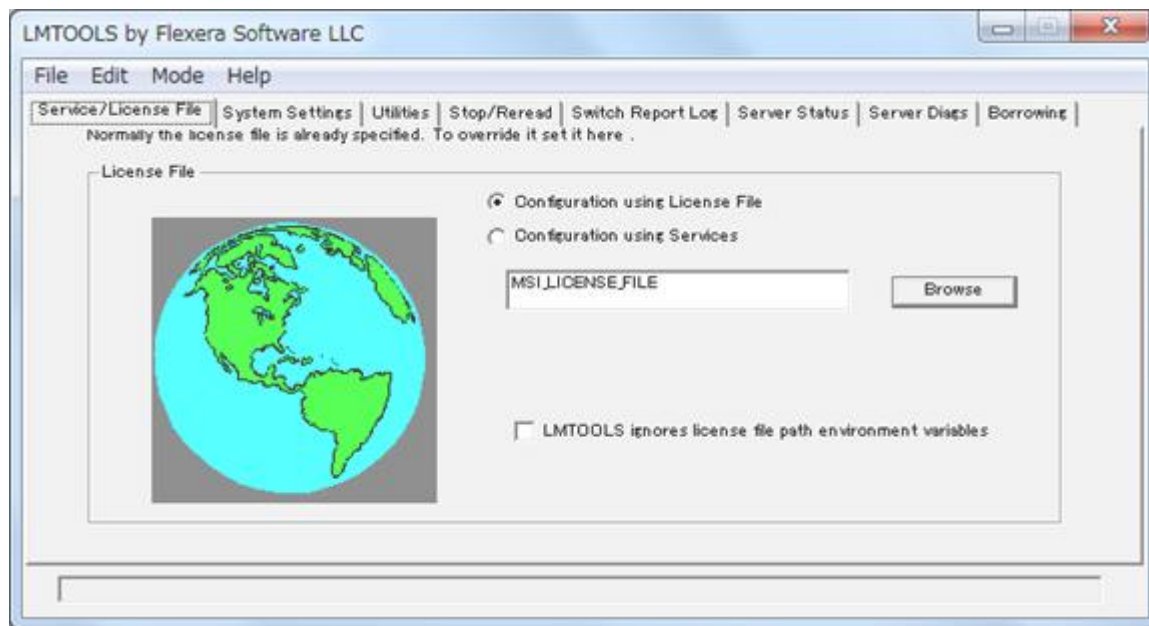
6.9.2. License Usage Status

6.9.2.1. How to check on Windows

BIOVIA > Licensing > License Administrator x.x.xx > Utilities (FLEXlm LMTTOOLS) from the Windows [Start menu] .

Open [Service/License File] tab and select [Configuration using License File] .

Make sure that MSI_LICENSE_FILE is displayed.



Open [Server Status] tab, click [Perform Status Enquiry] 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 Enquiry].

6.9.2.2. How to check on login nodes

oExecute the following command to view the usage status.

```
$ lmutil lmstat -S msi -c *****@kvm5,*****@kvm6,*****@ldap2
```



The port number for the license is visible after you purchase the application with [Application activation \(TSUBAME4\)](#).
/apps/t4/rhel9/ism/materialsstudio/t4-license
The port number is changed in mid April every year.

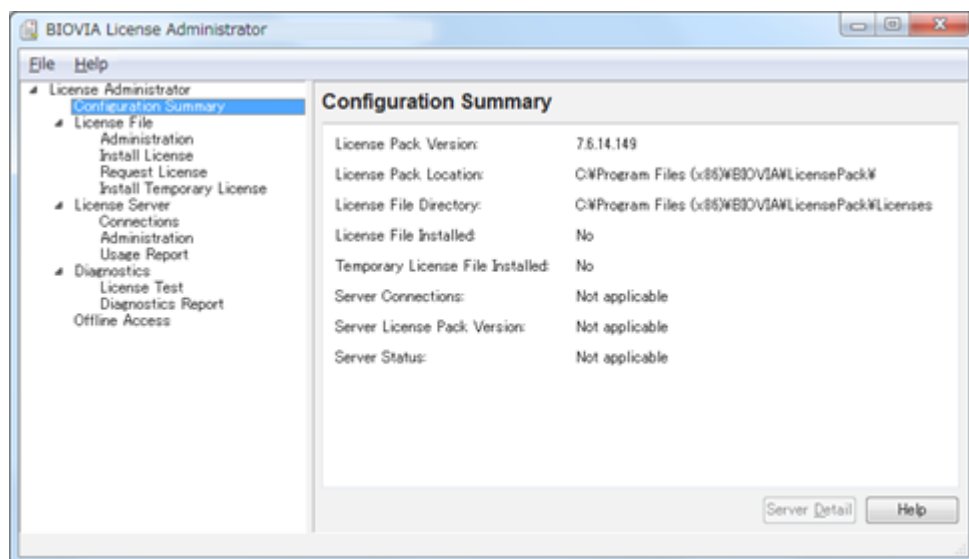
6.9.3. Launching Materials Studio

On Windows that Materials Studio installed, go to Start menu and click BIOVIA > Materials Studio 2024 to launch it.

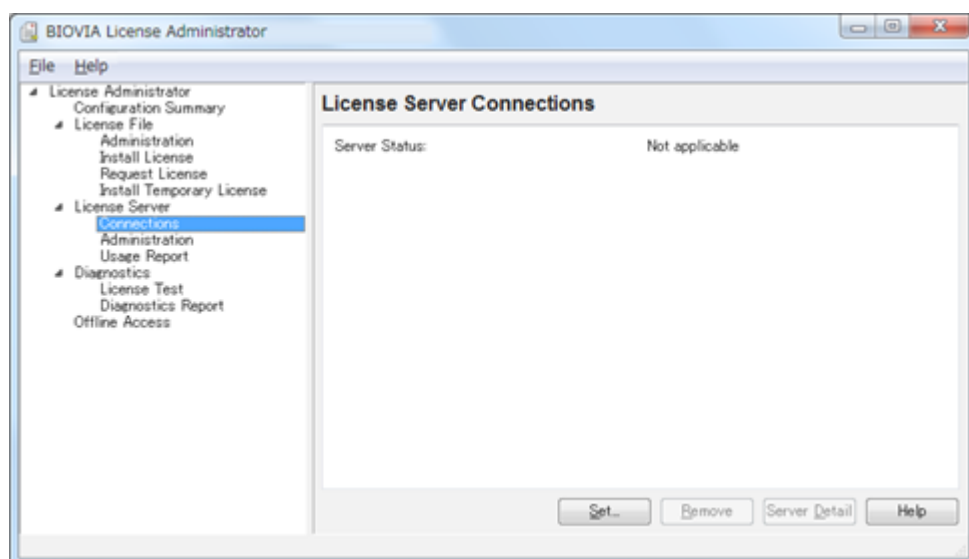
6.10. Discovery Studio

6.10.1. How to set up a license connection

From the Start menu, run BIOVIA > Licensing > License Administrator X.X.X as administrator.



Open [Connections] and click [Set] to open the Set License Server dialog.



Check Redundant servers, enter the host name and port number as shown below, and click OK.

Host name	kvm5
Host name	kvm6
Host name	ldap2



The port number for the license is seen after you purchase the application with [Application distribution for Labs](#). They are described in the install manual. The port number is changed in mid April every year.

If the Server Status shows "Connected", the setup is complete.

In order to use Discovery Studio, a connection must be established to at least two license server hosts.

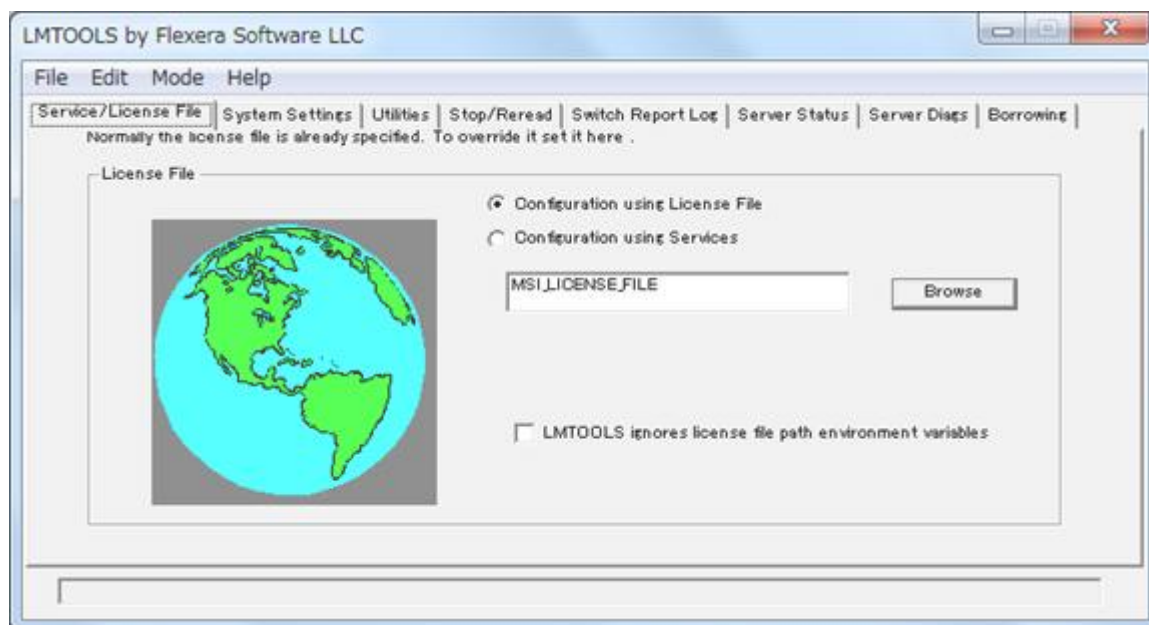
6.10.2. How to check license usage

6.10.2.1. How to check on Windows

Run BIOVIA > Licensing > License Administrator X.X.X > Utilities (FLEXlm LMTTOOLS) from Windows start menu.

Open [Service/License File] tab, select [Configuration using License File].

Make sure MSI_LICENSE_FILE is displayed.



Open [Server Status] tab, click [Perform Status Enquiry] then the status of license usage is shown.

If you want to check only specific license, input the license name into [Individual Feature], then run [Perform Status Enquiry].

6.10.2.2. How to check on login nodes

Execute the following command to view the usage status.

```
$ lmtutil lmstat -S msi -c *****@kvm5,*****@kvm6,*****@ldap2
```



The port number for the license is visible after you purchase the application with [Application activation \(TSUBAME4\)](#).
/apps/t4/rhel9/lsv/discoverystudio/t4-license
The port number is changed in mid April every year.

6.10.3. Launching Discovery Studio

On Windows that Discovery Studio is installed, click BIOVIA > Discovery Studio 2024 64-bit Client from Start menu.

6.11. Mathematica

The following is the procedure for starting with the CLI.

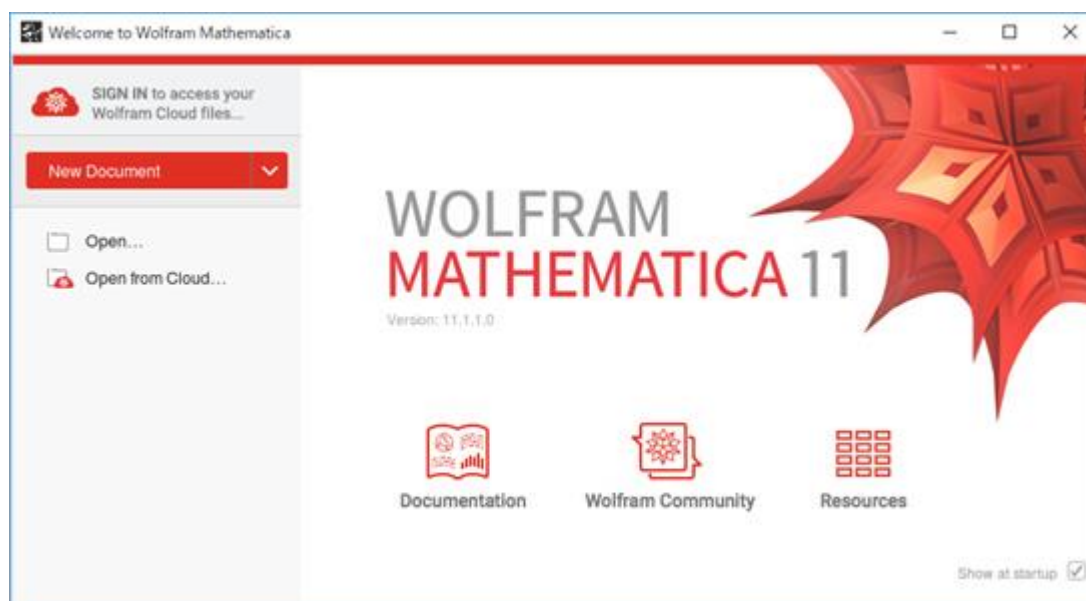
```
$ module load mathematica
$ math
Mathematica 11.1.1 Kernel for Linux x86 (64-bit)
Copyright 1988-2024 Wolfram Research, Inc.

In[1]:=
```

Entering Quit ends the program.

You can use it with GUI as follows.

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



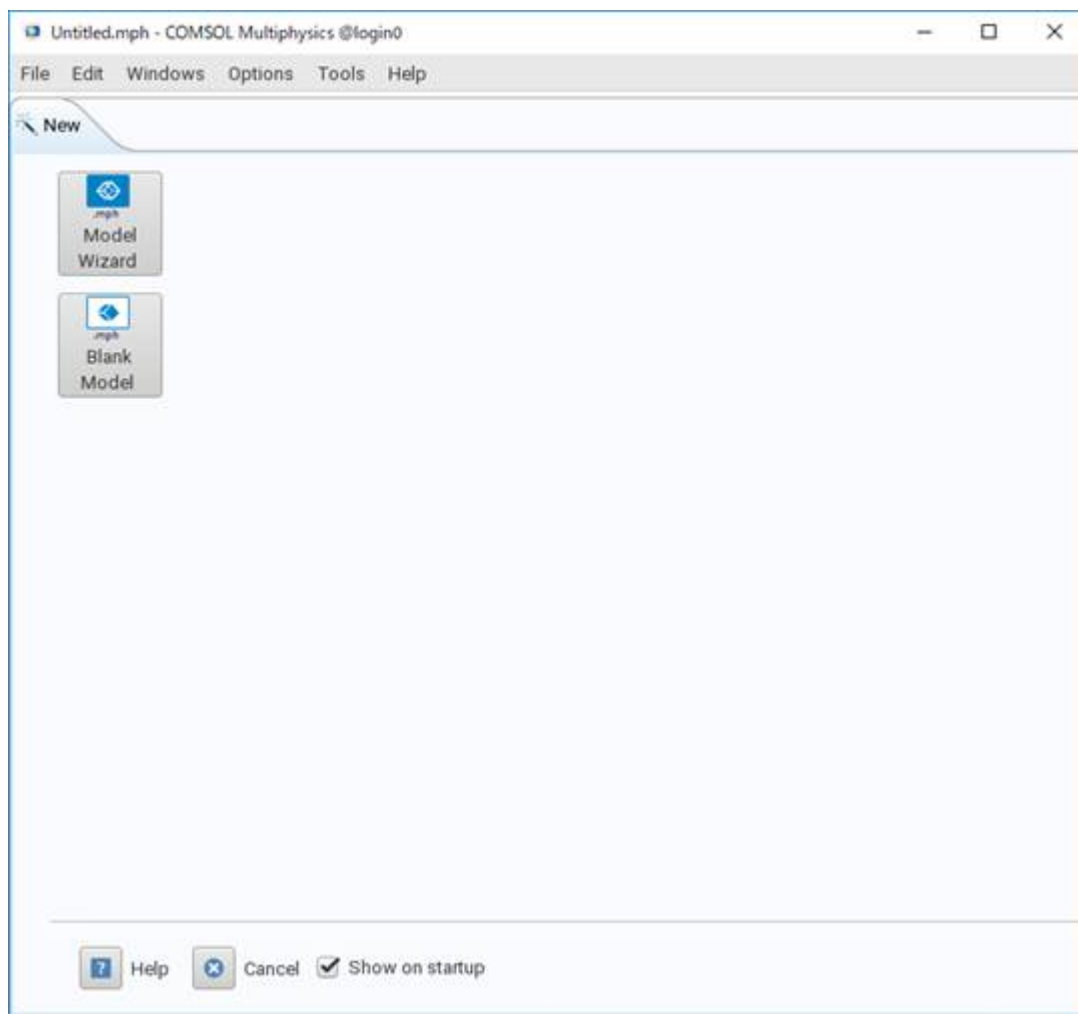
When you stop the program, select [File] from menu bar of notebook, then [exit].

6.12. COMSOL

COMSOL은 3D 모델링, 시뮬레이션, 최적화를 위한 소프트웨어입니다.

You can launch COMSOL as follows.

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



Click File > Exit from menu bar, then the program ends.

You can check the status of COMSOL licenses as follows.

```
$ lmutil lmstat -S LMCOSOL -c *****@kvm5:*****@kvm6:*****@ldap2
```



The port number for the license is visible after you purchase the application with [Application activation \(TSUBAME4\)](#).

/apps/t4/rhel9/isv/comsol/t4-license

The port number is changed in mid April every year.

6.13. Schrodinger

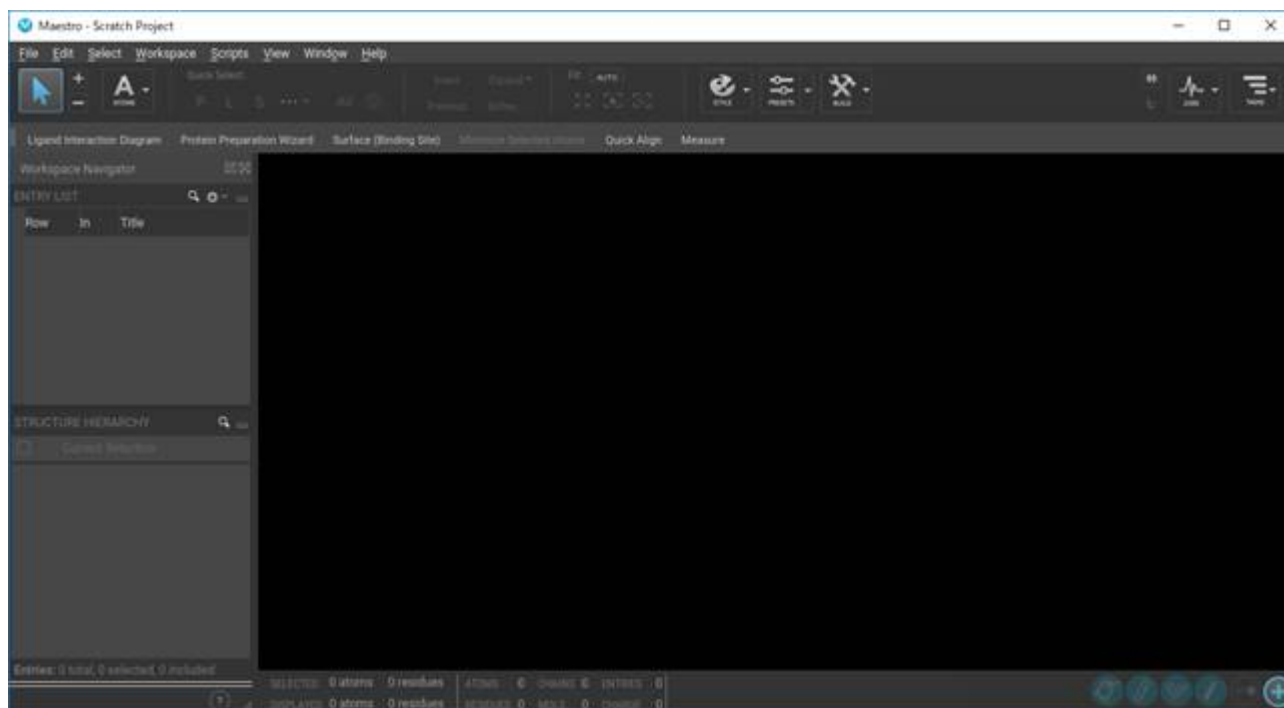
The procedure for using Schrodinger is shown below.

Ligprep with CLI :

```
$ module load schrodinger
SMILES format input file, MAE format output file
$ ligprep -ismiinputfile -omaeoutputfile
```

If you want to use it with GUI, launch Maestro.

```
$ module load schrodinger
$ maestro
```



Click File > Exit on menu bar will ends the program.

You can check license usage of Schrodinger as follows.

```
$ lmtutil lmstat -S SCHROD -c *****@kvm5:*****@kvm6:*****@ldap2
```



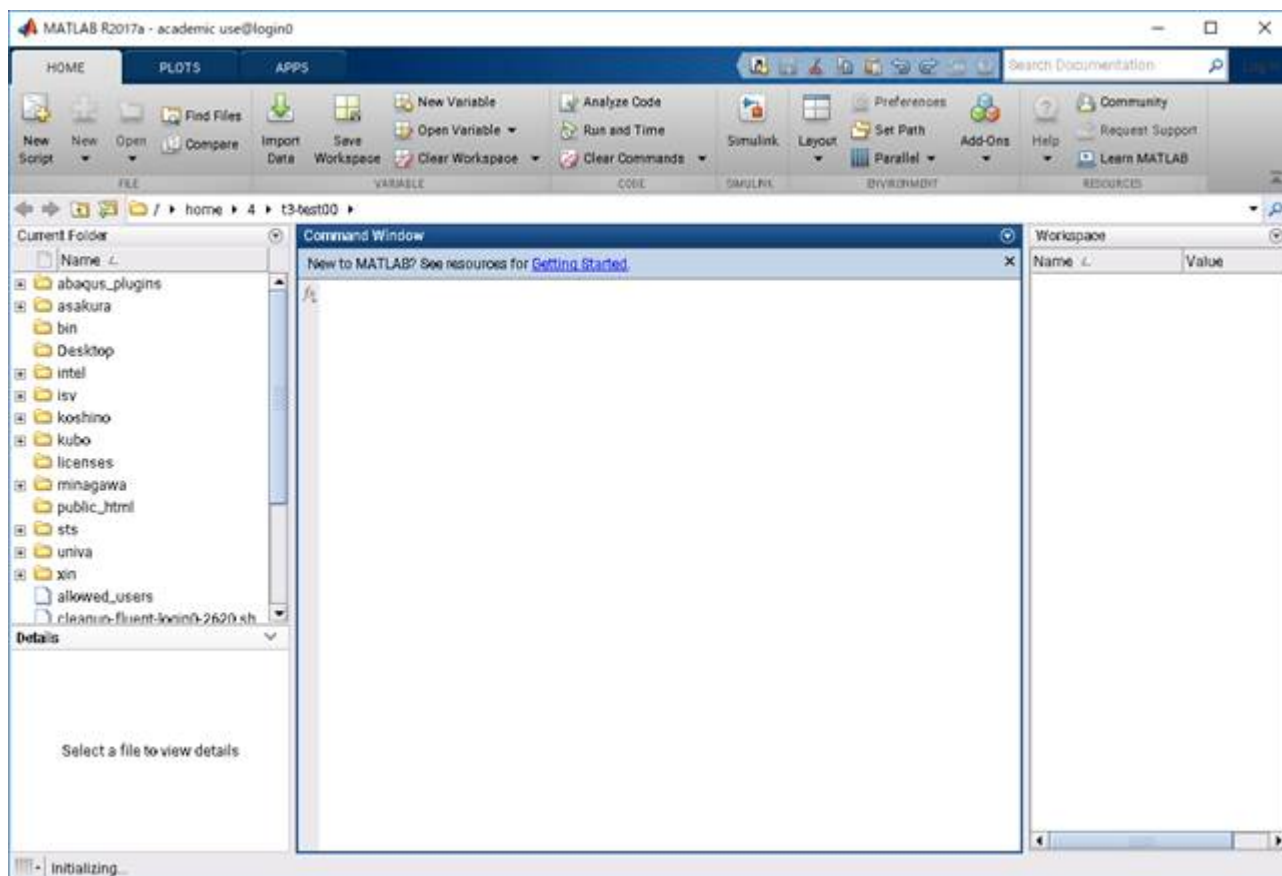
The port number for the license is visible after you purchase the application with [Application activation \(TSUBAME4\)](#).
/apps/t4/rhel9/isv/schrodinger/t4-license
The port number is changed in mid April every year.

6.14. MATLAB

MATLAB is an application that enables numerical calculations such as matrix calculations and data visualization.

Examples of how to use MATLAB are shown below.

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



CLI

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

Enter exit when you stop using the application.

You can check MATLAB license usage status as follows.

```
$ lutil lmstat -S MLM -c *****kvm5:*****kvm6:*****ldap2
```



The port number for the license is visible after you purchase the application with **Application activation (TSUBAME4)**.

/apps/t4/rhel9/isv/matlab/t4-license

The port number is changed in mid April every year.

6.15. VASP



When using VASP, users must have their own license.



For VASP inquiries, the response by TSUBAME4.0 contacts is limited due to licensing terms.
Please refer to [TSUBAME4.0 FAQ](#) or refer to and post on [VASP Forum](#).

There are two ways to use VASP with TSUBAME 4.0.

- **Use pre-compiled binaries**

We've acquired VASP licenses for supercomputers and provide compiled binaries to VASP license holders.

If you have a valid VASP6 license and want to use compiled binaries on TSUBAME4.0, please register [through this application form](#).

After the procedure is completed, set permissions to access binaries on TSUBAME. (It may take some time as confirmation with the license issuer is required.)

VASP compiled binary usage is shown below.

```
$ module load VASP
$ vasp_std (or vasp_ncl etc...)
```

- **Build the source code yourself**

If you build the source code yourself, no application is required. Please prepare the source code yourself.

Also, you do not need to specify the module command. Specify environment variables according to your own environment.

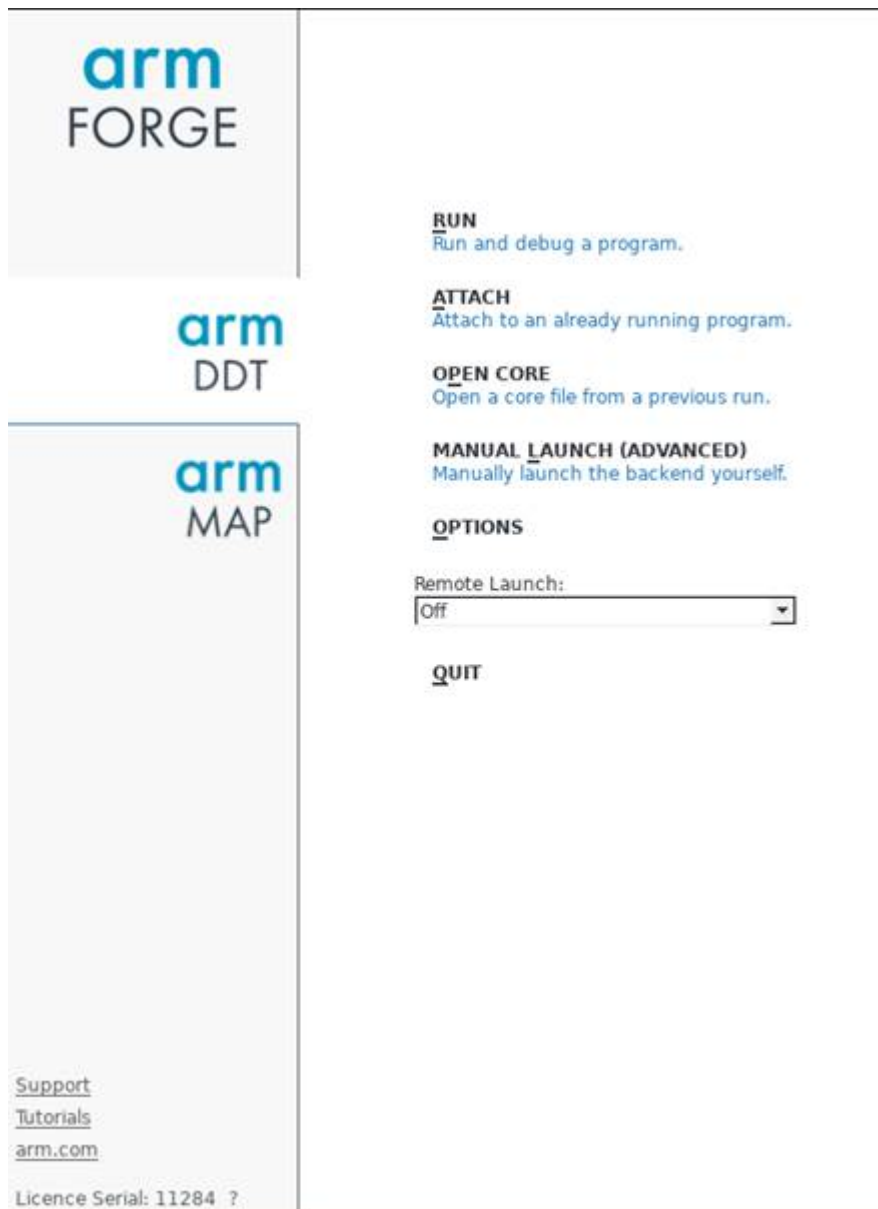
As reference information, the procedure for building VASP6.4.2 on TSUBAME4.0 is available.

- [I would like to know the procedure to build a VASP with TSUBAME4.0.](#)

6.16. Arm Forge

Run Arm Forge as follows.

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

Select File > Exit will end the program.

7. Freeware

Freesoftware available in this system is listed below.

Software	Description
GAMESS	Solver Simulator
Tinker	Solver Simulator
GROMACS	Solver Simulator
LAMMPS	Solver Simulator
NAMMD	Solver Simulator
QUANTUM ESPRESSO	Solver Simulator
CP2K	Solver Simulator
OpenFOAM	Solver Simulator, Visualization
CUDA	GPU library
CuDNN	GPU library
NCCL	GPU library
TensorFlow	DeepLearning framework
DeePMD-kit	MD DeepLearning framework
PyTorch	Machine larning
R	Interpreter (Rmpi,rpud)
Hadoop	Distributed Data Processing Tools
POV-Ray	Visualization
ParaView	Visualization
VisIt	Visualization
vmd	Visualization
VESTA	Visualization
turbovnc	Remote GUI(X11)
VirtualGL	Remote GUI
Open OnDemand	Web poral site for HPC
gnuplot	Data grapher/visualization
GIMP	Image Viewing and Editing
Tgif	Image Viewing and Editing
ImageMagick	Image Viewing and Editing
TeX Live	TeX distribution
OpenJDK	Development tool
python	Development tool
ruby	Development tool
perl	Development tool
PHP	Development tool
golang	Development tool

Software	Description
Emacs	Editor
vim	Editor
PETSc	Linear system solvers, libraries
FFTW	Fast Fourier Transform Library
Apptainer	Container platform
Spack	Software package management
miniconda	Software package management
PyPI	Software package management
Rbenv	Software package management
Alphafold	Bio
tmux	Terminal mutiplexer
NetCDF	Multidimensional data format
HDF5	Hierarchical data format
ffmpeg	Video and audio processing



For "module" command, please refer to [Switch User Environment](#).

7.1. Quantum chemistry/MD related software

7.1.1. GAMESS

GAMESS is an open source ab initio molecular quantum chemistry calculation application.

An example of how GAMESS can be used with the batch queue system is shown below.

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

module load gamess
$GAMESS_DIR/rungms exam01 00 2 2 1
```

A detailed description is provided below.

<http://www.msg.ameslab.gov/gamess/index.html>

7.1.2. Tinker

Tinker is a modeling software for molecular dynamics with special features for biopolymers.

An example of how Tinker can be used with the batch queue system is shown below.

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

```
module load tinkler
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
```

A detailed description is provided below.

<https://dasher.wustl.edu/tinker/>

7.1.3. GROMACS

GROMACS is an engine for molecular dynamics simulation and energy minimization.

An example of how to use GROMACS with the batch queue system is shown below.

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

module load 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 -np 4 gmx_mpi mdrun
cp -rp $TMPDIR/water-cut1.0_GMX50_bare $HOME
```

A detailed description is provided below.

<http://www.gromacs.org/>

7.1.4. LAMMPS

LAMMPS is a classical molecular dynamics code that models populations of liquid, solid, and gaseous particles.

An example of how LAMMPS can be used with a batch queue system is described below.

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

module load lammps
cp -rp $LAMMPS_DIR/examples/VISCOSITY $TMPDIR
cd $TMPDIR/VISCOSITY
mpirun -x PATH -x LD_LIBRARY_PATH -np 4 lmp -pk gpu 0 -in in.gk.2d
cp -rp $TMPDIR/VISCOSITY $HOME
```

A detailed description is provided below.

<http://lammps.sandia.gov/>

7.1.5. NAMD

NAMD is an object-oriented parallel molecular dynamics code designed for high-performance simulations of large biomolecular systems.

An example of how NAMD can be used with a batch queue system is described below.

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

module load 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
```

A detailed description is provided below.

<https://www.ks.uiuc.edu/Research/namd/3.0/ug/>

7.1.6. CP2K

CP2K is a quantum chemistry and solid state physics software package that can run atomic simulations of solid, liquid, molecular, periodic, material, crystalline, and biological systems.

An example of how CP2K can be used with a batch queue system is described below.

```
#!/bin/bash
#$ -cwd
#$ -l node_f=1
#$ -l h_rt=0:10:0
#$ -N cp2k

module load cp2k
cp -rp $CP2K_DIR/benchmarks/QS $TMPDIR
cd $TMPDIR/QS
export OMP_NUM_THREADS=1
mpirun -x PATH -x LD_LIBRARY_PATH -np 4 cp2k.psmf -i H2O-32.inp -o H2O-32.out
cp -rp $TMPDIR/QS $HOME
```

A detailed description is provided below.

<https://www.cp2k.org/>

7.1.7. QUANTUM ESPRESSO

QUANTUM ESPRESSO is a suite for first-principles electronic structure calculations and materials modeling.

An example of how to use QUANTUM ESPRESSO with the batch queue system is described below.

```
#!/bin/sh
#$ -cwd
#$ -l h_rt=00:10:00
#$ -l node_f=1
#$ -N q-e

module purge
module load 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 -np 4 pw.x < scf.in
```

A detailed description is provided below.

<https://www.quantum-espresso.org/>

7.2. CFD related software

7.2.1. OpenFOAM

OpenFOAM is an open source fluid/continuum simulation code. There are two versions installed: the Foundation version (openfoam) and the ESI version (openfoam-esi).

An example of how to use OpenFOAM with the batch queue system is described below.

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

```
# $ -N openform

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

If you are using the ESI version of OpenFOAM, replace `module load` above with `module openfoam-esi`.

A detailed explanation is given below.

<https://openfoam.org/resources/>

<http://www.openfoam.com/documentation/>

7.3. Numerical calculation library for GPU

7.3.1. cuBLAS

cuBLAS is a BLAS (Basic Linear Algebra Subprograms) library that runs on GPUs.

How to use

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

When calling cuBLAS in a normal C program, it must be specified with `-l`, `-L`, or `-I` at compile time.

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

7.3.2. cuSPARSE

cuSPARSE is a library for sparse matrix calculations on NVIDIA GPUs.

How to use

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

When calling cuSPARSE in a normal C program, it must be specified with `-l`, `-L`, or `-I` at compile time.

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

7.3.3. cuFFT

cuFFT is a library for parallel FFT (Fast Fourier Transform) on NVIDIA GPUs.

How to use

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

If cuFFT is called during a normal C program, it must be specified with `-l`, `-L`, or `-I` at compile time.

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


7.4. Machine learning and big data analysis related software

7.4.1. CuDNN

CuDNN is a library for GPU-based Deep Neural Networks.

The usage of CuDNN is described below.

```
$ module load cuda cudnn
```

7.4.2. NCCL

NCCL is a collective communication library for multiple GPUs.

An example of how to use NCCL is shown below.

```
$ module load cuda nccl
```

7.4.3. PyTorch

PyTorch is an open source machine learning library available for machine learning in Python.

The installation procedure for PyTorch is described below.

```
$ python3 -m pip install --user torch
```



PyTorch is installed in the user environment.

7.4.4. TensorFlow

TensorFlow is an open source library for machine learning and AI using data flow graphs.

Instructions for installing TensorFlow are provided below.

```
$ python3 -m pip install --user tensorflow
```



TensorFlow is installed in the user environment.

A detailed description is provided below.

<https://www.tensorflow.org/>

7.4.5. DeePMD-kit

DeePMD-kit is a machine learning framework for MD. An example job script for DeePMD-kit is shown below.

7.4.5.1. 1 DeePMD-kit + LAMMPS

7.4.5.1.1. DEEPM-D-KIT LAMMPS 1 NODE

An example job script for DeePMD-kit + LAMMPS (1 node, 4 GPUs) is shown below.

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

```

#$ -l node_f=1
#$ -cwd

module purge
module load deepmd-kit lammps
module li 2>&1

# enable DeePMD-kit for lammps/2aug2023_u3
export LAMMPS_PLUGIN_PATH=$DEEPMKIT_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

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

```

7.4.5.1.2. DEEPMKIT LAMMPS 2 NODES

An example job script for DeePMD-kit + LAMMPS (2 nodes, 8 GPUs) is shown below.

```

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

module purge
module load deepmd-kit lammps
module li 2>&1

# enable DeePMD-kit
export LAMMPS_PLUGIN_PATH=$DEEPMKIT_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

cd ../01.train
mpirun -x PATH -x LD_LIBRARY_PATH -x PYTHONPATH -x NCCL_BUFFSIZE=1048576 -npernode 4 -np 8 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
mpirun -x PATH -x LD_LIBRARY_PATH -x PYTHONPATH -x LAMMPS_PLUGIN_PATH -npnnode 4 -np 8 lmp -i in.lammps
```

A detailed description is provided below. <https://docs.deepmodeling.com/projects/deepmd/en/master/index.html>

7.4.6. R

R is an interpreted programming language for data analysis and graphics.

Rmpi is installed for parallel processing and rpub for GPU.

Examples of how to use R are described below.

```
$ module load R
$ mpirun -np 2 Rscript test.R
```

7.4.7. Apache Hadoop

The Apache Hadoop software library is a framework for distributed processing of large data sets using a simple programming model.

Examples of how to use Apache Hadoop are described below.

```
$ module load hadoop
$ mkdir input
$ cp -p $HADOOP_HOME/etc/hadoop/*.xml input
$ hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar grep input output 'dfs[a-z.]+'
$ cat output/part-r-00000
1      dfsadmin
```

The following is the procedure for using the system in the case of a batch queue system.

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

module load hadoop
cd $TMPDIR
mkdir input
cp -p $HADOOP_HOME/etc/hadoop/*.xml input
hadoop jar $HADOOP_HOME/share/hadoop/mapreduce/hadoop-mapreduce-examples-3.3.6.jar grep input output 'dfs[a-z.]+'
cp -rp output $HOME
```

7.5. Visualization related software

7.5.1. POV-Ray

POV-Ray is a free ray-tracing software.

Examples of how POV-Ray can be used are described below.

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

A detailed description is provided below.

<http://www.povray.org/>

7.5.2. ParaView

ParaView is an open source, multi-platform data analysis and visualization application.

Examples of how ParaView can be used are described below.

```
$ module load paraview
$ paraview
```

7.5.2.1. For visualization with multiple GPUs

You can use `paraview/5.12.0`, `paraview/5.12.0-egl`, to visualize with multiple GPUs on multiple nodes. Note that `paraview/5.12.0-egl` does not include the `paraview` command.

Here is an example of using 8 GPUs with `node_f=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 node_f=2

module purge
module load paraview

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

Don't forget to give `wrap.sh` execute permission. (`chmod 755 wrap.sh`)

With the above job script

```
qsub -g <グループ名> job.sh
```

and submit the job.

Confirm that the job is flowing with `qstat`.

```
yyyyyyyy@login1:~$ qstat
job-ID      prior    name         user          state submit/start at   queue                jclass                               slots ja-task-ID
-----
xxxxxxx 0.55354 job.sh      yyyyyyy      r       05/31/2024 09:24:19 all.q@rXnY                               56
```

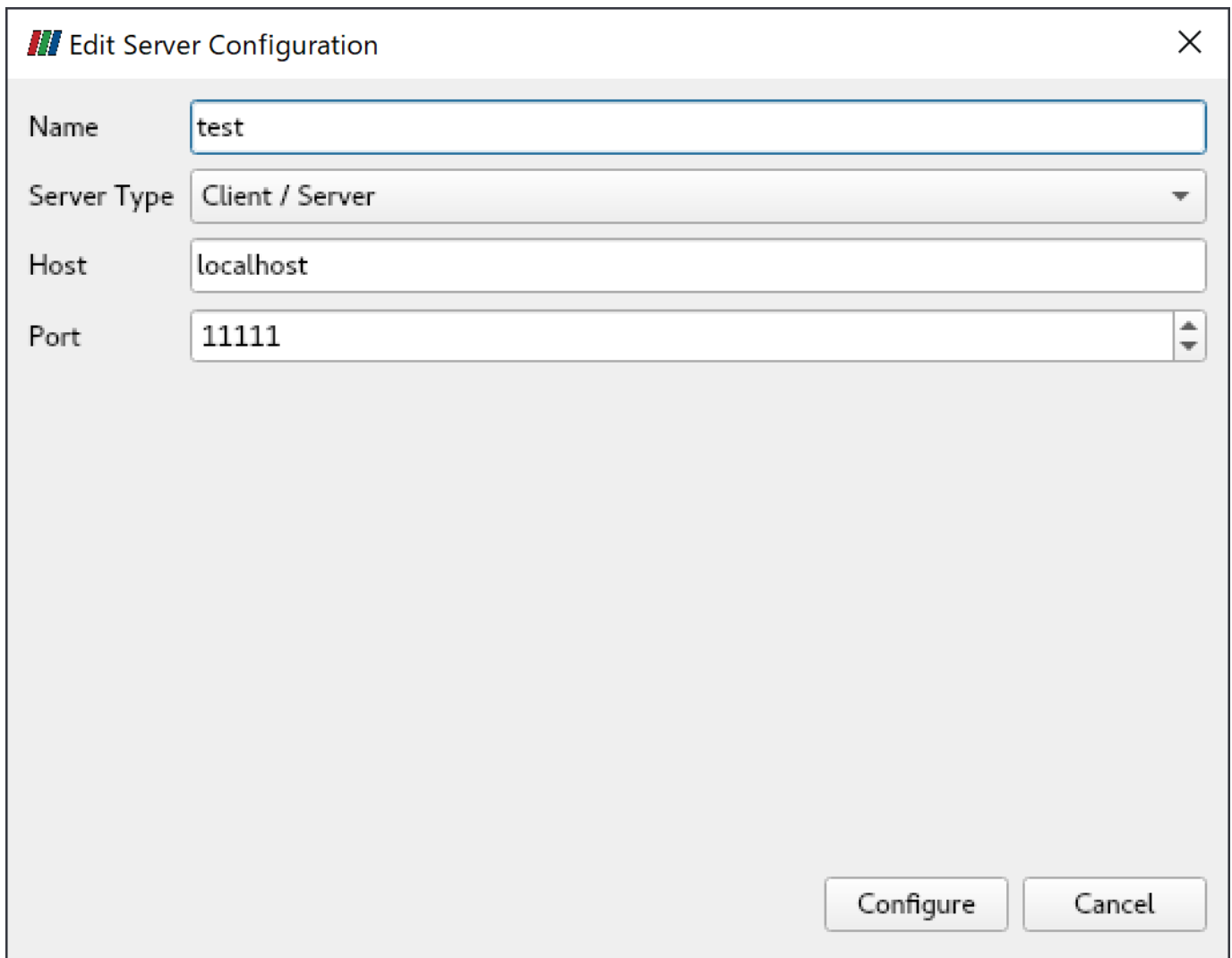
Ssh to the node where the job is flowing via X forwarding and start paraview.

```
yyyyyyyy@login1:~$ ssh -CY rXnY
yyyyyyyy@rXnY:~$ module load paraview
paraview
```

`turbovnc` can also be used.

After starting, click "File"-">"Connect" and click "Add Server".

Enter "Name" appropriately (in this case, "test") and click "Configure".



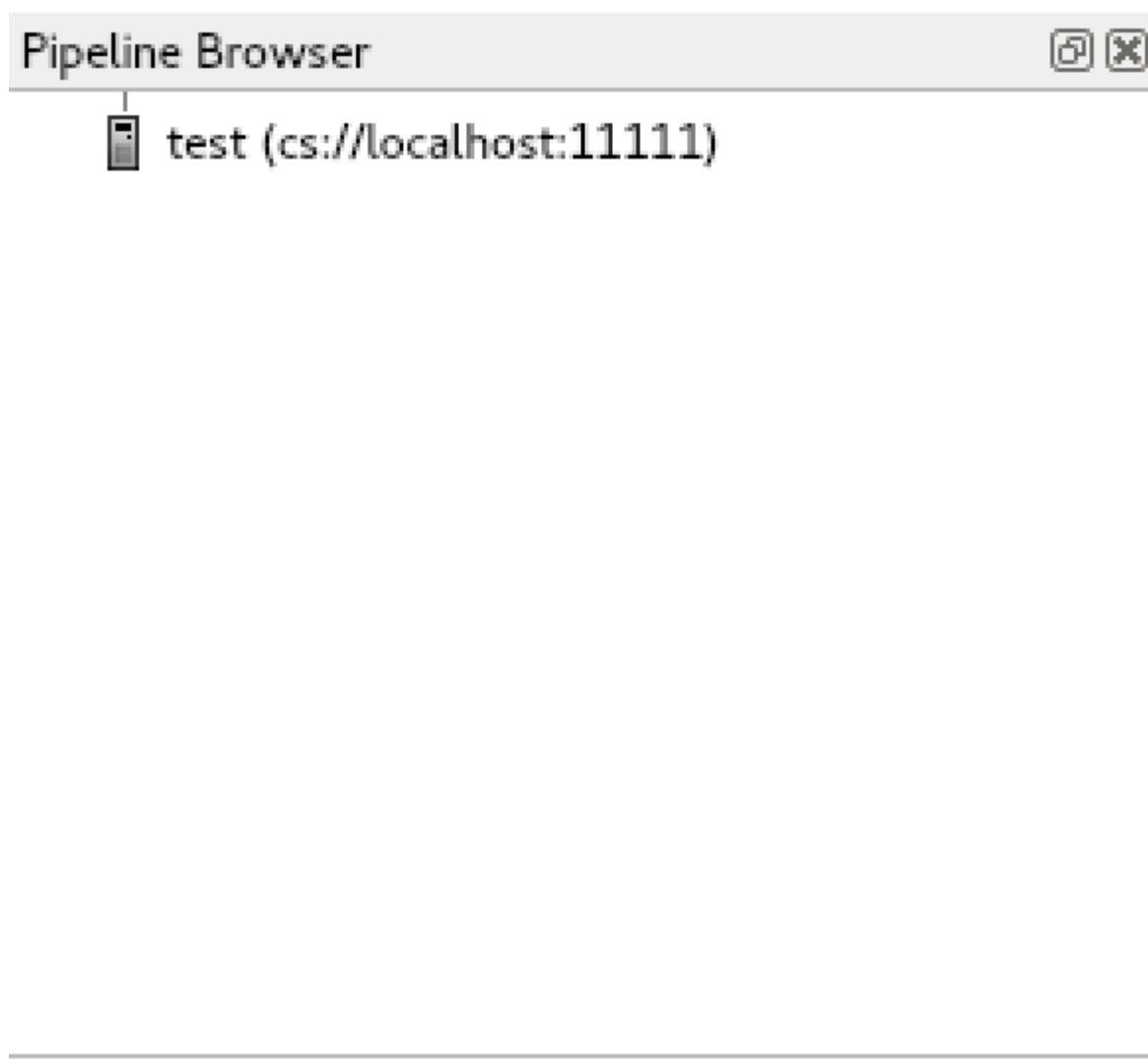
The image shows a dialog box titled "Edit Server Configuration" with a close button (X) in the top right corner. The dialog contains four input fields: "Name" with the value "test", "Server Type" with a dropdown menu showing "Client / Server", "Host" with the value "localhost", and "Port" with the value "11111". At the bottom right, there are two buttons: "Configure" and "Cancel".

Name	test
Server Type	Client / Server
Host	localhost
Port	11111

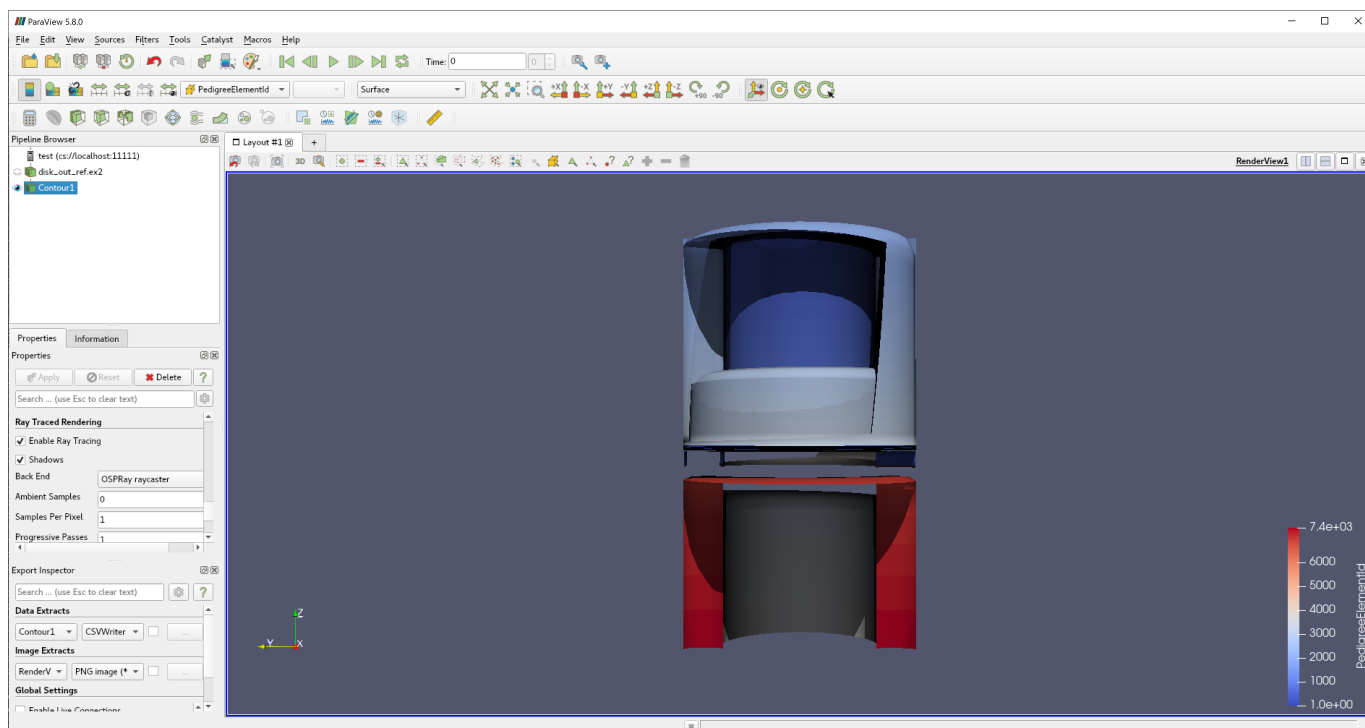
Configure Cancel

Then click "Connect".

Once connected, `test(cs://localhost:11111)` will appear in the "Pipeline Browser" field.



The sample data of paraview can be downloaded from [here](#).



A detailed description is provided below.

<https://www.paraview.org/>

7.5.3. VisIt

VisIt is an open source visualization application.

Examples of how VisIt can be used are described below.

```
$ module visit
$ visit
```

A detailed description is provided below.

<https://wci.llnl.gov/simulation/computer-codes/visit/>

7.6. Other freeware

7.6.1. turbovnc

turbovnc is open source VNC software. An example of how to use turbovnc is shown below.

Please allocate a computation node with qrsh and execute on the computation node.

- Get compute nodes

```
$ qrsh -g <TSUBAME group> -l <Resource type>=<number> -l h_rt=<Maximum run time>
```

- Execute the following on the secured compute node and start vncserver

```
$ module load turbovnc
$ vncserver -xstartup xfce.sh

You will require a password to access your desktops.

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

```
Desktop 'TurboVNC: rXnY:1 ()' started on display rXnY:1 # <-- Make sure to memorize 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/rXnYnZ:1.log
```

If you want to increase the screen size, specify the size as `vncserver -geometry <WIDTH>x<HEIGHT>.`

- Then download the installer for your own PC from <https://sourceforge.net/projects/turbovnc/files/>, Install turbovnc viewer
- From the terminal software connected to the compute node, configure SSH port forwarding to forward local port 5901 to port 5901 on the compute node (if the display number is rXnYnZ:n, the port number for port forwarding is set to 5900+n).
- Start turbovnc viewer from your own PC, connect to localhost:5901 and enter the password you set.



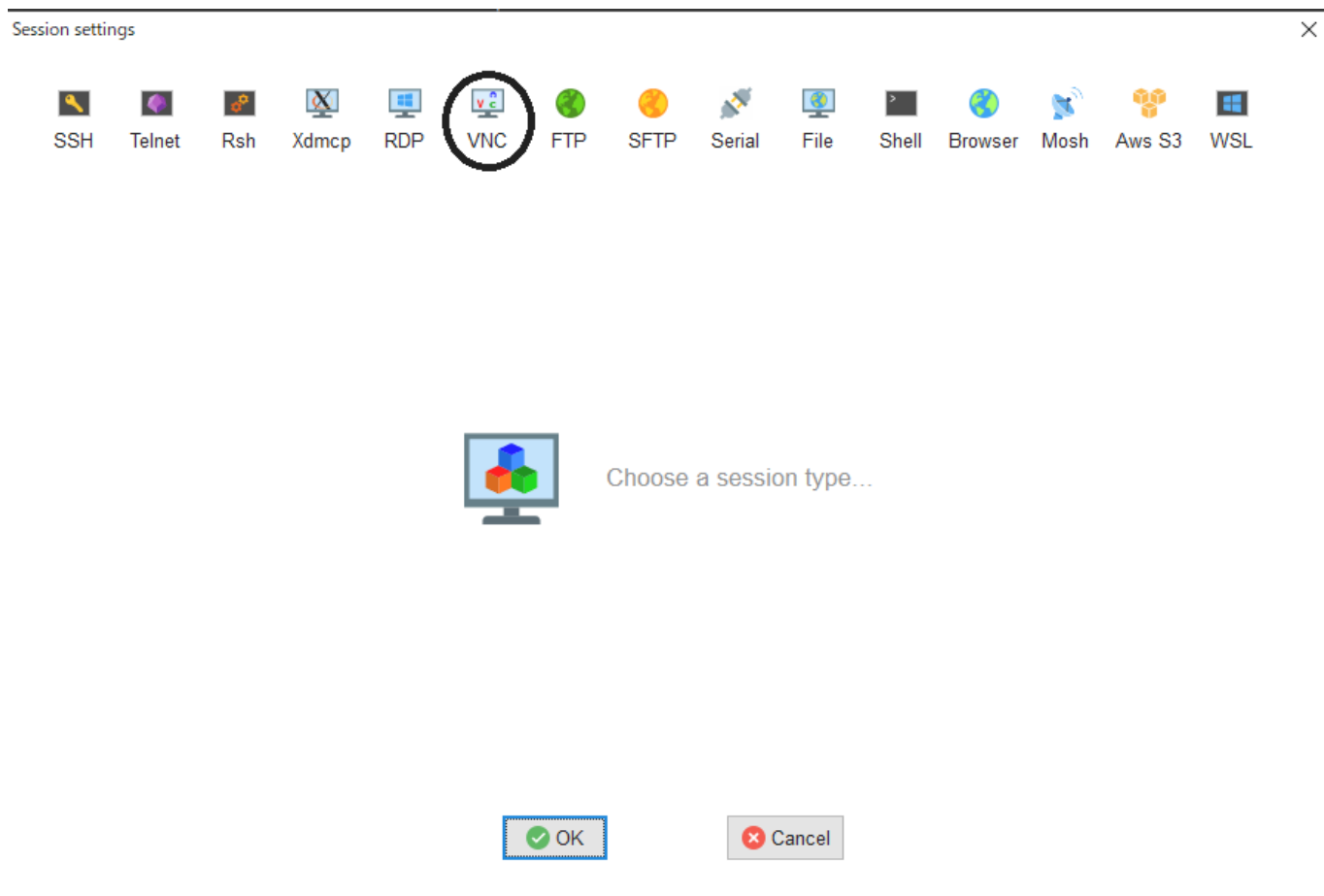
Tips

The VNC display number counts up each time vncserver is started; check the port number in the SSH port forwarding configuration each time.

7.6.1.1. How to use VNC client from MobaXterm

MobaXterm has a built-in VNC client, so you can use VNC connections without installing a VNC client.

- After securing the node with qrsh, select "Sessions"->"New session"->"VNC" from MobaXterm.



- Then, in "Basic Vnc settings", enter the hostname of the secured compute node in "Remote hostname or IP address" and 5900+n in "Port", click "Connect through SSH gateway(jump host)" in "Network settings", enter login.t3.gsic.titech.ac.jp in "Gateway SSH server", and click "Connect through SSH server" in "Network settings". Enter login.t3.gsic.titech.ac.jp in "Gateway SSH server", leave "Port" as 22, enter your TSUBAME login name in "User", check "Use private key" and enter your Check "Use private key" and enter your private key.

Session settings ✕

SSH Telnet Rsh Xdmcp RDP **VNC** FTP SFTP Serial File Shell Browser Mosh Aws S3 WSL

Basic Vnc settings


Remote hostname or IP address * Port

Advanced Vnc settings **Network settings** **Bookmark settings**

☒ Connect through SSH gateway (jump host)

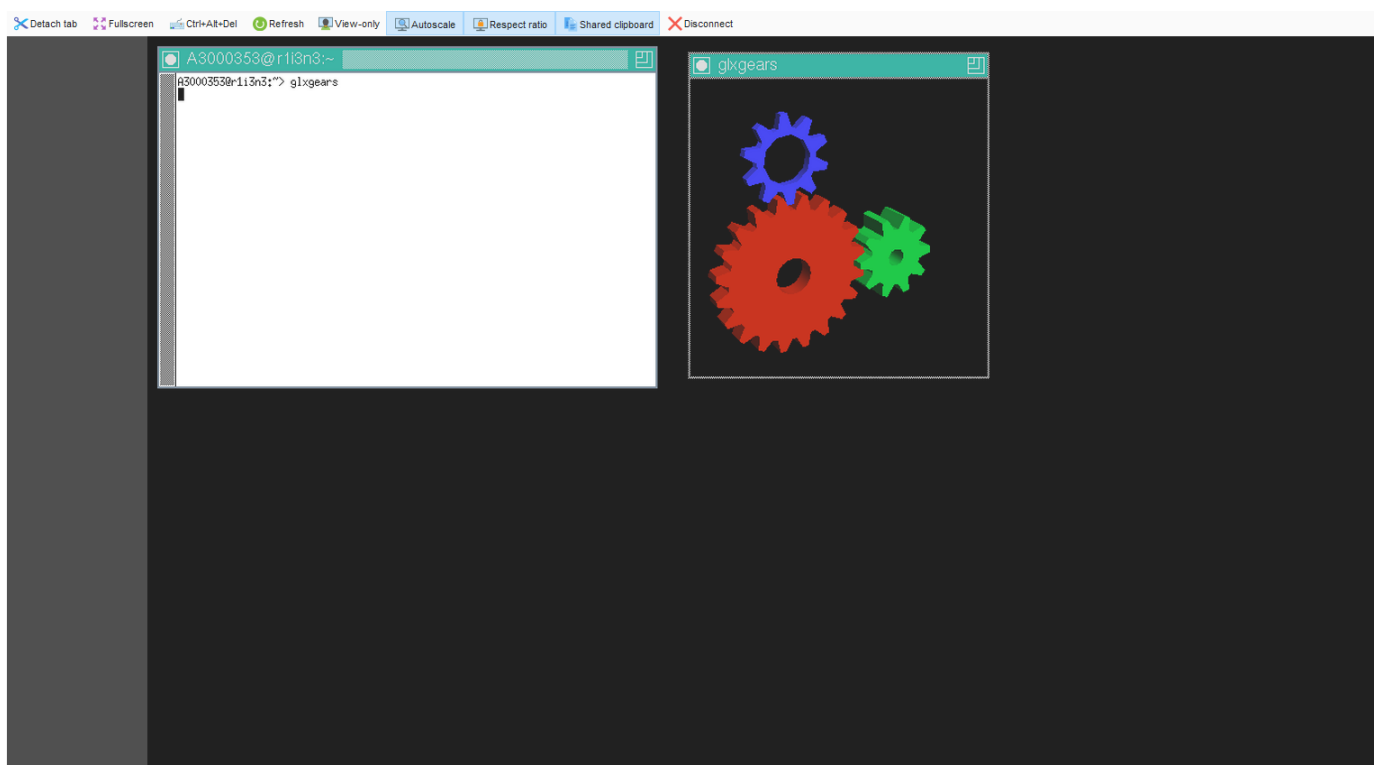
Gateway SSH server Port User

☒ Use private key



☒ OK ☐ Cancel

Click OK to start the VNC client.



7.6.1.2. turbovnc + VirtualGL

If you are using a resource type (node_f, node_h, node_q, gpu_1) that allocates one or more GPUs when using turbovnc, you can use VirtualGL to visualize using GPUs. As an example, the use of VirtualGL in the case of gpu_1 is shown below.

```
$ qrsh ... -l gpu_1=1
$ module load turbovnc
$ vncserver -xstartup xfce.sh
```

- Connect with a VNC client and perform the following

```
$ vglrun -d /dev/dri/card<N> <OpenGL application>
```

<N> is a number from 1 to 4, corresponding to GPU 0 to 3. The correspondence between the GPU number and card<N> is as follows

Device name	GPU number
/dev/dri/card1	GPU1(64:00)
/dev/dri/card2	GPU0(04:00)
/dev/dri/card3	GPU3(E4:00)
/dev/dri/card4	GPU2(84:00)

7.6.2. gnuplot

gnuplot is a command line interactive graph drawing program.

In addition to the standard features, it is built to support X11, latex, PDFlib-lite and Qt4.

Examples of how to use gnuplot are given below.

```
$ gnuplot
```

7.6.3. Tgif

tgif is an open source drawing tool.

The following is a description of how to use tgif.

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

If you found an error saying Cannot open the Default(Msg) Font '-*-courier-medium-r-normal-*-*-*-*-*iso8859-1', 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

GIMP is an open source image manipulation program.

Examples of how to use GIMP are described below.

```
$ gimp
```

7.6.5. ImageMagick

ImageMagick is an image processing tool.

In addition to the standard features, it is built to support X11, HDRI, libwmf, and jpeg.

An example of how to use ImageMagick is shown below.

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

7.6.6. Tex Live

TeX Live is an integrated TeX package.

An example of how to use TeX Live is shown below.

```
$ lualatex test.tex
```



A PDF file will be created.

7.6.7. Java SDK

Open JDK 1.8 is installed as the Java SDK.

An example of how to use the Java SDK is shown below.

```
$ javac Test.java
$ java Test
```

7.6.8. PETSc

PETSc is an open source parallel numerical library. It can perform linear equation solving, etc.

Two versions, one for real numbers and the other for complex numbers, are installed.

Examples of how to use PETSc are described below.

```
$ module load petsc/3.20.4-real      ← real number usage
or
$ module load petsc/3.20.4-complex  ← for complex numbers
$ mpiifort test.F -lpetsc
```

7.6.9. FFTW

FFTW is an open source library for Fast Fourier Transform.

Since the FFTW 2x and 3x series are incompatible, two versions, Series 2 and Series 3, are installed.

Examples of how to use FFTW are described below.

```
$ module load fftw/3.3.10-intel intel-mpi/2021.11  ← Intel MPI
or
$ module load fftw/3.3.10-gcc      ← Open MPI
$ gfortran test.f90 -lfftw3
```

7.6.10. Apptainer

Apptainer is a Linux container for HPC.

An example of how to use Apptainer is shown below. Use the latest Ubuntu Docker image.

- Image file: ubuntu_latest.sif
- -nv : with GPU
- -B : mout filesystem

```
$ mkdir $HOME/apptainer
$ cd $HOME/apptainer
$ apptainer pull docker://ubuntu:latest
$ apptainer shell --nv -B /gs -B /apps -B /home ubuntu_latest.sif
Apptainer> uname -a
Linux r1n11 5.14.0-362.18.1.el9_3.x86_64 #1 SMP PREEMPT_DYNAMIC Wed Jan 3 15:54:45 EST 2024 x86_64 x86_64 x86_64 GNU/Linux
Apptainer> cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.4 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.4 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
Apptainer>
```

7.6.11. Alphafold

Alphafold is a protein structure prediction program using machine learning. An example of using Alphafold is shown below.

- Initialization (login node or compute node)

```
module purge
module load alphafold

cp -pr $ALPHAFOLD_DIR .
cd alphafold
git pull # Update latest version
# If you want to use a specific version (in this case v2.3.3), add the following
# git checkout -b v2.3.3 v2.3.3
```

- At runtime (example job script for alphafold/2.3.2)

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

module purge
module load alphafold
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
```

- At runtime (example job script for alphafold/2.1.1)

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

module purge
module load alphafold
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
```

Due to the large size of the database files, please avoid **individual downloads** whenever possible.

For a detailed description of Alphafold, please see below. <https://github.com/deepmind/alphafold>

7.6.12. miniconda

miniconda is a python virtual environment creation software. An example of using miniconda is shown below.

```
module load miniconda
eval "$(/apps/t4/rhel9/free/miniconda/24.1.2/bin/conda shell.bash hook)"
conda create -n test
conda activate test
```

For a detailed description of miniconda, please see below. <https://docs.anaconda.com/free/miniconda/index.html>

7.6.13. spack

spack is a package manager for HPC. An example of using spack is shown below.

```
module load spack
spack install tree
```

For more information on spack, please see below. <https://spack.io/>

Appendix. Comparison of old and new TSUBAME

Appendix.1. Architecture

	TSUBAME3.0	TSUBAME4.0
Theoretical peak double precision performance	12.5PFLOPS	66.8PFLOPS
Theoretical peak half precision performance	47.2PFLOPS	952PFLOPS
Total main memory	135TiB	180TiB
Total HDD capacity	15.9PB	44.2PB
Total SSD capacity		327TB
Compute nodes	540	240
Total cores	15,120	23,040
Total GPUs	2,160	960
Interconnect	Intel Omni-Path HFI 100Gbps	InfiniBand NDR200 200Gbps
Interenet	SINET5 100Gbps	SINET6 100Gbps

Appendix.2. Compute nodes

	TSUBAME3.0	TSUBAME4.0
Computing Unit	Compute node HPE SGI ICE-XA 540 nodes	Compute node HPE Cray XD665 240 nodes
Components (per node)		
CPU	Intel Xeon E5-2680 v4 2.4GHz x 2 Socket	AMD EPYC 9654 2.4GHz x 2 Socket
Cores/Threads	14cores / 28threads x 2CPU	96cores / 192threads x 2CPU
Memory	256GiB	768GiB (DDR5-4800)
GPU	NVIDIA TESLA P100 for NVlink-Optimized Servers x 4	NVIDIA H100 SXM5 94GB HBM2e x 4
SSD	2TB	1.92TB NVMe U.2 SSD
Interconnect	Intel Omni-Path HFI 100Gbps x 4	InfiniBand NDR200 200Gbps x 4

Appendix.3. Software

Appendix.3.1 System software

	TSUBAME3.0 (as of 2023.4.6)	TSUBAME4.0
OS	SUSE Linux Enterprise Server 12 SP5	RedHat Enterprise Linux Server 9.3
Job scheduler	Univa Grid Engine 8.6.11	Altair Grid Engine 2023.1.1
Compilers	GCC 4.8.5, 12.2.0 Intel Compiler 23.0.0 NVIDIA HPC SDK 22.2	GCC 11.4.1 Intel oneAPI compiler 2024.0 and MKL NVIDIA HPC SDK 24.1 AOCC 4.1.0
MPI	Intel MPI 21.8.0 SGI MPT 2.16 OpenMPI 3.1.4	Intel MPI 2021.11 OpeMPI 5.0.2
CUDA library	11.0.3 (12.1.0)	12.3.2
CUDA driver	450.172.01	545.23.08
OmniPath driver (OPA)	10.10.3.1.1	
InfiniBand driver (OFED)		23.10-1.1.9

Appendix.3.2 Commercial application

Software	Description	TSUBAME3.0	TSUBAME4.0
ANSYS	Analysis Software	○	○
ABAQUS	Analysis Software	○	○
ABACUS CAE	Analysis Software	○	○
MSC Nastran	Analysis Software	○	
MSC Patran	Analysis Software	○	
Gaussian	Quantum chemistry calculation program	○	○
GaussView	Quantum chemistry calculation program Pre-Post tool	○	○
AMBER	Molecular dynamics calculation program	○	○
Materials Studio	Chemical Simulation Software	○	○
Discovery Studio	Chemical Simulation Software	○	○
Mathematica	Mathematical Processing Software	○	○
Maple	Mathematical Processing Software	○	
AVS/Express	Visualization	○	
AVS/Express PCE	Visualization	○	
LS-DYNA	Analysis Software	○	○ Included in ANSYS
COMSOL	Analysis Software	○	○
Schrodinger	Chemical Simulation Software	○	○
MATLAB	Numerical calculation software	○	○
VASP	Quantum molecular dynamics calculation program	○	○
Arm Forge	Debugger	○	○
Intel Compiler	Compiler	○	○ oneAPI
PGI Compiler	Compiler	○	○ NVIDIA HPC SDK

Appendix.3.3 Freeware

Software	Description	TSUBAME3.0	TSUBAME4.0
GAMESS	Solver Simulator	○	○
Tinker	Solver Simulator	○	○
GROMACS	Solver Simulator	○	○
LAMMPS	Solver Simulator	○	○
NAMMD	Solver Simulator	○	○
QUANTUM ESPRESSO	Solver Simulator	○	○
CP2K	Solver Simulator	○	○
OpenFOAM	Solver Simulator, Visualization	○	○
CuDNN	GPU library	○	○
NCCL	GPU library	○	○
Caffe	DeepLearning framework	○	
Chainer	DeepLearning framework	○	
TensorFlow	DeepLearning framework	○	○
DeePMD-kit	MD DeepLearning framework	○	○
R	Interpreter (Rmpi, rpud)	○	○
clang	Compiler	○	○ AOCC
Apache Hadoop	Distributed Data Processing Tools	○	○
POV-Ray	Visualization	○	○
ParaView	Visualization	○	○
VisIt	Visualization	○	○
turbovnc	Remote GUI(X11)	○	○
gnuplot	Data grapher/visualization	○	○
Tgif	Image Viewing and Editing	○	○
GIMP	Image Viewing and Editing	○	○
ImageMagick	Image Viewing and Editing	○	○
TeX Live	TeX distribution	○	○
Java SDK	Development tool	○	○
PETSc	Linear system solvers, libraries	○	○
FFTW	Fast Fourier Transform Library	○	○
DMTCP	Checkpoint Restart	○	○
Singularity	Linux container for HPC	○	○ Apptainer
Open OnDemand	Web portal site for HPC		○

Appendx.4. Storage

TSUBAME3.0	Storage	Mount point	Capacity	Filesystem
	Home directory	/home	40TB	GPFS+cNFS
	Shared application deployment (HDD)	/apps		
	Hight-speed storage area 1 (HDD)	/gs/hs0	4.8PB	Lustre
	Hight-speed storage area 2 (HDD)	/gs/hs1	4.8PB	Lustre
	Hight-speed storage area 3 (HDD)	/gs/hs2	4.8PB	Lustre
	Local scratch area (SSD)	/scr	1.9TB/node	xfs
TSUBAME4.0	Storage	Mount point	Capacity	Filesystem
	High-speed storage area	/gs/fs	372TB	Lustre
	Home directory (SSD)	/home		
	Large-scale (Big) storage area	/gs/bs	44.2PB	Lustre
	Shared application deployment (HDD)	/apps		
	Local scratch area (SSD)	/local	1.92TB/node	xfs

Revision History

Date	Change
2024/04/21	First edition