This document is for the users that use high performance computing resources in UNIST Supercomputing Center.

It starts on page 17 for the foreign researchers.

UNIST HPC-Tachyon System User Guide

1. HPC Resources

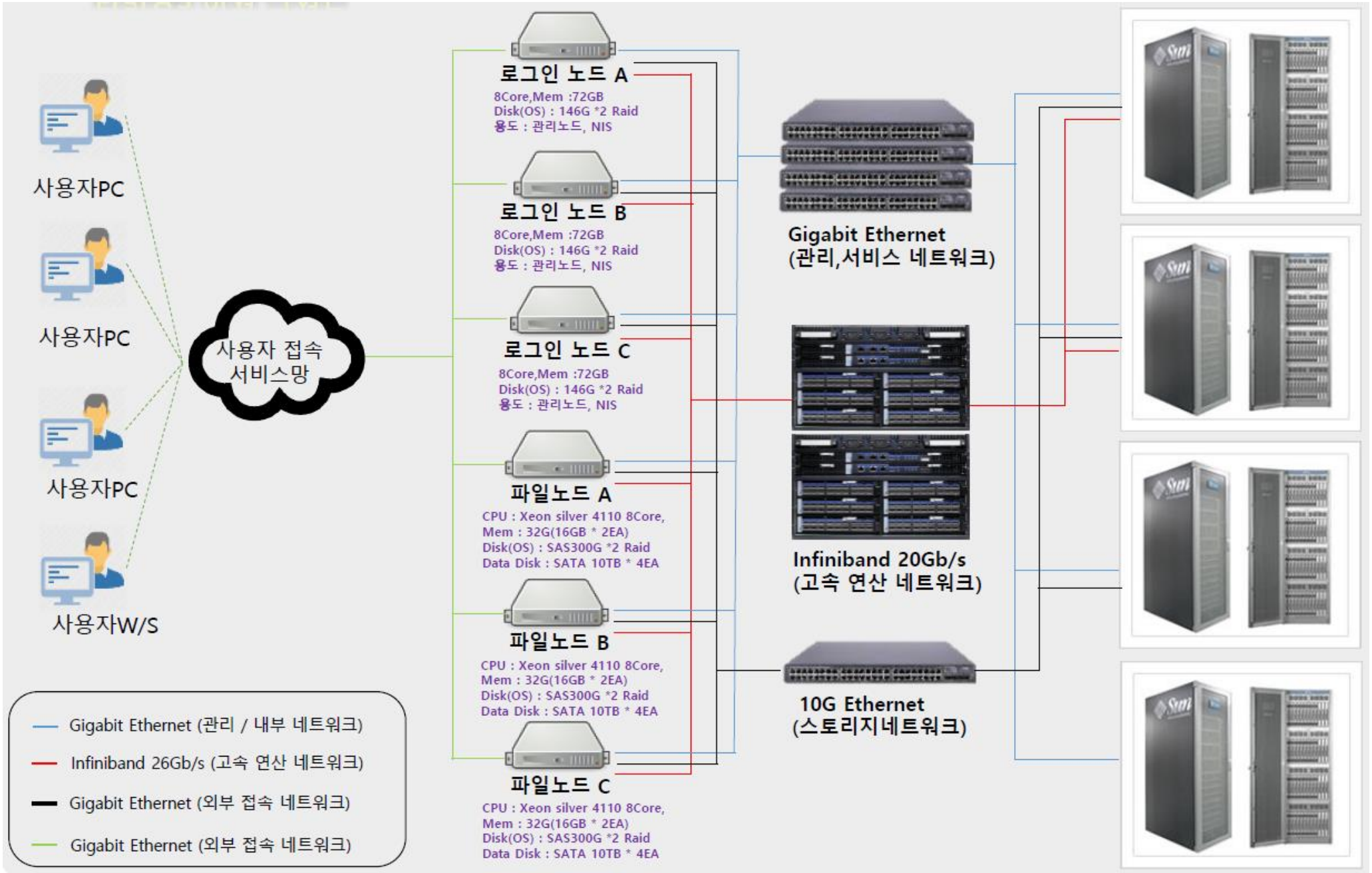
A. Hardware

i. Configuration Overview

UNIST Supercomputing Center(이하 USC)에서는 다양한 연구 지원을 위해 High Performance Computing(이하 HPC) system 과 병렬 파일 시스템을 지원합니다. USC에서는 HPC x86_64 Linux 클러스터로 Tachyon, Leopard, Lion, Falcon, Eagle이 있으며 파일 시스템으로 Lustre parallel filesystem을 보유하고 있습니다.

Tachyon 시스템은 563개의 계산 노드로 구성되어 있고, 각 node 당 8개의 코어를 장착하고 있습니다. CPU architecture는 Nehalem입니다. Tachyon HPC 시스템들의 총 코어의 수는 4,504 코어이고, 각 계산 노드는 24GB 또는 48GB의 메모리가 꼽혀 있습니다. 각 노드 간 네트워크는 521노드(4,168 코어)가 IB 4X QDR(40Gbps)로, 42노드(336코어)가 1G 이더넷으로 글로벌 공유스토리지에 NFS를 통해 1G Gigabit Ethernet로 구성되어 있습니다.

자세한 HPC system 구성은 다음 그림을 통해 확인할 수 있습니다.


사용자PC
사용자PC
사용자PC
사용자W/S
사용자 접속
서비스망
로그인 노드 A
8Core,Mem :72GB
Disk(OS) : 146G *2 Raid
용도 : 관리노드, NIS
로그인 노드 B
8Core,Mem :72GB
Disk(OS) : 146G *2 Raid
용도 : 관리노드, NIS
로그인 노드 C
8Core,Mem :72GB
Disk(OS) : 146G *2 Raid
용도 : 관리노드, NIS
파일노드 A
CPU : Xeon silver 4110 8Core,
Mem : 32G(16GB * 2EA)
Disk(OS) : SAS300G *2 Raid
Data Disk : SATA 10TB * 4EA
파일노드 B
CPU : Xeon silver 4110 8Core,
Mem : 32G(16GB * 2EA)
Disk(OS) : SAS300G *2 Raid
Data Disk : SATA 10TB * 4EA
파일노드 C
CPU : Xeon silver 4110 8Core,
Mem : 32G(16GB * 2EA)
Disk(OS) : SAS300G *2 Raid
Data Disk : SATA 10TB * 4EA
Gigabit Ethernet
(관리,서비스 네트워크)
Infiniband 20Gb/s
(고속 연산 네트워크)
10G Ethernet
(스토리지네트워크)
Gigabit Ethernet (관리 / 내부 네트워크)
Infiniband 26Gb/s (고속 연산 네트워크)
Gigabit Ethernet (외부 접속 네트워크)
Gigabit Ethernet (외부 접속 네트워크)

ii. How to access HPC systems

HPC-Tachyon시스템에는 2개의 login node가 있습니다. 사용자는 Login node를 통해서 HPC 시스템에 접근할 수 있습니다. 사용자 PC와 login node 간 연결은 SSH protocol(2022번 포트) 을 통해 이루어지고, 이것을 가능하게 해 주는 터미널 프로그램(PuTTY, Secure Shell Client, Xmanager 등)을 이용하시면 됩니다.

Hostname	login02	login03
DNS	**Tlogin02.usc.unist.ac.kr**	**Tlogin03.usc.unist.ac.kr**

※ UNIST supercomputing center에 설치된 시스템은 UNIST 외부에서 접속이 불가합니다. (특수한 경우, 허용된 IP에 의해서만 접근 허용)

iii. Disk quota

HPC-Tachyon의 스토리지는 NFS를 통해 사용자의 홈디렉토리가 전체 노드에서 공유됩니다. Tachyon 시스템에는 스크래치 디렉토리가 없습니다. 할당받은 홈디렉토리 공간에서 자유롭게 작업을 하시면 됩니다.

디렉토리 이름	용량
/home1	22 TB
/home2	22 TB
/home3	22 TB

iv. HPC-Tachyon

1) System Overview

A) Infiniband Nodes

Hostname	com[001] – com[528]
Number of node(# of core)	522node (4,176 cores), 8cores/node
Processor	Intel Xeon X5570 2.93GHz
Memory	24GB or 48GB per node
OS	Linux CentOS 6.9
Interconnection	Infiniband(40G) network between computing nodes 1G Ethernet network to the NFS file system
Storage	/home1, /home2, /home3 in NFS file system

B) Ethernet network Nodes

Hostname	com[529] – com[624]
Number of node(# of core)	42node (336 cores), 8cores/node
Processor	Intel Xeon X5570 2.93GHz
Memory	24GB or 48GB per node
OS	Linux CentOS 6.9
Interconnection	1G Ehternet network between computing nodes 1G Ethernet network to the NFS file system
Storage	/home1, /home2, /home3 in NFS file system

2) Job Queues and Pes

A) Job Queues and PEs

i) Queues

queue name	wall clock time	available cores	available PEs	Remarks
1g.q	168 hours	Up to group quota	make	For Ethernet Nodes
ib.q	168 hours	Up to group quota	make, mpi_2 ~ mpi_8	For Infiniband Nodes

ii) Parallel Environment(PE)s

PE의 사용 문법은 pe_name [digit]이며 [digit]는 계산에 사용할 전체 코어 수를 의미합니다. 자세한 내용은 매뉴얼 후반부의 작업 스크립트 예시를 참고하세요.

iii) Resource quota

사용 가능한 코어의 Quota는 연구그룹별로 주어지며, ib.q를 사용하는 계산노드에서는 그룹별 3,000 코어를 '동시 사용'이 가능합니다.

Job status가 qw(queue wait)상태일 시 resource를 얼마나 사용하고 있는지 확인하시기 바랍니다.

B. Software

i. Compilers

Compilers	Version
GNU(GCC)	GCC 4.4.7(Linux CentOS 6.9 Default)
	GCC 7.4.0
Intel compiler	Intel composer xe 2016u4
	Intel composer xe 2019u4

ii. Libraries

Classification	Libraries	Libraries built with follow packages.
Parallel libraries	OpenMPI 3.1.4	GCC 4.4.7(Linux CentOS 6.9 Default) GCC 7.4.0 Intel composer xe 2016u4
	Intel-impi xe2016u4	Intel composer xe 2016u4
	Intel-impi xe2019u4	Intel composer xe 2019u4
Mathematical libraries	FFTW 3.3.8	gcc 4.4.7(Linux CentOS 6.9 Default)
	Intel MKL xe2016u4	Intel composer xe 2016u4
	Intel MKL xe2019u4	Intel composer xe 2016u4

iii. Computational Packages

Classification	Applications	Installed Version(set up compiles)
Commercial	ABAQUS	abaqus 2019
	ANSYS CFD, Fluent	ansys 2019 R3
	COMSOL Multiphysics 5.0	comsol 54
	VASP	vasp 5.4.1 (intel 2019u4 –impi)
	Gaussian 09	will be installed soon
Open Source	GROMACS	will be installed soon
	Quantum Espresso	will be installed soon
	LAMMPS	31Mar2017
	R Source	R 3.6.1 (intel 2019u4-mkl)

※ NOTICE : 사용자가 연구를 위해 상용 package 사용을 원하는 경우 불법 license사용 방지를 위해 사용 전 연락 주시기 바랍니다.

C. Environment Settings

i. Overview

설치된 모든 software의 사용 편의를 위해 Environment Module이 설정되어 있습니다. 사용자들은 프로그램 동작을 위해 HPC에서 환경 설정 자주 사용하는데 기존의 방법대로 설정하자면 실행파일, 경로, 라이브러리 경로 등을 추가해줘야 하는데 UNIX를 자주 쓰던 사용자는 익숙할 수 있지만 처음 사용하는 사용자에게는 번거로운 작업입니다. 때문에 저희 HPC system에서는 이와 같은 수고스러움을 방지하고자 module system을 사용하고 있습니다. Module file을 생성해 놓아서 필요한 module을 "load"하거나 "unload"하면 됩니다. 추가로 사용자가 필요로 할 시 ".file"을 수정하여 개별 module file을 변경할 수 있습니다.

module 을 사용하게 되면 경로, 라이브러리 경로, MAN경로 등을 업데이트 할 때 쉽게 변경할 수 있습니다.

HPC system에서 계정이 생성되었을 때 ".file"을 사용하여 user environment를 구성 및 변경해야 되지만(default는 bash shell), module을 사용하면 좀더 쉽게 user environment를 설정할 수 있습니다.

1) Module commands

아래는 module에 대한 command입니다.

```
$ module help
```

"help" 옵션은 아래와 같이 출력됩니다. (help를 입력하지 않아도 동일하게 출력됩니다.)

```
Available SubCommands and Args:
 + add|load            modulefile [modulefile ...]
 + rm|unload           modulefile [modulefile ...]
 + switch|swap          [modulefile1] modulefile2
```

```
+ display|show     modulefile [modulefile ...]
+ avail           [modulefile [modulefile ...]]
+ purge
+ list
```

$ module list

현재 modulefile 을 출력합니다

$ module avail

사용할 수 있는 모든 modulefiled 을 출력합니다.

$ module purge

현재 load 된 modulefile 을 모두 unload 합니다.

$ module load *modulefile*

"*modulefile*"을 load 합니다.

$ module unload *modulefile*

"*modulefile*"을 unload 합니다.

$ module switch modulefile_old modulefile_new

"*modulefile_old*"을 "*modulefile_new*"로 변경 합니다.

$ module show *modulefile*

"*modulefile*"을 PATH, LD_LIBRARY_PATH, MANPATH, etc 로 변경하는 방법을 출력합니다.

D. Job Submission

i. Overview

USC에서는 사용자가 code를 실행하기 위해서 queue system을 사용해야 합니다. 저희 USC에서는 사용자의 효과적인 사용 및 편의를 위해 SGE(Sun Grid Engine)를 사용하여 job을 관리 합니다.

ii. SGE commands

Commands	Example	Description
qsub	qsub job_script_file_name	submit a job.
qstat	qstat	show job status oneself
	qstat -u '*'	show job status for all users
qhost	qhost	show computing node status
qdel	qdel job_ID qdel -u user_ID (user's all job cancelled)	cancel a job
qconf	qconf -sql	show all queue list
	qconf -spl	show all pe list
	qconf –sq queuename	show about "queuename" detailed
	qconf -sul	show all user list
	qconf -srqs	show resource quota policy
	qconf -shgrpl	show host group list
	qconf -sc	show complex attributes

iii. 시스템별 작업 스크립트 예제

1) Parallel Job

```
#!/bin/bash
#$ -V                          # job submit node의 shell 환경변수를 computing node에도 적용(default)
#$ -pe pe_name(make 또는 mpi_2 ~ mpi_8 중 선택) 8 # parallel environment(pe)로 설정, 계산에 사용하는 전체 코어 수
입력
#$ -N parallel_job             # Job name. 명시하지 않으면 job_script이름을 가져옵니다.
#$ -q queue_name(ib.q 또는 1g.q) # 사용할 queue name
#$ -S /bin/bash                # shell selection users want to use
#$ -cwd                        # 현재 directory를 작업 directory로 사용
#$ -l h_rt=24:00:00            # 작업 경과시간(hh:mm:ss)(wall time clock).
                               # 미 기입 시 작업이 강제 종료 됩니다.
                               # job이 wall time clock에 도달 시 자동으로 중지됩니다.

## Intel MPI를 사용하는 경우 아래 3개 라인의 코멘트표시(#)를 해제하세요.
#export I_MPI_FABRICS=shm:dapl
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load mpi/intel-xe2016/ompi-3.1.4    # ex) in case of OpenMPI 3.1.4
mpirun -machinefile $TMPDIR/machines -np $NSLOTS ./execution_file
```

iv. 어플리케이션(프로그램)별 작업 스크립트 예제

1) ANSYS CFD (FLUENT)

```
#!/bin/bash
#$ -V
#$ -pe pe_name(make 또는 mpi_2 ~ mpi_8 중 선택) 8 # parallel environment(pe)로 설정, 계산에 사용하는 전체 코어 수
```

```
입력
#$ -N MXTTA
#$ -q ib.q
#$ -S /bin/bash
#$ -cwd
#$ -l h_rt=24:00:00

## Intel MPI를 사용하는 경우 아래 3개 라인의 코멘트표시(#)를 해제하세요.
#export I_MPI_FABRICS=shm:dapl
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load ansys/2019

INPUT_FILE=MyFluentJobInputFileName

fluent 3ddp -t$NSLOTS -g -cnf=$TMPDIR/machines -sge -pinfiniband -mpi=openmpi -i $INPUT_FILE >
output_log
```

2) COMSOL Multiphysics

```
#!/bin/bash
#$ -V
#$ -pe pe_name(make 또는 mpi_2 ~ mpi_8 중 선택) 8 # parallel environment(pe)로 설정, 계산에 사용하는 전체 코어 수 입력
#$ -N CMSXMPL
#$ -q ib.q
#$ -S /bin/bash
#$ -cwd
#$ -l h_rt=24:00:00

## Intel MPI를 사용하는 경우 아래 3개 라인의 코멘트표시(#)를 해제하세요.
#export I_MPI_FABRICS=shm:dapl
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load comsol/54

NUM_CORE=8                                  ## Tachyon System per node's core is 8
NUM_NODE=`expr $NSLOTS / $NUM_CORE`     ## Total number of compute nodes

comsol batch -nn $NUM_NODE -np $NUM_CORE -f $TMPDIR/machines -inputfile MyInputFileName -outputfile MyOutputFileName -batchlog MyLogFileName -tmpdir /uwork/p0xxxxx/
```

3) ABAQUS

```
#!/bin/bash
#$ -V
#$ -pe pe_name(make 또는 mpi_2 ~ mpi_8 중 선택) 8 # parallel environment(pe)로 설정, 계산에 사용하는 전체 코어 수 입력
#$ -N CMSXMPL
#$ -q ib.q
#$ -S /bin/bash
```

```
#$ -cwd
#$ -l h_rt=24:00:00

## Intel MPI를 사용하는 경우 아래 3개 라인의 코멘트표시(#)를 해제하세요.
#export I_MPI_FABRICS=shm:dapl
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load abaqus/2019

# Define particulars of this run:
INPUT_FILENAME=AQS.inp
JOBNAME=${JOB_NAME}
ABAQUS_ARGS=
SCRATCH_DIR=PathForTheTempraryFile
#
# To manage abaqus jobs, you need to catch signals
# and use "abaqus terminate" to stop the job
#
exit_gracefully () {
abaqus terminate job=$JOBNAME
echo Abaqus job $JOBNAME terminated
exit
}

# invoke abaqus in the background on the compute node:
trap exit_gracefully SIGUSR2

abaqus cpus=$NSLOTS mp_mode=mpi job=$JOBNAME input=$INPUT_FILENAME scratch=$SCRATCH_DIR
$ABAQUS_ARGS interactive analysis

# Report some useful info
/bin/uname \-a
```

```
#
# now sleep until lock file disappears
#
sleep 60
while [ -f ${JOBNAME}.lck ]; do
  sleep 5
done
```

last updated December 24, 2019 by Park, Sangmin

UNIST HPC User Guide

1. HPC Resources

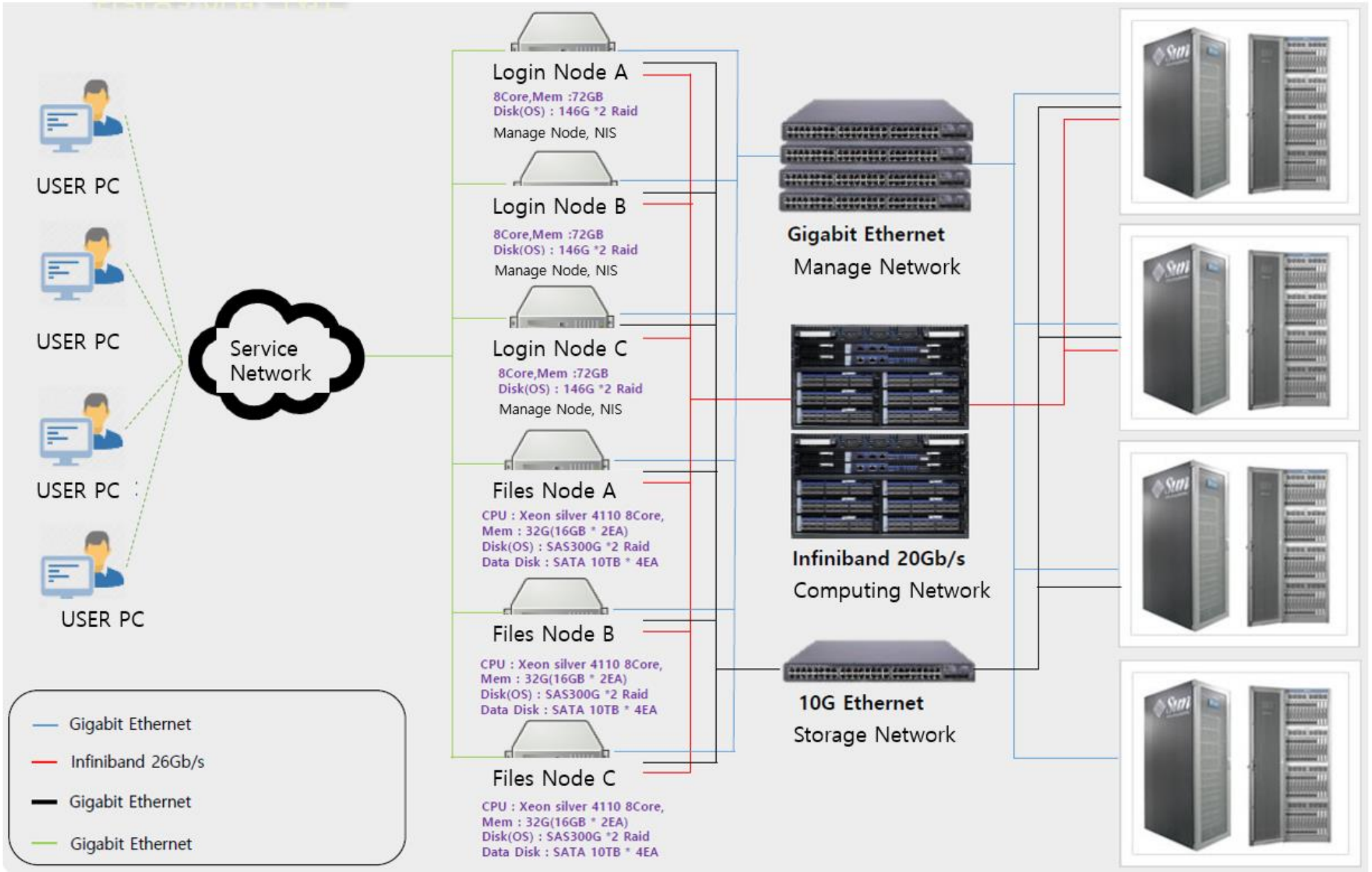
A. Hardware

i. Configuration Overview

UNIST Supercomputing Center(USC) has several High Performance Computing(HPC) systems and parallel file system to support a variety of researches that find the secret in the universe. For those, we have four HPC x86 Linux clusters named it leopard, lion, falcon, and eagle and Lustre parallel file system to be able to support high I/O throughput calculations.

the Tachyon system consists of approximately 500 computing nodes and each node has 8 cores. The microarchitecture of Tachyon system is Nehalem. Total number of cores in all Tachyon HPC cluster are approximately 4,000. The HPC systems vary in the interconnection. Those are connected from the 1G gigabit to the 40G infiniband connection.

Detailed HPC system configuration in USC-Tachyon could be shown by the following figure.


USER PC
USER PC
USER PC
USER PC
Service Network
Login Node A
8Core,Mem :72GB
Disk(OS) : 146G *2 Raid
Manage Node, NIS
Login Node B
8Core,Mem :72GB
Disk(OS) : 146G *2 Raid
Manage Node, NIS
Login Node C
8Core,Mem :72GB
Disk(OS) : 146G *2 Raid
Manage Node, NIS
Files Node A
CPU : Xeon silver 4110 8Core,
Mem : 32G(16GB * 2EA)
Disk(OS) : SAS300G *2 Raid
Data Disk : SATA 10TB * 4EA
Files Node B
CPU : Xeon silver 4110 8Core,
Mem : 32G(16GB * 2EA)
Disk(OS) : SAS300G *2 Raid
Data Disk : SATA 10TB * 4EA
Files Node C
CPU : Xeon silver 4110 8Core,
Mem : 32G(16GB * 2EA)
Disk(OS) : SAS300G *2 Raid
Data Disk : SATA 10TB * 4EA
Gigabit Ethernet
Manage Network
Infiniband 20Gb/s
Computing Network
10G Ethernet
Storage Network
Gigabit Ethernet
Infiniband 26Gb/s
Gigabit Ethernet
Gigabit Ethernet

ii. How to access HPC systems

We have two login nodes to access HPC-Tachyon systems. Login nodes are the main gateway to access to HPC-Tachyon systems. So, users need to go through the login nodes to access HPC-Tachyon systems. (2022 port)

hostname	login02	login03
DNS	**Tlogin02.usc.unist.ac.kr**	**Tlogin03.usc.unist.ac.kr**

iii. Disk quota

Tachyon has NFS file system. So, all user's home directory locates in the NFS file system server. It is shared to the all Tachyon system nodes.

Directory Name	Capacity
/home1	22 TB
/home2	22 TB
/home3	22 TB

iv. HPCs

1) System Overview

A) Infiniband Nodes

Hostname	com[001] – com[528]
Number of node(# of core)	522node (4,176 cores), 8cores/node
Processor	Intel Xeon X5570 2.93GHz
Memory	24GB or 48GB per node
OS	Linux CentOS 6.9
Interconnection	Infiniband(40G) network between computing nodes 1G Ethernet network to the NFS file system
Storage	/home1, /home2, /home3 in NFS file system

B) Ethernet network Nodes

Hostname	com[529] – com[624]
Number of node(# of core)	42node (336 cores), 8cores/node
Processor	Intel Xeon X5570 2.93GHz
Memory	24GB or 48GB per node
OS	Linux CentOS 6.9
Interconnection	1G Ehternet network between computing nodes 1G Ethernet network to the NFS file system
Storage	/home1, /home2, /home3 in NFS file system

2) Job Queues and PEs

A) Job Queues and PEs

B) Parallel Environment(PE)s

make PE의 사용 syntax는 make [digit]이며 [digit]는 solving할 total core 수를 의미 합니다.

C) Resource quota

HPC의 resource독점사용을 막기 위해 resource 할당 정책이 설정되어 있습니다. Quota는 연구그룹별 quota로 주어지며, 병렬계산 노드에서는 그룹별 3000core까지 '동시 사용'이 가능합니다.

Job status가 wait상태일 시 resource를 얼마나 사용하고 있는지 확인하시기 바랍니다.

i) Queues

queue name	wall clock time	available cores	available PEs	Remarks
tachyon-1g.q	168 hours	Up to group quota	make	Ethernet network Nodes
tachyon-ib.q	168 hours	Up to group quota	make	Infiniband Nodes

ii) Parallel Environment(PE)s

The usage syntax of make PE is "make [digit]", and [digit] refers to the total core that are used during the computation.

iii) Resource quota

The resource allocation policy is set to prevent exclusive use of HPC resources. The quota is given by research group quota, and for parallel computational nodes up to 3000 cores per group.

If your Job status is wait, Please check group resource usages.

B. Software

i. Compilers

Compilers	Version
GNU(GCC)	GCC 4.4.7(Linux CentOS 6.9 Default)
	GCC 7.4.0
Intel compiler	Intel composer xe 2016u4
	Intel composer xe 2019u4

ii. Libraries

Classification	Libraries	Libraries Install compiles list
Parallel libraries	OpenMPI 3.1.4	GCC 4.4.7(Linux CentOS 6.9 Default) GCC 7.4.0 Intel composer xe 2016u4
	Intel-impi xe2016u4	Intel composer xe 2016u4
	Intel-impi xe2019u4	Intel composer xe 2019u4
Mathematical libraries	FFTW 3.3.8	GCC 4.4.7(Linux CentOS 6.9 Default)
	Intel MKL xe2016u4	Intel composer xe 2016u4
	Intel MKL xe2019u4	Intel composer xe 2016u4

iii. Computational Packages

Classification	Applications	Installed Version(set up compiles)
Commercial	ABAQUS	abaqus 2019
	ANSYS CFD, Fluent	ansys 2019 R3
	COMSOL Multiphysics 5.0	comsol 54
	VASP	vasp 5.4.1 (intel 2019u4 –impi)
	Gaussian 09	None
Open Source	GROMACS	None
	Quantum Espresso	None
	LAMMPS	31Mar2017
	R Source	R 3.6.1 (intel 2019u4-mkl)

※ NOTICE : When you want to use a commercial package for your research, please contact us first, due to prevent the illegal usage of license.

C. Environment Settings

i. Overview

USC has implemented module environment to manage users' environment for all software installed in our HPC systems. Users are required very often to define some variables that affect the environment on each HPCs. And the operation of the program is executed under this environment. These environment variables are used to inform the HPC system the location of execution files, documentation, or related libraries. This would make new user or users unfamiliar with UNIX-like system very annoying. Through the module approach, however, users are no longer painful with configuring user's environment. The scripts(modulefiles) are made by the staff and users simply "load" and "unload" modules to configure the environment, also user can make individual module files.

Environment module approach updates users' environment very easily, especially **PATH**,**LD_LIBRARY_PATH**, **MANPATH** variables, etc.

When your account is made in HPC systems, "dot-files" should be set up for user's environment(default – bash shell). You can modify this "dot-files" to configure your environment firsthand. But, there is a way to set your environment up more conveniently.

1) Module commands

To know how to use environment module approach, type the following:

```
$ module help
```

"help" option displays as below. (not full result, select useful options)

```
Available SubCommands and Args:
 + add|load         modulefile [modulefile ...]
 + rm|unload        modulefile [modulefile ...]
```

```
+ switch|swap      [modulefile1] modulefile2
+ display|show     modulefile [modulefile ...]
+ avail           [modulefile [modulefile ...]]
+ purge
+ list
```

$ module list

prints a list of the currently loaded modulefile.

$ module avail

lists all the modulefiles which are available to be loaded.

$ module purge

unloads all loaded modulefiles currently.

$ module load *modulefile*

loads *the modulefile.*

$ module unload *modulefile*

unload *the modulefile.*

$ module switch modulefile_old modulefile_new

switches *the modulefile_old* with *modulefile_new*.

$ module show *modulefile*

displays how the *modulefile* changes the environment such as PATH, LD_LIBRARY_PATH, MANPATH, etc.

D. Job Submission

i. Overview

USC uses the queuing system to run users' code. We are using SGE(Sun Grid Engine) to handle user's job. This makes HPCs more effective and gives users convenience within HPCs use.

ii. SGE commands

Commands	Example	Description
qsub	qsub job_script_file_name	submit a job.
qstat	qstat	show job status oneself
	qstat -u '*'	show job status for all users
qhost	qhost	show computing node status
qdel	qdel job_ID qdel -u user_ID (user's all job cancelled)	cancel a job
qconf	qconf -sql	show all queue list
	qconf -spl	show all pe list
	qconf -sq short.q	show about short.q detailed
	qconf -sul	show all user list
	qconf -srqs	show resource quota policy
	qconf -shgrpl	show host group list
	qconf -sc	show complex attributes

iii. Job Script Examples

1) Parallel Job

```
#!/bin/bash
#$ -V                          # exporting environment of master node to the compute nodes (default).
#$ -pe pe_name(make or mpi_2~mpi_8) 8 # set parallel environment(pe), set number of total cores you need
#$ -N parallel_job             # Job Name. if you don't set this option,
                               # default value is job_script name.
#$ -q queue_name(ib.q or 1g.q)   # queue name (ib.q , 1g.q)
#$ -S /bin/bash                # shell selection users want to use
#$ -cwd                        # use current directory as working directory.
#$ -l h_rt=24:00:00            # resource time for job. (hh:mm:ss)(wall time clock).
                               # if you don't set this option, Job won't work.
                               # Job will be automatically stopped when it reaches the wall time clock.


## If you want to use intel mpi, should be uncommented below three lines.
#export I_MPI_FABRICS=shm:dapl
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load mpi/intel-xe2016/ompi-3.1.4     # ex) in case of OpenMPI 3.1.4
mpirun -machinefile $TMPDIR/machines -np $NSLOTS ./execution_file
```

iv. Exampl for each applications job script

1) ANSYS CFD (FLUENT)

```
#!/bin/bash
#$ -V
```

```
#$ -pe pe_name(make or mpi_2~mpi_8) 8 # set parallel environment(pe), set number of total cores you need
#$ -N MXTTA
#$ -q ib.q
#$ -S /bin/bash
#$ -cwd
#$ -l h_rt=24:00:00

## If you want to use intel mpi, should be uncommented below three lines.
#export I_MPI_FABRICS=shm:dapl
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load ansys/17.0

INPUT_FILE=MyFluentJobInputFileName

fluent 3ddp -t$NSLOTS -g -cnf=$TMPDIR/machines -sge -pinfiniband -mpi=openmpi -i $INPUT_FILE >
output_log
```

2) COMSOL Multiphysics

```
#!/bin/bash
#$ -V
#$ -pe pe_name(make or mpi_2~mpi_8) 8 # set parallel environment(pe), set number of total cores you need
#$ -N CMSXMPL
#$ -q ib.q
#$ -S /bin/bash
#$ -cwd
#$ -l h_rt=24:00:00

## If you want to use intel mpi, should be uncommented below three lines.
#export I_MPI_FABRICS=shm:dapl
```

```
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load comsol/54

NUM_CORE=8                                  ## Tachyon System per node's core is 8
NUM_NODE=`expr $NSLOTS / $NUM_CORE`     ## Total number of compute nodes

comsol batch -nn $NUM_NODE -np $NUM_CORE -f $TMPDIR/machines -inputfile MyInputFileName -
outputfile MyOutputFileName -batchlog MyLogFileName -tmpdir /uwork/p0xxxxx/
```

3) ABAQUS

```
#!/bin/bash
#$ -V
#$ -pe pe_name(make or mpi_2~mpi_8) 8 # set parallel environment(pe), set number of total cores you need
#$ -N CMSXMPL
#$ -q ib.q
#$ -S /bin/bash
#$ -cwd
#$ -l h_rt=24:00:00


## If you want to use intel mpi, should be uncommented below three lines.
#export I_MPI_FABRICS=shm:dapl
#export I_MPI_DAPL_PROVIDER=ofa-v2-ib0
#export I_MPI_DYNAMIC_CONNECTION=0

source /etc/profile.d/modules.sh
module load abaqus/2019

# Define particulars of this run:
```

```
INPUT_FILENAME=AQS.inp
JOBNAME=${JOB_NAME}
ABAQUS_ARGS=
SCRATCH_DIR=PathForTheTempraryFile
#
# To manage abaqus jobs, you need to catch signals
# and use "abaqus terminate" to stop the job
#
exit_gracefully () {
abaqus terminate job=$JOBNAME
echo Abaqus job $JOBNAME terminated
exit
}

# invoke abaqus in the background on the compute node:
trap exit_gracefully SIGUSR2

abaqus cpus=$NSLOTS mp_mode=mpi job=$JOBNAME input=$INPUT_FILENAME scratch=$SCRATCH_DIR
$ABAQUS_ARGS interactive analysis

# Report some useful info
/bin/uname \-a

#
# now sleep until lock file disappears
#
sleep 60
while [ -f ${JOBNAME}.lck ]; do
  sleep 5
done
```