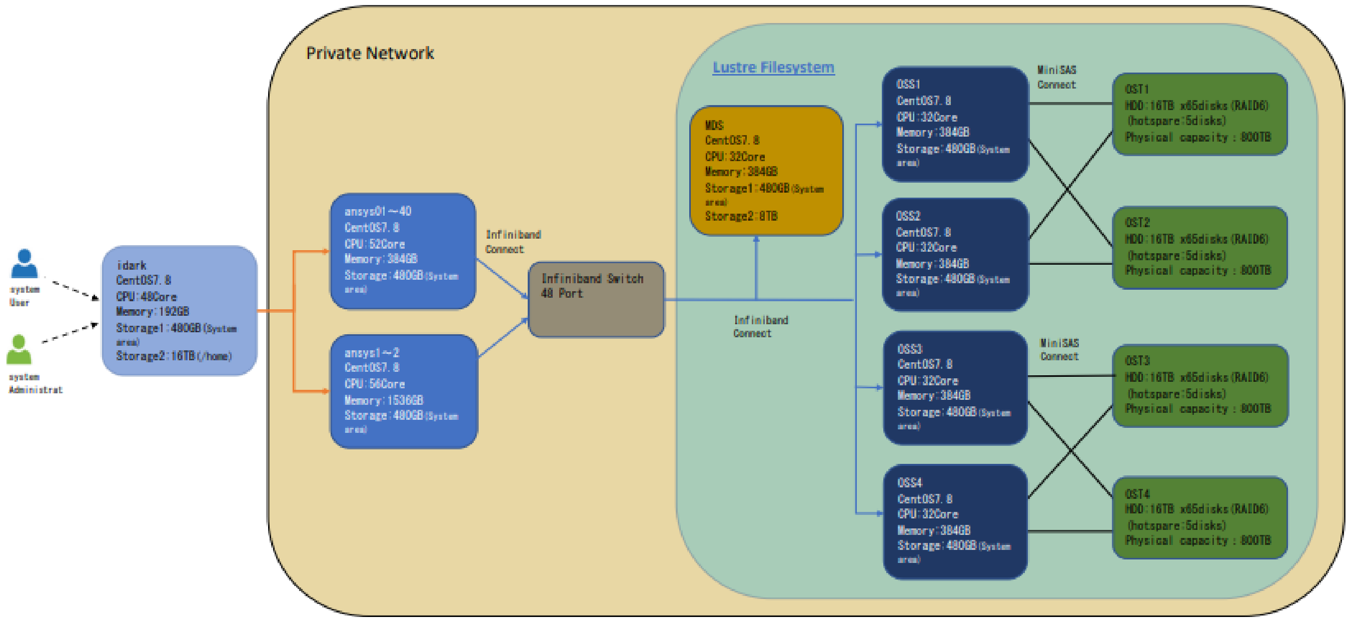
User operation manual

Contents

1 Overall system configuration diagram

The overall configuration diagram of this system is shown below.



2 How to manage user accounts

Our cluster products use Network Information Service (NIS)、User account Are collectively managed by the head node.

This section describes how to manage user accounts using NIS.

2.1 Add user account

(1) Log in to the head node as root user.

(2) Execute the following to add a user account.

- When setting the user shell to "tcsh "

```
# useradd␣-m␣-g␣users␣　「Additional user account name」・・・①
```

- When setting the user shell to " bcsh "

```
# useradd␣-m␣-g␣users␣　「Additional user account name」・・・①
```

```
# passwd␣　「Additional user account name」・・・②
Changing password for user　「Additional user account name」
New password:　・・・③
Retype new password:　・・・④
passwd: all authentication tokens updated successfully.　・・・⑤
# cd␣/var/yp␣;␣make　・・・⑥
```

① Add a user account.

② Set the password for the added user account.

③ Enter the password string.

※The password string is not displayed here.

④ Re-enter the password character string entered in ③.

※The password string is not displayed here.

⑤ This message is displayed when the password is set successfully.

⑥ Update the NIS database.

2.2 Delete user account

(1) Log in to the head node as root user.

(2) Execute the following to delete a user account.

- To leave the home area of the user to be deleted

```
# userdel␣ 「Deleted user account name」・・・①
# cd␣/var/yp␣;␣make ・・・②
```

① Delete a user account.

② Update the NIS database.

- When deleting the home area of the user to be deleted

```
# userdel␣-r␣ 「Deleted user account name」・・・①
# cd␣/var/yp␣;␣make ・・・②
```

③ Delete a user account

④ Update the NIS database.

2.3 Password change by users

After the administrator creates an account, the user changes the password.

(1) Log in to the head node as root user.

(2) Run the yppasswd command to change the password.

```
$ yppasswd ・・・①
Changing NIS account information for user account name on hostname.
Please enter old password: ・・・②
Changing NIS password for user account name on hostname.
Please enter new password: ・・・③
Please retype new password: ・・・④

The NIS password has been changed on hostnamen. ・・・⑤
```

① Run the yppasswd command.

② Enter the currently set password.

※The password string is not displayed here.

③ Enter the new password.

※The password string is not displayed here.

④ Re-enter the password character string entered in ③.

※The password string is not displayed here.

⑤ This message is displayed when the password is set successfully.

3 Intel Parallel Studio XE

3.1 Install Directory

As of 2019, Intel application development software represented by Intel Compiler

Is integrated into a development package called Intel Parallel Studio XE.

Intel Parallel Studio XE ships with multiple development software, but in this article,

I will focus on Intel Compiler and MKL included with Intel Compiler.

The Intel Parallel Studio XE package is available from the Intel Registration Center.

It is distributed on the homepage. If you have a license, Intel Parallel Studio XE is:

You can download the package from the site.

https://registrationcenter.intel.com/regcenter/registe

Intel Parallel Studio XE is updated annually, with the year of release being the Version.

Also, the Intel Compiler included there is the end of the Intel Parallel Studio XE Version (year). The two digits are the Major Version.

Ex: Intel Parallel Studio XE 2018 → Intel Compiler 18.0.X X has the number of updates

Intel Parallel Studio XE set up on the computer is installed in the following directory

It has been tolled.

Package	**Directory**
Intel Parallel Studio XE 20XX EX)　Intel Parallel Studio XE 2018	/opt/intel/psxe20XX /opt/intel/psxe2018

This installation directory is different from the default for Intel Parallel Studio XE.

Su. The reason for the change is when multiple Intel Compilers (Parallel Studio XE) are mixed. This is because the environment variables of the part are duplicated and a problem occurs .

3.2 Install Directory

To use Intel Compiler, use the following command. By default, all users except root

Can be used.

Command	Details
ifort	Fortran language compiler
icc	C language compiler
icpc	C++ compiler

You can check the compiler version with the "-V" option. EX: ifort -V

The factory settings have the following alias settings for the icc / ifort command. icc / ifort

The following options are automatically applied when you use the command.

Open	Details
-D_FILE_OFFSET_BITS=64	Programmatically take a file larger than 2GB Used to handle. (Standard use with icc.)
-D_LARGEFILE_SOURCE	Programmatically take a file larger than 2GB Used to handle. (Standard use with icc.)
-assume buffered_io	When writing to a file, write to disk immediately,First, instruct it to accumulate in the buffer. (Standard use with ifort.)

This is a compilation option often used by Intel Composer.

Open	Details
-O0 -O1 -O2 -O3	Controls source code optimization. Select and specify one of -O0 to -O3. If not explicitly specified (default setting), -O2 Become. -O0: No optimization. Performance is very poor,You need to be careful. -O1: Enable some optimizations. -O2: Default setting. Vectorization is enabled. -O3: Performs stronger optimization than -O2.
-parallel	The created executable file works in parallel with threads Will come to do.
-qopenmp	Compile a program that uses OpenMP It is used when doing. In older versions of Intel Compiler, -openmp It was an option name, but it has changed.
-mcmodel	Specifies the memory model to use. (Ex) -mcmodel = large Data with a size of 2GB or more can be used.
-fp-model	Specifies how to handle floating point numbers. (Ex) -fp-model strict Disables reduction and allows you to change the floating point environment Enable the property.

3.3 Change compiler version

The Intel Compiler environment settings are made in the files in each user's home directory.

Specifically, if the user's shell script is bash, is ~ / .bashrc, and if it is tcsh,

In the ~ / .cshrc file, the package of the compiler to be used is set.

If multiple compilers are installed, change the compiler package to be used.

To do this, make the following changes.

- If the user's shell is bash, modify COMPILER in ~ / .bashrc

Ex) When changing Intel Compiler from Version 18.0 to 17.0

Before correction	**Revised**
COMPILER=INTEL18.0	#COMPILER=INTEL18.0
#COMPILER=INTEL17.0	COMPILER=INTEL17.0
#COMPILER=INTEL15.0	#COMPILER=INTEL15.0
#COMPILER=PGI17	#COMPILER=PGI17
#COMPILER=PGI16	#COMPILER=PGI16
#COMPILER=PGI15	#COMPILER=PGI15

- If the user's shell is tcsh, modify COMPILER in ~ / .cshrc.

Ex) When changing Intel Compiler from Version 17.0 to 18.0

Before correction	**Revised**
set COMPILER=INTEL18.0	#set COMPILER=INTEL18.0
#set COMPILER=INTEL17.0	set COMPILER=INTEL17.0
#set COMPILER=INTEL15.0	#set COMPILER=INTEL15.0
#set COMPILER=PGI17	#set COMPILER=PGI17
#set COMPILER=PGI16	#set COMPILER=PGI16
#set COMPILER=PGI15	#set COMPILER=PGI15

※You can check the shell used by each user with the "echo $ 0" command.

The compiler version will change the next time the user logs in.

Please use "ifort -V" or "icc -V" to confirm that the version has changed.

⚠ Precautions when changing the compiler version

If you change the version of the above Intel compiler, the environment variables

LD_LIBRARY_PATH etc. will change.

If you run a previously compiled program after changing the compiler version,

The line file may not be able to reference the correct library and may not work properly.

In that case, Recompile the program you wnt to run, or a new user for each version of the compiler you use

3.4 Intel Math Kernel Library

The Intel Compiler ships with the Intel Math Kernel Library (MKL) package.

MKL is often used for scientific and technological calculations such as vector arithmetic functions and fast Fourier transform (FFT).

A group of libraries that processing at high speed, with extremely high performance, especially when running on an Intel CPU.

The compilation example when using MKL's blas / lapack etc. is as follows.

Ex1) Compiling with dynamic link using Intel LP64 interface

```
ifort myprog.f -L${MKLPATH} -I${MKLINCLUDE}
-lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -liomp5 -lpthread -lm
```

Ex2) Compiling with static links using Intel LP64 interface

```
i ifort myprog.f -L${MKLPATH} -I${MKLINCLUDE} -Wl,--start-group
${MKLPATH}/libmkl_intel_lp64.a ${MKLPATH}/libmkl_intel_thread.a
${MKLPATH}/libmkl_core.a -Wl,--end-group -liomp5 -lpthread -lm
```

※$ {MKL_PATH} and $ {MKLINCLUDE} vary depending on the version of Intel Compiler

used.

MKL needs to select and change options and libraries to link with depending on the build status. The link options are very complicated.

Intel is the official site, a site that navigates the link options according to the situation,

The Intel® Math Kernel Library Link Line Advisor site has been opened.

Please refer to the following site for specific link options.

https://software.intel.com/en-us/articles/intel-mkl-link-line-advisor

3.5 Document

The Intel Parallel Studio XE documentation is installed below.

Please refer to it if you want to know the details of usage and specifications.

Package	Directory
Intel Parallel Studio XE 20XX EX)　Intel Parallel Studio XE 2018	/opt/intel/psxe20XX/documentation_20XX /opt/intel/psxe2018/documentation_2018

The documents in the above directory are basically Intel's online manuals.

It is in the form of a reference. Therefore, refer to it in an environment where the computer cannot connect to the Internet. Please note that there is not much information available.

In addition, various documents can be found on the website of Excel Software Co., Ltd., an agency of Intel Parallel Studio XE. Is distributed. Please refer to the URL below.

https://www.xlsoft.com/jp/products/intel/studio_xe/index.html#documents

3.6 Document

The license for Intel Parallel Studio XE sold by us has a one-year support period.

You have the following rights during the warranty period.

(1) Log in to the head node as root user Get a new package when the Intel Parallel Studio XE package is upgraded. You can install it on your calculator. New package is Intel Registray. It will be distributed from the Shon Center.

(2) Get Intel's technical support.

If the license expires, you will not be able to receive the above support. This will continue even after the support deadline. If you want to receive these measures, you need to renew the license for a fee.

Use the compiler already installed on your computer after the support expires

Or you can use the compiled program.

Support period for Intel Parallel Studio XE is logged at Intel Registration Center

You can check it. It is also described in the license file used by the calculator.

Please check the following for the license file.

- License files are located under /opt /intel /licenses.

Please refer to the following example for the confirmation points.

- File ex: (In the example below, it is until 2019/2/19)

```
PACKAGE IC45FB71A INTEL 2019.0219 5B5F03F3893A COMPONENTS="ArBBL ¥
```

4 IntelMPI

IntelMPI is a software that implements the MPI standard that provides message functionality between parallel and distributed processes.
It is one of the software. As a feature,

- Paid software developed by Intel based on MPICH3 and MVAPICH2.
- Compliant with MPI-3.1 standard.
- Adjustments have been made specifically for Intel hardware.

That is the point.

In the past, with mpich and lam compliant with the MPI-1 standard, mpich2 compliant with the MPI-2 standard, and openmpi In most cases, the source code that was working can be compiled as is without any modification.
Note that the application binaries that were built with mpich or openmpi in the past remain as they are. Does not work in the Intel MPI environment. To run with IntelMPI, from the source code IntelMPI, You need to rebuild.
This section outlines the Intel MPI set up on your computer.

4.1 Install Directory

Currently, Intel MPI has a unified directory structure with Intel Parallel Studio XE
The Major Version of IntelMPI and the Version of Intel Parallel Studio XE (year) are common.
So, for example, Intel MPI 2018 has the same directory structure as Intel Parallel Studio XE 18.0.
The Intel MPI set up on the computer is the Intel Parallel Studio XE described in Chapter 2.1. It is installed in the following directory for consistency

Package	**Directory**
IntelMPI20XX EX)　IntelMPI2018	/opt/intel/psxe20XX /opt/intel/psxe2018

4.2 IntelMPI compile command

You can compile the source code using MPI with the following command.
IntelMPI has different compile commands depending on the type of compiler used internally. With the traditionally used mpif90 and mpicc, the Intel Compiler is used. Please note that there is no such thing.

Command	Details
mpiifort	Commands for Fortran using the Intel Fortran Compiler (ifort)
mpiicc	Commands for C using Intel C Compiler (icc)
mpiicpc	Commands for C ++ using Intel C / C ++ Compiler (icpc)
mpif77	Commands for Fortran using the GNU Fortran compiler g77
mpif90	GNU Fortran compiler Commands for Fortran using gfortran
mpicc	If specified by the I_MPI_CC variable, etc., a command for C language using the specified C Compiler. If no variable is specified, a command for C language using the GNU C compiler (gcc)
mpigcc	Commands for the C language using the GNU C compiler (gcc)
mpigxx	Commands for the C/C ++ language using the GNU C/C ++ compiler (g ++)

As for the options at the time of compiler, basically the options of the compiler used internally can be used as they are.
However, you need to be careful about static links.
In recent years, due to OS system relations or application development reasons,
Library link very often we use shared links (dynamic links) at times.
It is possible to use static options as a compiler option, Application in some cases, it is difficult to deal with the case where the link is assumed to be shared. In some cases, the libraries required for linking are only packages for shared links. There are many cases where it is difficult to create a binary that links all libraries statically.
In such a case, even if you build the application using the static command,
The application may not operate normally in cases such as executing outside the created node or changing (updating) the node environment.

4.3 Execution command

Run the program using the mpirun command. The format is as follows.

```
mpirun -np [Parallel number] [option] [Executable file]
```

[parallel Number] contains the number of processes used for parallel computing.
Usually that number of CPU cores is used it will be use. For example, for parallel computing using 4core, specify mpirun -np 4…
The [option] item is a run-time option. A detailed explanation of the options is, man command on the OS run man mpirun on the screen or check the Intel MPI documentation.
When specifying [Executable file], it is necessary to specify the path correctly.
Especially when there are multiple executable files with the same name or when starting a process between nodes, It is highly recommended to specify an absolute path for safe execution.
For example, the program test exists in / usr / local / bin /, Input file with -i option,
-o Consider a case where you can specify the output file with the option.
On the input file use / home / hpc / test_data / inp and specify / home / hpc / test_out / out as the output destination, If you want to, the command line will be as follows.

```
mpirun -np 4 /usr/local/bin/test -i /home/hpc/test_data/inp
-o /home/hpc/test_out/out
```

Also, when the program test is placed in the current directory in the above example, the command line will be as follows.

```
mpirun -np 4 ./test -i /home/hpc/test_data/inp -o /home/hpc/test_out/out
```

4.4 Specifying a calculation node when executing a calculation

In IntelMPI, it is necessary to specify the node used for calculation each time at runtime.
Nodes are specified by run-time commands.
Program test in the current directory, 1 process on compute node node01,
To execute 3 processes on the calculation node node02, use the following command.

```
mpirun -np 1 -host node01 ./test : -np 3 -host node02 ./test
```

If you want to evenly allocate and execute the calculation process for multiple nodes used for parallel computing, you can also use the -ppn option.
Program test in the current directory, compute nodes node01, node02, node03, node04
2 processes each, If you want to execute it in a total of 8 processes, use the following command.

```
mpirun -ppn 2 -hosts node01,node02,node03,node04 ./test
```

※Simultaneous use of the -np and -ppn options can cause malfunctions and is not recommended.
In Intel MPI, the process assigned to each node at runtime is bound to the CPU (core),is the default.
There is no need to explicitly set the option to bind the process to the CPU (core).

It is also possible to set the specification of the node used for calculation in the file.
First, create a file that specifies the node to use for the calculation and the process to allocate to it.
For example, if you want node01 to run 2 processes, node02 to run 4 processes, and node03 to run 2 processes, the configuration file is as follows.

```
node01:2
node02:4
node03:2
```

After creating the configuration file, specify the created file with the -machinefile option.
Save this config file in the current directory as machinelist, to execute the program test in the current directory, use the following command.

```
mpirun -machinefile ./machinelist ./test
```

In this case, parallel computing is performed using the allocation method described in the configuration file.
(In the case of the machinelist shown in the concrete example, parallel calculation is executed in a total of 8 processes.)
The -machinefile option can also be combined with the -np option.
Specifically, specify as follows.

```
mpirun -machinefile ./machinelist -np 4 ./test
```

If these two options coexist, the number of parallel processes specified by -np will be increased. The calculation is performed by allocating in order from the top of the configuration file specified by -machinefile.
In the case of the above example, parallel calculation is performed in a total of 4 processes, 2 processes each for node01 and node02. It will be executed.
As mentioned above, Intel MPI does not have a function to automatically refer to the configuration file specified by -machinefile when executing the calculation.
If you do not use the job scheduler, sorry to trouble you, but you will specify it every time you execute the calculation.

4.5 Tips for parallel computing

Depending on the characteristics of the program to be executed, how to allocate the execution process to the nodes of the program, It can have a significant impact on performance. An example is shown below.

(Example) A program that is rate-determining by transferring a large amount of data between the CPU core and memory.

In order to fully utilize the high processing performance of the CPU core, data must be retrieved from memory at a speed commensurate with that. In the case of a program whose performance is dominated by the data transfer rate from memory to the CPU core (register) due to the algorithm. It is important to choose the process group assignment to the node, taking into account the transfer rates available to each process.

On the node, the processes running on the node share the data transfer rate between CPU and memory, Therefore, the transfer speed that can be used by each process is one severalth of the running processes.

For example, in a 4-node cluster environment with 8 cores on each node, a total of 16 processes can be parallelized, Consider the following two types of execution examples.

Execution example A: When using 2 nodes and starting 8 processes on each node

Execution example B: When using 4 nodes and starting 4 processes on each node

If the total data transfer rate between CPU and memory is set to 100, each CPU core in each-The data transfer speed between memories is as follows.

	Execution example A	Execution example B
node01	100/8 = 12.5	100/4 = 25
node02	100/8 = 12.5	100/4 = 25
node03		100/4 = 25
node04		100/4 = 25

Execution example A has twice the data transfer speed between CPU and memory of each process as compared to execution example B.

For programs that are rate-determining by transferring large amounts of data between the CPU core and memory, since this data transfer speed greatly affects the calculation speed of the entire program, carefully allocate the process group.

The choice is important.

The above example is a general theory, so please check the actual operation by yourself before using it.

4.6 How to link with the job scheduler

When you submit an MPI program as a job using the job scheduler (LSF / (Open) PBS / GridEngine, etc.), Jobscheduler automatically selects the node to execute the program.
For information on how to submit an MPI job using the job scheduler, Each job schedule Please check the manual of the Jurer.

■appendix

(1) Placement of shell configuration files.

The shell environment setting file of each application set up by us is /home/.common
It is located in the directory.
Configuration files for bash (extension .sh) and csh (extension .csh) Two types are available. The directories are arranged as follows, depending on the compiler and MPI version.

① /home/.common/INTEL*,/home/.common/PGI * directory.

This directory is created for each compiler name and compiler version, and contains the shell environment setting files for Intel Parallel Studio XE and PGI Compiler. In addition, a shell configuration file for the application compiled and linked using that compiler will be placed.

② /home/.common/INTEL*/IntelMPI directory.

On the environment where Intel Parallel Studio XE is set, the shell environment setting file for using Intel MPI is placed. In addition, the shell configuration file of the application compiled and linked using Intel Parallel Studio XE and IntelMPI will be placed.

③ /home/.common directory.

The shell environment setting file of the application that does not depend on the compiler / MPI is placed directly under this directory.

(2) Flow of reading the shell environment setting file.

The variables COMPILER and MPI are defined in ~ / .cshrc and ~ / .bashrc for each user. Then, the compiler specified by the variable COMPILER and the shell environment setting file related to the MPI specified by the variable MPI are read from under /home/.common in ascending file name order. The order of reading is ① in the bullet point (1) above, then ② according to the MPI, and finally ③.

5 PBS Professional User Manual

5.1 Installation overview

(1) Package

The source files for the PBS Professional Open Source batch queuing system distributed by https://github.com/pbspro/pbspro are being built and set up.

(2) Installation directory

/opt/pbs

(3) Daemon

The daemon runs automatically when the OS starts.

/etc/init.d/pbs

… Works on all nodes of the PBS Professional Open Source cluster.

(4) Environment setting file

The user preferences of PBS Professional batch queuing system are configured in the files in the following directory.

```
/etc/profile.d/pbs.sh
/etc/profile.d/pbs.csh
```

Command example

Outlines the commands for users to use the PBS Professional Open Source batch queuing system.

(1) pbsnodes -a

Shows the status of all nodes in the cluster.

```
$ pbsnodes -a
hpc01.local
 Mom = hpc01.local
 ntype = PBS
 state = free
 pcpus = 56
 resources_available.arch = linux
 resources_available.host = hpc01
 resources_available.mem = 131497160kb
 resources_available.ncpus = 56
 resources_available.vnode = hpc01.local
 resources_assigned.accelerator_memory = 0kb
 resources_assigned.hbmem = 0kb
 resources_assigned.mem = 0kb
 resources_assigned.naccelerators = 0
 resources_assigned.ncpus = 0
 resources_assigned.vmem = 0kb
 resv_enable = True
 sharing = default_shared
 last_state_change_time = Mon Aug 6 16:33:57 2018
hpc02.local
...
```

(2) qsub

PBS Professional Open Source Submits a job onto the batch queuing system. When a job is submitted, the job ID is displayed. Please note that the specified job must be in script format. Please refer to Chapter 4.4for creating scripts.

```
$ qsub␣./test.sh
6.hpc01.local
```

(3) qdel

Deletes the job with the specified job ID

```
$ qdel␣6
```

(4) qstat

PBS Professional Open Source Displays the status of the batch queuing system.

```
# qstat -f
Job id                    Name              User            Time Use    S    Queue
--------------------------  ----------------  ----------------  ---------   -    -----
195. hpcw01.local         testjob1.sh       hpc             00:17:05    R    workq
```

The display contents of qstat -a are as follows.

job id: Job ID and the node that submitted the job

Name: Submitted job

User: The user who submitted the job

Time Use: Job execution time

S: Current status of the job

R = Running Q = Waiting for queue H = Pending E = Ending T = Migrating W = Waiting

Queue: Queue assigned when submitting a job

Displays the details of the job ID specified by qstat -f JOBID.

```
# qstat -f 195
Job Id: 195. hpcw01.local
  Job_Name = test-openmp.sh
  Job_Owner = hpc@ hpcw01.local
  resources_used.cpupercent = 90
  resources_used.cput = 00:00:09
  resources_used.mem = 4523272kb
  resources_used.ncpus = 28
  resources_used.vmem = 19908072kb
  resources_used.walltime = 00:00:11
  job_state = R
  queue = workq
  server = hpcw01.local
  Checkpoint = u
  ctime = Tue Nov 22 10:22:35 2016
  Error_Path = hpcw01.local:/home/hpc/test-xhpl-openmp/test-openmp.sh.e195
  exec_host = hpcw01/0*12
  exec_vnode = (hpcw01:ncpus=12)
  Hold_Types = n
  Join_Path = n
  Keep_Files = n
  Mail_Points = a
  mtime = Tue Nov 22 10:22:36 2016
  Output_Path          =          hpcw01.local:/home/hpc/test-xhpl-openmp/test-
openmp.sh.o195
  Priority = 0
  qtime = Tue Nov 22 10:22:35 2016
  Rerunable = True
  Resource_List.ncpus = 12
```

In addition, qstat can check other information with the following options.

-q : Show queue status
-Q : Show queue status
-B : Displaying the contents of PBS Server
-a : Display details of job list

(5) tracejob

View job details

```
$ tracejob 195
Job: 195. hpcw01.local
11/22/2016 10:22:35 L Considering job to run
11/22/2016 10:22:35 S Job Queued at request of hpc@hpcw01.local, owner =
 hpc@hpcw01.local, job name = test-openmp.sh, queue
 = workq
11/22/2016 10:22:35 S Job Run at request of Scheduler@hpcw01.local on
 exec_vnode (hpcw01:ncpus=12)
11/22/2016 10:22:35 S Job Modified at request of Scheduler@hpcw01.local
11/22/2016 10:22:35 L Job run
11/22/2016 10:22:35 S enqueuing into workq, state 1 hop 1
11/22/2016 10:22:35 A queue=workq
11/22/2016 10:22:35 A user=hpc group=users project=_pbs_project_default
 jobname=test-openmp.sh queue=workq
 ctime=1479777755 qtime=1479777755 etime=1479777755
 start=1479777755 exec_host=hpcw01/0*12
 exec_vnode=(hpcw01:ncpus=12) Resource_List.ncpus=12
 Resource_List.nodect=1 Resource_List.place=free
 Resource_List.select=1:ncpus=12
 resource_assigned.ncpus=12 project=_pbs_project_default
 jobname=test-openmp.sh queue=workq
 ctime=1479777755 qtime=1479777755 etime=1479777755
 start=1479777755 exec_host=hpcw01/0*12
 exec_vnode=(hpcw01:ncpus=12) Resource_List.ncpus=12
 Resource_List.nodect=1 Resource_List.place=free
 Resource_List.select=1:ncpus=12
 resource_assigned.ncpus=12
```

(6) qmgr

PBS Professional Open Source You can display the settings of the batch queuing system. If you are root, you can change the settings of PBS Professional Open Source.

```
$ qmgr -c "print queue workq"
#
# Create queues and set their attributes.
#
#
# Create and define queue workq
#
create queue workq
set queue workq queue_type = Execution
set queue workq enabled = True
set queue workq started = True
```

Example of use

- qmgr␣-c␣"print␣queue␣queuename"

  Displays information about the specified queue.
- qmgr␣-c␣"print␣queue␣queuename"

  Displays information about the specified node.
- qmgr␣-c␣"print␣server".
  PBS Professional Open Source Displays configuration information for the batch queuing system.

5.2 Queue settings

(1) queue settings

The PBS queue is divided into Group A and Group B, where Group A is ansys [01-40], Group B will be ansys [1-2].
TThe queue settings are as follows.
tiny queue

```
# qmgr -c "print queue tiny"
#
# Create queues and set their attributes.
#
#
# Create and define queue tiny
#
create queue tiny
set queue tiny queue_type = Execution
set queue tiny resources_max.ncpus = 1
set queue tiny resources_max.nodect = 1
set queue tiny default_chunk.host_group = GroupA
set queue tiny max_user_run = 256
set queue tiny enabled = True
set queue tiny started = True
```

mini queue

```
# qmgr -c "print queue mini"
#
# Create queues and set their attributes.
#
# Create and define queue mini
#
create queue mini
set queue mini queue_type = Execution
set queue mini resources_max.ncpus = 52
set queue mini resources_max.nodect = 1
set queue mini default_chunk.host_group = GroupA
set queue mini max_user_run = 6
set queue mini enabled = True
set queue mini started = True
```

small queue

```
# qmgr -c "print queue small"
#
# Create queues and set their attributes.
#
#
# Create and define queue small
#
create queue small
set queue small queue_type = Execution
set queue small resources_max.ncpus = 208
set queue small resources_max.nodect = 4
set queue small default_chunk.host_group = GroupA
set queue small max_user_run = 3
set queue small enabled = True
set queue small started = True
```

large queue

```
# qmgr -c "print queue large"
#
# Create queues and set their attributes.
#
#
# Create and define queue large
#
create queue large
set queue large queue_type = Execution
set queue large resources_max.ncpus = 1040
set queue large resources_max.nodect = 20
set queue large default_chunk.host_group = GroupA
set queue large max_user_run = 1
set queue large enabled = True
set queue large started = True
```

mini2 queue

```
# qmgr -c "print queue mini2"
#
# Create queues and set their attributes.
#
#
# Create and define queue mini2
#
create queue mini2
set queue mini2 queue_type = Execution
set queue mini2 resources_max.ncpus = 56
set queue mini2 resources_max.nodect = 1
set queue mini2 default_chunk.host_group = GroupB
set queue mini2 max_user_run = 1
set queue mini2 enabled = True
set queue mini2 started = True
```

queue	Maximum number of jobs to be executed /user	Maximum number of cores in use /job	Maximum number of nodes in use /job
tiny	256	1	1
mini	6	52	1
small	3	208	4
large	1	1040	20
mini2	1	56	1

(2) Node settings

It is described in /var/spool/pbs/server_priv/nodes of the host machine. This phi
The contents set in the file are displayed by the pbsnodes command. np, gpu, properties, etc.Please change this file when you change the value of.
※queue GroupA : ansys[01-40]
※queue GroupB : ansys[01-40]

```
# cat␣/var/spool/pbs/server_priv/nodes
ansys01 np=52 GroupA
ansys02 np=52 GroupA
ansys03 np=52 GroupA
ansys04 np=52 GroupA
ansys05 np=52 GroupA
ansys06 np=52 GroupA
ansys07 np=52 GroupA
ansys08 np=52 GroupA
ansys09 np=52 GroupA
ansys10 np=52 GroupA
    (abbreviation)
ansys31 np=52 GroupA
ansys32 np=52 GroupA
ansys33 np=52 GroupA
ansys34 np=52 GroupA
ansys35 np=52 GroupA
ansys36 np=52 GroupA
ansys37 np=52 GroupA
ansys38 np=52 GroupA
ansys39 np=52 GroupA
ansys40 np=52 GroupA
db01 np=48 GroupA
db02 np=48 GroupA
db03 np=48 GroupA
db04 np=48 GroupA
db05 np=48 GroupA
db06 np=48 GroupA
ansys1 np=56 GroupB
ansys2 np=56 GroupB
```

5.3 Job submission example (CPU version)

The following is an example of the CPU version of the job script.

(1) Node settings

```
#! / bin / tcsh
Make #PBS␣-j␣oe    #stdout and stderr the same file
# PBS ␣ -o ␣ logfile Specify the file name of      #stdout
# PBS␣-q␣workq     #Specify queue
cd ␣ $ PBS_O_WORKDIR
./a.out
```

※ When using the executable file in the current directory, the current directory Cd to $ PBS_O_WORKDIR before running.

(2) OpenMP job (example of 8-core parallelism)

```
#!/bin/tcsh
#PBS␣-l␣nodes=1:ppn=8   #1 node used・8 CPU core used
cd␣$PBS_O_WORKDIR
export OMP_NUM_THREADS=8
./a.out
```

(3) IntelMPI job (example of using 2 nodes with 4 cores each)

```
#!/bin/tcsh
#PBS -l nodes=2:ppn=4   #2 Use nodes・Use 4 CPU cores each
cd $PBS_O_WORKDIR
mpirun -np 8 ./a.out
```

(4) Specifying the node to execute the job

```
#!/bin/tcsh
#PBS -l nodes=2:ppn=4   #2 Use nodes・Use 4 CPU cores each
cd $PBS_O_WORKDIR
mpirun -np 8 ./a.out
```

- When specifying by the host name of the node

```
#!/bin/csh
#PBS -q workq
#PBS -l select=1:ncpus=7:host=hpc01:mpiprocs=7 (Continue)
+1:ncpus=1:host=hpc02:mpiprocs=1
cd $PBS_O_WORKDIR
mpirun -np 8 ./a.out
```

The variables that can be used in the job script are excerpted in the table below.

Variable	**Description**
PBS_JOBNAME	User specified jobname
PBS_O_WORKDIR	Job's submission directory
PBS_TASKNUM	Number of tasks requested
PBS_O_HOME	Home directory of submitting user
PBS_MOMPORT	Active port for mom daemon
PBS_O_LOGNAME	name of submitting user
PBS_NODENUM	Node offset number
PBS_O_SHELL	Script shell
PBS_O_JOBID	Unique pbs job id
PBS_O_HOST	Host on which job script is currently running
PBS_QUEUE	Job queue
PBS_NODEFILE	File containing line delimited list on nodes allocated to the job
PBS_O_PATH	Path variable used to locate executableswithin job script

5.4 Documents

The official manual for the paid version of PBS Professional is available on the website. See below for more information. It is also a reference material for the PBS Professional Open Source version.

・ PBS Works Documentation
https://pbsworks.com/SupportGT.aspx?d=PBS-Professional,-Documentation