Chemdynm User Guide

Important System and Programming Notes

File system changed

The file system on new chemdynm is running cluster file system – Lustre from SUN.
http://wiki.lustre.org/index.php?title=Main_Page#What_is_Lustre.3F

The home directory is /luster/home instead of /export/home/chemdynm. Be aware if you still try to submit old script to queuing system.

Scratch directory

/state/partition1/your_user_name in each computer node is the scratch directory for your running job. /luster/scratch is NOT a local disk file system on each node. This is a global Lustre files system for storing temporary files. If you are using this directory for your job, not only the performance of your job will be four times slower than the performance that using local scratch /state/partition1/your_user_name, but also effect the performance of others job.

Login

ssh login. It may update to eraider login for the user who has eraider account.

Computer Node Name Changed

Computer node named from c1-1 to c1-32 instead of c0-0 to c0-30

User Quota and Backup Policy

/luster/home/username – 100GB/account, backup once/week

/luster/work/username – 100GB/account, no backup

/luster/scratch/username – no quota limit, no backup

Parallel Environment (InfiniBand Related)

There are two IB 24-ports switches which will take care of parallel computing. Each switch connects 16 computer nodes and uses the rest of 8 ports for cross-over connections to another IB switch. The bottle neck may occurs on those cross-over connections when lots of job run on two section of computer nodes and require communication between two IB switches.

Memory Intensive Job

The computer nodes c1-1 ~ c1-4 are 32GB memory, the rest of nodes c1-5 ~ c1-32 are 16GB only. If you want your job to occupy all the resource in one computer nodes for some reason, you have to add following line into your qsub script

```
#qsub -n 8                    ! tell the queue you will need 8 processors
#qsub -R "span[ptile=8]"      ! tell the queue that find those 8 processors in one node.
```

Queuing System

There are two queues in Chemdynm queuing system, HiMem queue and Normal queue. HiMem queue is subset of Normal queue which means that c1-1 ~ c1-4 are not separated from the rest of computer nodes by queuing system.

1. If you want the job to go to c1-1 ~ c1-4 only, please add #$ -q HiMem into the header of your qsub script.

2. If you want the job to go to c1-5 ~ c1-32 only, please add #$ -q all.q into the header of your qsub script.

3. If you want the job to go to any computer nodes available, then do nothing.

CPU Information of Computer Nodes

Dual Quad-core Intel(R) Xeon(R) CPU E5430 at 2.66GHz, L2 cache 12MB

Venus-NWChem

NA

MPI (mvapich) Options for Scalable code

See the mvapich User Guides http://mvapich.cse.ohio-state.edu/support/.

Introduction

Chemdynm is computer cluster of Prof. Bill Hase research group in the department of Chemistry and Biochemistry, Texas Tech University, serving NSF PIRE researcher throughout University of Vienna, Austria, University of Pisa, Italy, and University of Santiago de Compostela, Spain.

The Dell Linux cluster, Chemdynm, is configured with 32 Dell Power Edge 1950 computer nodes (rack mount), total 606GB memory and 21.5TB disk space, The theoretical peak performance is 2.7 TFLOPS. Nodes are interconnected with Cisco InfiniBand technology.

Architecture

The Chemdynm compute and login nodes run a Linux CentOS and are managed by the Rocks 5.0 cluster toolkit. One Dell 48-ports GigE manageable switch provides access between frontend, computer nodes, metadata server, storage servers. Parallel Lustre files systems have been configured to target different storage needs. Each compute node contains 8 cores as a dual-socket, quad-core platform. The configuration and features for the compute nodes, interconnect and I/O systems are described below, and summarized in Tables 1-3.

Compute Nodes: Chemdynm is a rack-mount 1U system. Each node is a Dell PowerEdge

1950 running a 2.6 x86_64 Linux kernel from www.rockclusters.org. Each node contains two Intel Xeon Quad-Core 64-bit processors (8 cores in all) on a single board, as an SMP unit and utilizes four 45-nm Hi-k next generation Intel® Core™ microarchitecture cores. The processor is manufactured on Intel's 45 nanometer process technology combining high performance with the power efficiencies of a low-power microarchitecture.. The core frequency is 2.66GHz with a peak performance 83 GFLOPS/node. Each node contains 16GB of memory. The memory subsystem has a 1333MHz Front Side Bus, and 2 channels with 533MHz Fully Buffered DIMMS. Each socket possesses 12 MB (2 x 6MB) Level 2 cache with Intel® Advanced Smart Cache Architecture.

Filesystems: Chemdynm's filesystems are built on three SuperMicro storage servers, each containing 8 SATA drives, and one Dell metadata servers. From this aggregate space of 19TB, several filesystems will be partitioned (see Table 4).

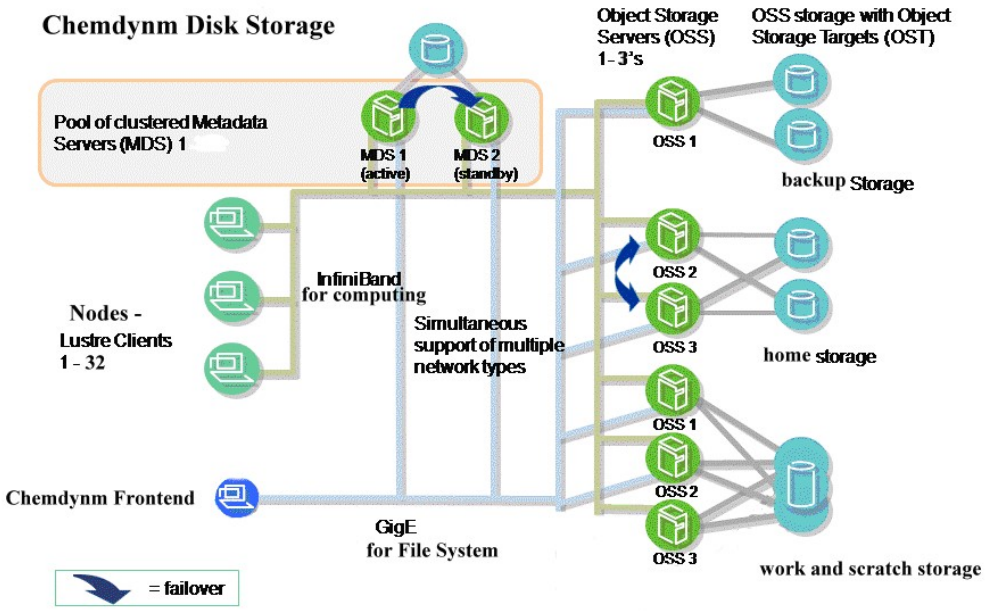

Table 1. System Configuration & Performance

Component	Technology	Performance/Size
Peak Floating Point Operations		2.7 TFLOPS (Theoretical)
Nodes(blades)	dual Quad-Core Intel Xeon E5430 processors @ 2.66GHz	32 Nodes / 256 Cores
Memory	Distributed	606GB (Aggregate)
Shared Disk	Lustre, parallel File System	19TB
Local Disk	SATA 80GB	2.5TB (Aggregate)
Interconnect	InfiniBand Switch	1 GB/s P-2-P Bandwidth

Table 2. Dell PowerEdge 1950 Compute Nodes / Frondend

Component	Technology
Sockets per Node/Cores per Socket	2/4 (Harpertown)

Clock Speed	2.66GHz
Memory Per Node	32GB memory from c1-1 to c1-4 16GB memory from c1-5 to c1-32
Front Side Bus	1333MHz

Table 3. Quad-Core Intel Xeon Harpertown Processor

Technology	64-bit
Clock Speed	2.66GHz
Peak Performance/core	10.5GFLOPS/core
L2 Cache	2 x 6MB
L2 Cache Speed	2.66GHz
Thermal Design Power	80W
Thermal Specification	67°C

Table 4. Storage Systems

Storage Class	Size	Architecture	Features
Local	80GB/node	SATA 3.5 HDD	Local scratch
Parallel	19TB	Lustre, SuperMicro storage servers	3 SuperMicro data storage servers, 1 Dell Metadata servers (See Table 5 for breakdown of the parallel filesystems)
Backup Tape (TOSM)	NA	NA	10Gb/s connection through TTU backbone

Table 5. Parallel Filesystems

Storage Class	Size	Quota (per User)	Features
Home	1.4TB	100GB	Backed up weekly; Not purged
Work	5TB	100GB	Not backed up; Not purged
Scratch	10TB	No Quota	not backed up; Purged when it is full
Local Backup	2.6TB	NA	No available for user

System Access

SSH

To ensure a secure login session, users must connect to machines using the secure shell, ssh program. Telnet is not allowed because of the security vulnerabilities associated with it. The "r" commands rlogin, rsh, and rcp, as well as ftp, are also disabled on this machine for similar reasons. These commands are replaced by the more secure alternatives included in SSH --- ssh, scp, and

sftp.

Before any login sessions can be initiated using ssh, a working SSH client needs to be present in the local machine. Go to http://www.hpcc.ttu.edu/connecting.php TTU HPCC introduction to SSH for information on downloading and installing SSH.

To initiate an ssh connection to a Chemdynm frontend (login node), execute the following command on your local workstation

ssh <login-name> @ chemdynm.chem.ttu.edu

Password changes (with the passwd command) are required to provide 8 char password.

Login Info

Login Shell

The most important component of a user's environment is the login shell that interprets text on each interactive command line and statements in shell scripts. Each login has a line entry in the /etc/passwd file, and the last field contains the shell launched at login. To determine your login shell, execute:

echo $SHELL {to see your login shell}

You can use the chsh command to change your login shell; instructions are in the man page. Available shells are listed in the /etc/shells file with their full-path. To change your login shell, execute:

cat /etc/shells {select a <shell> from list}

chsh -s <shell> <username> {use full path of the shell}

User Environment

The next most important component of a user's environment is the set of environment variables. Many of the Unix commands and tools, such as the compilers, debuggers, profilers, editors, and just about all applications that have GUIs (Graphical User Interfaces), look in the environment for variables that specify information they may need to access. To see the variables in your environment execute the command:

env {to see environment variables}

The variables are listed as keyword/value pairs separated by an equal (=) sign, as illustrated below by the HOME and PATH variables.

HOME=/home/utexas/staff/milfeld

PATH=/bin:/usr/bin:/usr/local/apps:/opt/intel/bin

(PATH has a colon (:) separated list of paths for its value.) It is important to realize that variables set in the environment (with setenv for C shells and export for Bourne shells) are "carried" to the environment of shell scripts and new shell invocations, while normal "shell" variables (created with the set command) are useful only in the present shell. Only environment variables are seen in the env (or printenv) command; execute set to see the (normal) shell variables.

File Systems

The Chemdynm have several different file systems with distinct storage characteristics. There

are predefined, user-owned directories in these file systems for users to store their data. Of course, these file systems are shared with other users, so they are managed by either a quota limit, a purge policy (time-residency) limit, or a migration policy.

Three Lustre file systems are available to users: $HOME, $WORK and $SCRATCH. Users have 100GB for $HOME. $WORK on our Chemdynm is NOT a purged file system, but is limited by a reasonable quota. Use $SCRATCH for temporary, large file storage; this file system is purged periodically (TBD), and has a very large quota. All file systems also impose a inode limit.

To determine the size of a file system, cd to the directory of interest and execute the "df -k ." command. Without the "dot" all file systems are reported. In the df command output below, the file system name appears on the left (luster target object) , and the used and available space (-k, in units of 1KBytes) appear in the middle columns followed by the percent used and the mount point:

```
% df -k .
Filesystem            1K-blocks      Used Available Use% Mounted on
chem-mds:/home        1378301968 379607204 928680964  30% /lustre/home
```

To determine the amount of space occupied in a user-owned directory, cd to the directory and execute the du command with the -sb option (s=summary, b=units in bytes):

```
du -sb
```

To determine quota limits and usage on execute the lfs quota command with your username and the directory of interest:

```
lfs quota -u <username> $HOME
lfs quota -u <username> $WORK
lfs quota -u <username> $SCRATCH
```

The four major file systems available on Chemdynm are:

home directory

At login, the system automatically changes to your home directory. This is the recommended location to store your source codes and build your executables. The quota limit on home is 100GB. A user's home directory is accessible from the frontend node and any compute node.

Use $HOME to reference your home directory in scripts.

Use cd to change to $HOME.

work directory

Store large files here. Often users change to this directory in their batch scripts and run their jobs in this file system. A user's work directory is accessible from the frontend node and any compute node.

The quota is 100GB(TBD) for each Users.

Purge Policy: There is no purging on this file system.

This file system is not backed up.

scratch or temporary directory

This is NOT a local disk file system on each node. This is a global Lustre files system for storing temporary files.

Purge Policy: Files on this system are purged when a file's access time exceeds TBD days.

PLEASE NOTE: Admin may delete files from scratch if the scratch file system becomes full and directories consume an inordinately large amount of disk space, even if files are less than TBD days old. A full work file system inhibits use of the file system for all users. The use of programs or scripts to actively circumvent the file purge policy will not be tolerated.

Often, in batch jobs it is more efficient to use and store files directly in $WORK (to avoid moving files from scratch later before they are purged).
The quota on this system is None.

Compilation

Please note that we currently do not have a cluster license for intel fortran compiler for the entire cluster. How ever individual users can get a academic license for free.

Compiler currently available to all users

C Compiler	gcc
Fortran	gfortran (formerly g77)
MPICH	IB-gcc-gfortan (location ./share/apps)
OPENMPI	IB-ICC-IFORT , IB-pathcc-pathf90 Location (/share/apps)

At this time only users with license for icc,ifort and pathscale can compile application using mpicc. Until we purchase cluster version of intel compilers you can compile your applications on hrothgar or other system and move the binary back to chemdynm.

To use a particular version of mpicc or mpirun use the following step to set path.

Eg: To use mpicc in /share/apps/mpi/openmpi/IB-icc-ifort

```
export PATH=/share/apps/mpi/openmpi/IB-icc-ifort-64/bin:$PATH
export LD_LIBRARY_PATH=/share/apps/mpi/openmpi/IB-icc-ifort-64/lib:$LD_LIBRARY_PATH
```

to compile a helloworld program

```
mpicc helloworld.c –o test.exe
```

Running Code

All jobs should be submitted on chemdynm using a SGE job submission script.
Following are examples/templates for submitting serial and parallel jobs. We will be adding to our documentation.

Serial job submission

```
[sysadm1@frontend-0 sysadm1]$ cat sleep.sh
#!/bin/bash
#$ -cwd
#$ -j y
```

```
#$ -S /bin/bash
#
date
sleep 10
date
```

Entries which start with #$ will be treated as SGE options.
-cwd means to execute the job for the current working directory.
-j y means to merge the standard error stream into the standard output stream instead of having two separate error and output streams.
-S /bin/bash specifies the interpreting shell for this job to be the Bash shell.

Parallel Job

For a parallel MPI job script, take a look at this script, linpack.sh. Note that you need to put in two SGE variables, $NSLOTS and $TMP/machines within the job script.

```
[sysadm1@frontend-0 sysadm1]$ cat linpack.sh
#!/bin/bash
#
#$ -cwd
#$ -j y
#$ -S /bin/bash
#
MPI_DIR=/opt/mpich/gnu/
HPL_DIR=/opt/hpl/mpich-hpl/

# OpenMPI part. Uncomment the following code and comment the above code
# to use OpemMPI rather than MPICH

# MPI_DIR=/opt/openmpi/
# HPL_DIR=/opt/hpl/openmpi-hpl/

$MPI_DIR/bin/mpirun -np $NSLOTS -machinefile $TMP/machines \
        $HPL_DIR/bin/xhpl
```

The command to submit a MPI parallel job script is similar to submitting a serial job script but you will need to use the -pe mpich N. N refers to the number of processes that you want to allocate to the MPI program. Here's an example of submitting a 2 processes linpack program using this HPL.dat file:

```
[sysadm1@frontend-0 sysadm1]$ qsub -pe mpich 2 linpack.sh
```

your job 17 ("linpack.sh") has been submitted

If you need to delete an already submitted job, you can use qdel given it's job id. Here's an example of deleting a fluent job under SGE:

```
[sysadm1@frontend-0 sysadm1]$ qsub fluent.sh
your job 31 ("fluent.sh") has been submitted
[sysadm1@frontend-0 sysadm1]$ qstat
job-ID  prior name       user         state submit/start at     queue                          master
ja-task-ID
------------------------------------------------------------------------------------------------
     31 0 fluent.sh    sysadm1      t     12/24/2003 01:10:28 comp-pvfs-                     MASTER
[sysadm1@frontend-0 sysadm1]$ qdel 31
sysadm1 has registered the job 31 for deletion
[sysadm1@frontend-0 sysadm1]$ qstat
[sysadm1@frontend-0 sysadm1]$
```

Although the example job scripts are bash scripts, SGE can also accept other types of shell scripts. It is trivial to wrap serial programs into a SGE job script. Similarly, for MPI parallel jobs, you just need to use the correct mpirun launcher and to also add in the two SGE variables, $NSLOTS and $TMP/machines within the job script. For other parallel jobs other than MPI, a Parallel Environment or PE needs to be defined. This is covered withn the SGE documentation

Submitting a specific queue

Chemdynm has two queues and two different host groups.
Nodes: C1-1, C1-2,C1-3,C1-4 have 32 G memory each and are grouped ad HiMem
Nodes:C1-5 to C1-32 have 16 G memory each and are grouped as All Hosts.

When a user sumits a job it is by default run on the default queue (that runs on host group All hosts).

To submit a job to the HiMemory systems

Please add the following line to your job submission script.

#$ -q HiMem