CIC cluster structure and management policies

Nguyen Tuan Duc, Keita Makino, Hiroki Asakawa
{duc, maki, ask}@nue.ci.i.u-tokyo.ac.jp

October 4, 2007

1 Introduction

As the result of the project "Parallel and distributed programming environment construction and management", the CIC cluster has been installed. The cluster is located in Engineering building 8 at Hongo campus.
This document describes the CIC cluster's [1] hardware / software structure as well as system performance, security and reliability.

2 Hardware structure

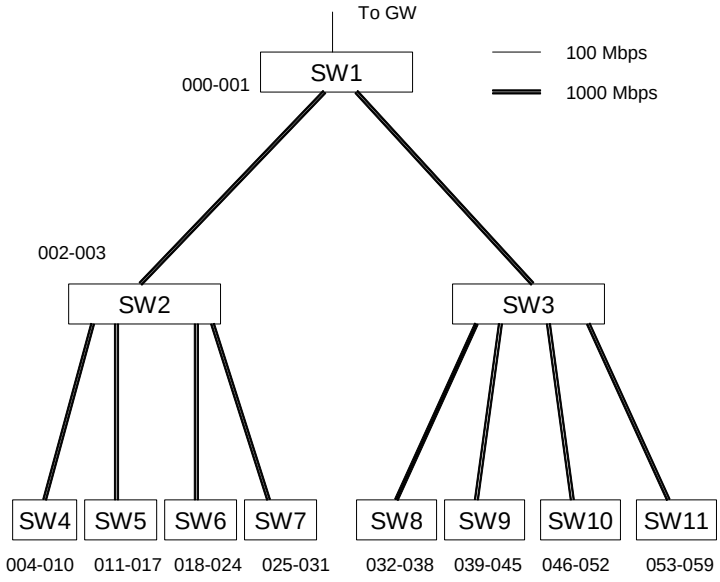

Figure 1: CIC cluster network structure

The cluster contains of about 60 PCs (most of these PCs have Intel HT 2.8GHz processor, 1GB of RAM and 120GB of HDD). The PCs are connected via gigabit-ethernet through a hierarchy of switches.
At the installation time, the cluster uses 11 switches (SW1 - SW11) as shown in figure 1. The switches form a tree structure to make sure that no loop exists. However, the tree structure may slow down system performance because of bottleneck at the root node (SW1 in figure 1). The remedy should be a change in

network topology or using high performance switches with many ports for the entire cluster.
Figure 1 also shows which PC is connected to which switch: host numbers are placed beside or bellow each switch, host name associated with host number in the form of cic{host_number}.cic.ci.i.u-tokyo.ac.jp. Users may use this information to reduce the communication bottleneck when execute applications.
The cluster uses the IP range of 133.11.213.128 - 133.11.213.255 (133.11.213.128/25).

3 Software structure

This section describes software structure of the cluster and the installation process including the installation of the Linux OS, NFS, NIS server etc..

3.1 Softwares used by CIC

CIC uses Fedora Core 6 as the operating system. Since FC6 supports kickstart installation, the installation process of the entire cluster is straightforward.
Each node has it own file system, and a mount point to share users' home directories (path /home in each node) via NFS. Currently, only cic000 exports its home folder, thus users may only use hard disks attached to cic000. A method to distribute the storage should be future work on the cluster.
User accounts are managed by the Network Information Service (Yellow Page service). Each user has the same account name, password and path to home directory at all nodes.
In addition to the above services, the cluster uses DNS to map host names to IP addresses. The DNS server is not contained in computation nodes (cic000 - cic059). Furthermore, the monitoring system (Ganglia), manual web pages etc. are hosted by a WWW server, this server is also not contained in the above computation nodes.
For parallel processing, LAM-MPI is installed at the shared folder /home/lammpi. All users may use this version of MPI to compile and execute their programs (of course, users can install MPI themselves, if they do not want to use LAM-MPI)

3.2 Installation of the Linux Operating System

The Fedora Core 6 distribution is first installed at a node by using installation CD-ROM. After the installation, this node exports a directory contains ISO images of the installation discs. The following line in /etc/exports file does the job:

```
/share/install    133.11.213.128/255.255.255.128(ro)
```

To automate the installation process, we use kickstart installation. The packages for installation, hostname, IP address ... information is stored in a kickstart configuration file (ks.cfg) and put to a HTTP server (http://install_server/ks.cfg). Here is some important information in the file ks.cfg:

```
auth  --useshadow  --enablemd5  --enablenis
--nisdomain=myname --nisserver=133.11.x.x
# Use NFS installation media
nfs --server=133.11.x.x --dir=/share/install/
# Network information
network --bootproto=static --device=eth6 --gateway=133.11.y.y
--ip=133.11.z.z --nameserver=133.11.x.x
--netmask=255.255.255.0 --onboot=on --hostname=myhostname
```

Since many machines do not have CD-ROM, we need to use network boot. To do this, the server node has to have a DHCP and TFTP service. At the server node, the root folder of TFTP contains PXE Linux configuration file, PXE boot loader, the Linux kernel (vmlinuz) image and the Initial Ram Disk (initrd.img). The PXE Linux configuration is as follows:

```
label linux
        kernel vmlinuz
        append initrd=initrd.img ramdisk_size=10000 root=/dev/ram
        ks=http://install_server/ks.cfg
```

To maintain user accounts information, the NIS server is installed at server node, all other nodes use the server node to obtain accounts information.
After such preparation, we can turn on all nodes and begin the automated installation process. The PXE boot client will find DHCP server to obtain IP address and TFTP server to download the PXE boot loader and Linux kernel. After loading the kernel into memory, the PXE boot loader will move control to the kernel entry point. The kernel will boot the machine as normal and start the Fedora Anaconda installer, provide it with the URL specified in the append string above (http://install_server/ks.cfg). The installer uses the kickstart configuration file as specified, mount the /share/install folder as type NFS and launch the installation process.
After the installation, each node has its hostname and IP address correctly assigned. The NIS domain name is also declared and users can remote login to every node from the server node.

3.3 Network File System (NFS) configuration

As the end of the above procedures, we can easily ssh-login to each node from other nodes. This is the time to initially mount the Network File System, to share the /home directory.
First, we need to determine which node should exports the /home directory. In CIC cluster, cic000 is used, because it is connected to SW1, the top-level switch. Thus, NFS server is installed at cic000, and a shared folder is exported. All nodes mount this folder as /home.
At this time, we can create user accounts and their home directory will be shared among nodes.

3.4 MPI installation

LAM-MPI is compiled at a node, and installed to a sub-directory of /home (/home/lammpi)
All users may use this version of MPI implementation to compile and execute their programs.

3.5 Ganglia installation

Ganglia is cluster monitoring system. It runs a daemon (gmond) at each nodes to collect information about the node (such as CPU utilization, memory usage, ...), and uses another daemon, called gmetad, to aggregate all nodes information to a central node. A web interface front-end is provided, which queries gmetad for the aggregated data and displays to web pages.
The installation method of Ganglia is thus gmond at each node, and gmetad at a node. A HTTP server supports PHP script is also needed to run the front-end of the software.
In our system, Ganglia web interface is at http://www.cic.ci.i.u-tokyo.ac.jp/ganglia/

. Users can use this interface to obtain information about the entire cluster, as well as each node.

4 Performance

In this section, we present some empirical results in evaluation of the CIC cluster.

4.1 CPU speed

Single CPU speed is measured. The benchmark performs a sequence of multiplications of 2 `double` variables.
The speed obtained (with optimizations such as loop un-rolling ...) is 1077 MFLOPS on cic039 (Pentium 4, 2.8GHz) and 1155 MFLOPS on cic006 (with Pentium 4, 3.0GHz). We believe this is the typical speed of CPUs in the cluster.
The performance for division of 2 `double` variables (with optimizations the same as above) is 65 MFLOPS and 70 MFLOPS on cic039 and cic006, respectively.
Thus, multiplication can be done 16 times faster than division of `double` numbers.

4.2 Peer-to-peer communication bandwidth

Table 1 shows bandwidth between a pair of nodes. The lines from 1 - 8 are results when 2 nodes connected to the same switch (switch number is in bracket).

Table 1: Bandwidth between pair of nodes

Pair	Bandwidth (Mbps)
cic004 - cic005 (SW4)	798.5
cic011 - cic012 (SW5)	719.5
cic018 - cic019 (SW6)	731.1
cic029 - cic030 (SW7)	721.0
cic032 - cic033 (SW8)	745.3
cic039 - cic040 (SW9)	673.0
cic046 - cic047 (SW10)	709.6
cic053 - cic054 (SW11)	680.4
cic004 - cic011	722.8
cic032 - cic039	682.2
cic004 - cic032	751.7

Even NICs and switches support gigabit-ethernet, bandwidth between 2 peers is approximately 700 Mbps because of overheads such as packet header.

4.3 Bi-section bandwidth

We measured bi-section bandwidth with a cut through SW1. The nodes participated in the experiment are cic004 - cic059, of which, nodes in the left branch of SW1 (as shown in figure 1) send and receive 100 MB (using TCP) to their counterpart nodes in the right branch.
The following is pseudo-code of the experiment:

```
if ( node_number < 32 ) then
     send_to( node_number + 28 );
     recv_from( node_number + 28 );
else
     recv_from( node_number - 28 );
     send_to( node_number - 28 );
endif
bisection_bandwidth = bandwidth_of_pair * number_of_pairs;
```

Bisection bandwidth is approximately 1050 Mbps, bigger than speed of the switch SW1 (1000Mbps). This is because many pairs of nodes communicate in parallel, but not synchronous (i.e., pairs may communicate in overlapping manner or pipelining manner). When the number of pairs becomes large, the chance for overlapping their work increases and we gain the speed. Though, the maximum bi-section bandwidth is around bandwidth of SW1 (1000Mbps), this means that if their are 20 pairs of processes communicate in parallel, peer-to-peer bandwidth is only 1000 / 20 = 50 Mbps. Thus, even the NIC at each node is gigabit-ethernet NIC, we can only reach the speed of 50Mbps on each node. This may slow down the performance of parallel applications.

4.4 Linpack benchmark result

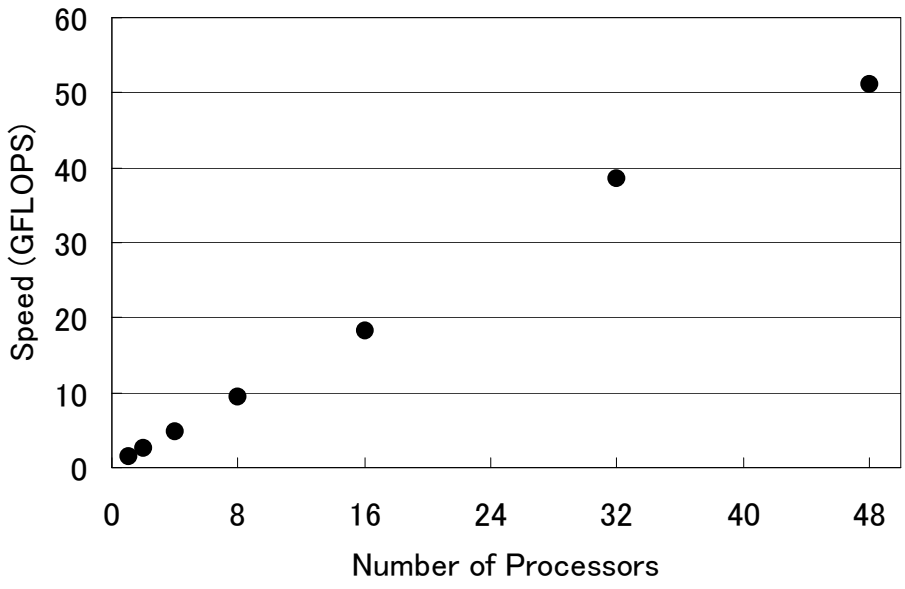

Figure 2: Linpack benchmark result

In this experiment we used the Linpack benchmark [3] to measure the speed of performing floating point operations. We executed the benchmark with 1, 2, 4, 8, 16, 32 and 48 processors and get the best speed while changing benchmark's parameters.
Figure 2 shows the result of Linpack benchmark. The speed obtained with 48 processors is 51.2 GFLOPS.

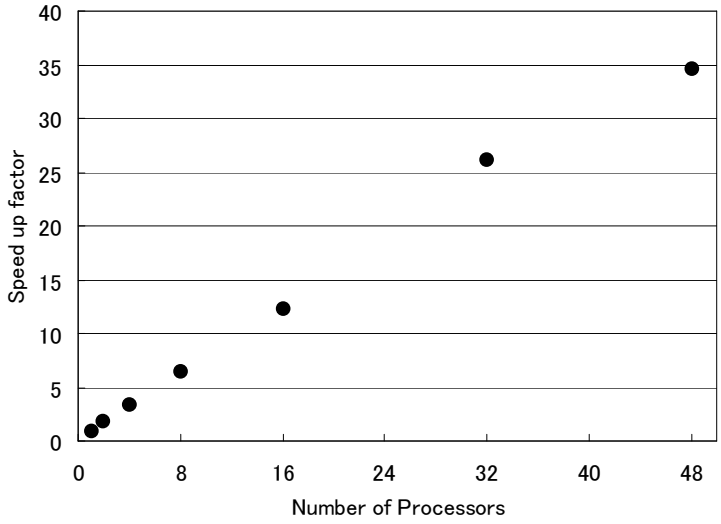

Figure 3: Speed-up factor for Linpack benchmark

Figure 3 shows the speed-up factor of the Linpack benchmark. The speed-up factor with 32 processors is 26.1 and with 48 processors is 34.5.

5 Security

To maintain the security and prevent attacks from outside networks, IP-table service is activated. The INPUT policy at each node is DROP, and accept only packets from the CIC-subnet (133.11.213.128/25).
Still, we need to provide users the ability to remote-login from outside networks, thus we allow a node to open its TCP port 22, for ssh from everywhere. After login to this node, users may freely go to other nodes in the cluster.
At the server node, we also open port 53 for DNS server and port 80 for HTTP server.

6 Reliability

This system is created with the hope that it will be reliable, but we can not guarantee anything. The hard disk at cic000 may corrupt at any time, and users' data may be lost. Furthermore, nodes may crash suddenly and users' processes may die without any warning. This cause many intricacies, especially for parallel programs. Parallel applications can fall to infinite-loop to wait for a crashed process to response something.
Thus, the users should conceive this problem clearly and check / backup their computation / data frequently to reduce the damage to the least when disaster occurs.

7 System management

Users who want to use CIC are required to register to get an account. The registration may be done simply by sending an email to admin@cic.ci.i.u-tokyo.ac.jp, with the information about Username and SSH DSA public key.

The cluster will be shutdown in some condition (e.g., power cut off, etc.). It may take up to a week to recover from such condition because the cluster is at Hongo campus, when members of the project are at Akihabara. We are looking for volunteers in Hongo campus to become cluster's administrator.
Users should read the User Manual [2] to clearly understand how to use the cluster (e.g., how to run MPI programs, Java programs, ...).

Acknowledgment

We would like to express our gratitude to many people who support us in this IST Hands-on project. We are grateful to our project supervisor, Professor Ikuo Takeuchi, who gave us the chance to participate in a very interesting project and help us to complete the cluster's infrastructure. We are also grateful to Professor Kei Hiraki and Associate Professor Mary Inaba for their precious advice and support in implementation of the cluster.
We are deeply indebted to our Engineering Partners, Mr. Yuura Katsuhiko and Dr. Kumeno Fumihiro who guide us through the project and provide valuable methodologies in group-working, documentation and management of the project.
Mr. Junji Tamatsukuri, Sasada Koichi and Seiichi Yamamoto gave us valuable advice, suggestions, technical support and solutions when we are posed with difficult problems.
Finally, we would like to thanks all people in the IST Hands-on program for their contribution to our project and for providing an interesting course and financial support to IST students like us.

References

[1] http://www.cic.ci.i.u-tokyo.ac.jp

[2] H. Asakawa et al., CIC cluster user manual. http://www.cic.ci.i.u-tokyo.ac.jp/manual/

[3] Jack Dongarra, Piotr Luszczek, Antoine Petitet. The LINPACK Benchmark: Past, Present, and Future. http://www.netlib.org/utk/people/JackDongarra/PAPERS/hpl.pdf.