Cluster Based Performance enhancement of plasma simulation codes

Major Project Report

Submitted in partial fulfillment of the requirements
For the degree of

Master of Technology
in
Information & Communication Technology

Prepared By

Pranavkumar Rameshbhai Manvar
11MICT09

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY
AHMEDABAD-382481
May 2013

Cluster Based Performance enhancement of plasma simulation codes

Major Project Report

Submitted in partial fulfillment of the requirements

For the degree of

Master of Technology in Information & Communication Technology

Prepared By

Pranavkumar Rameshbhai Manvar

11MICT09

Under the guidance of

External Project Guide	Internal Project Guide
Mr. N. Ravi Prakash	Dr. Madhuri Bhavsar
Engineer SF,	Sr Associate Professor, Section Head IT,
RHRTD Division,	Department of Information Technology,
Institute for Plasma Research,	Nirma University,

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

INSTITUTE OF TECHNOLOGY

NIRMA UNIVERSITY

AHMEDABAD-382481

Declaration

I,Pranavkumar Rameshbhai Manvar, 11MICT09, give undertaking that the Major Project entitled "Cluster Based Performance enhancement of plasma simulation codes " submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in Institute of Technology of Nirma University, Ahmedabad, is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

-Pranavkumar Rameshbhai Manvar

Certificate

This is to certify that the Major Project entitled"**Cluster Based Performance enhancement of plasma simulation codes** "submitted by Pranavkumar Rameshbhai Manvar(11MICT09), towards the partial fulfillment of the requirements for the degree of Master of Technology in Information and Communication Technology of Nirma University, Ahmedabad is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination. The results embodied in this major project , to the best of my knowledge, have not been submitted to any other university or institution for award of any degree.

....
Mr. N. RaviPrakash
External Guide
Institute for Plasma Research

....
Dr. Madhuri Bhavsar
Internal Guide
Nirma University

....
Prof. Gaurang Raval
PG Coordinator - ICT
Nirma University

....
Dr. Sanjay Garg
HOD CSE
Nirma University

....
Dr. K. Kotecha
Director, Institute of Technology
Nirma University

Acknowledgments

With immense pleasure, I would like to present this report on the part of my dissertation work related to **"Cluster Based Performance enhancement of plasma simulation codes "**. I am very thankful to all those who helped me for the successful completion of the first phase of the dissertation and for providing valuable guidance throughout the project work. This work is carried out and financially supported by **INSTITUTE FOR PLASMA RESEARCH,GANDHINAGAR**.
I hereby take the opportunities to express my deep sense of gratitude to my respected industry guides, **Mr. Ravi Prakash N.**, Engineer 'SF'; **Mr. Naveen Rastogi** and **Mr. Nisarg Patel** for their valuable, precious guidance and active support during the project. I heartily thankful to them for their time to time suggestion and the clarity of the concepts of the topic that helped me a lot during the study.
I would like to offer thanks to **Dr. Madhuri Bhavsar**, Institute Guide who helped me in pointing the need of the project, suggestions and encouragement during the time of research and also for writing of this report.
My sincere thanks and gratitude to **Prof. Gaurang Raval**, PG-Coordinator, ICT , Computer Department, Nirma University ; **Dr. Sanjay Garg**, HOD ,Computer Science and Engineering Department, Nirma University ; **Dr. K. Kotecha**, Director, Institute of Technology, Nirma University for their continual kind words of encouragement and motivation throughout the Dissertation work.
Finally, I would like to thank The Almighty, my family members for supporting and encouraging me in all possible ways. I would also like to thank my friends who have provided continuous encouragement in making this dissertation work successful.

-Pranavkumar Rameshbhai Manvar
11MICT09

Abstract

Simulation codes are presently running on standalone systems. They are all compute intensive which needs large number of resources like memory and high processing power. Some highly compute intensive codes are running on super-computer. This may not be a cost effective option. Thus there is a need to create an HPC environment compatible to run them. These codes need a cluster where each nodes are dedicated to it. It is difficult to have dedicated clusters for each of these codes and hence a grid environment is required which may have non-dedicated systems attached together to generate high processing power. Also all these codes are to be converted into cluster compatible code.

Each codes may require different operating systems and different parallel processing environment,thus each nodes in cluster may have different operating systems like Ubuntu, fedora, red het etc. Therefore configuration steps may differ based on the operating systems running on each node . Also some code may use PVM and some may use MPI. MPI used by each code may also differ, vise Intel Mpi, LAM Mpi, HP Mpi etc. These led to the requirement of developing an automatic cluster configuration tool and management tool,as the number of nodes in cluster configurations grows, installation, configuration, and administration becomes more challenging.

Objective

Cluster configuration requires changes to be made in system files of operating system. Cluster configuration differs for each operating system. As number of nodes increases in the cluster, configuration becomes very difficult. Hence a tool is required which may take care of the configuration process for each node independent of the OS running on that node.

Managing a cluster having large number of nodes, and keeping track of load on each node, up time and down time of a node, jobs assigned to individual nodes or jobs successfully completed by them is a cumbersome task. Cluster management tool can reduce the overhead of monitoring and managing the load on the system and individual nodes in the cluster.

These Simulation codes may take several hours to several days to complete the execution and hence failure of a single node in the system may require to restart the job which has already completed its run of several hours from the beginning. Restarting the job from the beginning will consume double the amount of time it actually needed. Failure management with check-pointing can safely remove the malfunctioning node and restart the job from last committed checkpoint.

Contents

Declaration **iii**

Certificate **iv**

Acknowledgments **v**

Abstract **vii**

List of Figures **x**

1 Introduction **1**

1.1 Supercomputer or Dedicated Cluster: 2

1.2 Grid Environment: 2

2 Grid Environment Setup **5**

2.1 OS installation: 5

2.2 Establishment of Network: 8

2.3 Remote Login via ssh: 9

2.4 Network File Sharing: 10

2.5 MPI installation 11

3 Monitoring Tool Setup **14**

3.1 Pre-requisites 14

3.2 Assumptions 14

3.3 Steps 15

4 Cluster Configuration Tools Comparison **19**

4.1 OSCAR 19

4.2 CLUSTERIT 20

5 IPR Cluster Tool **23**

5.1 How Cluster Tool Works? 23

5.2 Bash files 27

5.3 Monitoring Tool : 30

6 Neutronics Installation Process **35**

7 Failover Managment **40**

7.1 How Neutronics Works? 41

8 Result **44**

References **47**

List of Figures

2.1 Cluster 6

5.1 Flow Chart of Cluster Tool 23
5.2 GUI at Client Side 24
5.3 GUI at Server Side 25
5.4 GUI of Management Tool 26
5.5 Bash Files 27
5.6 Gnaglia Monitoring Tool snapshot 31
5.7 Gnaglia Monitoring Tool snapshot:CPU metrics 32
5.8 Gnaglia Monitoring Tool snapshot:Disk metrics 32
5.9 Gnaglia Monitoring Tool snapshot:Memory metrics 33
5.10 Gnaglia Monitoring Tool snapshot:Network metrics 34
5.11 Gnaglia Monitoring Tool snapshot:Process metrics 34

7.1 Flow of Failover Management 41
7.2 Neutronics 41
7.3 Tally Table 42

8.1 Neutronics Result 44

Chapter 1

Introduction

Ansys is running on single workstation but if we want to run it in distributed environment then we need same type of workstations with same architecture and 10gbps Ethernet Speed. In grid environment it is not possible that all workstations have same architecture. So one have to modify its source files and make it grid compatible. Ansys has collaboration with intel MPI and HP-MPI which are not opensource .In readymade cluster, it uses LAM MPI and some other mpi.

Same as Ansys, some of the plasma simulation code which uses PVM is running on single workstations but if we want to run it in distributed environment, it will use all the cores of a node. That is if workstation has two processors with 8 core, it will use 4 core or 8 core as it is not possible to define how many cores per node is to be used in PVM. It is running on supercomputer because all nodes are dedicated nodes, have similar architecture and operating system. So we cannot define no. of subtasks for execution in PVM.

Hence a generalized and an automated tool for cluster or grid configuration is required. In that tool user will enter only root password and IP addresses of nodes to be connected in cluster, type of communication mechanism required irrespective of the type of OS.

1.1 Supercomputer or Dedicated Cluster:

It's expensive, most likely made out of proprietary hardware and software, small in size and most likely all coupled with the application, backend, etc.We need and must have this type of infrastructure in order to meet our SLA requirements. HPC or Cluster Computing (I use them interchangeably despite the bad comments that I will receive after writing this article) is "small" in size – a few hundred nodes and as high as a couple of thousand nodes. Each node is highly optimized to perform at its peak, and the application is configured such that it takes advantage of the node. For example, if one have four cores per node and 4GB RAM per Core, the chances are that application was designed (or modified) to take this bit of information into account. We do not like to implement any security with this type of infrastructure. We like security to be only at the edges, and the nodes to do what they are good at: computation and computation only.

1.2 Grid Environment:

Grids, however, benefit from the fact that they are heterogeneous in nature. One may have nodes that are a few years old, and one may have nodes that are brand new. It is up to the resource manager, which we will discuss at a later article, to take advantage and find the best fit for a given request. The main point of Grid is to create a single system image (SSI) with the nodes that it has access to. From a client's perspective, they are accessing a single node; a single image which is all powerful. This makes the resource manager's job very challenging, and one of the main reasons that Grid nodes go underutilized for much of their life span.

- **Advantages:**
  - No need to buy large six figure SMP servers for applications that can be split up and farmed out to smaller commodity type servers. Results can then be concatenated and analyzed upon job(s) completion.

  - Much more efficient use of idle resources. Jobs can be farmed out to idle servers or even idle desktops. Many of these resources sit idle especially during off business hours. Policies can be in place that allows jobs to only go to servers that are lightly loaded or have the appropriate amount of memory/cpu characteristics for the particular application.
  - Grid environments are much more modular and don't have single points of failure. If one of the servers/desktops within the grid fail there are plenty of other resources able to pick the load. Jobs can automatically restart if a failure occurs.
  - Policies can be managed by the grid software. The software is really the brains behind the grid. A client will reside on each server which sends information back to the master telling it what type of availability or resources it has to complete incoming jobs.
  - This model scales very well. Need more compute resources? Just plug them in by installing grid client on additional desktops or servers. They can be removed just as easily on the fly. This modular environment really scales well.
  - Upgrading can be done on the fly without scheduling downtime. Since there are so many resources some can be taken offline while leaving enough for work to continue. This way upgrades can be cascaded as to not effect ongoing projects.
  - Jobs can be executed in parallel speeding performance. Grid environments are extremely well suited to run jobs that can be split into smaller chunks and run concurrently on many nodes. Using things like MPI will allow message passing to occur among compute resources.

- **Disadvantages:**
  - For memory hungry applications that can't take advantage of MPI one

may be forced to run on a large SMP.

- One may need to have a fast interconnect between compute resources (gigabit ethernet at a minimum). Infiband for MPI intense applications
- Some applications may need to be tweaked to take full advantage of the new model.
- Licensing across many servers may make it prohibitive for some apps. Vendors are starting to be more flexible with environment like this.

Chapter 2

Grid Environment Setup

Cluster Environment Setup is divided into five steps:

- OS installation
- Establishment of Network
- Remote login
- Network File Sharing
- MPI installation

2.1 OS installation:

Hardware Requirement: Atleast 2 deskstops/workstations/servers

Operating System: Fedora Full Version not desktop Version

Ethernet Card: 100Mbps/1Gbps/10Gbps

Ethernet Cable: 100Mbps/1Gbps/10Gbps

Ethernet Switch: Depends on Ethernet Card and Cable

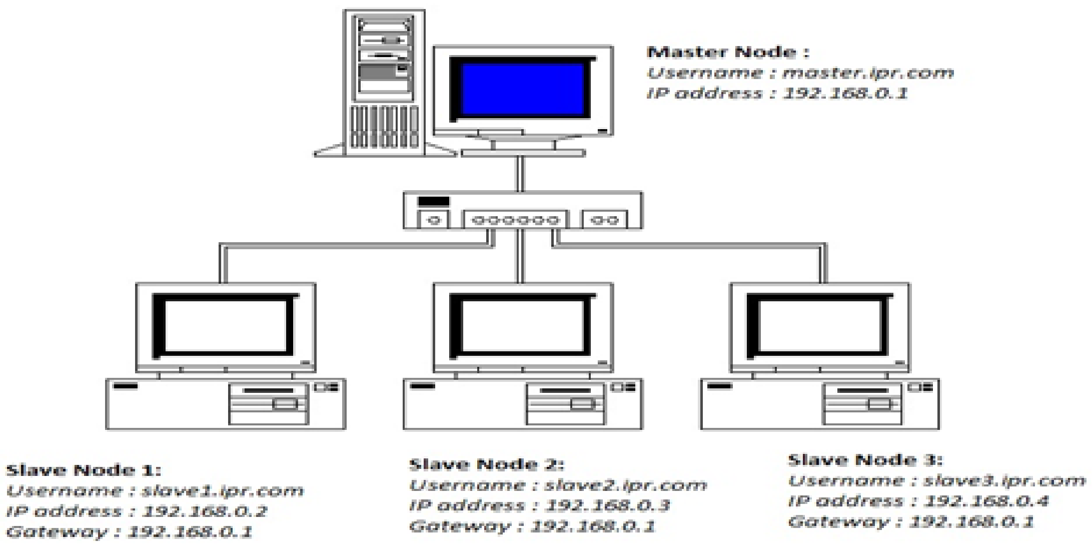


Figure 2.1: Cluster

Define Master node of your Cluster and remaining nodes are Slave node.

Make partions during installation for /soft and /home1 in all nodes. (*/ Here, All nodes means master node and slave nodes, If I want to talk about master node or slave node I will specify. /*)

Master Node:

Username: master

Full Name: master.ipr.com

IP address: 192.168.0.1 (we are use switch so master node itself work as Gateway)

Slave Nodes:

Node 1:

Username: slave1

Full name: slave1.ipr.com

IP address: 192.168.0.2

Gateway: 192.168.0.1

Node 2:

Username: slave2

Full name: slave2.ipr.com

IP address: 192.168.0.3

Gateway: 192.168.0.1

Turn off Firewall in all nodes:

(System − >Administration − > Firewall − > Disable) Go to the System -− >Administration − > SELinux Management (If SELinux Management is present then do next)

System Default Enforcing Mode − > Disable

System Default Policy Type − >targeted

Do Check mark on Relabel on next reboot

Install GCC, G++, G77 (necessary for MPI)

Create user:

For Master node:

System − > Administration − > Users and Groups − > Add User

User Name: srv

Full Name: master.ipr.com

Login Shell: /bin/bash

Home Directory: /home1/srv

For Slave nodes:

System − > Administration − > Users and Groups − > Add User

User Name: srv

Full Name: slave (1/2).ipr.com (1/2 means 1 or 2)

Login Shell: /bin/bash

Home Directory: /home1/cli(1/2)

2.2 Establishment of Network:

Check the master node is connected with Slave node or not, using **ping** command.
From **master node:** ping 192.168.0.2(for slave1) and ping 192.168.0.3(for slave2)
If it is giving unreachable then check the connection and solve it.
Edit /etc/hosts for all nodes: **Master node:**

```
192.168.0.1 master.ipr.com master
192.168.0.1 master.ipr.com srv
127.0.0.1 localhost.localdomain localhost localhost4
::1 master.ipr.com master localhost6.localdomain6 localhost6
192.168.0.2 slave1.ipr.com cli1
192.168.0.3 slave2.ipr.com cli2
```

Slave Node1 (slave1 or cli1):

```
192.168.0.2 slave1.ipr.com slave1
192.168.0.2 slave1.ipr.com cli1
127.0.0.1 localhost.localdomain localhost localhost4
::1 slave1.ipr.com cli1 localhost6.localdomain6 localhost6
192.168.0.1 master.ipr.com srv
192.168.0.3 slave2.ipr.com cli2
```

Slave Node2 (slave2 or cli2):

```
192.168.0.3 slave2.ipr.com slave2
192.168.0.3 slave2.ipr.com cli2
127.0.0.1 localhost.localdomain localhost localhost4
::1 slave2.ipr.com cli2 localhost6.localdomain6 localhost6
192.168.0.1 master.ipr.com srv
```

192.168.0.2 slave1.ipr.com cli1

Try command from master node(srv) ping cli1 and ping cli2

2.3 Remote Login via ssh:

Master Node:

In Terminal:

$ ssh-keygen

Enter file in which to save the key (/home1/srv/.ssh/id_rsa):**/home1/srv/.ssh/id_rsa_srv**

Enter passphrase (empty for no passphrase): **just press enter key**

Enter same passphrase again: **just press enter key**

(You public and private key will be generated in **/home1/srv/.ssh**)

Slave Nodes:

In Terminal:

$ ssh-keygen

Enter file in which to save the key (/home1/srv/.ssh/id_rsa):**/home1/srv/.ssh/id_rsa_cli1/2** (1 or 2)

Enter passphrase (empty for no passphrase): **just press enter key**

Enter same passphrase again: **just press enter key**

(You public and private key will be generated in /home1/srv/.ssh)

Continue in cli(1/2)

$ cat /home1/cli1/.ssh/id_rsa_cli1.pub >> **/home1/cli1/.ssh/authorized_keys**

Now copy **id_rsa_srv.pub** file content from master node and paste it in **authorized_keys** of slaves.

So Now in authorized_keys has two public keys one is its own public key and another one is masters public key.

For all Nodes:

Now go to **/etc/ssh/sshd_config**

RSAAuthentication yes

AuthorizedKeyFile /home1/srv/.ssh/authorized_keys

(this is for master node,same for slaves but difference is /home1/cli(1 or 2)/.ssh/authorized_keys)

Now **restart service sshd** in all node.

(System − > Administration − > Services − > sshd)

Now try to login Remotely from master node to slave node using

From master node:

$ ssh x cli1 (or cli2)

2.4 Network File Sharing:

For master node only:

$ su

Password: enter root password

$ vi /etc/exports

(this file will be empty,you have to add one line)

/soft 192.168.0.2/24 (rw,no_root_squash,sync)

$ exportfs -avr

Now restart service **NFS**

(System − > Administration − > Services − > nfs)

Slave nodes:

$su

Password:enter root password

$ mount t 192.168.0.1:/soft /soft

(for unmount use **umount /soft**)

```
$ umount /soft
```

2.5 MPI installation

First of extract lam-mpi file in **/soft**

```
$ cd /soft/lam-mpi
$ ./configure -prefix=/soft/lam
```

(lam is installation folder and lam-mpi is source folder)

```
$ make
$ make install
```

For All nodes:

```
$ vi /home1/srv/.bash_profile
# .bash_profile
# Get the aliases and functions
if [ -f ~/.bashrc ]; then
    . ~/.bashrc
Fi
# User specific environment and startup programs
PATH=$PATH:$HOME/bin:/soft/lam/bin:

export PATH
export LAMRSH='ssh -x'
export LAM=/soft/lam/bin
```

Make same .bash_profile in slave nodes.

Make .bashrc

```
$vi /home1/srv/.bashrc
# .bashrc
# Source global definitions
```

```
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi
export PATH=/soft/lam/bin:$PATH
# User specific aliases and functions
```

Make same .bashrc in slave nodes.

Make .cshrc

$vi /home1/srv/.cshrc

```
set path = (/soft/lam/bin $path)
```

Make same .cshrc in slave nodes.

Edit /etc/bashrc

At the end of file add below line:

```
export LAM=/soft/lam/bin
```

Make same in slave nodes.

Now check on nodes using $ mpirun command that lam-mpi is installed or not.

Now on Master node only :

$ vi /home1/srv/lamhosts

srv (cpu=8)

cli1 (cpu=8)

cli2 (cpu=2)

How to run mpi program in cluster ?

Slave nodes: (login with cli(1 or 2))

```
$ mount 192.168.0.1:/soft /soft
```

Master nodes: (copy hello.c in /home1/srv) (login with srv)

```
$ cd /home1/srv
$ lamboot v lamhosts (// root cannot boot lam mpi //)
$ mpicc o hello hello.c
$ mpirun hello ( or ./hello)
```

Chapter 3

Monitoring Tool Setup

3.1 Pre-requisites

- You must have NTP configured on your cluster and the time on all nodes must be synchronized (note: after starting ntpd this may take half an hour)
- You must have the Ganglia source code file downloaded (available at http://sourceforge.net /projects /ganglia/files/ganglia monitoring core/3.2.0/ganglia-3.2.0.tar.gz/download [mirror]) on the head node
- You must have the RRDtool source code file downloaded (available at http://oss.oetiker.ch /rrdtool/pub/rrdtool-1.4.5.tar.gz [mirror]) on the head node
- You must have the libConfuse source code file downloaded (available at http://savannah.non .org/download/confuse/confuse-2.7.tar.gz [mirror]) on the head node

3.2 Assumptions

- We will assume that there are 7 nodes which are available, node1 through node7.
- All of the source code files are located in /root/.

Overview

Ganglia will allow you to monitor your cluster's performance in an easy to use web interface. This will help let you know whether your software is utilizing your cluster appropriately as well as provide a debugging and alert tool for any problems which occur.

Ganglia itself consists of three separate parts: gmond is the process which will run on each node, collecting and aggregating statistics about the node, gmetad will format and save the statistics for final display, and web will provide the user with a web interface containing all of the data for the cluster.

3.3 Steps

Note: The hash sign (#) serves as a reminder to type the command in as the root user.

Installing RRDtool

Extract the source code

1. # cd /root

   # tar xzvf rrdtool-1.4.5.tar.gz

   Then, install to /opt/rrdtool

2. # cd rrdtool-1.4.5
3. # ./configure –prefix=/opt/rrdtool
4. # make

   # make install

Installing libConfuse

Extract the source code

1. # cd /root

   # tar xzvf confuse-2.7.tar.gz

   Then, install to /opt/libconfuse

2. # cd confuse-2.7

3. # ./configure –enable-shared –prefix=/opt/libconfuse
4. # make
5. # make install

   # ln -vs /opt/libconfuse/lib /opt/libconfuse/lib64

Building Ganglia

First, configure the development environment

1. # yum install -y httpd-devel

   # yum install -y pcre-devel

   Then, extract the source code

2. # cd /root
3. # tar xzvf ganglia-3.2.0.tar.gz

   Now, build and install Ganglia with the custom RRDtool and libConfuse directories

4. # cd ganglia-3.2.0
5. # ./configure –with-librrd=/opt/rrdtool/

   –with-gmetad –with-libconfuse=/opt/libconfuse

   –prefix=/opt/ganglia

6. # make
7. # make install

Installing and Configuring Ganglia (gmond)

First, gmond should be configured to start at boot time

1. # cp /root/ganglia-3.2.0/gmond/gmond.init /opt/ganglia
2. # sed -i 's@GMOND=/usr@GMOND=/opt/ganglia@' /opt/ganglia/gmond.init
3. # cp /opt/ganglia/gmond.init /etc/rc.d/init.d/gmond
4. # chkconfig –add gmond

   Then, on each node

5. node# cp /opt/ganglia/gmond.init /etc/rc.d/init.d/gmond

   node# chkconfig –add gmond

   Now, configure gmond itself. First, a default configuration file should be created

6. # cd /opt/ganglia/etc

   # /opt/ganglia/sbin/gmond -t > gmond.conf

   Then, edit /opt/ganglia/etc/gmond.conf to change the cluster name from unspecified to uhssc. You may change any of the other information as desired (location, owner, etc)

   Now, you should be ready to start gmond on your cluster! Since it uses a multicast setup you do not need to explicitly tell it what nodes there are

7. # chkconfig gmond on

   # service gmond start

   and on all of the nodes, run

   node# yum install -y apr

   node# chkconfig gmond on

   node# service gmond start

Installing and Configuring Ganglia (gmetad)

gmetad will collect the information from your entire cluster and save it in a clever format (RRD). First, create the directory to store the RRD files

1. # mkdir -p /opt/ganglia/var/lib/ganglia/rrds

   # chown nobody:nobody /opt/ganglia/var/lib/ganglia/rrds

2. Then, modify /opt/ganglia/etc/gmetad.conf with the following changes:

   1. Change the line

   data_*source*"*mycluster*" localhost

   to

   data_*source*"*uhssc*" localhost

   2. Add to the bottom of the file

   rrd_*rootdir* "/opt/ganglia/var/lib/ganglia/rrds"

   Configure gmetad to start when the head node boots up

3. # cp /root/ganglia-3.2.0/gmetad/gmetad.init /etc/rc.d/init.d/gmetad

4. # sed -i 's@GMETAD=/usr@GMETAD=/opt/ganglia@' /etc/rc.d/init.d/gmetad

   # chkconfig –add gmetad

Finally, start gmetad!

5. # chkconfig gmetad on

   # service gmetad start

Installing and Configuring Ganglia (web)

For the last part, you need to configure Ganglia to display the information. First, install a web browser and PHP

1. # yum install -y php php-gd
2. # yum install -y firefox

   # service httpd restart

   Copy the Ganglia web folder to be accessible for the web server

   # cp -a /root/ganglia-3.2.0/web /var/www/html/ganglia

   Configure web to use the /opt prefix. In the file /var/www/html/ganglia/conf.php, make the following changes

   1. Change the line

   $gmetad_*root* = "/var/lib/ganglia";

   to

   $gmetad _*root* = "/opt/ganglia/var/lib/ganglia";

   2. Change the line

   define("RRDTOOL", "/usr/bin/rrdtool");

   to

   define("RRDTOOL", "/opt/rrdtool/bin/rrdtool");

Now, create the template directory

3. # mkdir -p /opt/ganglia/var/lib/ganglia/dwoo

   # chown apache:apache /opt/ganglia/var/lib/ganglia/dwoo

4. Finally, you should be able to open Firefox and browse to http://localhost/ganglia to

   view your cluster information.

Chapter 4

Cluster Configuration Tools Comparison

4.1 OSCAR

Open Source Cluster Application Resources ,it is an open-source project software.Main purpose of this tool is to build high-performance clusters. OSCAR is available for Linux only. The goal of OSCAR is to make cluster easily with configurable option for systems based on Linux.

OSCAR contains a number of facilities, including cluster-management tools, a message passing interface based on the MPI standard, a job queuing system, and a batch scheduler. System images can be built on an OSCAR server and downloaded to OSCAR clients for execution using the OSCAR Installation Wizard. The environment is straightforward to install and configure, providing good scalability at minimal cost.

Drawbacks:

- Nodes are dedicated nodes.
- Supports SSH login only.
- Supports only MPI.

- Same Linux distribution and version as the server node
- All clients must have the same architecture (e.g., ia32 vs. ia64)
- OSCAR runs under RedHat Linux 7.1 and later or Mandrake Linux 8.2 and later
- Does not include failover management.

4.2 CLUSTERIT

This is a collection of clustering tools, to turn your ordinary everyday pile of UNIX workstations into a speedy parallel beast.[4] Fast parallel execution of remote commands. C vs. Perl. You do the math.

Heterogeneous cluster makeup

This makes it very easy to administer a large number of machines, of varying architectures, and operating systems. The fact that my tools are completely architecture independent, make it possible to dsh commands out to machines that aren't even running the same OS! This can be useful for a variety of mass administration tasks an admin may have to undertake.

Choice of authentication

IBM forces you to use kerberos 4 for authentication on the SP's. This is actually fine for a closed environment like an SP, but for something to be run on just a stack of otherwise useful boxes, you need more freedom. This suite allows you to do whatever you like.. ssh, kerberos, .rhosts. Whatever suits your security needs best.

Sequential node, and random node execution

The idea here is that these dsh-like programs allow you to do something akin to load balanced scripting. For example one could set up an NFS shared build directory, and issue the command:

make -j4 CC="seq 'cd /usr/src/foo ; gcc'"

Which would execute a build in parallel, on 4 nodes in your cluster, assigning processes to each node in sequence. The run command is equivalent to saying: "I don't care where you run, just run and tell me how things turned out." Generally speaking, the run command will achieve better results as the size of the cluster increases. If you have only three nodes, the odds of getting the same node over and over are fairly good.

Job sequencing

It is possible using this package to schedule processes on the remote machines, so that no more than one process per machine is active at any one time. This was designed to combat problems with using sequence for parallel builds. When building in parallel with sequence, it is possible that a node receives a task that will take it much longer than the other nodes to complete. It is also possible that as other nodes finish their jobs faster, the node which has been bogged down is handed another job. When performing large parallel builds, eventually very slow machines will stall the entire build, as they are attempting to compile many objects at once, and are usually at this point near-death from swapping.

The Job Scheduling in ClusterIt can prevent this in two ways. First, the job scheduling will not allow a node to process any more than one command at a time. If more commands than nodes are requested, the excess commands will block until a node has freed up. Second, the scheduler has the ability to register a benchmark number of some sort for each node. This allows the scheduler to always give out the fastest of the remaining nodes whenever one is requested. This allows a parallel build to more efficiently utilize a heterogeneous cluster.

Barrier sync for shell scripting

This is a new idea. The barrier mechanism consists of a daemon run on a host, and a client which can be used to barrier sync with. An example of use would be:

```
#!/bin/sh
do something
barrier -h host -k token -s 5
do something else
```

You would then dsh the execution of this script to your hosts. The barrier makes sure that all hosts have completed the first ”something” before the continue on to the next something. The -s, is the level of parallelism for the script, ie: how many processes to wait for before continuing.

Distributed Virtual Terminals

This is a parallel interactive execution environment. The user is given windows for each host in the cluster, and a central management window. Keystrokes typed on the central management window, will be relayed to all of the subordinate windows. This allows the user to vi a file on 20 machines simultaneously, for example. You can also select a window, and use it like a normal xterm, to perform actions on just that host.

What my solution does not provide:

A parallel programming API Use MPI, or PVM, or whatever for that, thats outside the scope of this suite.[4]

Chapter 5

IPR Cluster Tool

5.1 How Cluster Tool Works?

This is a flow diagram of Cluster Tool.First of all it checks User is root or not,if user it not root then it generate error message.After that user has to enter Server name and IP address.then it will check it is connected with server or not.MPI installation is not possible in root so at least one user should be created.After that user enters user name and its location.After that user selects MPI from drop down list and then select Remote Login option.At last give path for shared folder from programs will run and will be mounted from client to server.

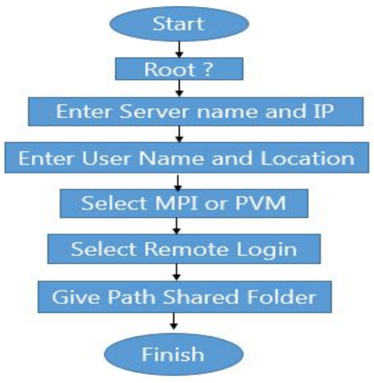
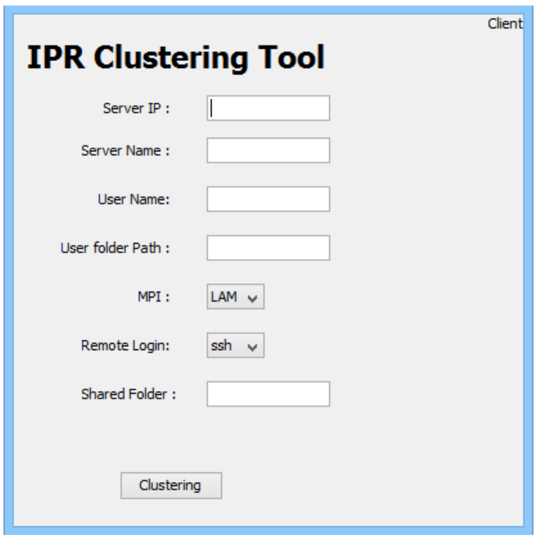


Figure 5.1: Flow Chart of Cluster Tool

Client Side:


Client

IPR Clustering Tool

Server IP :

Server Name :

User Name:

User folder Path :

MPI : LAM

Remote Login: ssh

Shared Folder :

Figure 5.2: GUI at Client Side

Steps:

- Enter the Server IP and Server Name.
- Enter the Username and User Folder Path of System.
- Select MPI or PVM (Lam,Intel,HP).
- Select Remote Login (ssh,rsh).
- Provide shared folder path for network sharing.

Behind the GUI :

- Using Server IP and Name it will check whether it is connected to the server or not?

- Using Username and User folder path it will check for cluster user and its directory.
- Tool have to modify .cshrc file which is can be found in the user folder
- Using Selected MPI it will select bash file which is for MPI installation
- Before MPI installation, it will check whether gcc and fortran are installed or not
- Using selected remote login option, it will run bash file and configure that remote login service

Server Side:

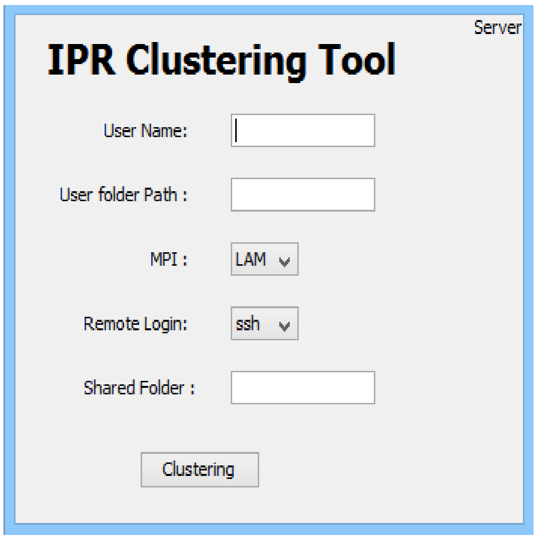


Figure 5.3: GUI at Server Side

Steps:

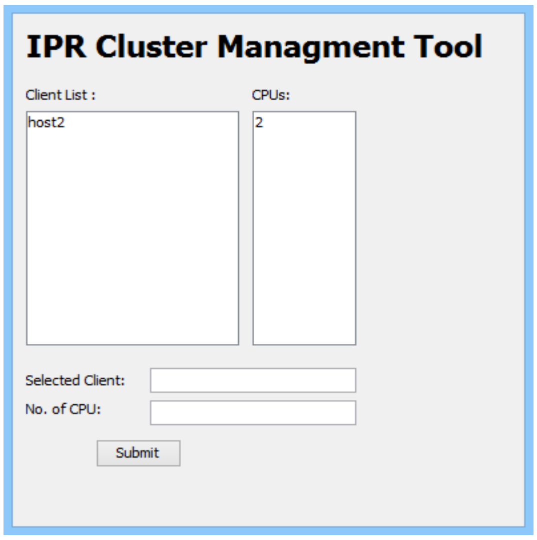
- Enter the Username and User Folder Path of System.
- Select MPI or PVM (Lam,Intel,HP).

- Select Remote Login (ssh,rsh).
- Provide shared folder path for network sharing.

Behind the GUI :

- Using Username and User folder path it will check for cluster user and its directory.
- Tool have to modify .cshrc file which is can be found in the user folder
- Using Selected MPI it will select bash file which is for MPI installation
- Before MPI installation, it will check whether gcc and fortran are installed or not
- Using selected remote login option, it will run bash file and configure that remote login service

Cluster Management Tool:



host2
2

Figure 5.4: GUI of Management Tool

In IPR cluster Management tool user will see how many CPUs are shared with its client name.
Client selected by admin will be automatically printed in the textbox which is not editable and admin can edit no. of CPUs needed.
Behind GUI:

- It will read lamhost file to get Client list and CPUs.
- Click on submit button it will modifiy lamhost file.

5.2 Bash files

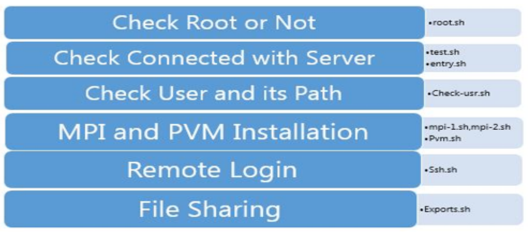


Figure 5.5: Bash Files

Root.sh

```
#!/bin/bash
# Init
FILE="/tmp/out.$$"
GREP="/bin/grep"
#....
# Make sure only root can run our script
if [ "$(id -u)" == "0" ]; then
echo "root" 1 &2
exit 1
```

fi

# ... **Test.sh**

- Get a server name and server IP from user
- Create a dynamic bash file
- Ping server name for 5 times and generate log file
- IF log file does not contain ttl string ,then generate error message
- Enter full name of server,IP in clients hosts file

Entry .sh Add server name and IP address in /etc/hosts file. **Check-usr.sh**

- Take a cluster user name and path where it has been created
- It will check using egrep i username /etc/passwd

If not then it will give error message and check this user path MPI installation

Mpi-1.sh

```
mkdir /mpi
cp -r lam-mpi /mpi/
chmod -R 777 /var/cache/*
chmod -R 777 /mpi/*
```

mpI-2.SH

```
"echo \"PATH=$PATH:$HOME/bin:/soft/lam/bin:\" >>
"+ user_path+"\.bash_profile"
"echo \"export PATH\" >> "+ user_path+"\.bash_profile
"echo \"export LAMRSH=ssh -x\" >> "+ user_path+"/.bash_profile
"echo \"export LAM=/soft/lam/bin\" >> "+ user_path+"/.bash_profile
"echo \"export PATH=/soft/lam/bin:$PATH\" >>"+ user_path+"/.bashrc
"echo \"set path = (/soft/lam/bin $path)\" >> "+ user_path+"/.cshrc
"echo \"export LAM=/soft/lam/bin\" >> "+ "/etc/bashrc
```

”su ”+user+” -c”+” /mpi/lam-3.sh”

Mpi-3.sh

cd /mpi/lam-mpi

./configure -prefix=/mpi/lam

make

make install

Remote Login(ssh)

- Copy the pair of private key and public key in /home/.ssh
- Modify public key with user@hostname
- Modified /etc/sshd/sshd.config file

How it is better than other cluster tools ?

- Supports Heterogeneous Environment(Fedora,Ubuntu,Red Hat)
- Failover Management
- Remote Login via (ssh and rsh both)
- Supports more than One MPI (LAM,INTEL,HP)
- Supports PVM
- Load Balancing Supports
- Automatically generate patch for neutronics code

Pre Requests:

- Login with Root
- User should be placed in one partition
- Gcc and Fortran should be installed

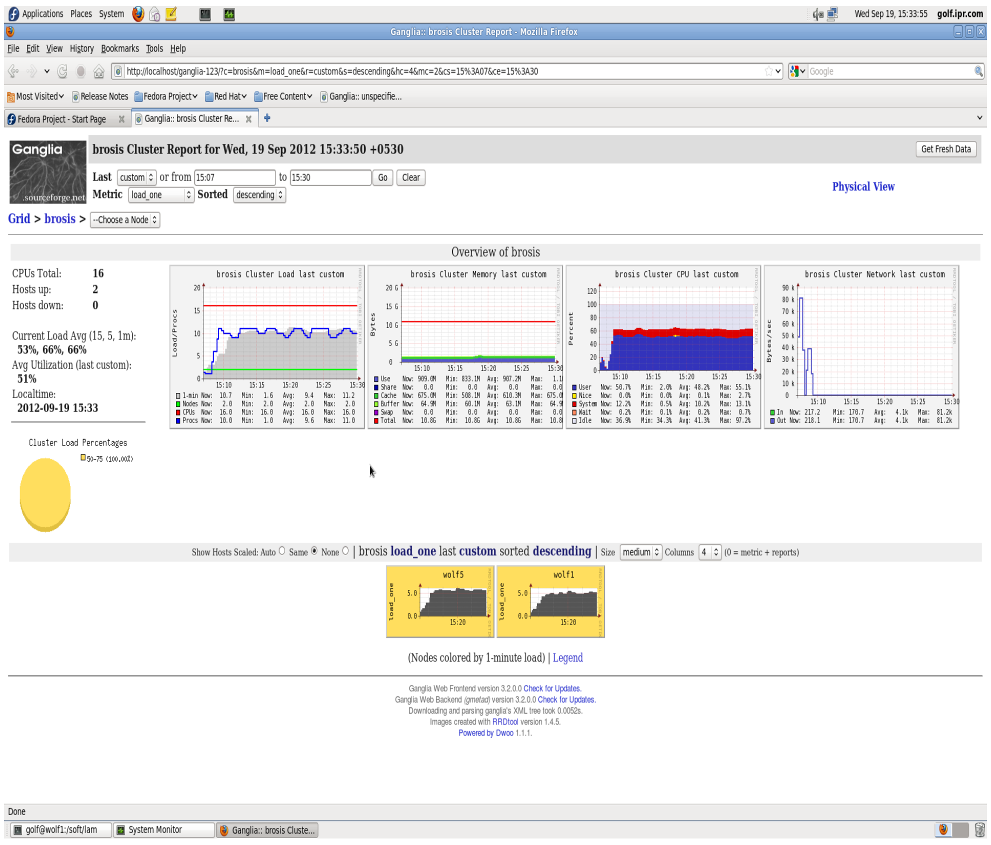
How it works?

- Run as a root
- Enter name and IP address of servers(Two servers for failover management)
- It will update host file of client node
- Select ssh or rsh
- It will automatically update its related files
- Both ssh and rsh can be possible
- Select MPI (LAM,HP,INTEL)
- You can select PVM also
- After that it will be install ganglia monitoring tool in background user
- Do not have to choose any option for it
- It will automatically configure fail over management (if one server is gone dead)
- Generate patch for neutronics code

5.3 Monitoring Tool :

Cluster Monitoring Tool Snapshots :
Overview :

Here there are two node (wolf5 and wolf1(head)) and we are running infinite pi program
Using LAM-MPI we are able to create only 10 threads(processes)


Ganglia:: brosis Cluster Report - Mozilla Firefox

http://localhost/ganglia-123/?c=brosis&m=load_one&r=custom&s=descending&hc=4&mc=2&cs=15%3A07&ce=15%3A30

brosis Cluster Report for Wed, 19 Sep 2012 15:33:50 +0530

Last custom or from 15:07 to 15:30
Metric load_one **Sorted** descending

Grid > brosis > --Choose a Node

Overview of brosis

CPUs Total: **16**
Hosts up: **2**
Hosts down: **0**

Current Load Avg (15, 5, 1m):
53%, 66%, 66%
Avg Utilization (last custom):
51%
Localtime:
2012-09-19 15:33

```
brosis Cluster Load last custom
1-min  Now: 10.7   Min:  1.6   Avg:  9.4   Max: 11.2
Nodes  Now:  2.0   Min:  2.0   Avg:  2.0   Max:  2.0
CPUs   Now: 16.0   Min: 16.0   Avg: 16.0   Max: 16.0
Procs  Now: 10.0   Min:  1.0   Avg:  9.6   Max: 11.0

brosis Cluster Memory last custom
Use    Now: 909.0M  Min: 833.1M  Avg: 907.2M  Max:   1.1
Share  Now:   0.0   Min:   0.0   Avg:   0.0   Max:   0.0
Cache  Now: 675.0M  Min: 508.1M  Avg: 610.3M  Max: 675.0
Buffer Now:  64.9M  Min:  60.1M  Avg:  63.1M  Max:  64.9
Swap   Now:   0.0   Min:   0.0   Avg:   0.0   Max:   0.0
Total  Now:  10.8G  Min:  10.8G  Avg:  10.8G  Max:  10.8

brosis Cluster CPU last custom
User   Now: 50.7%  Min:  2.0%  Avg: 48.2%  Max: 55.1%
Nice   Now:  0.0%  Min:  0.0%  Avg:  0.1%  Max:  2.7%
System Now: 12.2%  Min:  0.5%  Avg: 10.2%  Max: 13.1%
Wait   Now:  0.2%  Min:  0.1%  Avg:  0.2%  Max:  0.7%
Idle   Now: 36.9%  Min: 34.3%  Avg: 41.3%  Max: 97.2%

brosis Cluster Network last custom
In  Now: 217.2  Min: 170.7  Avg:  4.1k  Max: 81.2k
Out Now: 218.1  Min: 170.7  Avg:  4.1k  Max: 81.2k
```

Cluster Load Percentages
50-75 (100.00%)

Show Hosts Scaled: Auto ○ Same ◉ None ○ | brosis **load_one** last **custom** sorted **descending** | Size medium Columns 4 (0 = metric + reports)

(Nodes colored by 1-minute load) | Legend

Ganglia Web Frontend version 3.2.0.0 Check for Updates.
Ganglia Web Backend *(gmetad)* version 3.2.0.0 Check for Updates.
Downloading and parsing ganglia's XML tree took 0.0052s.
Images created with RRDtool version 1.4.5.
Powered by Dwoo 1.1.1.

Applications Places System
Wed Sep 19, 15:33:55 golf.ipr.com
File Edit View History Bookmarks Tools Help
Most Visited Release Notes Fedora Project Red Hat Free Content Ganglia:: unspecifie...
Fedora Project - Start Page
Ganglia:: brosis Cluster Re...
Get Fresh Data
Go
Clear
Physical View
wolf5
wolf1
load_one
Done
golf@wolf1:/soft/lam
System Monitor
Ganglia:: brosis Cluste...

Figure 5.6: Gnaglia Monitoring Tool snapshot

Maximum thread in Lam mapi = 5 * no. of hosts = 5*2=10

So we can use only 5 core out of 8 core per cpu.

Both have 8 coresso total no. of CPU =16 and Host Up =2

Current Load of last 15 minutes :53 percentage

Current Load of last 5 minutes :66 percentage

Current Load of last 1 minutes :66 percentage

Yellow colour shows cpu load is between 50-75 percentage

Head node (wolf1.ipr.com) :

CPU metrics :

cpu_ aidle -¿ Percent of time since boot idle CPU (0 %)

cpu_ idle -¿ Percent CPU idle (40 % because we are use 5 cores out of 8)

cpu_ nice -¿ Percent CPU nice (nice is zero because we are not changing process priority)

cpu_ system -¿Percent CPU system kernel time is while executing code in kernel space

cpu_ user -¿Percent CPU user (user time is while executing code in user space)

cpu_ wio -¿CPU time spent waiting for I/O

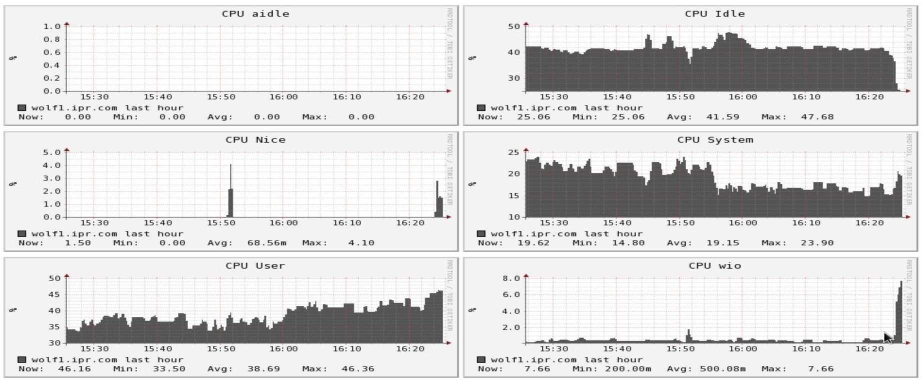


Figure 5.7: Gnaglia Monitoring Tool snapshot:CPU metrics

Disk metrics :

disk_ available-¿ Total free disk space

disk_ total -¿Total available disk space

maximum_ disk_ space_ used-¿ maximum used disk space (we have mounted /soft folder of 30gb)

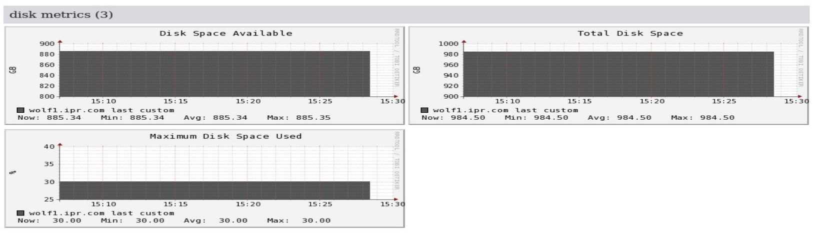


Figure 5.8: Gnaglia Monitoring Tool snapshot:Disk metrics

Memory metrics :

mem_ buffers -¿Amount of buffered memory

mem_ cached -¿Amount of cached memory

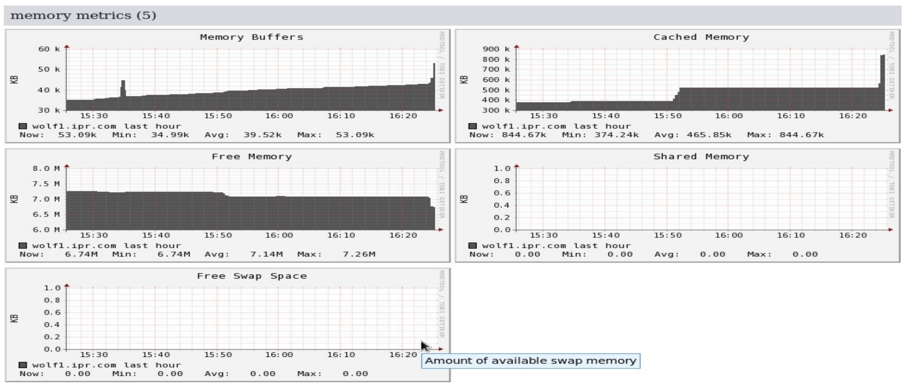
mem_ free -¿Amount of available memory

mem_ shared -¿Amount of shared memory

mem_ total -¿Amount of available memory

swap_ free -¿Amount of available swap memory

swap_ total -¿Total amount of swap memory



```
Memory Buffers
■ wolf1.ipr.com last hour
Now:  53.09k   Min:  34.99k   Avg:  39.52k   Max:  53.09k

Cached Memory
■ wolf1.ipr.com last hour
Now: 844.67k   Min: 374.24k   Avg: 465.85k   Max: 844.67k

Free Memory
■ wolf1.ipr.com last hour
Now:   6.74M   Min:   6.74M   Avg:   7.14M   Max:   7.26M

Shared Memory
■ wolf1.ipr.com last hour
Now:   0.00    Min:   0.00    Avg:   0.00    Max:   0.00

Free Swap Space
■ wolf1.ipr.com last hour
Now:   0.00    Min:   0.00    Avg:   0.00    Max:   0.00
```

Figure 5.9: Gnaglia Monitoring Tool snapshot:Memory metrics

Network metrics :

bytes_ in -¿Number of bytes in per second

bytes_ out -¿Number of bytes out per second

pkts_ in -¿Packets in per second

pkts_ out -¿ Packets out per second

proc_ run-¿ Total number of running processes

proc_ total -¿Total number of processes

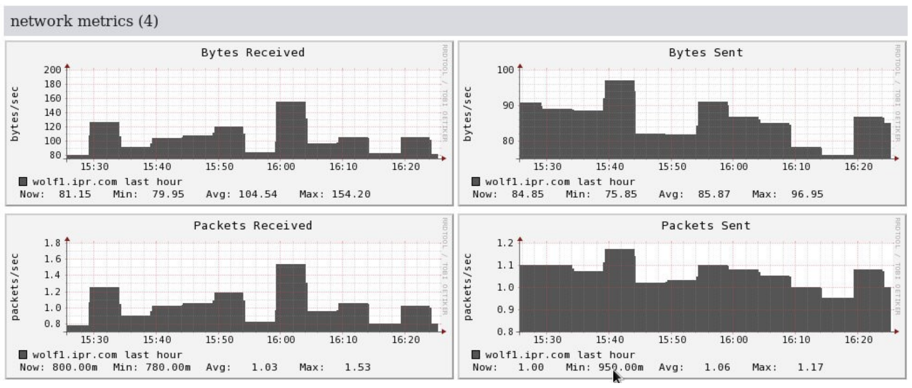


Figure 5.10: Gnaglia Monitoring Tool snapshot:Network metrics

Process metrics :

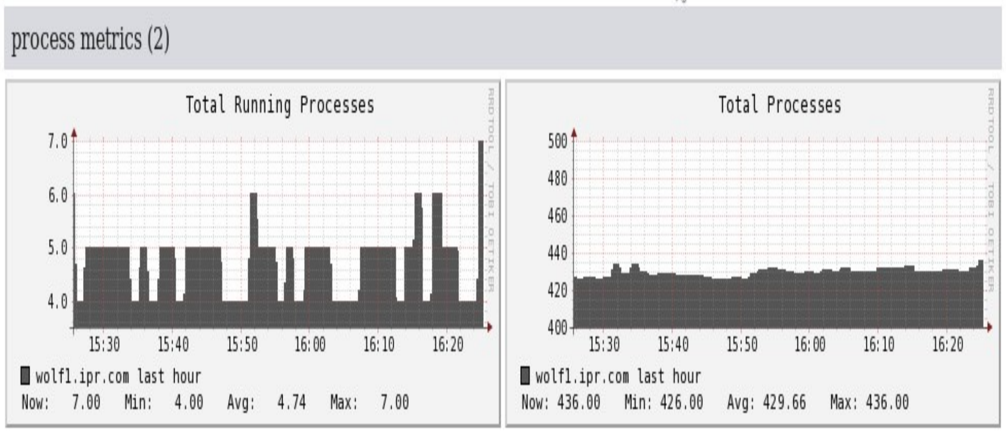


Figure 5.11: Gnaglia Monitoring Tool snapshot:Process metrics

Chapter 6

Neutronics Installation Process

The following environment variables should be set

export PATH=$ PATH:myPath/myNEUTRONICSXdirectory/myNEUTRONICSXrelease/bin

export DATAPATH= myPath/myNEUTRONICSXdirectory/data/

export DATAPATH=Z̈:/myPath/myNEUTRONICSXdirectory/data ̈

under cygwin !

export DISPLAY=":0"

Watch out, this last export can cause problem with ssh -X—Y. It may be commented if no geometry visualization (¿Neutronicsx i=myFile.mcn ixz) is needed.

Compiling NEUTRONICS :

Modify prpr.id so it will compile with g77.

Remove comments from install.fix for the LINUX compile.

(You may want to change the optimization level, see below)

Copy testoutp.lg77.tar to testoutp.dec and testmctl.lg77.tar to testmctl.dec If you do NOT do this, then expect many diff files.

Run ./install dec (or ./install dec ¡ answer if you grab an answer file from below). Specify the location of your X11 libraries using option 3.3. Choose option (1). Set the location of your X-libraries (/usr/X11R6/lib on my machine). Enter the name of

your graphics library (libX11.so on my machine). Set the location of the X-include files (/usr/X11R6/include)
Set the location of your data libraries.
Set the pvm options.
Process the file.

This is the answer file produced for a pvm version of Neutronics:

- .1
- 5
- 2.1
- 1
- 2.2
- 1
- 2.3
- 2
- mdas=4000000
- 3.1
- 1
- 3.2
- 1
- 3.3
- 1

- /usr/X11R6/lib
- libX11.so
- /usr/X11R6/include
- 4.1
- 1
- /usr/local/lanl/dlc189
- 4.2
- 2
- 5.1
- 2
- /usr/local/pvm3/lib/LINUX
- libfpvm3.a,libpvm3.a
- p

This is the makeNeutronics script file for a pvm version of Neutronics

```
# !/bin/sh
# Script file to make NEUTRONICS 4B on the DEC UNIX.
# Files needed:
prpr.id,makxs.id,patch?,Neutronicsc.id,Neutronicsf.id.
set -ex
rm -f compile newid patch
cp prpr.id prpr.f
f77 -o prpr prpr.f
cp makxs.id codef
```

```
grep *define patchc ¿ patch
prpr
mv compile makxsf.f
f77 -o makxsf makxsf.f
rm -f newid *.f *.o
cp Neutronicsc.id codef
cp patchc.pvm patch
prpr
mv compile Neutronicsc.c
cc -c -I/usr/X11R6/include Neutronicsc.c
rm -f codef patch newid
cp Neutronicsf.id codef
cp patchf.pvm patch
prpr
fsplit compile ¿ clog
rm -f compile codef patch newid clog
mkdir flibpvm
mkdir olibpvm
f77 -O0 -c *.f
mv *.f *.c flibpvm
f77 -o Neutronics.pvm *.o -L/usr/X11R6/lib -lX11 -L/usr/local/pvm3/lib/LINUX -
lfpvm3 -lpvm3
mv *.o olibpvm
```

Optimization Issues

The Neutronics test suite above was run for optimization levels of -O0 -O1 -O2 and -O3. Optimizations -O0 produced the output shown above. Optimization -O1 differed from the -O0 file in problems 1,12, and 23. Problem 1 differed due to a

round off errors, and problem 23 due to a single particle being in a different tally bin. Problem 12 had extensive differences. When optimization levels -O2 and -O3 were used, problems 2, 4, 6, and 23 failed to complete.
A benchmark test problem (of my choosing) produced the following timing results:
Optimization Time

-O0 - 15.92 min

-O1 - 11.41 min

-O2 - 10.57 min

-O3 - 10.57 min

Optimization levels -O0 and -O1 produce acceptable results for Neutronics. The ideal level of optimization may depend upon your problem. -O2 and -O3 do not pass the test suite.

Chapter 7

Failover Managment

When a node running in a cluster fails , the specific job that are running at that node begin to run on another node in the failover cluster. The steps in failing over are as follows:

- Detection: A failure is detected.
- Failover: Any node fails over to another node in the failover cluster.
- Job Restarts: Following a failure, first the job is terminated , than all the clients running the thread of the same job are reassigned the job from last checkpoint.

Failure detection in a failover clusters
The servers in a failover cluster monitor one another through periodic network signals, called heartbeats. If a client misses five heartbeats, communication with that client is considered to have failed. Failover clustering also monitors some of the services (for example, the HPC Job Scheduler Service on the head node) to ensure that they are running.
Ganglia Monitoring tool can easily detect the failed node. Once a failed node is detected, it is safely removed from the cluster. And all the jobs assigned to it can be

found from the lamhosts file. Once the jobs assigned to it are found they are listed in failed jobs file. Threads of those jobs running on other nodes can also be found. Those jobs are terminated. Nodes per job is also listed in the jobs_ node file.
These jobs are restarted from the last successful checkpoint. And redistributed to the nodes having minimum load or load less or equal to 50 percent.

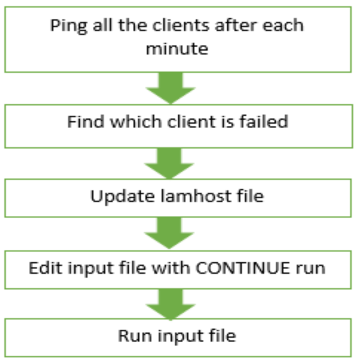

Ping all the clients after each minute
Find which client is failed
Update lamhost file
Edit input file with CONTINUE run
Run input file

Figure 7.1: Flow of Failover Management

7.1 How Neutronics Works?

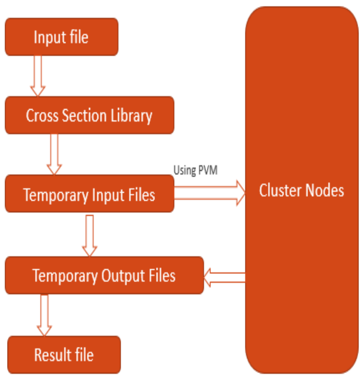
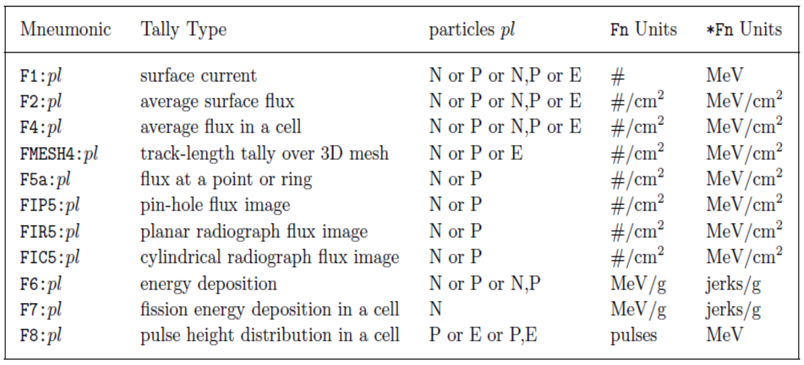


Figure 7.2: Neutronics

- Once an input file is submitted to Neutronics it does following steps
- It runs through nps=1 to nps =10x̂ (user defined)
- It divides run in to several dumps
- Eg upper limit =105̂
- Dumps will be :
  - Dump1 200
  - Dump2 10000
  - Dump3 20000
- Dumps are nothing but calculations of effects on system per y amount of neutrons
- It is sequential item Each dump contains following calculations



Mneumonic	Tally Type	particles *pl*	Fn Units	*Fn Units
F1:*pl*	surface current	N or P or N,P or E	#	MeV
F2:*pl*	average surface flux	N or P or N,P or E	$\#/cm^2$	MeV/cm^2
F4:*pl*	average flux in a cell	N or P or N,P or E	$\#/cm^2$	MeV/cm^2
FMESH4:*pl*	track-length tally over 3D mesh	N or P or E	$\#/cm^2$	MeV/cm^2
F5a:*pl*	flux at a point or ring	N or P	$\#/cm^2$	MeV/cm^2
FIP5:*pl*	pin-hole flux image	N or P	$\#/cm^2$	MeV/cm^2
FIR5:*pl*	planar radiograph flux image	N or P	$\#/cm^2$	MeV/cm^2
FIC5:*pl*	cylindrical radiograph flux image	N or P	$\#/cm^2$	MeV/cm^2
F6:*pl*	energy deposition	N or P or N,P	MeV/g	jerks/g
F7:*pl*	fission energy deposition in a cell	N	MeV/g	jerks/g
F8:*pl*	pulse height distribution in a cell	P or E or P,E	pulses	MeV

Figure 7.3: Tally Table

[7]

Checkpointing:

Checkpoints are created at the end of every dump. They are then stored in Runtpe file. Runtpe file consists of the dump information and all the results required to start next dump. Continue Run command is written to the input file which is to be re-run.Once the failover occurs and the jobs failed are detected their respective runtpe file is moved to the failed job directory. Each dump is read and respective mathematical functions are replaced by iterative functions.

- We can create checkpoints at the end of every dumps
- If system fails we can start again with the last completed dump
- Hence we used concept of Continue Run
- If any input file is appended with CONTINUE Command than it is under checkpointing state.

Continue Run:

- If system is under continue run than calculations will be sequential and store every dumps regularly in RUNTPE file.
- When a person submits continue run file to system it search for RUNTPE file if exist it reads the file and start from last successful completed dump which can be searched from output file.
- If doesnot exist it creates RUNTPE file.

Requirement:

- Execution Command should be appended by option C
- Two files are important for continue run
  - RUNTPE (Restart file)
  - Continue-run input file.

Chapter 8

Result

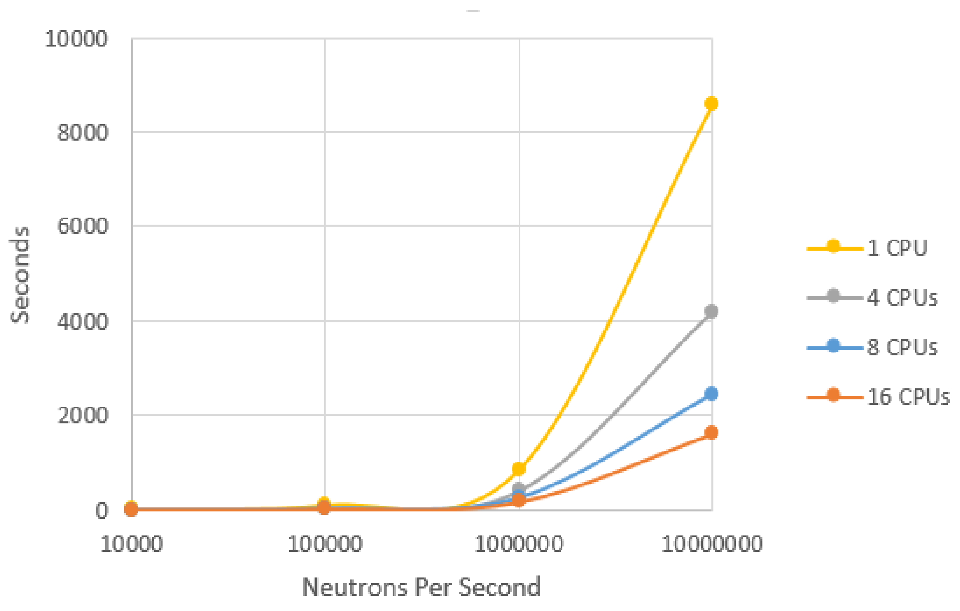


Figure 8.1: Neutronics Result

One can limit the execution time of neutronics by specifying the maximum number of source particle histories or by the maximum execution time. The number of source particle histories N is specified on the card by user.

NPS N

Eventually Neutronics run is terminated when a certain number of particle histories have been run or a desired computing time is reached. These are the cutoffs specified through the commands like NPS and CTME by the user for example

NPS 1000000 $ stop after a million source particles have been run

CTME 20.0 $ stop run after twenty minutes

In case both the cutoffs are specified, run is terminated when first cutoff is met. Usually NPS is of more importance and is widely used.

Above graph shows the performance of cluster , with varying nps and varying number of cpus. As number of cpu increases execution time decreases. Major performance enhancement is not achieved when nps is small, but as nps increases by the power of tens one can see visible difference in the execution time. Table indicates execution time per nps per cluster(varying number of cpu).

Conclusion

Cluster configuration tool made, can configure the cluster on different operating systems based on linux. It has a server side configuration module and a client side configuration module which automates the process of configuring the cluster by making necessary changes in the system files irrespective of the operating system. Here cluster being build is not the dedicated cluster, hence users of the nodes in the cluster may use the system for other tasks.

Cluster monitoring and management tool has the facility to view the processing load and network load on each node, as well as on the entire cluster. It keeps track of jobs submitted to each node, completed jobs, as well as status of each node. Malfunctioning node can easily be detected from the down time of a node.

Failure management module in this software is dedicated for neutronics code. This module is based on check-pointing method where by checkpoints are created at the end of each successful dump. If a failure occurs the job can be restarted from the last successful dump. Tests were conducted on various input files and results shows no performance degradation after failure. Check-pointing mechanism eliminates the need of restarting the entire process again and thus saves the major computing time. Execution time of a process on standalone system and that on a cluster showed major time reduction as shown in results.

References

[1] Electric flux distribution in photodetachment of heteronuclear diatomic molecular negative ion, Wang De-Hua,2010

[2] C. Leangsuksun, T. Liu, T. Rao1, S. Scott, and R. Libby, A failure predictive and policy-based high availability strategy for linux high performance computing cluster, in Proc. of 5th LCI International Conference on Linux Clusters, 2004

[3] http://software.intel.com/en-us/articles/building-clusters-the-easy-way-with-oscar

[4] http://www.garbled.net/clusterit.html

[5] http://www.ehu.es/AC/ABC.htm

[6] Distributed ansys manual

[7] Neotronics code manual