

## INDEX

<u>Sr. No.</u>	<u>Date</u>	<u>Title</u>	<u>Page No.</u>	<u>Signature</u>
1		To study the basic commands of linux.	3	
2		To establish Beowulf Cluster using MPI(Message Passing Interface) Library.	5	
3		To configure and install PVM (Version 3.4)	9	
4		To study distributed file system over a network using NFS protocol.	14	
5		Installation and configuration of Alchemi Grid.	17	
6		Running a sample application on Alchemi Grid and analysing it.	22	
7		To study a Grid Simulation Toolkit.	34	
8		To run two sample programs using GridSim Toolkit.	36	
9		Case study on the following: 1. Google Docs 2. Amazon EC2	39	
10		Case study on Google App Engine.	45	

## PRACTICAL 1

**AIM: To study the basic commands of linux.**

### 1. **ls :-**

- lists directory contents
- Syntax: `ls [OPTION]... [FILE]...`
- `-a(all)` : lists all the directory content of file system
- `-d(directory)` : lists directory entries instead of contents, and do not dereference symbolic links.

Example: - `ls -a` or `ls -a -d`

### 2. **Mkdir :-**

- It will create directory
- Syntax: `mkdir [OPTION] DIRECTORY...`

Example: - `mkdir a`

### 3. **chmod :-**

- Changes file access permissions
- Syntax:
 

<code>chmod</code>	<code>[OPTION]...</code>	<code>MODE[,MODE]...</code>	<code>FILE...</code>
<code>chmod</code>	<code>[OPTION]...</code>	<code>OCTAL-MODE</code>	<code>FILE...</code>
<code>chmod</code>	<code>[OPTION]... --reference=RFILE FILE...</code>		
- `-v(verbose)` : output a diagnostic for every file processed
- `-c(change)` : like verbose but report only when a change is made
- `u` is for user
- `g` is for group
- `o` is for others
- Following are the symbolic representation of three different permissions:
  - `r` is for read permission
  - `w` is for write permission
  - `x` is for execute permission.

Example: - `chmod u+x filename`

- If you want to change a file permission same as another file, use the reference option as shown below. In this example, file2's permission will be set exactly same as file1's permission.

Example:- `$ chmod --reference=file1 file2`

#### 4. **pwd :-**

- Prints name of current/working directory
- Syntax: `pwd [OPTION]`

Example: - `pwd`

#### 5. **rm :-**

- Removes files or directories
- Syntax: `rm [OPTION]... FILE...`
- `-f(force)`: ignore nonexistent files, never prompt
- `-r, -R(recursive)`: remove directories and their contents recursively

Example: - `rm -rf a`

#### 6. **Cp :-**

- Copies files from one location to another
- Syntax:
 

<code>cp</code>	<code>[OPTION]...</code>	<code>SOURCE</code>	<code>DEST</code>
<code>cp</code>	<code>[OPTION]...</code>	<code>SOURCE...</code>	<code>DIRECTORY</code>
<code>cp [OPTION]...</code>	<code>--target-directory=DIRECTORY SOURCE...</code>		

Example: – `cp a.txt` or `cp root`

#### 7. **Make :-**

- Looks for a file name (whatever name is) in your directory, and then execute it.
- Syntax: `make [OPTION]... Filename`

Example: - `make -f MyMakefile`

## **PRACTICAL 2**

**AIM: To establish Bewoulf Cluster using MPI(Message Passing Interface) Library.**

### **DESCRIPTION:**

A cluster is a group of computers which work together toward a final goal like to get faster processing, increasing processing speed. MPI is most popular package for message passing with in parallel program. It is a simply a library of definition and functions that can be used in C/C++ (Fortran) programs.

### **CONFIGURATION:**

**STEP 1:** Installing linux in all nodes of Cluster.

- For Master node, it is recommended to use separate partitions, while slave node swap partition can be used.

**STEP 2:** Installing MPICH in all nodes of Cluster.

- Create a directory MPI(we can use any name) in the home directory.  
\$ cd \$HOME  
\$ mkdir MPI
- Unpack the tar file.  
gunzip mpich.tar.gz  
\$ tar xzf mpich2-1.4.tar
- Rename folder that contain mpich.  
\$ mv mpich2-1.4 mpich
- Configure(activate ) services for access authentication.  
./configure rsh=ssh
- Build MPICH2  
\$ make
- Install the MPICH2 commands.

```
$ make install
```

- Add the bin directory to your path.  

```
$ export PATH=$PATH:/usr/local/mpich/bin
```

```
$ which mpd
```

```
$ which mpicc
```

```
$ which mpiexec
```

```
$ which mpirun
```

All should refer commands in the bin subdirectory of our install directory.

- The MPI has been successfully installed now. We can follow the same steps to install MPI in other machines.
- You can check whether it is properly installed or not by following command.

```
mpicc <.c file>
./<outfile>
```

### STEP 3: Configuring nodes of Cluster.

- /etc/hosts

Edit this file on every cluster node, adding the names and IP addresses of every node in the cluster. This allows these machines to be accessed by name instead of by IP number.

```
127.0.0.1                                localhost
192.168.0.1 master

192.168.0.2 slave-1

192.168.0.3 slave-2
```

- /etc/hosts.equiv

This means that users of these machines can access the localhost without supplying a password.

```
192.168.0.1 master

192.168.0.2 slave-1

192.168.0.3 slave-2
```

- .rhosts

This file should exist in each user's home directory. This file is also required so users can use RSH to connect to each node without supplying a password.

```
master
slave-1
slave-2
```

- /etc/securetty

This allows for easier administration of the nodes and is highly recommended. Simply add "rsh", "rexec", and "rlogin" to the end of the file.

- /etc/pam.d

This directory contains configuration files that effect logins of the various services defined here.

Modify the rsh, rlogin, and rexec files by rearranging the lines to have the line with "rhosts" as the first line and the line with "securetty" as the second line. An example of these files after modification is given below:

```
auth    required /lib/security/pam_rhosts_auth.so
auth    required /lib/security/pam_securetty.so
auth    required /lib/security/pam_nologin.so
auth    required /lib/security/pam_env.so
account required /lib/security/pam_stack.so service=system-auth
session required /lib/security/pam_stack.so service=system-auth
```

- /etc/exports

This file should only be modified on the master node.

This will allow every host to access these directories and will eliminate the need to replicate work on every node. Typically, it is a good idea to export each user's home directory. This is accomplished by adding the following line to /etc/exports.

```
/home 192.168.0.0/255.255.255.0(rw,no_root_squash)
```

It is also a good idea to export /usr/local as a great deal of user programs will be installed here. Do this by adding the following line to /etc/exports.

```
/usr/local 192.168.0.0/255.255.255.0(rw,no_root_squash)
```

- /etc/fstab

/etc/fstab is a list of devices and directories that will be mounted at boot time. Since NFS is being used, it is necessary to modify /etc/fstab to mount these NFS directories.

The following lines should be added to mount the two exports set up above.

```
[hostname of master node]:/usr/local /usr/local nfs  
[hostname of master node]:/home /home nfs
```

On the master node, type `../nfs start`

On each node, type `../network start`.

#### **STEP 4:** Testing nodes of Cluster.

You can check MPICH by checking clock time by following command.

- `Mpirun -np <c file>`  
Ex. `Mpirun -3 cpi.c`

Clock time for cluster for same application will be less than single node.

## **PRACTICAL 3**

### **AIM: To configure and install PVM (Version 3.4)**

#### **WHAT IS PVM?**

PVM is a software system that enables a collection of heterogeneous computers to be used as a coherent and flexible concurrent computational resource, or a "parallel virtual machine".

The individual computers may be shared- or local-memory multiprocessors, vector supercomputers, specialized graphics engines, or scalar workstations and PCs, that may be interconnected by a variety of networks, such as Ethernet or FDDI.

PVM consists of a run-time environment and library for message-passing, task and resource management, and fault notification. While PVM will not automatically make a commercial software package run faster, it *\*does\** provide a powerful set of functions for manually parallelizing an existing source program, or for writing new parallel / distributed programs.

The PVM software must be specifically installed on every machine that is to be used in your "virtual machine". There is no "automatic" installation of executables onto remote machines in PVM, although simply copying the pvm3/lib and pvm3/bin directories to another *\*similar\** machine (and setting \$PVM\_ROOT and \$PVM\_ARCH - see below) is sufficient for running PVM programs. Compiling or building PVM programs requires the full PVM installation.

User programs written in C, C++ or Fortran can access PVM through provided library routines.

#### **UNPACKING**

The files in the source distribution unpack into a "pvm3" directory. The pvm3 directory can reside in either a private or shared disk area. Installations for multiple machine architectures can coexist in the same shared filesystem because compiled files are placed in different subdirectories named for each architecture (\$PVM\_ARCH).



Some of the more important directories are:

Directory	Contains
bin/\$PVM_ARCH/ PVM	User program executables (examples & your programs)
conf/	Make configuration files for all PVM architectures
console/	Source for the pvm console
doc/	Miscellaneous documentation
examples/	Example PVM programs source files
gexamples/	More example PVM programs - for group library
hoster/	An example "hoster" program
include/	Header files for PVM programs
lib/	Generic system executables (scripts, rc file stubs)
lib/\$PVM_ARCH/	System executables (pvmd, console, etc.)
libfpvm/	Source for the libfpvm Fortran library
man/man[13]/	Online manual pages (nroff format)
misc/	Some miscellaneous PVM examples and utilities
patches/	Patch files and instructions, as they are released
pvmgs/	Source for the libgpvm library and group nameserver
rm/	An example resource manager for PVM
shmd/	A special daemon for shared memory systems (*MP)
src/	Source for the libpvm library and pvmd daemon
src/\$PVM_ARCH/	Additional source code for specific machines
tasker/	An example "tasker" program for PVM
tracer/	An example "tracer" program for PVM
xep/	An example interactive X-Window program

## BUILDING AND INSTALLING

To build the full PVM software, you will need to have a "make" utility, a C or C++ compiler, and a Fortran compiler installed on your system.

Before building or running PVM, you must set the environment variable "PVM\_ROOT" to the path where PVM resides, i.e. the path of the directory with this "Readme" file. This can be in a private area, for example \$HOME/pvm3, or a public one, such as /usr/local/pvm3.

If your shell is csh, add a line such as:

```
setenv PVM_ROOT $HOME/pvm3
```

to your .cshrc file. If you use a shell that reads .profile, such as sh or ksh, or .bashrc for bash, add the following lines to that file:

```
PVM_ROOT=$HOME/pvm3
export PVM_ROOT
```

The use of this variable and others is explained more fully in the pvm\_intro man page.

You can also include an appropriate shell startup file stub to set other PVM environment variables and to add PVM directories to your execution path. Inert the matching stub file, pvm3/lib/cshrc.stub, pvm3/lib/kshrc.stub or pvm3/lib/bashrc.stub, after your declaration of PVM\_ROOT in your shell startup file.

To build PVM for your system, type "make" in this directory. Make will use the "aimk" in the pvm3/lib directory to build the daemon executable (pvmd3), the C library (libpvm3.a), the Fortran library (libfpvm3.a) and the console client program (pvm).

The libraries and executables are installed in \$PVM\_ROOT/lib/\$PVM\_ARCH, where \$PVM\_ARCH is the host architecture name, e.g. "CRAY" or "LINUX".

\$PVM\_ROOT/conf/\$PVM\_ARCH.def

for comments regarding alternative configurations for your system. The provided scripts \$PVM\_ROOT/lib/pvm and \$PVM\_ROOT/lib/pvmd are used to start the PVM console and the PVM daemon, respectively. They determine the machine architecture and run the actual programs in the \$PVM\_ROOT/lib/\$PVM\_ARCH directory. You can either copy these scripts to your bin directory or add \$PVM\_ROOT/lib to your shell search path (using the above \*.stub files).

You may wish to add \$PVM\_ROOT/man to your MANPATH environment variable, if it's supported on your system.

## ALTERNATIVES TO RSH

To use "ssh" instead of "rsh" on your system, you can either:

1. Modify the \$PVM\_ROOT/conf/\$PVM\_ARCH.def file to change the absolute path specified for RSHCOMMAND in the ARCHCFLAGS define. Replace the path to rsh with the absolute path to "ssh" on your system and then recompile PVM and your applications. Or,
2. Set the "PVM\_RSH" environment variable to point to the absolute path to "ssh" on your system. This does not require re-compilation but must be done in every shell from which PVM is executed. The "PVM\_RSH" environment variable can be set in your \$HOME/.cshrc or equivalent shell startup file to take affect in all new shells.

Once either of these approaches has been applied, the \$HOME/.rhosts file is then no longer necessary for PVM, thereby eliminating a fundamental security concern. Now, to add PVM hosts (using ssh), you can either manually enter your password each time you add a host, or else you can set up an ssh-agent session. (You can always just use a \$HOME/.shosts file, but then you're not much more secure than with standard rsh...)

When manually entering your password, you will not receive a prompt from PVM or ssh - the text prompts normally returned by ssh are automatically captured and written to the local /tmp/pvml.<uid> log file. However, simply typing your password from inside the "pvm" console or when running the "pvmd" script will work. You can bypass this manual password entry for a given session by executing the following commands (example is for bash, could use for any shell):

```
$ ssh-agent bash
$ ssh-add
<now enter your passphrase ONCE>
$ pvm
$ pvm> add host2 . . .
```

For this to work, you must have run "ssh-keygen" on each host to create a private/public key pair for your login id. These keys are typically stored in the \$HOME/.ssh directory on each system. The \$HOME/.ssh/identity.pub public key must be copied to each remote host and added as a line in the \$HOME/.ssh/authorized\_keys file on the remote system. When this has been set up, any PVM hosts can be added without password or passphrase entry using the above ssh-agent session.

## STARTING AND STOPPING PVM

To start PVM, run \$PVM\_ROOT/lib/pvm. This starts the console task, which in turn starts a pvmd if one is not already running. More hosts can be started and added to your "virtual machine" by using the console "add" command.

To start the pvmd without starting the console, you can also run the \$PVM\_ROOT/lib/pvmd directly. A number of hosts can all be started at once by supplying the pvmd with a host file, as in:

```
$PVM_ROOT/lib/pvmd my_hosts
```

where "my\_hosts" contains the names of the hosts you wish to add, one host per line. See the pvmd man page for other host file options.

To stop PVM, use the PVM console command "halt". From within your user programs, you can use the pvm\_halt() function. You can also kill the pvmd3 process (always use a catchable signal such as SIGTERM). But, if you do kill the pvmd, or if it fails for some other

reason, always be sure to remove any leftover `/tmp/pvmd.<uid>` pvmd socket files, where `<uid>` is your numerical user id. These files will make PVM think a pvmd is running, and can cause the dreaded "Can't Start Pvmd" message.

## TROUBLESHOOTING

If you ever have trouble starting PVM or adding a new host to your virtual machine, verify that there are no leftover `/tmp/pvmd.<uid>` daemon socket files on the machines where you are trying to start PVM (as described above) and then check the local `/tmp/pvml.<uid>` log file for any error messages which may help you to determine the problem.

A common problem is caused by restricted access to a remote machine via "rsh". PVM uses "rsh" to start the pvmd on a remote host. If you cannot do this:

```
% rsh remote_host 'echo $PVM_ROOT'
```

and successfully get back the correct value of `$PVM_ROOT` on that remote host, without typing your password, then that is the problem. You will need to set up permissions on the remote host to allow rsh access without a password. On Unix systems, this is accomplished by creating a `$HOME/.rhosts` file on the *\*remote\** machine that provides access to your *\*local\** machine. (Note: the `$HOME/.rhosts` file is *\*NOT\** a PVM host file...) The format of a `$HOME/.rhosts` file is as follows:

```
your_host_1 your_login_on_host_1
your_host_2 your_login_on_host_2
your_host_3 your_login_on_host_3
...
```

## PRACTICAL 4

**AIM: To study distributed file system over a network using NFS protocol.**

### DESCRIPTION

Network File System (NFS) is a network file system protocol allows user on a client computer to access files over a network in a manner similar to how local storage is accessed. NFS, like many other protocols, builds on the Open Network Computing Remote Procedure Call system.

### CONFIGURATION

#### ➤ SETTING UP NFS SERVER

For setting up NFS server we need to first edit EXPORTS file, which is in etc directory. This file contains a list of entries; each entry indicates a volume that is shared and how it is shared.

An entry in exports file will be like this:

Directory client1(options) client2(options)

Where the options can be:

- **ro:** The directory is shared read only; the client machine will not be able to write it. This is the default.
- **rw:** The client machine will have read and write access to the directory.
- **no\_root\_squash:** By default, any file request made by user root on the client machine is treated as if it is made by user nobody on the server. **no\_root\_squash** is selected, then root on the client machine will have the same level of access to the files on the system as root on the server. This can have serious security implications, although it may be necessary if you want to perform any administrative work on the client machine that involves the exported directories. You should not specify this option without a good reason.
- **no\_subtree\_check:** If only part of a volume is exported, a routine called subtree checking verifies that a file that is requested from the client is in the appropriate part of the volume. If the entire volume is exported, disabling this check will speed up transfers.

For example if you want to export home directory to a client having ip address 192.168.100.2 with read only permission, then the entry in exports file will be:

/home 192.168.100.2/24(ro)

Now, we need to start some services like nfs and portmap.

- **nfs** starts the NFS server and the appropriate RPC processes to service requests for shared NFS file systems. It can be started using '**service nfs start**' command.
- **portmap** — accepts port reservations from local RPC services. These ports are then made available (or advertised) so the corresponding remote RPC services access them. **portmap** responds to requests for RPC services and sets up connections to the requested RPC service. It can be started using '**service portmap start**' command.

Once all this is done, we can see the list of exported files using following command.

```
Showmount -e
```

## ➤ SETTING UP NFS CLIENT

On client side again we need to start the same services other then this we need to use rpc.lockd and rpc.statd services. They are used for following purpose:

- **rpc.lockd** — allows NFS clients to lock files on the server. If rpc.lockd is not started, file locking will fail. rpc.lockd implements the *Network Lock Manager (NLM)* protocol.
- **rpc.statd** — This process implements the *Network Status Monitor (NSM)* RPC protocol which notifies NFS clients when an NFS server is restarted without being gracefully brought down. This process is started automatically by the nfslock service and does not require user configuration.

With portmap, lockd, and statd running, you should now be able to mount the remote directory from your server just the way you mount a local hard drive, with the mount command.

### Mount Directory On Client Machine

To mount home directory on client with ip address 192.168.100.2 we can use following command:

```
Mount 192.168.100.2:/home /mount/home
```

Here this directory will appear at /mount/home on client machine.

### Getting NFS File Systems To Be Mounted At Boot Time

NFS file systems can be added to your /etc/fstab file the same way local file systems can, so that they mount when your system starts up. Entry in /etc/fstab looks like :

device	mountpoint	fs-type	options	dump	fsckorder
--------	------------	---------	---------	------	-----------

For our example it will be:

```
192.168.100.2:/home/mount      nfs      ro      0      0
```

Here in the options we can also define whether the mounting is soft or hard. In the soft mounting If a file request fails, the NFS client will report an error to the process on the client machine requesting the file access. While in a hard mounting the program accessing a file on a NFS mounted file system will hang when the server crashes. The process cannot be interrupted or killed (except by a "sure kill"). When the NFS server is back online the program will continue undisturbed from where it was.

## PRACTICAL 5

### **AIM: Installation and configuration of Alchemi Grid**

#### **Common Requirements**

Microsoft .NET Framework 1.0 or .NET Framework 1.1

#### **Manager**

The Manager should be installed on a stable and reasonably capable machine. The Manager requires:

- SQL Server 2000 or MSDE 2000

#### **Installation**

Install the Manager via the Manager installer. Use the sa password noted previously to install the database during the installation:

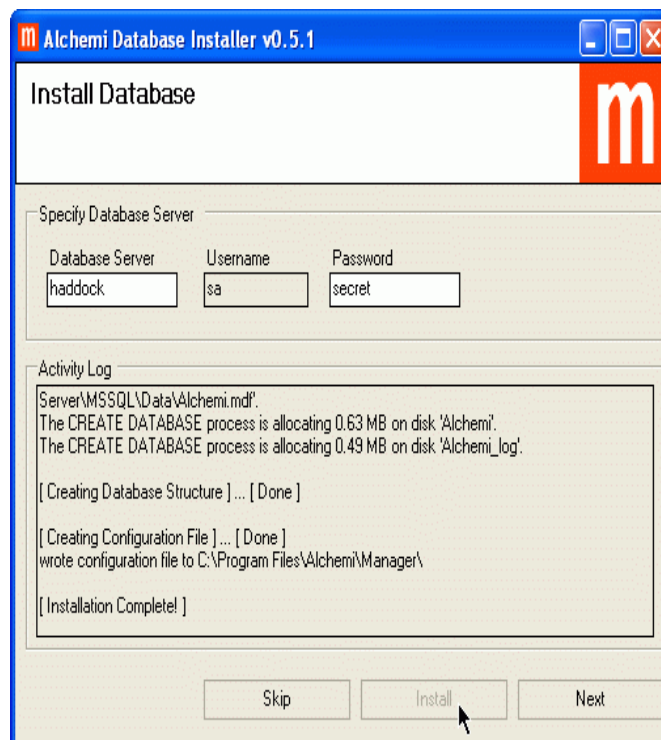


Fig 3.1. Database installation during Manager Installation.



## Configuration And Operation

- The Manager is configured from the application itself.
- The Manager can be run from the desktop or Start -> Programs -> Alchemi -> Alchemi Manager. The database configuration settings used during installation automatically appear when the Manager is first started.
- For a **stand-alone Manager** (for a uni-level grid or the highest-level multi-level grid Manager) you only need to designate the "OwnPort" setting, which is the port that Manager listens on for all communication.
- Click the "Start" button to start the Manager.

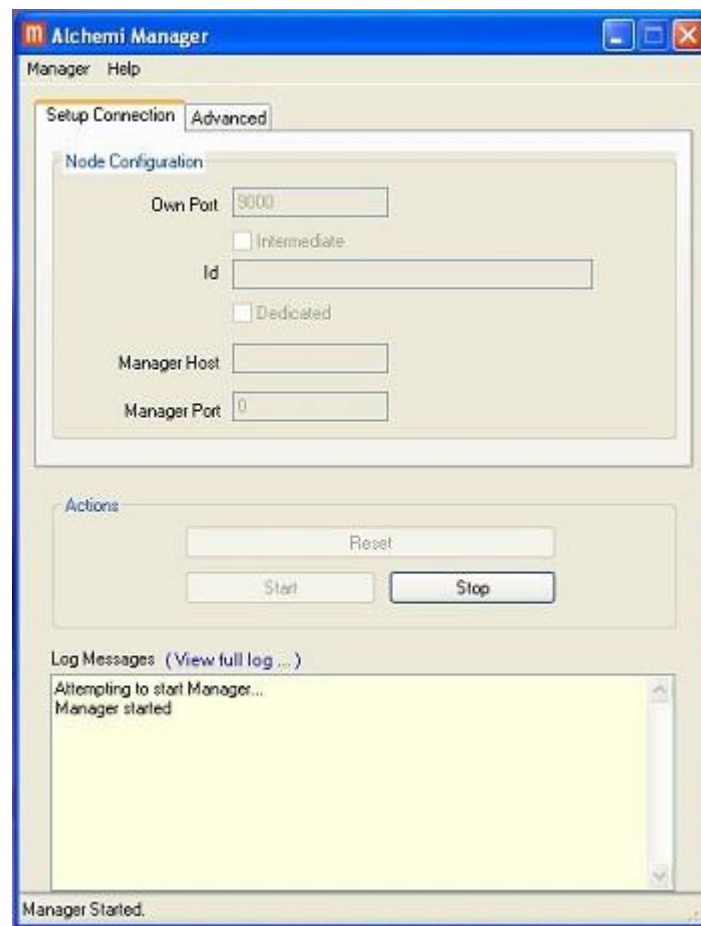


Fig 3.2. Stand-alone Manager configuration.

Since the Manager behaves like an Executor towards the higher-level Manager, you also need to specify whether it is a dedicated or non-dedicated "Executor" by checking/unchecking the "Dedicated" box.

# Executor

## Installation

- Install the Executor via the Executor installer.
- The Executor comes with a screensaver, which automatically activates the Executor. During installation, the "Screen Saver" properties are displayed with the screen saver selected.

## Configuration And Operation

- The Executor is configured from the application itself.
- The Executor can be run from the desktop or Start -> Programs -> Alchemi -> Alchemi Executor.

You need to configure 2 aspects of the Executor:

- The host and port of the Manager to connect to.
- Dedicated / non-dedicated execution. A non-dedicated Executor executes grid threads on a voluntary basis (it requests threads to execute from the Manager), while a dedicated Executor is always executing grid threads (it is directly provided grid threads to execute by the Manager).

Now, to connect the Executor to the Manager following steps need to be followed:

- Click the "Connect" button to connect the Executor to the Manager.
- If the Executor is configured for non-dedicated execution, you can start executing by clicking the "Start Executing" button in the "Manage Execution" tab. **Note:** When the Alchemi screen saver is activated, it automatically "presses" this button.

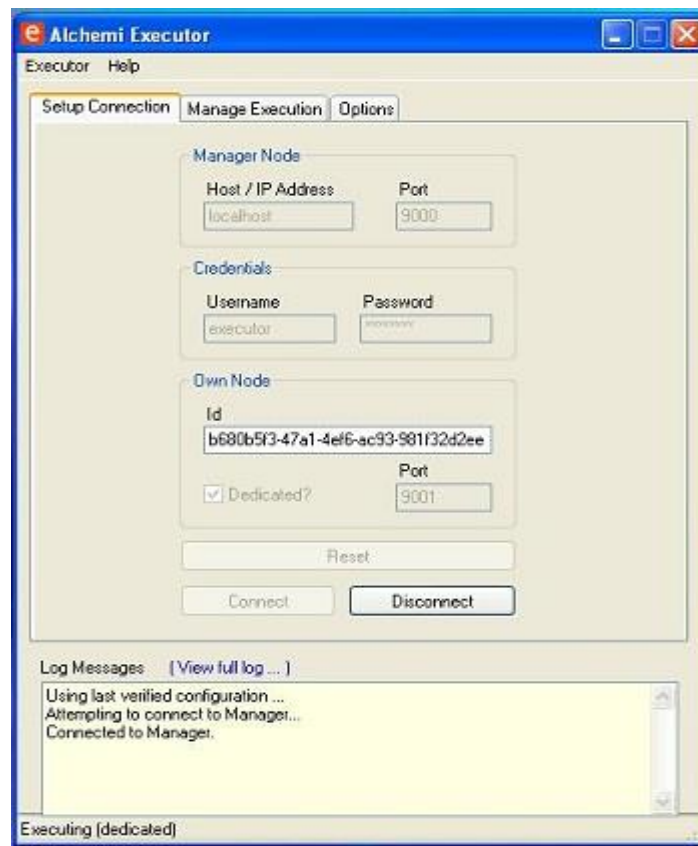


Fig 3.5. Executor connected to a Manager.

## Verifying Grid Installation

You can verify successful setup of a grid by running a sample application on it. The Alchemi SDK contains a sample application "Prime Number Generator" that calculates the prime number in the range specified by the user.

Configure PrimeNumberGenerator to point to a Manager:

- Examples\PrimeNumberGenerator\bin\Debug\PrimeNumberGenerator.exe.config

Run it:

- Examples\PrimeNumberGenerator\bin\Debug\PrimeNumberGenerator.exe

```

D:\COLLEGE\7th Sem\ACT\ACT_ORI\alchemi6\Alchemi-1.0.6-sdk-net-2.0\Alchemi-1.0.6-sdk-net-2.0\...
[PrimeNumber Checker Grid Application]
-----
Enter a maximum limit for Prime Number checking [default=10000000] :10000000
Connecting to Alchemi Grid...
Host [default=localhost] :
Port [default=9000] :
Username [default=user] :
Password [default=user] :

Creating a grid thread to check if 5997131 is prime...
Creating a grid thread to check if 1412226 is prime...
Creating a grid thread to check if 1524697 is prime...
Creating a grid thread to check if 5066904 is prime...
Creating a grid thread to check if 754878 is prime...
Creating a grid thread to check if 5819137 is prime...
Creating a grid thread to check if 7527944 is prime...
Creating a grid thread to check if 5600439 is prime...
Creating a grid thread to check if 1345857 is prime...
Creating a grid thread to check if 7543204 is prime...
Prime Number Generator completed.
5997131 is prime? False <4 factors>
1412226 is prime? False <24 factors>
1524697 is prime? True <2 factors>
5066904 is prime? False <32 factors>
754878 is prime? False <8 factors>
5819137 is prime? False <4 factors>
7527944 is prime? False <8 factors>
5600439 is prime? False <24 factors>
1345857 is prime? False <8 factors>
7543204 is prime? False <6 factors>
Application finished.
Random primes found: 1. Total time taken : 00:00:02.8481629

```

Fig 3.6. "PrimeNumberGenerator" running on a grid.

## PRACTICAL 6

**AIM: Running a sample application on Alchemi Grid and analysing it.**

### Configure

Configure PrimeNumberGenerator to point to a Manager:

- Examples\PrimeNumberGenerator\bin\Debug\PrimeNumberGenerator.exe.config

If the bin folder does not appear in the PrimeNumberGenerator folder, then it is necessary to follow the following steps for the first time:

- Run the PrimeNumberGenerator in the Visual Studio for the first time.
- Then, the bin folder will appear in the PrimeNumberGenerator folder.

Now, it is possible to run the PrimeNumberGenerator.exe inside the Debug folder whenever it is needed to run.

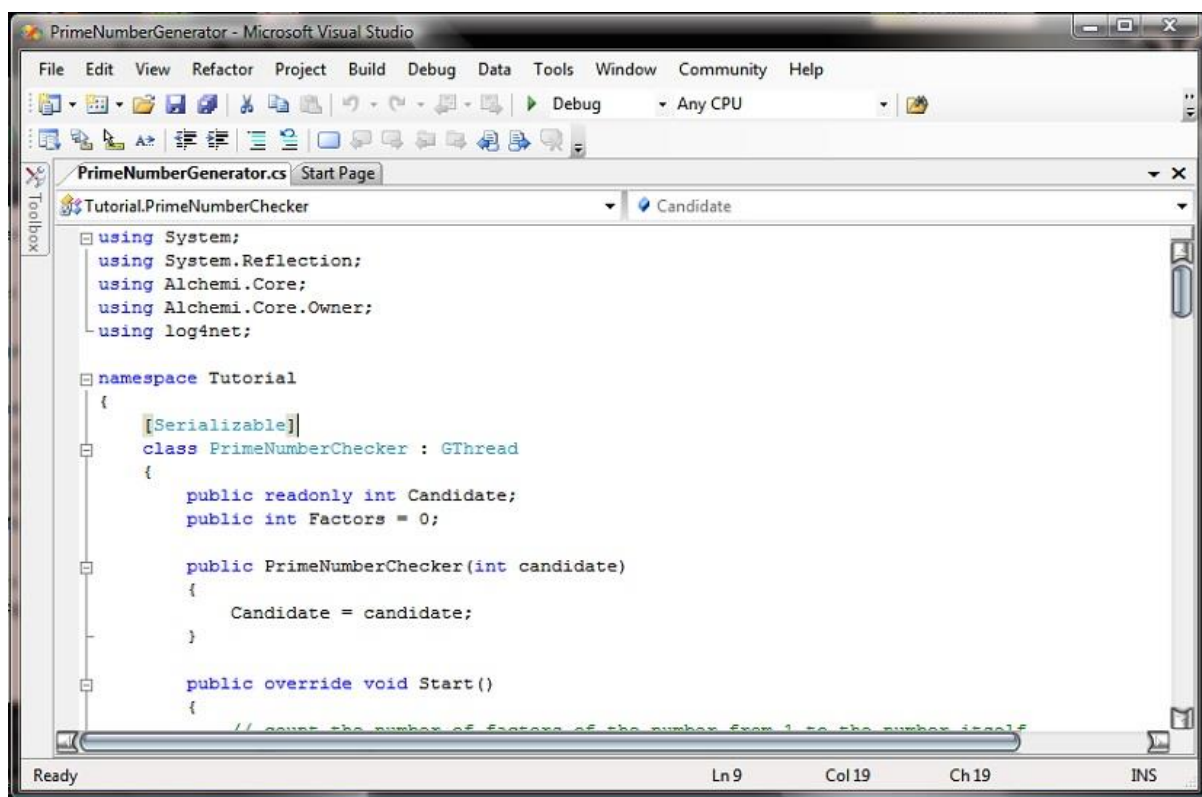


Fig. PrimeNumberGenerator program in Visual Studio

After running the program in visual studio once, the 'bin\debug' folder will appear. There will be PrimeNumberGenerator.exe inside the debug folder. Now, whenever needed, by double clicking the PrimeNumberGenerator.exe the program would run by itself.

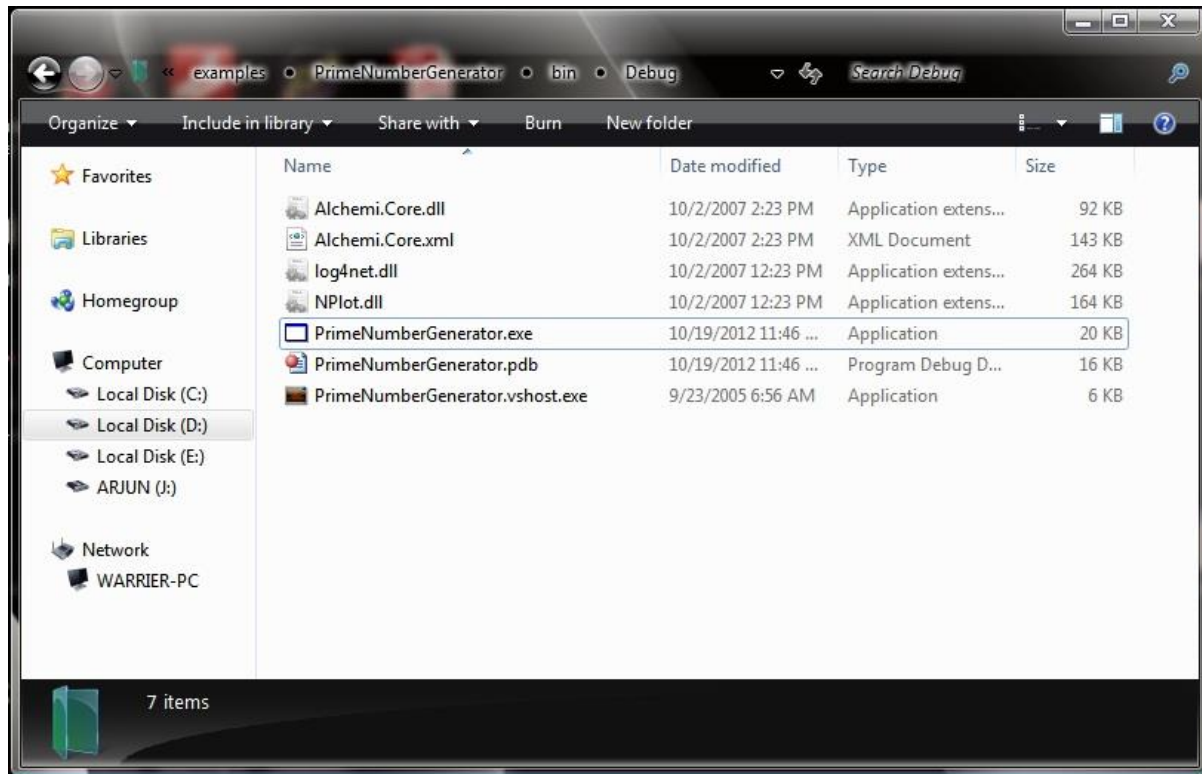


Fig. The PrimeNumberGenerator in the debug folder.

## Alchemi Console

Alchemi console is opened by double clicking the Alchemi.Consol.exe. On making a grid connection the console would look like as shown in figure below.

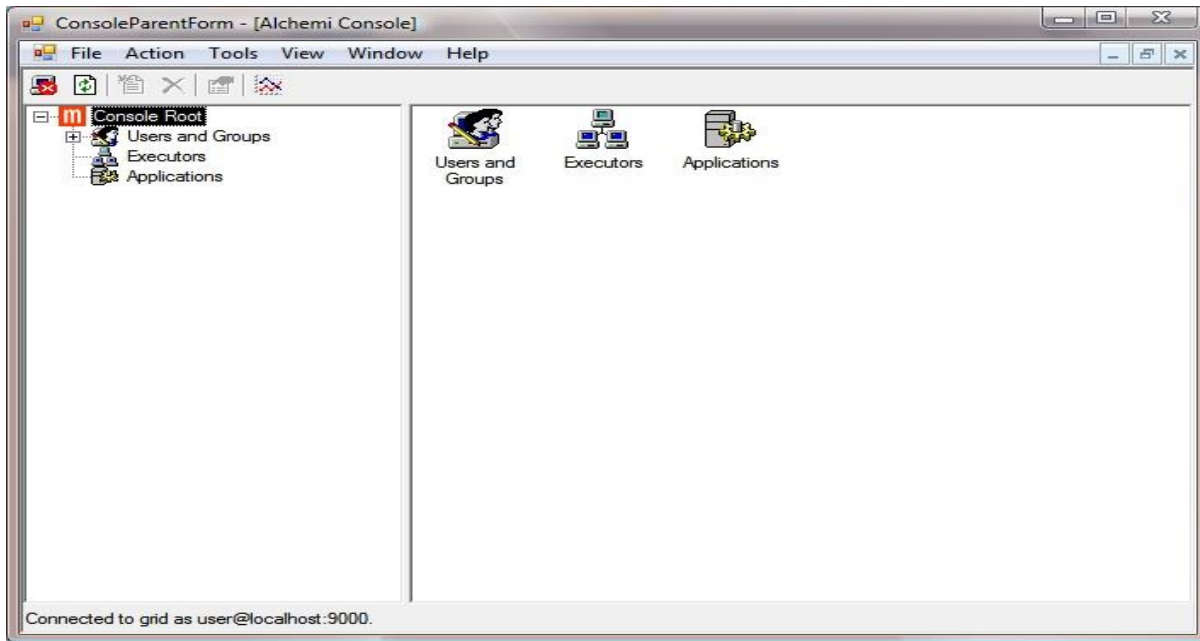


Fig. Alchemi Console

### Executors Console

On clicking on the Executors, the console will show the number of executors connected to the manager. Here, there are 3 executors connected to the manager, hence, the console will show as in figure given below.



Fig. Console showing number of executors.

## Executor Properties

On double clicking on any of the executor, will show the properties owned by that specific executor.

It would show up with a popup box as shown below in figure.

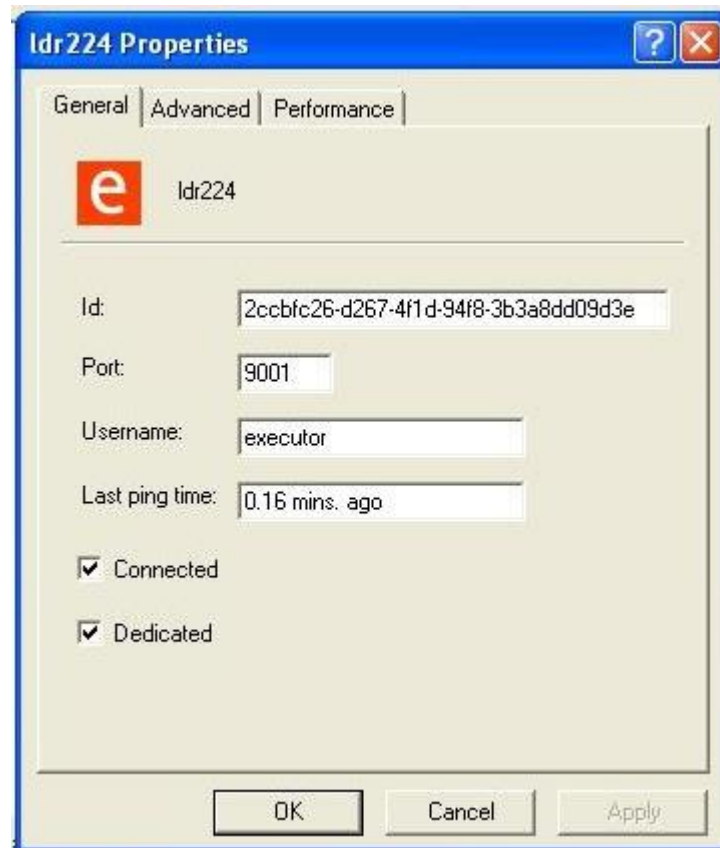


Fig. Executor Properties



## Threads

Threads are the light weight processes made by the manager by dividing the application to be run.

On clicking the application tab the threads would appear as show below in figure.

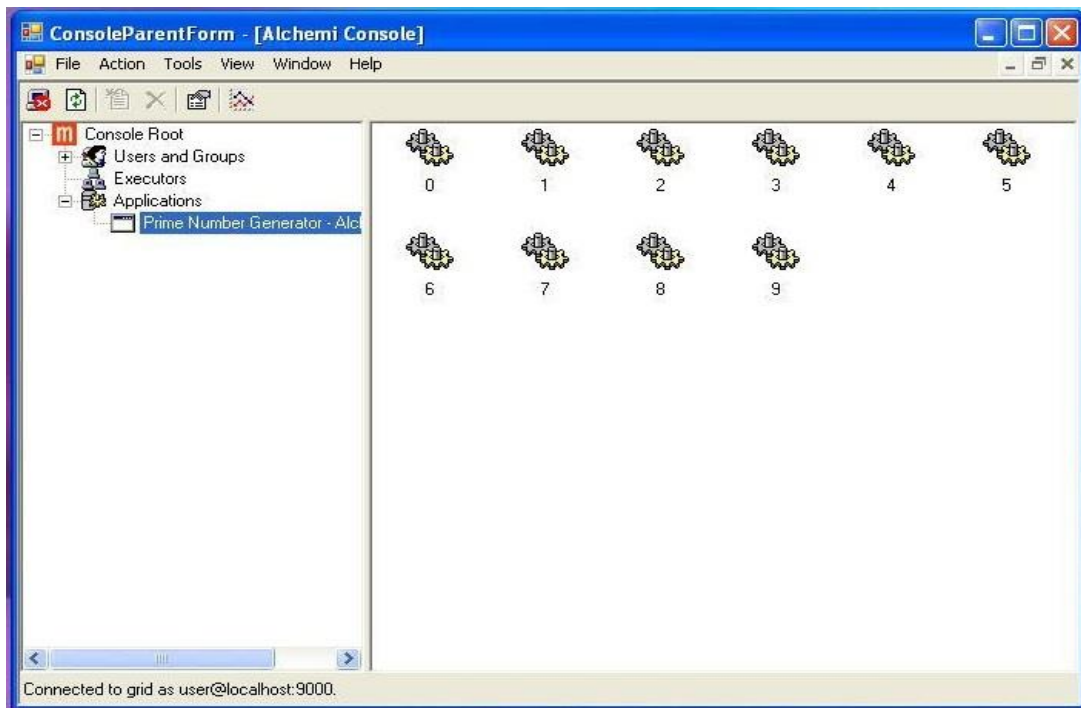


Fig. Threads

## Thread Properties

On double clicking on any of the threads it would show up the properties of the thread as shown in figure below.

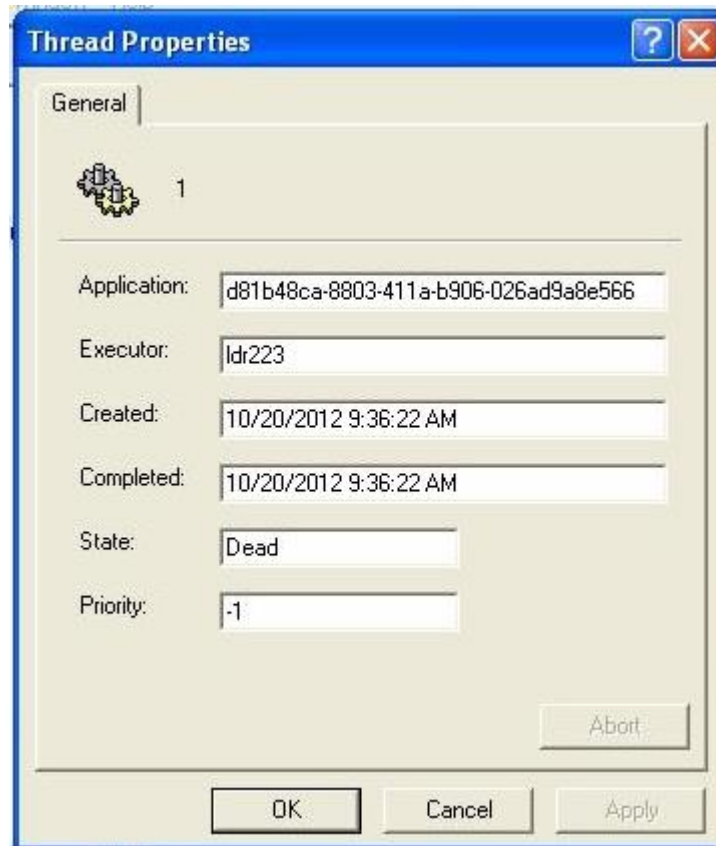


Fig. Thread Properties

## System Performance Monitor

On clicking the Performance Summary tab the console will display as below.

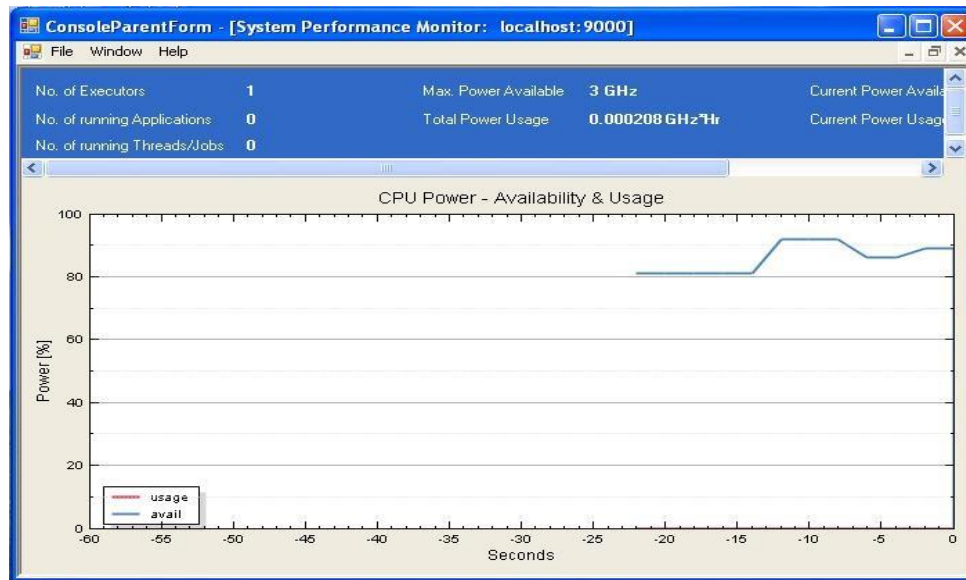


Fig. System Performance Monitor

These were the main parts of the Alchemi Console in brief.

## Execute the program

As said earlier, the PrimeNumberGenerator application can be run by double clicking PrimeNumberGenerator.exe.

On doing so a popup console would show up as shown in figure below.

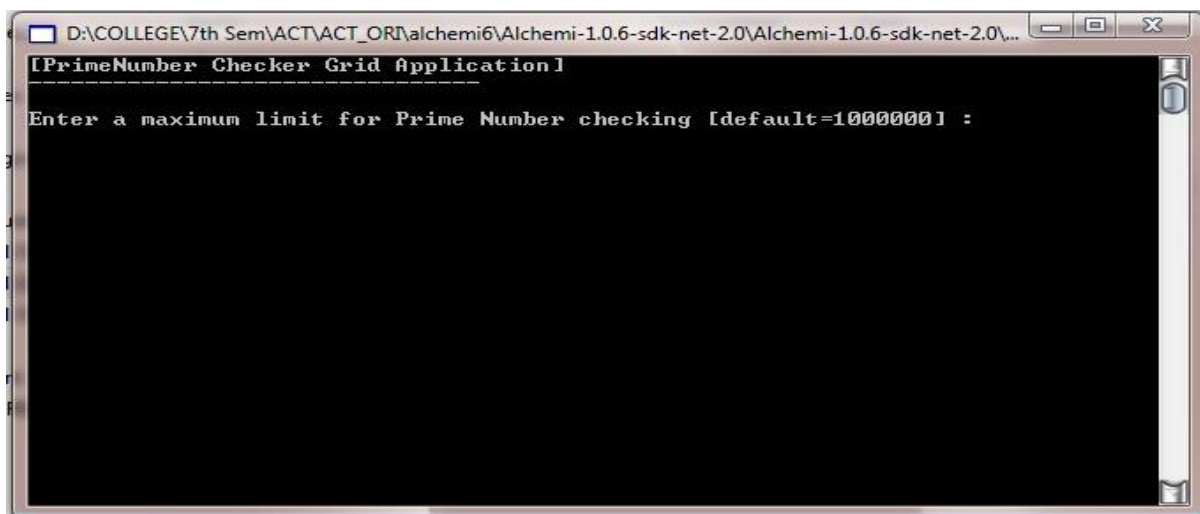


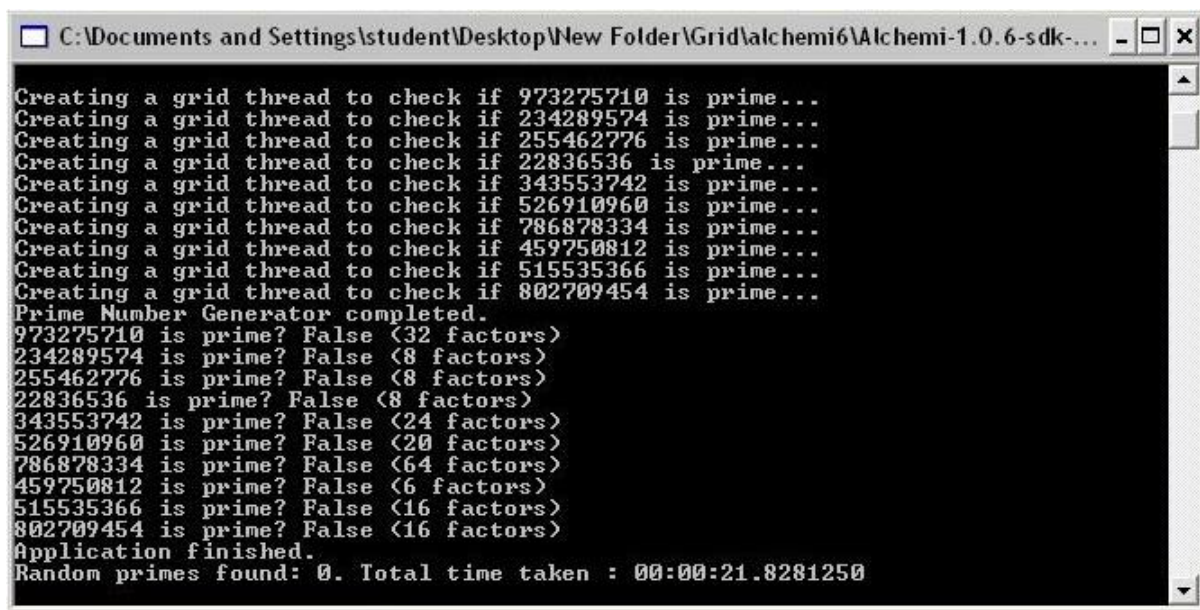
Fig. Console appeared.

Here, we have considered the value to be entered as the limit is '1000000000'. Now, we will run it in different cases, such as,

- With 1 executor
- With 2 executor
- With 3 executor

#### With One Executor

In this case only ONE EXECUTOR is connected to the MANAGER. Now on running the application, PrimeNumberGenerator application console and the Performance Monitor will be displayed as follows.



```

C:\Documents and Settings\student\Desktop\New Folder\Grid\alchemi6\Alchemi-1.0.6-sdk-...
Creating a grid thread to check if 973275710 is prime...
Creating a grid thread to check if 234289574 is prime...
Creating a grid thread to check if 255462776 is prime...
Creating a grid thread to check if 22836536 is prime...
Creating a grid thread to check if 343553742 is prime...
Creating a grid thread to check if 526910960 is prime...
Creating a grid thread to check if 786878334 is prime...
Creating a grid thread to check if 459750812 is prime...
Creating a grid thread to check if 515535366 is prime...
Creating a grid thread to check if 802709454 is prime...
Prime Number Generator completed.
973275710 is prime? False <32 factors>
234289574 is prime? False <8 factors>
255462776 is prime? False <8 factors>
22836536 is prime? False <8 factors>
343553742 is prime? False <24 factors>
526910960 is prime? False <20 factors>
786878334 is prime? False <64 factors>
459750812 is prime? False <6 factors>
515535366 is prime? False <16 factors>
802709454 is prime? False <16 factors>
Application finished.
Random primes found: 0. Total time taken : 00:00:21.8281250

```

Fig. Application Console – PrimeNumberGenerator

Here, on running the application using one executor the reading of the 'Total Time Taken' resulted as 21.8281250 seconds.

Hence, accordingly the system performance reading is also shown in System Performance Monitor.

There you could see that the 'No. of Executors' show 2, even though the 'Max. Power Available' shows power of one executor. This is so because, 'No. of Executors' shows the total number of executors in the network than the active ones.

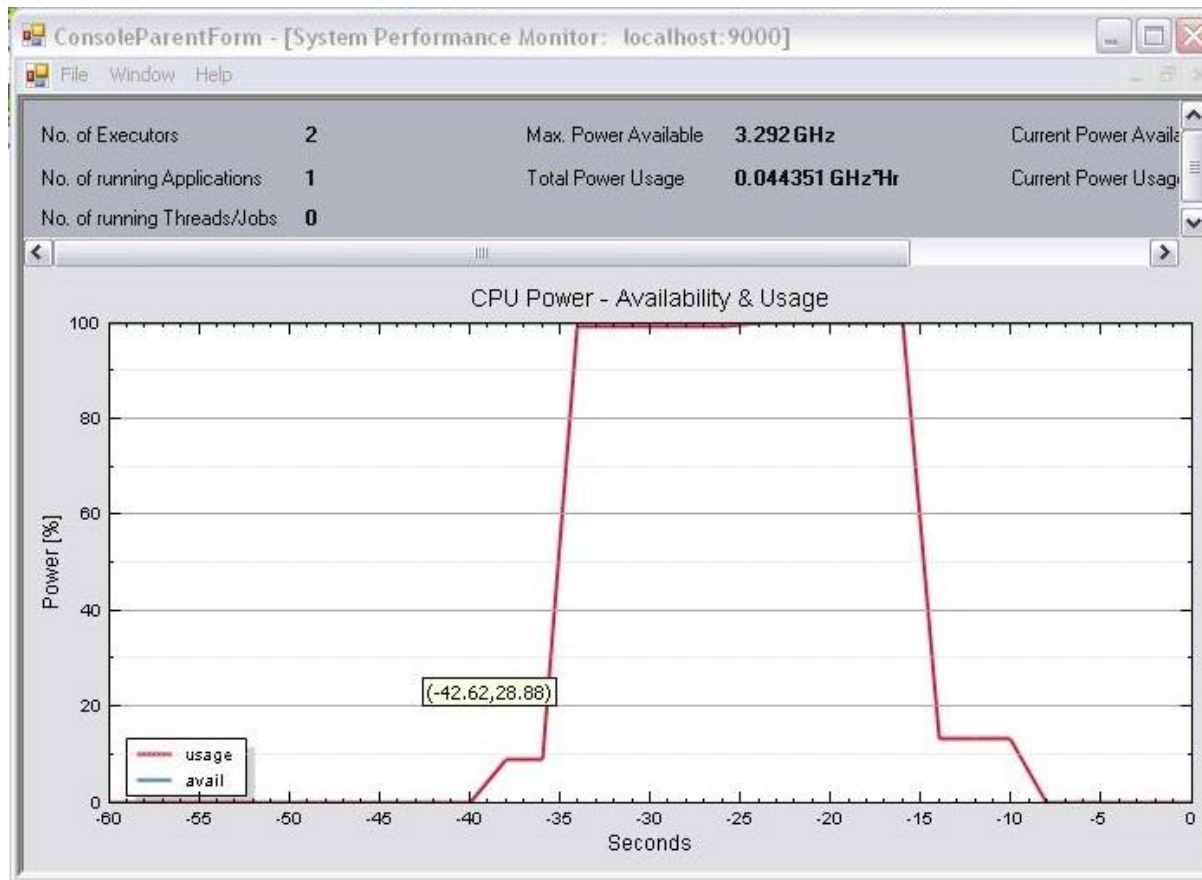


Fig. System Performance Monitor

### With Two Executor

In this case TWO EXECUTORS are connected to the MANAGER. Now on running the application, PrimeNumberGenerator application console and the Performance Monitor will be displayed as follows. Here also we will use the value of 'Limit' as '1000000000', for consistency in the result.

```

C:\Documents and Settings\student\Desktop\New Folder\Grid\alchemi6\Alchemi-1.0.6-sdk-...
Creating a grid thread to check if 339277810 is prime...
Creating a grid thread to check if 635134784 is prime...
Creating a grid thread to check if 227057103 is prime...
Creating a grid thread to check if 289319948 is prime...
Creating a grid thread to check if 927224302 is prime...
Creating a grid thread to check if 325473311 is prime...
Creating a grid thread to check if 133483220 is prime...
Creating a grid thread to check if 50864148 is prime...
Creating a grid thread to check if 320205730 is prime...
Creating a grid thread to check if 116125106 is prime...
Prime Number Generator completed.
339277810 is prime? False (8 factors)
635134784 is prime? False (14 factors)
227057103 is prime? False (48 factors)
289319948 is prime? False (6 factors)
927224302 is prime? False (8 factors)
325473311 is prime? False (4 factors)
133483220 is prime? False (24 factors)
50864148 is prime? False (18 factors)
320205730 is prime? False (64 factors)
116125106 is prime? False (8 factors)
Application finished.
Random primes found: 0. Total time taken : 00:00:14.8281250

```

Fig. Application Console – PrimeNumberGenerator

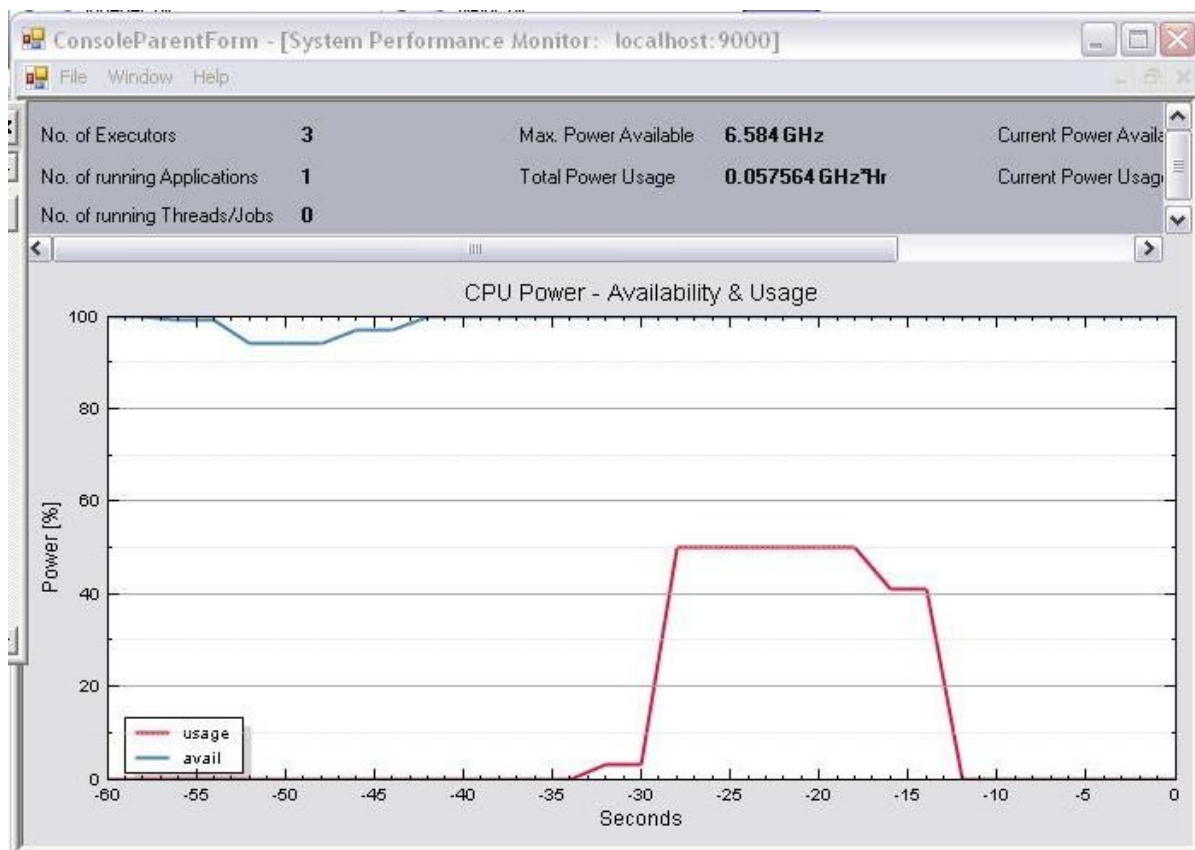
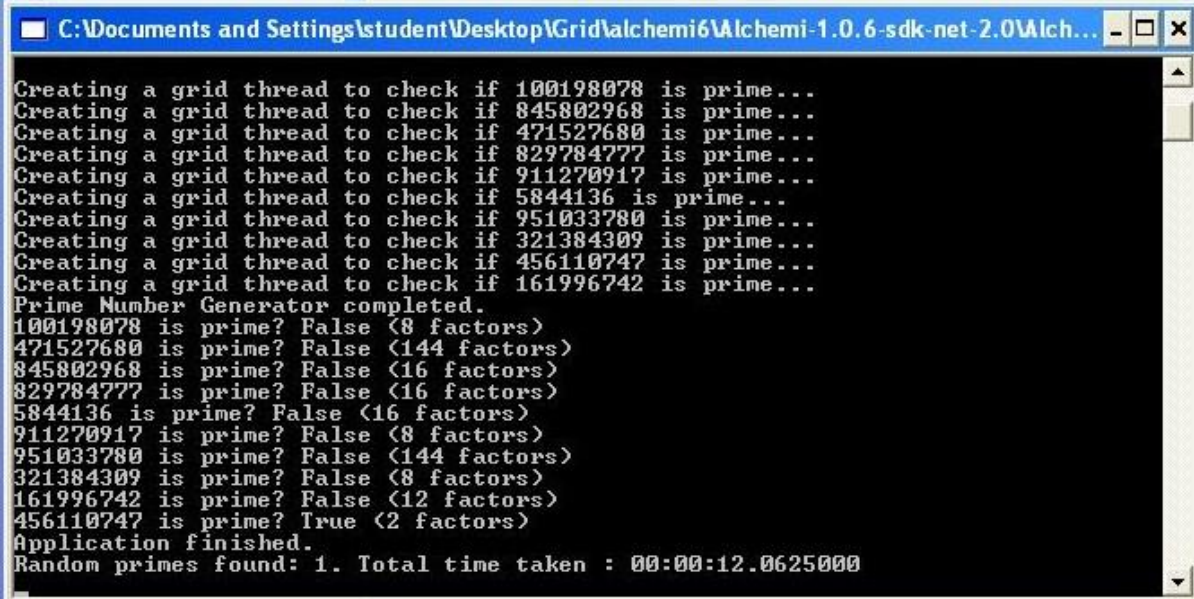


Fig. System Performance Monitoring



### ✚ With Three Executor

In this case THREE EXECUTORS are connected to the MANAGER. Now on running the application, PrimeNumberGenerator application console and the Performance Monitor will be displayed as follows. Here also we will use the value of 'Limit' as '1000000000', for consistency in the result.



```

C:\Documents and Settings\student\Desktop\Grid\alchemi6\Alchemi-1.0.6-sdk-net-2.0\Alch...
Creating a grid thread to check if 100198078 is prime...
Creating a grid thread to check if 845802968 is prime...
Creating a grid thread to check if 471527680 is prime...
Creating a grid thread to check if 829784777 is prime...
Creating a grid thread to check if 911270917 is prime...
Creating a grid thread to check if 5844136 is prime...
Creating a grid thread to check if 951033780 is prime...
Creating a grid thread to check if 321384309 is prime...
Creating a grid thread to check if 456110747 is prime...
Creating a grid thread to check if 161996742 is prime...
Prime Number Generator completed.
100198078 is prime? False (8 factors)
471527680 is prime? False (144 factors)
845802968 is prime? False (16 factors)
829784777 is prime? False (16 factors)
5844136 is prime? False (16 factors)
911270917 is prime? False (8 factors)
951033780 is prime? False (144 factors)
321384309 is prime? False (8 factors)
161996742 is prime? False (12 factors)
456110747 is prime? True (2 factors)
Application finished.
Random primes found: 1. Total time taken : 00:00:12.0625000

```

Fig. Application Console - PrimeNumberGenerator

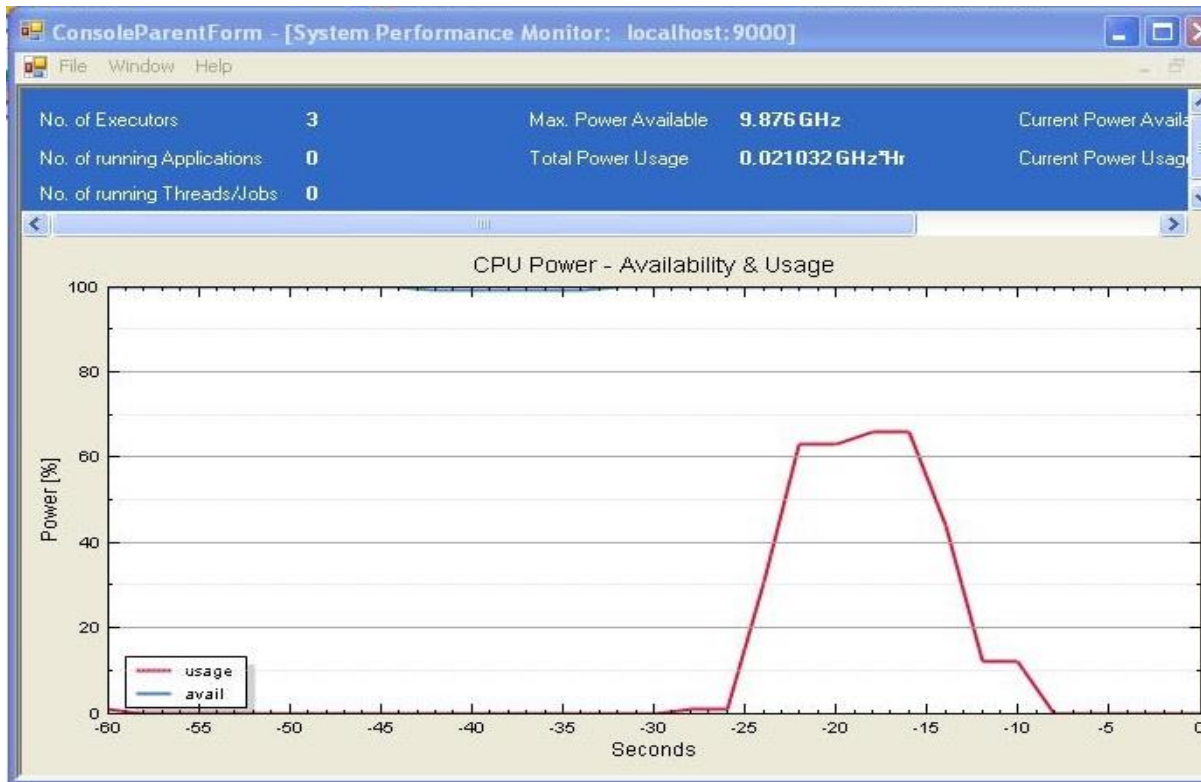


Fig. System Performance Monitor

### Resultant Readings

<u>No. of Executors</u>	<u>Max. Power</u>	<u>Limit (Input)</u>	<u>Time Taken</u>
One	3.292 GHz	10000000000	21.8281250
Two	6.584 GHz	10000000000	14.8281250
Three	9.876 GHz	10000000000	12.0625000



## **PRACTICAL 7**

**AIM: GridSim – To study a Grid Simulation Toolkit.**

### **Introduction**

Peer-to-Peer/Grid computing has emerged as a potential next generation platform for solving large-scale problems in science, engineering, and commerce. Simulation appears to be the only feasible way to analyse algorithms on large-scale distributed systems of heterogeneous resources. Unlike using the real system in real time, simulation works well, without making the analysis mechanism unnecessary complex, by avoiding the overhead of co-ordination of real resources. Simulation is also effective in working with very large hypothetical problems that would otherwise require involvement of a large number of active users and resources, which is very hard to coordinate and build at large-scale research environment for investigation purpose.

As a large-scale simulation consumes large amount of computing power, it is likely to use parallel and cluster computing systems. In this simulation it is possible to model applications in the areas of biotechnology, astrophysics, network design, and high-energy physics in order to study usefulness of our resource allocation techniques. The results will have significant impact on the way resource allocation is performed for solving problems on cluster and grid computing systems.

### **Features of GridSim**

The prominent features of GridSim toolkit can be stated as follows:

- The GridSim toolkit allows modeling and simulation of entities in parallel and distributed computing (PDC) systems-users, applications, resources, and resource brokers (schedulers) for design and evaluation of scheduling algorithms.
- It provides a comprehensive facility for creating different classes of heterogeneous resources that can be aggregated using resource brokers. for solving compute and data intensive applications.
- A resource can be a single processor or multi-processor with shared or distributed memory and managed by time or space shared schedulers.
- The processing nodes within a resource can be heterogeneous in terms of processing capability, configuration, and availability.
- The resource brokers use scheduling algorithms or policies for mapping jobs to resources to optimize system or user objectives depending on their goals.

## Functionalities of GridSim

The prominent functionalities of GridSim can be pointed as follows:

- Incorporates failures of Grid resources during runtime.
- New allocation policy can be made and integrated into the GridSim Toolkit, by extending from AllocPolicy class.
- Has the infrastructure or framework to support advance reservation of a grid system.
- Incorporates a functionality that reads workload traces taken from supercomputers for simulating a realistic grid environment.
- Incorporates an auction model into GridSim.
- Incorporates a datagrid extension into GridSim.
- Incorporates a network extension into GridSim. Now, resources and other entities can be linked in a network topology.
- Incorporates a background network traffic functionality based on a probabilistic distribution. This is useful for simulating over a public network where the network is congested.
- Incorporates multiple regional GridInformationService (GIS) entities connected in a network topology. Hence, you can simulate an experiment with multiple Virtual Organizations (VOs).
- Adds ant build file to compile GridSim source files.

## **PRACTICAL 8**

**AIM: To run two sample programs using GridSim Toolkit.**

### **Configure**

GridSim examples are generally run as normal java programs as these examples are themselves java programs.

Here, we will use Netbeans IDE to run these examples.

#### **Creating Project**

First of all, we need to create a project as generally done in Netbeans. Here, it would be a Java Application Project.

Then, create a new package in the Source Packages, and create a java class in that package as per the general java conventions.

Add the whole java code of the example into the class.

#### **Adding Libraries**

When the code is added to the class, IDE would indicate some errors based on the library functions.

This is so, because the libraries or jar files of GridSim is not added to the project. It is needed to run the program as well as to support the GridSim toolkit functionalities.

The GridSim libraries are:

- gridsim.jar
- simjava2.jar

Now, these can be added to the project in the Netbeans IDE by right clicking on the project, opening the properties tab and adding the jar files to the libraries.

## Run the Program

Now, the program is ready to be run without any errors. Then it can be run as a usual program.

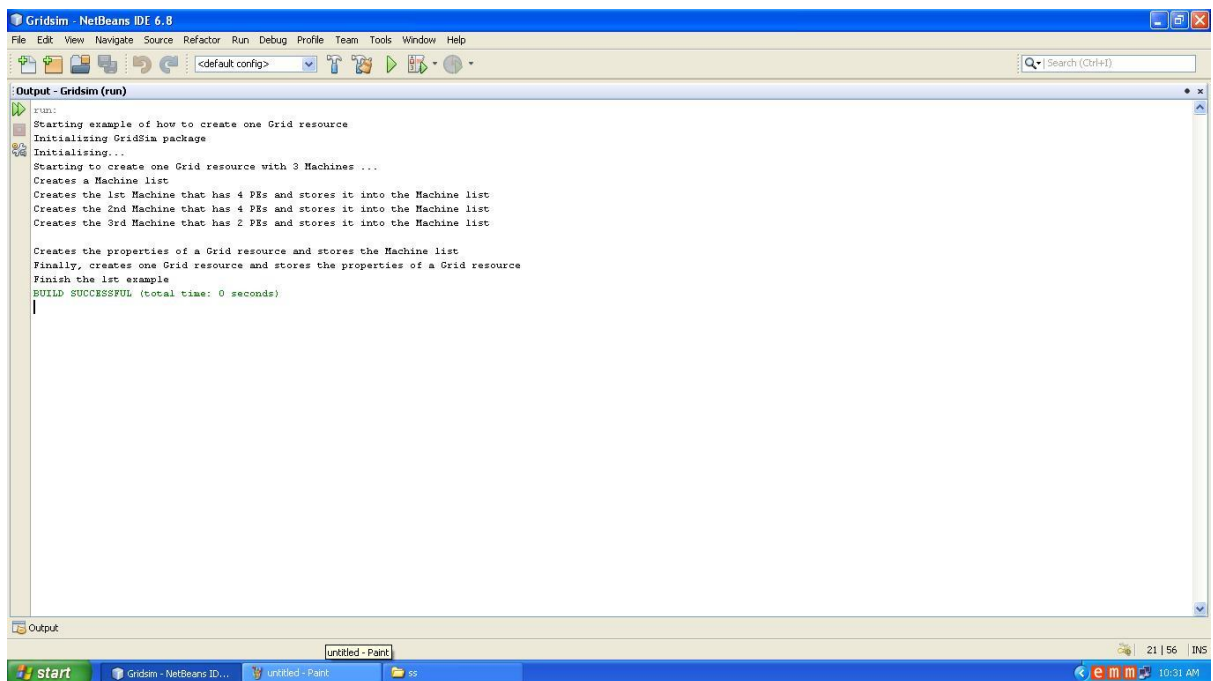
There are two examples, we have used to show the functionalities of GridSim Toolkit.

- **Example 1:**

Example 1 is a simple program to demonstrate how to use GridSim package. This example shows how to create one Grid resource with three machines.

This class creates one Grid resource with three machines. Before creating any of GridSim entities, you should remember to call `GridSim.Init()`.

The output obtained after running this example could be shown as below in the figure.



```

Gridsim - NetBeans IDE 6.8
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
<default config>
Output - Gridsim (run)
run:
Starting example of how to create one Grid resource
Initializing GridSim package
Starting to create one Grid resource with 3 Machines ...
Creates a Machine list
Creates the 1st Machine that has 4 PEs and stores it into the Machine list
Creates the 2nd Machine that has 4 PEs and stores it into the Machine list
Creates the 3rd Machine that has 2 PEs and stores it into the Machine list
Creates the properties of a Grid resource and stores the Machine list
Finally, creates one Grid resource and stores the properties of a Grid resource
Finish the 1st example
BUILD SUCCESSFUL (total time: 0 seconds)

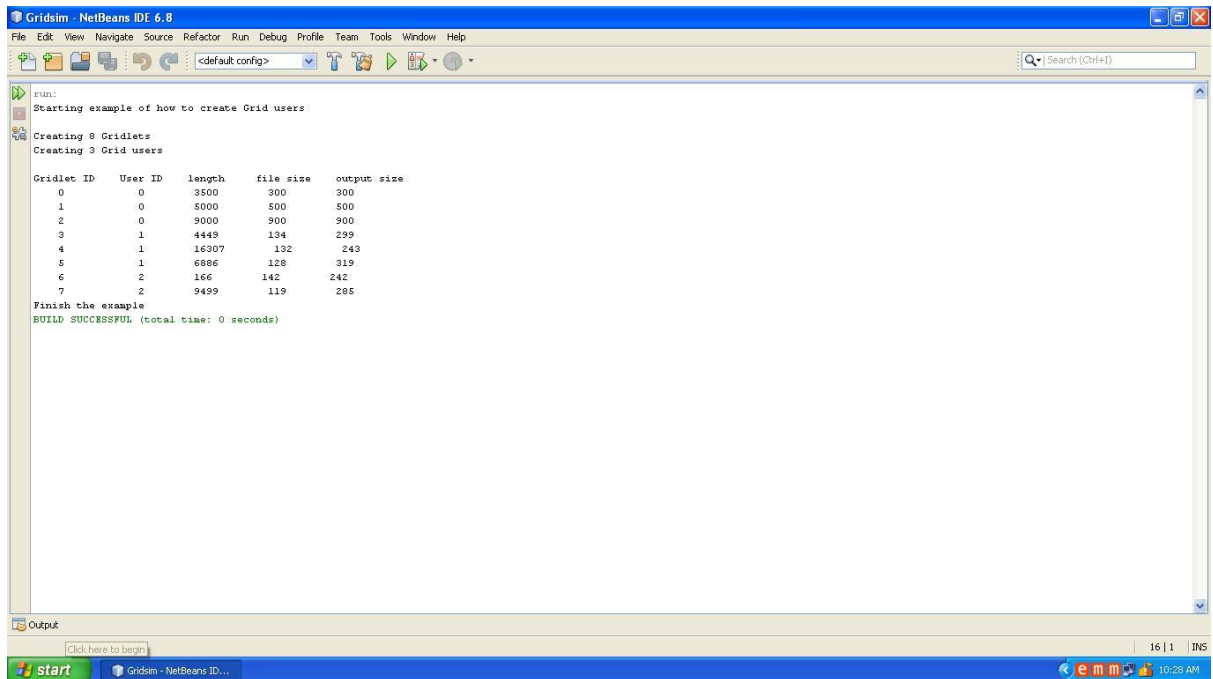
```

Fig. Output of Example 01

- **Example 2:**

Example 2 is a simple program to demonstrate how to use GridSim package. This example shows how to create one or more Grid users. A Grid user contains one or more Gridlets. Therefore, this example also shows how to create Gridlets with and without using GridSimRandom class.

The output of example 2 obtained after running it can be shown as in figure below.



```

run:
Starting example of how to create Grid users
Creating 8 Gridlets
Creating 3 Grid users

Gridlet ID   User ID   length   file size   output size
0            0         3500     300         300
1            0         5000     500         500
2            0         9000     900         900
3            1         4449     134         299
4            1         16307    132         243
5            1         6886     128         319
6            2         166      142         242
7            2         9499     119         285

Finish the example
BUILD SUCCESSFUL (total time: 0 seconds)
  
```

Fig. Output of Example 02

## **PRACTICAL 9**

**AIM: Case study on the following:**

- 3. Google Docs**
- 4. Amazon EC2**

### **1. GOOGLE DOCS**

Google Docs is a free, Web-based office suite, and data storage service offered by Google. It allows users to create and edit documents online while collaborating in real-time with other users. Google Docs combines the features of Writely and Spreadsheets with a presentation program incorporating technology designed by Tonic Systems. Data storage of files up to 1 GB.

### **What is Google Docs?**

Google Docs is

- an internet based office suite: the office programs, and the documents you create with them, are all kept on a Google server and accessed via the internet at [docs.google.com](https://docs.google.com)
- available to anyone with internet access whether through a PC, laptop or mobile device
- free with a Google account; if you have a Gmail account you already have access to Google Docs
- allows you to share documents for viewing and editing, and allows multiple users to collaborate simultaneously on a project over the internet
- 1 GB of free storage space; you can purchase extra storage space starting at 20 GB for \$5/year
- Google Docs has limited features; it is constantly under revision and open to change; it has very limited support and training opportunities

### **Features**

- Software as a service:

Google Docs is Google's "software as a service" office suite. Documents, spreadsheets, presentations can be created with Google Docs, imported through the web interface, or sent via email.

➤ Collaborative:

Google Docs serves as a collaborative tool for editing amongst in real time. Documents can be shared, opened, and edited by multiple users at the same time. Users cannot be notified of changes, but the application can notify users when a comment or discussion is made or replied to, facilitating collaboration.

➤ Sharing:

Google Docs is one of many cloud computing document-sharing services. The majority of document-sharing services require user fees, whereas Google Docs is free. Its popularity amongst businesses is growing due to enhanced sharing features and accessibility. In addition, Google Docs has enjoyed a rapid rise in popularity among students and educational institutions.

➤ Synchronize:

Google Cloud Connect is a plug-in for Windows Microsoft Office 2003, 2007 and 2010 that can automatically store and synchronize any for Microsoft Word document, PowerPoint presentation, or Excel spreadsheet to Google Docs in Google Docs or Microsoft Office formats. The Google Doc copy is automatically updated each time the Microsoft Office document is saved. Microsoft Office documents can be edited offline and synchronized later when online. Google Cloud Sync maintains previous Microsoft Office document versions and allows multiple users to collaborate by working on the same document at the same time.

➤ Scripting:

Google Spreadsheets and Google Sites also incorporate Google Apps Script to write code within documents in a similar way to VBA in Microsoft Office. The scripts can be activated either by user action or by a trigger in response to an event

➤ Supported file formats

Google Docs supports 15 file formats:

- Microsoft Word (.DOC and .DOCX)
- Microsoft Excel (.XLS and .XLSX)
- Microsoft PowerPoint (.PPT and .PPTX)
- Open Document Format (.ODT and .ODS)
- Adobe Portable Document Format (.PDF)
- Apple Pages (.PAGES)
- Adobe Illustrator (.AI)
- Adobe Photoshop (.PSD)
- Tagged Image File Format (.TIFF)
- Autodesk AutoCAD (.DXF)

- Scalable Vector Graphics (.SVG)
- PostScript (.EPS, .PS)
- TrueType (.TTF)
- XML Paper Specification (.XPS)
- Archive file types (.ZIP and .RAR)

➤ Data safety and privacy

On March 10, 2009, Google reported that a bug in Google Docs had allowed unintended access to some private documents. It was believed that 0.05% of documents stored via the service were affected by the bug, which Google claimed has been fixed.

➤ Mobile access

The Android Google Docs app allows users to view, edit, and create Google Docs documents, spreadsheets, and presentations. The Android Google Docs app can also take a photo of a document, sign, or other text and use Optical Character Recognition to convert to text that can be edited. (Help, Overview) The iPhone and iPad Safari Browser also allows users to view, edit, and create Google Docs documents, spreadsheets, and presentations. Furthermore, the Google App for iPhone and iPad allows users to view and edit Google Docs. Most other mobile devices can also view and edit Google Docs documents and spreadsheets using a mobile browser. PDF files can be viewed but not edited.

## 2. AMAZON EC2

Amazon Elastic Compute Cloud (EC2) is a central part of Amazon.com's cloud computing platform, Amazon Web Services (AWS). EC2 allows users to rent virtual computers on which to run their own computer applications. EC2 allows scalable deployment of applications by providing a Web service through which a user can boot an Amazon Machine Image to create a virtual machine, which Amazon calls an "instance", containing any software desired. A user can create, launch, and terminate server instances as needed, paying by the hour for active servers, hence the term "elastic". EC2 provides users with control over the geographical location of instances that allows for latency optimization and high levels of redundancy.

Amazon EC2's simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. Amazon EC2 reduces the time required to obtain and boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. Amazon EC2 changes the



economics of computing by allowing you to pay only for capacity that you actually use. Amazon EC2 provides developers the tools to build failure resilient applications and isolate themselves from common failure scenarios.

## Amazon EC2 Functionality

Amazon EC2 presents a true virtual computing environment, allowing you to use web service interfaces to launch instances with a variety of operating systems, load them with your custom application environment, manage your network's access permissions, and run your image using as many or few systems as you desire.

### To use Amazon EC2, you simply:

- Select a pre-configured, templated image to get up and running immediately. Or create an Amazon Machine Image (AMI) containing your applications, libraries, data, and associated configuration settings.
- Configure security and network access on your Amazon EC2 instance.
- Choose which instance type(s) and operating system you want, then start, terminate, and monitor as many instances of your AMI as needed, using the web service APIs or the variety of management tools provided.
- Determine whether you want to run in multiple locations, utilize static IP endpoints, or attach persistent block storage to your instances.
- Pay only for the resources that you actually consume, like instance-hours or data transfer.

## Service Highlights

### ➤ Elastic:

Amazon EC2 enables you to increase or decrease capacity within minutes, not hours or days.

### ➤ Completely Controlled:

You have complete control of your instances. You have root access to each one, and you can interact with them as you would any machine. You can stop your instance while retaining the data on your boot partition and then subsequently restart the same instance using web service APIs.

➤ Flexible:

You have the choice of multiple instance types, operating systems, and software packages. Amazon EC2 allows you to select a configuration of memory, CPU, instance storage, and the boot partition size that is optimal for your choice of operating system and application. For example, your choice of operating systems includes numerous Linux distributions, and Microsoft Windows Server.

➤ Designed for use with other Amazon Web Services:

Amazon EC2 works in conjunction with Amazon Simple Storage Service (Amazon S3), Amazon Relational Database Service (Amazon RDS), Amazon SimpleDB and Amazon Simple Queue Service (Amazon SQS) to provide a complete solution for computing, query processing and storage across a wide range of applications.

➤ Reliable:

Amazon EC2 offers a highly reliable environment where replacement instances can be rapidly and predictably commissioned. The service runs within Amazon's proven network infrastructure and datacenters. The Amazon EC2 Service Level Agreement commitment is 99.95% availability for each Amazon EC2 Region.

➤ Secure:

Amazon EC2 provides numerous mechanisms for securing your computer resources. Amazon EC2 includes web service interfaces to configure firewall settings that control network access to and between groups of instances.

➤ Inexpensive:

Amazon EC2 passes on to you the financial benefits of Amazon's scale. You pay a very low rate for the compute capacity you actually consume.

➤ Virtual Machines:

EC2 uses Xen virtualization. Each virtual machine, called an "instance", functions as a virtual private server. Amazon.com sizes instances based on "Elastic Compute Units".

➤ Free Tier:

As of December 2010 Amazon offers a bundle of free resource credits to new account holders. The credits are designed to run a "micro" sized server for one year. Charges are applied on demand so the credit need not be used in the first month.

➤ Reserved Instances:

Reserved instances enable EC2 service users to reserve an instance for one or three years. There is a fee associated with reserving an instance. The corresponding per hour rate charged by Amazon to operate the instance is much less than the rate charged for non-reserved instances.

## **PRACTICAL 10**

**AIM: Case study on Google App Engine.**

### **INTRODUCTION**

Google App Engine lets we run web applications on Google's infrastructure. App Engine applications are easy to build, easy to maintain, and easy to scale as our traffic and data storage needs grow. With App Engine, there are no servers to maintain. Just upload application, and it's ready to serve users.

We can serve our app from our own domain name (such as <http://www.example.com/>) using Google Apps. Or, we can serve our app using a free name on the appspot.com domain. We can share our application with the world, or limit access to members of our organization.

App Engine costs nothing to get started. All applications can use up to 1 GB of storage and enough CPU and bandwidth to support an efficient app serving around 5 million page views a month, absolutely free. When enable billing for application, free limits are raised, and only pay for resources used above the free levels. We can register up to 10 applications per developer account.

### **THE APPLICATION ENVIRONMENT**

#### **➤ The Sandbox**

Applications run in a secure environment that provides limited access to the underlying operating system. The sandbox isolates your application in its own secure, reliable environment that is independent of the hardware, operating system and physical location of the web server.

Example of the limitations of the secure sandbox environment is an application can only access other computers on the Internet through the provided URL fetch and email services. Other computers can only connect to the application by making HTTP (or HTTPS) requests on the standard ports.

### ➤ **Runtime environments**

Our application can run in one of three runtime environments:

1. Java : With App Engine's Java runtime environment, we can build our app using standard Java technologies, including the JVM, Java servlets, and the Java programming language—or any other language using a JVM-based interpreter or compiler, such as JavaScript or Ruby.
2. Python: App Engine features two dedicated Python runtime environments, each of which includes a fast Python interpreter and the Python standard library.
3. Go: App Engine provides a Go runtime environment that runs natively compiled Go code.

### ➤ **The Datastore**

App Engine provides a distributed data storage service that features a query engine and transactions. Just as the distributed web server grows with traffic, the distributed datastore grows with your data.

The App Engine datastore is not like a traditional relational database. Data objects, or "entities," have a kind and a set of properties. Datastore entities are "schemaless." The structure of data entities is provided by and enforced by your application code. The Java JDO/JPA interfaces and the Python datastore interface include features for applying and enforcing structure within your app. Our app can also access the datastore directly to apply as much or as little structure as it needs.

The datastore is strongly consistent and uses optimistic concurrency control. An update of a entity occurs in a transaction that is retried a fixed number of times if other processes are trying to update the same entity simultaneously.

### ➤ **App Engine Services**

App Engine provides a variety of services that enable to perform common operations when managing your application. The following APIs are provided to access these services:

### ➤ **URL Fetch**

Applications can access resources on the Internet, such as web services or other data, using App Engine's URL fetch service. The URL fetch service retrieves web resources using the same high-speed Google infrastructure that retrieves web pages for many other Google products.

### ➤ Mail

Applications can send email messages using App Engine's mail service. The mail service uses Google infrastructure to send email messages.

### ➤ Memcache

The Memcache service provides your application with a high performance in-memory key-value cache that is accessible by multiple instances of your application. Memcache is useful for data that does not need the persistence and transactional features of the datastore, such as temporary data or data copied from the datastore to the cache for high speed access.

### ➤ Image Manipulation

The Image service lets your application manipulate images. With this API, you can resize, crop, rotate and flip images in JPEG and PNG formats.

## FEATURES

- Google App Engine makes it easy to build an application that runs reliably, even under heavy load and with large amounts of data. App Engine includes the following features:
- dynamic web serving, with full support for common web technologies
- persistent storage with queries, sorting and transactions
- automatic scaling and load balancing
- APIs for authenticating users and sending email using Google Accounts
- a fully featured local development environment that simulates Google App Engine on your computer
- task queues for performing work outside of the scope of a web request
- scheduled tasks for triggering events at specified times and regular intervals

## CREATING APPLICATION USING JAVA

For developing application we require the App Engine software development kits (SDKs) for Java. It includes a web server application that emulates all of the App Engine services on your local computer. Each SDK includes all of the APIs and libraries available on App Engine. The web server also simulates the secure sandbox environment, including checks for attempts to access system resources disallowed in the App Engine runtime environment.

### **Following are the steps to create a java Application**

- Build an App Engine application using standard Java web technologies, such as servlets and JSPs
- Create an App Engine Java project with Eclipse, and without
- Use the Google Plugin for Eclipse for App Engine development
- Use the App Engine datastore with the Java Data Objects (JDO) standard interface
- Integrate an App Engine application with Google Accounts for user authentication
- Upload your app to App Engine