Easy clustering with openMosix

JAVIER BILBAO, GORKA GARATE, ANDONI OLOZAGA, ALVARO DEL PORTILLO
Department of Applied Mathematics
University of the Basque Country
School of Engineering Alda. Urkijo, s/n 48013 - Bilbao
SPAIN

Abstract: - In order to resolve some calculations and some equation systems, personal computer may be unsatisfactory because some of these operations can require supercomputation capabilities. Clustering is a very good option to avoid a very high cost, and to make a complete use of some PCs with different uses such as laboratory. We will present some easy ways to cluster some computers and some tests to compare processes running in a cluster and in only one computer.

Key-Words: - clustering, parallel computation, middleware, Linux, execution time

1 Introduction

Different types of engineering applications are implemented in order to calculate or resolve massive calculations. Some of these applications use commercial software such as Mathematica, Matlab, Ansys, Nastran, Cosmos, and so on. Workstations and personal computers are used to run the software, and, although the problem that has to be solved is not extremely difficult, the execution time is usually more than 1 hour, commonly several hours.

In spite of the great advance in the computer technology in the last years that has allowed the development of computers with calculation capacities that were inconceivable some years ago, the execution time is not reduced because of the increase of the difficulty and precision of the models.

Apart from the solution of buying supercomputers of thousands of euros, or using the future new technology of dual core microprocessors, there is another option: clustering of personal computers. A computer cluster is a group of locally connected computers that work together as a unit.

The way to implement a computer cluster is not unique. One of the more popular implementations is a cluster with nodes running Linux as the operating system and free software to implement the parallelism. A popular software to run the cluster and applications among the nodes of the cluster is openMosix [1].

Actually, openMosix is a Linux kernel extension for single-system image clustering. This kernel extension turns a network of ordinary computers into a supercomputer for Linux applications. Once you have installed openMosix, the nodes in the cluster start talking to one another and the cluster adapts itself to the workload. Processes originating from any one node, if that node is too busy compared to others, can migrate to any other node.

Fig. 1. Logo of openMosix project

In this paper, we present some easy ways to cluster some computers without the necessity of patching the operating system. In fact, it will not be necessary to install the operating system on the hard disk. Tests of compression and rendering are also presented.

2 Clustering

In general, clustering is a technology or a set of technologies that allow multiple computers to work together to solve common computing problems. The computing problems in question can be anything from complex CPU-intensive scientific computations to a horde of miscellaneous processes with no underlying commonality [2], [3].

At the beginning, clusters were developed in order to resolve problems of supercomputation, but nowadays it is not their only use [4]. The increase of the use of the web technology has caused the implementation of clusters in different servers with the aim of service to a high number of clients. This type of technology have been implemented in services such as web servers, email, e-commerce, or high performance data bases.

A cluster system may be considered as being made up of four major components, two hardware and two software [5]. The two hardware components are the nodes that perform the work and the network that interconnects the nodes to form a single system. The two software components are the collection of tools used to develop user parallel application programs and the software environment for managing the parallel resources of the cluster [6].

Clusters of computers can be classified in two groups: high availability clusters (such as web servers) and high performance clusters (supercomputation) [7].

An operating system allows an application to be launched from storage media into memory and to execute through to completion (termination). openMosix, as a "Single System Image", provides the ability for an application to be launched from any cluster node into memory and to execute on any node in the cluster (as a virtual instance of the application) [8]. Any given virtual application instance is migrated to the node with the most available capacity or resources.

A cluster can also be implemented in "normal" computers, PCs, that are used for example in a laboratory of a university. Traditional way began with the creation of a partition of the hard disk to install the Linux chosen by the user. Linux distribution was able to be chosen by the user, but not the version. The version had to be compatible with the patch of the openMosix, and the version of openMosix was usually lower that the last version of the Linux kernel. So if you had a Linux installed in your computer, you had to install another one and put the respective link in the Lilo or Grub menu (in order to select Windows, normal Linux or Linux-openMosix). When you had the openMosix last version, you had to patch the Linux kernel. And after that, configure the result.

Some developers think that looking at the problem from the technical perspective, managing a cluster of 30 nodes, where each node contains a full Linux distribution (including GUI) is not so different from managing a single node that contains a full Linux distribution. From the security perspectives, however, each additional non-essential software package that is stored on these drone or slave nodes adds both unnecessary risk and unwanted cost.

Sometimes this differential, from the technical perspective (administrate a cluster of 30 nodes or 1 single node with Linux), is not so small. It depends on the network (whether the cluster is part of a big network, because the generated traffic by the other computers of the net), the storage media (whether the operating system is on the hard disk, because if it is on a live-cd there is a task to translate the operating system to the memory and a diminution of the memory amount free to implemented the application), and other different situations (such as running the operating system by a virtual machine in order to not "touch" the hard disk or because the Linux-openMosix version is not supported by the "modern" computer –such as some recent Pentium IV and the clusterKnoppix 3.6).

On the other hand, not all distributions are the same. Heterogeneous cluster construction allows an organization to take advantage of the feature differences that separate the relevant Linux distributions. In other words, by planning for the cluster, selecting the right distribution for the master node(s) and the right distribution for drone nodes, you will build the right cluster for your organization.

Three are the easy ways to implement a cluster with openMosix as software to balance the nodes and to migrate the processes:

1. Use only clusterKnoppix distribution by a live-cd.
2. Use Chaos distribution with live-cd.
3. Use diskless.

2.1 clusterKnoppix

clusterKNOPPIX is a modified Knoppix distribution using the OpenMosix kernel. Features:

- openMosix terminal server - uses PXE, DHCP and tftp to boot linux clients via the network.
- No cdrom drive/harddisk/floppy needed for the clients openMosix autodiscovery - new nodes automatically join the cluster (no configuration needed).
- Cluster Management tools - openMosix userland/openMosixview. Every node has root access to every other node via ssh/RSAkeys MFS/dfsa support.
- Every node can run full blown X (PC-room/demo setup) or console only (more memory available).

When you boot your computer (where there is installed any operating system, including Windows), you only have to introduce the CD-ROM of clusterKnoppix and the Knoppix operating system will start (with openMosix). Linux will be loaded into the RAM memory. If there is more computers

with the clusterKnoppix started, they will see each other and will make up a cluster. By themselves. But there is one disadvantage and one previous task: the disadvantage is that the last version (3.6) of clusterKnoppix does not run very well with some Pentium IV; the previous task is creating a DHCP server and a domain (if they do not exist).

2.2 Chaos

The main goal of the Chaos Linux openMosix distribution is to provide the best union in the conflicting interest between the security and manageability of a compact operating system, and the need for a user friendly and fully featured operating system, in the cluster environment [11].

The cluster would consist of one or two home nodes (master nodes), furnished with the complete software suite required to effectively execute objective tasks on the cluster, and supplemented with slave or drone nodes that contain only enough software to start the host hardware and to join the cluster effectively.

Note that the home node (or master node) is the only node that needs the application existing on storage media. Each and every other node (drone node) in the cluster could run a distribution that is as small as the Linux kernel itself. That is to say, one of the computers of the cluster (master node) will have the clusterKnoppix and it will be the "manager" of the cluster; all the other nodes (drones) will have Chaos distribution and they will not access to the shell (all processes have to be sent from the master node).

Chaos is designed to be a compact Linux and openMosix distribution that is secure and highly distributable. Chaos does not interfere with the host operating system, running entirely from RAM it also frees the CDROM boot media [12], [13].

Chaos is not intended to be fully-laden with application software; it is well suited as a minimized "drone node" distribution. Though, the biggest advantage of using Chaos is its focus on security and deployment methodology.

Chaos is the first Linux and openMosix distribution to add native network security to the cluster deployment. Every node in the cluster does its own packet filtering, and establishes IPSEC tunnels as required. This is performed transparently, requiring no user intervention.

While Chaos is capable of PXE booting new nodes, CD booting will be used.

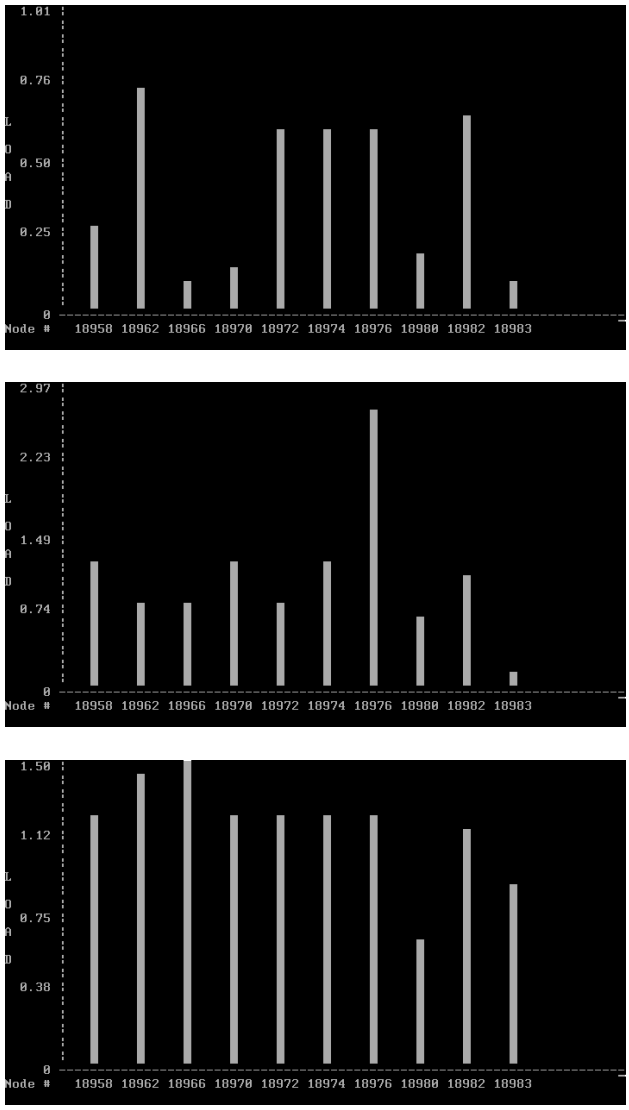

Fig. 2. Use of Chaos: monitorization of the working nodes using Mosmon

2.3 Diskless

The term diskless refers to the fact that slave nodes systems do not need a local file system device to boot and run.

When slave nodes of the cluster boot, they send a boot request to the network that can be intercepted by the appropriate managing computer, the master node. The master computer then sends an image to the drone nodes. This image carries a fully functional computer's most important programs along with instructions on how to load other programs off of the network. The slave nodes then create a virtual hard drive in their RAM and load the image there.

Usually, the boot is made by wake on LAN (WOL) process.

Diskless clusters use Network File System (NFS) to access to files among the nodes. NFS provides remote access to shared file systems across

networks. This means that a file system may actually be sitting on the master machine, but a drone machine can mount that file system and it will look on the slave machine like the file system resides on the local machine.

It is just a question of combining the idea of a diskless operation with openMosix and the result was an openMosix cluster that could be expanded as easily as plugging a new computer into the LAN and telling it to boot off the network.

But there are limitations on the type of computation that can be made on this kind of node: intensive I/O applications can be executed on such nodes but will not scale well and will slow down each calculation, resulting in inefficiency [14].

Besides, memory swapping is important because if an application's needs exceed the amount of available physical memory, the program could crash or the operating system might hang. Although memory can be swapped across the network to non-local drives, this action will drastically degrade the cluster's performance. For these reasons, if jobs with unknown memory requirements will run on the cluster, or we have nodes with very low memory, diskless nodes are not recommended [15].

But it is clear which the advantages are: efficient sharing of resources, easy administration, data security (since there is not hard disk, there is not physically-unsecured systems to contain data after they are powered off), no complaints from users of computers that are slave nodes in the cluster (computers remain non-upset after their use).

3 Tests

Some case studies have been implemented for performance evaluation of these easy ways. We will present tests with file compression and image rendering. When the cluster is made up of 15 computers, there are 10 Pentium III, 256 Mb RAM, and 5 Pentium IV, 256 Mb RAM, and all nodes are running clusterKnoppix or Chaos, that is, they use RAM memory to install the operating system image and to run the process. When the cluster is made up of 4 computers, they are Pentium IV, 2 with 512 Mb RAM, and 2 with 1 Gb RAM.

3.1 File compression

In order to do this probe, 19 .wav files of two music CDs were dumped to the hard disk.

After that, the compression process was implemented by means of a free software, BladeEnc.

The process has been programmed to do the compression of all .wav files of the CD in parallel, that is to say, each .wav file has been treated by an execution process of the BladeEnc software.

When this process has been carried out by only one Pentium IV computer, it has taken 33 minutes 27 seconds, but with a global time of 172 minutes and 20 seconds if the task is developed sequentially. But when the process has been implemented by a cluster of 15 Pentium IV and Pentium III computers, the execution time has been 25 minutes 48 seconds:

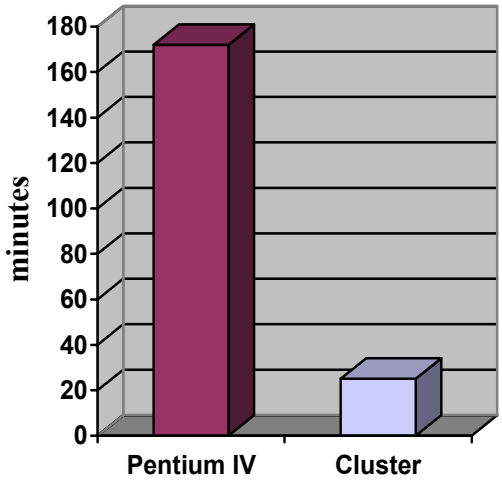

Fig. 3. Execution time for a cluster of fifteen computers and one Pentium IV to file compression

This last implementation has been monitorized by means of the openMosixview software. In the next figure it is shown how various processes have migrated to the rest of the nodes of the cluster:

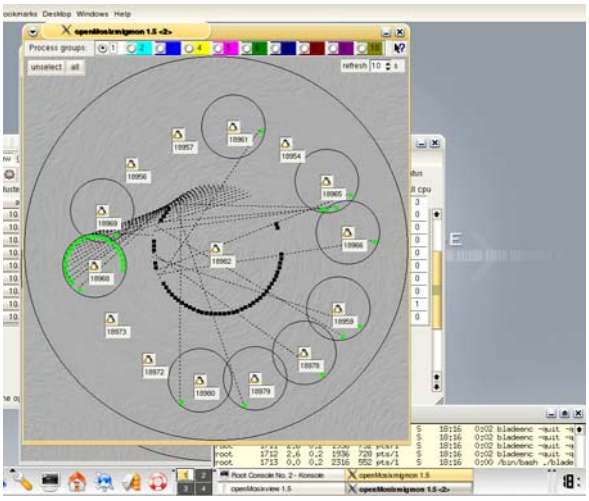

Fig. 4. Monitorization of the working nodes when the processes are migrating among the nodes of the cluster

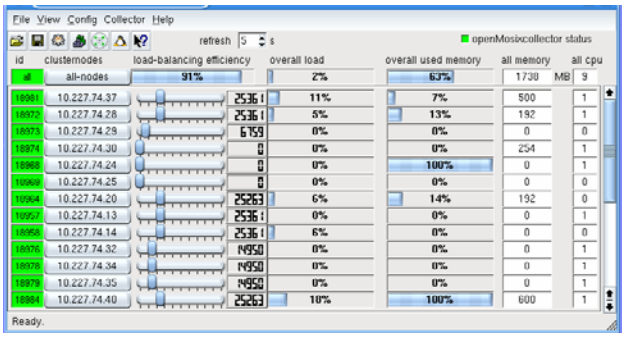

Fig. 5. Monitorization of the working nodes with the load balance and used memory

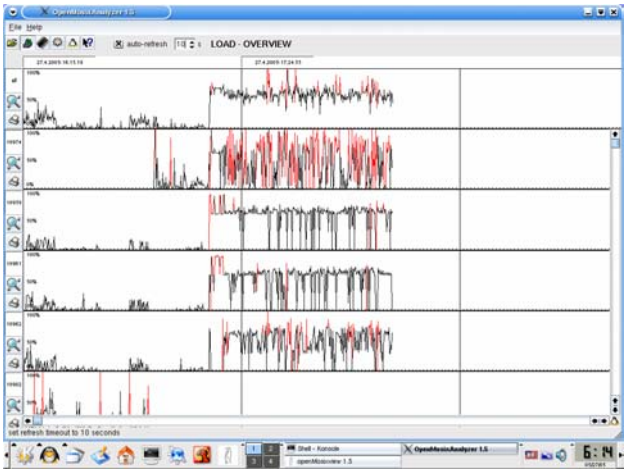

Fig. 6. Use of the memory and load of each node of the cluster when cluster runs processes

The cluster took approximately 15% time less to carry out the operation sequentially, but the time difference between only one computer and the cluster (with 4 computers) is increased when the number of files that have to be compressed is also increased. However, if the process is developed in parallel, the difference is reduced: the cluster took 77% time less than only one Pentium IV:

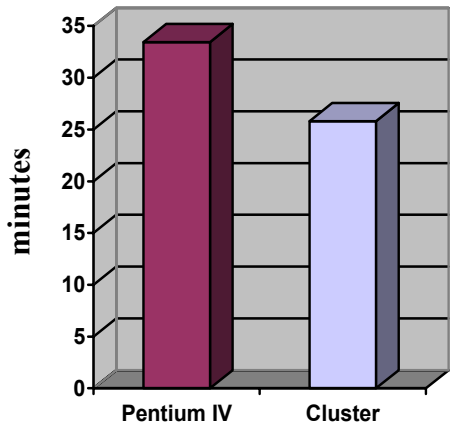

Fig. 7. Execution time for a cluster of fifteen computers and one Pentium IV to file compression

When the compression is implemented by the cluster, there is a constant direct access to the hard disk of the computer that launched the processes to the other nodes of the cluster. And although openMosix has a oMFS file system that allows any node of the cluster to access to the file system of the other nodes, this access time is higher than the time that a computer needs to access to its own hard disk.

On the other hand, the process (compression of a music file) is quite brief and an only Pentium IV is able to run various processes at the same time (in parallel), so the advantage of the cluster will be more notorious when the tasks will be bigger in time (sequentially).

Other aspect to take into account is the traffic of the network among the computers of the cluster. If the cluster is not isolated with a switch or hub, and the computers are interconnected with other computers out of the cluster, there is a constant and considerable traffic that reduces the efficiency of the operation. So, it is highly recommended to run the processes in a separated cluster, because the time gain can be up to 250%.

3.2 Rendering case study

We designed a simple scene made up of three cubes, and the effect was a turn. This scene has been programming to be rendered by the software POV-Ray (Persistence of Vision Ray Tracer).

The next shell script was done using Python language:

```
#!/usr/bin/python

import os, time

framesperproc=10
totalframes=50

filename='sc00'
infile=filename+'.pov'
outfile=filename+'.tga'
geometry='Width=640 Height=480 '

print time.localtime()
print '#'*80

for startframe in range(0,totalframes,framesperproc):
print "started rendering "+"from "+str(startframe)+' to '+str(startframe+framesperproc)+' ...'
os.system('time povray +A0.4 Display=false +geometry+'+KFI'+str(startframe)+'
+KFF'+str(startframe+framesperproc)+' +I'+infile+' +O'+outfile+' 2>/dev/null&')
```

It was specified that 50 images of the scene had to be obtained, and in each process 10 of these images would be dealt with. Therefore, 5 processes are created.
When the rendering process was done in a Pentium IV, it took 2 minutes and 42 seconds.
The same task done in a cluster of 4 Pentium IV computers took an execution time of 1 minute and 6 seconds.

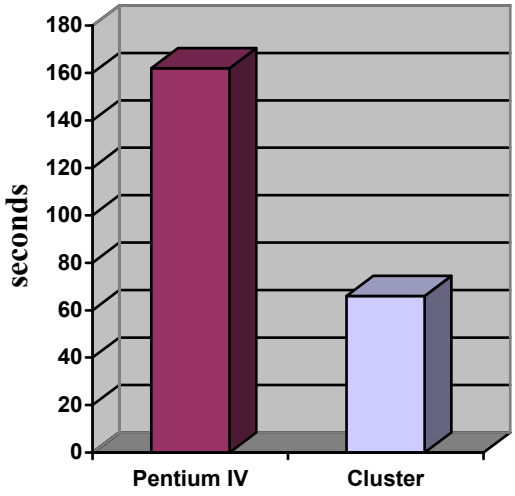

Fig. 8. Execution time for a cluster of fifteen computers and one Pentium IV to file compression

Fig. 9. Cluster made up of computers of a teaching laboratory

4 Conclusion

This paper presents three easy ways to implement a cluster with openMosix as software to balance the nodes and to migrate the processes: clusterKnoppix, Chaos and the use of diskless.

Results of some evaluations show that the use of a cluster requires lower execution time, but that it is conditioned by some external constraints such as the traffic of the network, etc.

References:

[1] J. Bilbao, G. Garate, First step in a PC cluster development with openMosix, *WSEAS Transactions on Computers*, 6, vol. 3, December 2004, pp. 2068-2072
[2] O. Pino, R. F. Arroyo, F. J. Nievas, *Los clusters como plataforma de procesamiento paralelo*
[3] R. M. Yañez, *Introducción a las tecnologías clustering*
[4] D. Santo, *El proyecto de cluster SSI openMosix*
[5] E. Plaza, *Cluster heterogéneo de computadoras*
[6] M. A. Perez, *Arquitecturas paralelas*
[7] M. Catalán i Coït, *Manual para Clustering con openMosix*
[8] M. Colomer, *Clustering con openMosix*
[9] T. Sterling, *Beowulf cluster computing with Linux*, MIT Press, Cambridge, 2001
[10] D. Robbins, *openMosix*, http://www.intel.com
[11] Ian Latter, *Security and openMosix; Securely deploying SSI cluster technology over untrusted networking infrastructure*, http://itsecurity.mq.edu.au/papers/White Paper - Security and openMosix.pdf, December 2003
[12] Moshe Bar, *HPC Computing Applied to Business Applications*, http://openmosix.sourceforge.net/Business_Applications_2003.pdf, 2003
[13]Giacomo Mulas, *Turning a group of independent GNU/Linux workstations into an (Open)Mosix cluster in an untrusted networking environment and surviving the experience*, http://www.democritos.it/events/openMosix/papers/cagliari.pdf, November 2002
[14] B. Des Ligneris, M. Barrette, M. Dagenais, *Thin-OSCAR: Diskless Clustering for All*, http://www.linuxworld.com/read/43718.htm
[15] C. Stanton, R. Ali, Y.C. Fang, M.A. Hussain, Installing Linux High-Performance Computing Clusters, *Power Solutions*, 4, 2001, pp. 11-16