Application for computational cluster performance tests configured in HTCONDOR

Vivian Orejuela[1,*], Álvaro Sánchez Ramirez[2], Andrés Felipe Toro[3], Andrés Felipe Gonzalez[4] And Diego Briñez[5]

[1]Profesor, Facultad de Ingeniería, UCEVA, Tuluá Colombia
[2]Estudent, Facultad de Ingeniería, UCEVA, Tuluá Colombia
[3]Estudent, Facultad de Ingeniería, UCEVA, Tuluá Colombia
[4]Estudent, Facultad de Ingeniería, UCEVA, Tuluá Colombia
[5]Estudent, Facultad de Ingeniería, UCEVA, Tuluá Colombia

Abstract. Increase the processing power in less time with high performance low-cost process is important for universities, for this reason computational clusters take paramount importance nowadays. In this paper it is shown the study of the performance time of a group that has been physically configured with a node and 3 nodes with the Core 2 Duo Quad processors, which have a Scientific Linux operating system and a cluster management software called HTCONDOR which is described time in the C language, to obtain the prime numbers in a range of 1 to 15 million and thus be able to launch the process from the point of destination to the worker nodes divided by groups of 5 million and take the measure of time. Achieving the difference between runtime on a PC and a high-performance cluster.

Index Terms— Computational hight performance, Cluster, HTCondor, Programing Pararell.

1 Introduction

The construction of the high performance cluster arises from the necessity of implementing a tool to support the projects and other research and academic activities of the different faculties of the university and other regional organizations.

Generally speaking, many researchers about high computing performance use technologies that allow us to increase the computing capacity, in order to gain extra speeding point's capacity to offer faster and more accurate results. It allows researchers to make use of these tools according to the focus of their research.

This article describes the cluster configuration and the tools used for its management and launching processes, then, the conducted study is presented to obtain the measurement performance, it uses a simple algorithm created in C language, it launches the work in the computational cluster and in a personal computer with average characteristics.

1.1 Background

Different cluster infrastructures have been implemented in the world; in fact, they have high performance characteristics, some of them are: Beowulf, Berkely NOW, Google, Cluster PS2, and Thunder, among others. Nowadays, there is an event or ranking called TOP500, this project started in 1993 as a need to find the definition of Supercomputer and conduct comparable statistics to measure the performance of these machines.

In Colombia, clusters are usually used to optimize the performance and availability of the devices, this type of supercomputing structure is more economic than individual computers of comparable speed and availability. It offers lots of resources towards the educational and research field of the computers such as the development of research projects. That is why some universities of the country are starting in the parallel and high performance computing field. Among the main universities that are part of this growing group are the Javeriana University, the University of the Andes, the Autonomous University of Manizales, the Industrial University of Santander, the Pontificia Bolivariana University, among others that are developing this tool to get into this great community, as well as the UCEVA. This list of universities that have a computer cluster, they have also used the setting tool, the management tool HTCONDOR, as a common factor. This can be attributed to a large extent to the training provided by Grid Colombia 2 and those universities that are part of this great network. Each one of these infrastructures has been created in order to encourage the development of research in your institution.

2 Theoretical Framework

A computing cluster is similar to a supercomputer but it has low price, where several computers are connected to achieve power and performance which is better than a normal computer .So, their individual power is added and grouped to carry tasks that require a great power

* Corresponding author: author@e-mail.org

potential, it is almost the same when forecasting to establish future predictions with high precision, simulating the behavior of protein synthesis, modeling large amounts of information etc. In fact, this article focuses specifically on high-performance clusters, which are designed to perform computing operations that require a large amount of processing and calculation, the implementation of this type of cluster allows the machines that make part of them work in parallel .Similarly, it reduces time in order to accomplish tasks comparison to computers of normal yield in which a task would take much more time. It is important to say that computing cluster is a great advantage regarding with costs which allows institutions and medium-sized companies to carry out the creation of a cluster of this type. Fig. 1. Caption of the Figure 1. Below the figure.

2.1. Cluster configuration

Generally speaking, the configuration of a high-performance computer cluster consists of two parts. The first is the software: an operating system specially designed for this task (for example a modified Linux kernel), compilers and special applications which allows the programs running on the system and take advantages of the cluster, the second part is the hardware that consists of computers connected network.

2.2 Physical computational Cluster configuration

In order to implement computer cluster in the UCEVA, the Scientific Linux operating system was used, which is a version of Linux created by Fermilab as the cluster administration tool HTC Condor. The following shows the configuration of three workers nodes and a main node called the c-head node that uses HTCondor to manage the tasks in the cluster.

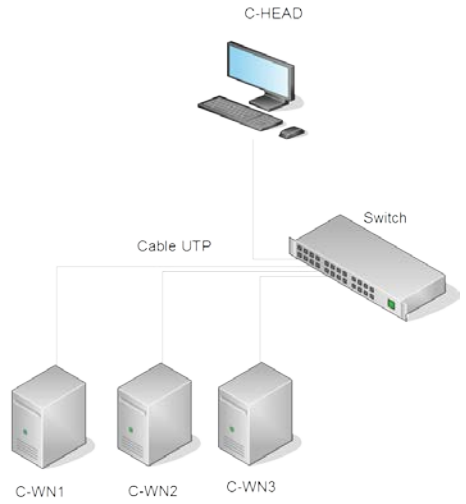

Figura 1. Diagrama de configuración del clúster computacional Uceva

2.3 HTCONDOR and its universes

The HTCONDOR is an Illinois University´s framework that is in charge of the administration of the high performance cluster with version 7.4.2 of the HTCondor. The basic operation of HTCondor consist of sending a work, it queues it, then it executes and finally it notifies the results.

It is described in more detail below

- Condor will usually be used to run tasks that will take a long time to obtain their results it is also to launch a program repeatedly.
- A shipping description file must be written with the Condor Submit command. Eventually, HTCondor looks for new jobs, and when it find them, HTCondor adjusts them according to the available resources.
- When there is an available machine, the work is sent for its execution .It may happen that in the best case it finishes correctly and the results are sent where the user specified it. In the event that the machine executing the work is occupied externally by a user, the state is saved if the "standard" universe is used through a "checkpoint".
- The right universe must be chosen for its execution.
- HTCondor will not always use all the computer equipment, it will only use them when the mouse or the keyboard are not used.

3. Calculation study of computational cluster performance

The theoretical performance of the computational cluster is obtained by adding the capacity in FLOPS (floating point operations per second) of each CPU of the worker nodes; it is given by the manufacturer of the processors for a total of 115.2 GFLOPS.In this study we are interested in finding to what extent is the performance of the cluster compared to a personal computer in a simple calculation task.

3.1 Structure

How can we measure the time difference between the computer cluster and a personal computer?

Is the result of the execution time of a simple algorithm in the computational cluster considerably smaller than that of a personal computer?

These questions were asked with the goal of obtaining visible and easy to show results to the academic community of the faculty.

3.2 Experimental Design

In this section we will explain in detail the performance time study of a cluster that has been physically configured with a head node and 3 worker nodes with core 2 duo quad processors which have a Scientific Linux operating system and a management software called HTCONDOR.It allowed to program an algorithm in C language, in order to obtain the prime numbers in a range from 1 to 15 million. This way, it is be able to launch the process from the head node to the worker nodes divided by groups of 5 million and take the measurement of time. In turn, the same algorithm was executed on a personal computer with common characteristics and it was taken at the time it took to complete the test. Finally, times were compared and plotted for a better visibility of the results.

3.3 Test configuration

In order to measure the cluster performance, the algorithm that calculates the prime numbers written in C language is going to carry out and that is found in a range of values defined by the user in the input parameters inserted in a job. Comparing it with the result of done by the same work on a personal computer with average characteristics.
The application is going to take the initial and final ranges of the calculation of the prime numbers that we want to find by taking the strategy of dividing the problem into multiple subproblems of equal complexity to be distributed in the cluster giving as a result, a greater speed in the processing of the data since they are the nodes responsible for processing and send the response to the main node.

The code of the application can be showed through the Nano NPrimos.c command:

```
#include <stdio.h>
#include <stdlib.h>
#define true  1
#define false 0

/**
 * Verifica si el número especificado es primo o no.
 *
 * @param    int    Número a verificar.
 * @return   int    true es primo, false no.
 */

int isPrime(int num)
{
    int i      = 0;
    int start  = 2;
    int end    = num;

    if(num == 2 || num == 3 || num == 5 || num == 7
|| num == 11)
    {
        return true;
    }

    if(num % 2 == 0)
    {
        return false;
    }

    for(i=start; i<=end; ++i)
    {
        if(i != num && num % i == 0)
        {
                    return false;
        }
    }
    return true;
}

int main(int argc, char *argv[])
{
    int start, end, i, val;
    if(argc != 3)
    {
        printf("Usage: %s <start> <end>\n", argv[0]);
        exit(1);
    }

    start = atoi(argv[1]);
    end   = atoi(argv[2]);

    for(i=start; i<=end; i++)
    {
      val = isPrime(i);

        if(val == true)
            printf("%d\n", i);
    }

    exit(0);
}
```

Figura 2. Código en lenguaje C, para generación de números primos

The code was done in C language and implemented in HTCondor under the commands and functions that this tool brings, the tests were made in the universe of standard configured in the cluster. Through the following command:

Condor_compile gcc NPrimos.c –o Primos

For the example case, we are goint to look for the prime numbers from 1 to 15,000,000 and divide them into 3 sub-works with ranges of 5,000,000 items each to be distributed in the different nodes

```
Executable      = findPrimes
Universe        = standard
Log             = _FindPrimes.log

Arguments       = 1 5000000
Output          = _FindPrimes-$(Process).out
Error           = _FindPrimes-$(Process).err
Queue

Arguments       = 5000000  10000000
Output          = _FindPrimes-$(Process).out
Error           = _FindPrimes-$(Process).err
Queue

Arguments       = 10000000  15000000
Output          = _FindPrimes-$(Process).out
Error           = _FindPrimes-$(Process).err
Queue
```

Figure 3.Prime submit

Once we have the file you must upload it to the cluster for processing, it is done with the following command: Condor_submit Primos.Submit
In order to check that the works have been successfully uploaded, we can see them through Condor_q. Once the jobs in the cluster have been processed, we will have several extension files .error, .out and. Log. As we can observe in the file Primos.Submit in these files will remain the errors or the answers of the nodes and we obtain the results of the prime numbers by the rank that was given to each one in the file Primos.Submit.

```
ID     OWNER          SUBMITTED     RUN_TIME ST PRI SIZE CMD
 13.0  condor         11/23 11:08   0+00:22:09 R  0   4.4  findPrimes 1 5000000
 13.1  condor         11/23 11:08   0+00:22:09 R  0   4.4  findPrimes  10000000
 13.2  condor         11/23 11:08   0+00:22:09 R  0   4.4  findPrimes  15000000
```

Figure 4. The condition work

In the previous image it is observed that each node took 22 minutes 09 seconds to process the assigned range of values, to obtain a total of 15 million processed numbers. Below,it is show the time that it took to run the same task on a personal computer that has a third-generation Intel CORE i5 processor and has 6 Gb of RAM running under the Windows 8 operating system. Which features can be consulted in the next web page http://www.asus.com/Notebooks_Ultrabooks/K55VD/specifications/.

Amount of processed data 1499999
(BUILD SUCCESSFUL (total time: 386 minutes 7 seconds)

Figure 5. Personal computer duration

In Figure 5 we can be observe that the time taken by the personal computer with average characteristics was 386 minutes 7 seconds.

Comparing the obtained results, we can be observe that a better time was achieved with the cluster that only took 22 minutes in consideration with the 386 minutes of the personal computer, which correspond to a performance greater than 18 times over the personal computer. This is due to the fact that each node of the cluster worked simultaneously and it is possible to process more data in a shorter time.

The comparative graphs are shown below:

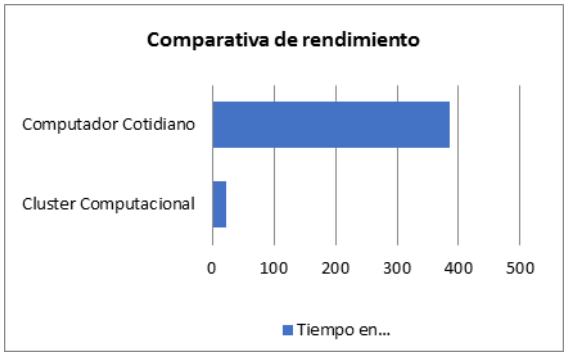

Figure 6. Comparative time chart

4. Conclusions

4.1 As a result of the study, an application was created whose objective is to use the calculation potential of the Computational Cluster, and execute it under one of the environments that allows HTCondor (standard.) The application also verifies its operation, and it shows that this tool can be used in an area of knowledge for educational purposes. This tool is able to demonstrate that the performance of the cluster is better in comparison to a daily use equipment with good characteristics.

4.2 The test consisted in the calculation of the prime numbers in a range of 1 to 15,000,000, they were selected and stored in a file for analysis in a computer of normal characteristics, this test lasted approximately 6:30 hours, in the same programming language and the pc only dedicated to this work. In the cluster, this same test lasted 22 minutes, it gave result in a time optimization of 18 times due to the processing power provided by this tool;it demonstrated the power computing that provides the union of several work teams performing a distributed task.

References

• S. T. Shah, R. J. W. E. Lahaye, S. A. A. Kazmi, M. Y. Chung, Syed Faraz Hasan. HTCondor System for Running ExtensiveSimulations Related to D2D Communication. 2014.

- Brennan John, Holmes Violeta, Bonner Stephen, Kureshi Ibad. PBStoHTCondor System for Campus Grids. 2015

- Joya A. Deri, Franz Franchetti, and Jose M.F. Moura. BIG DATA COMPUTATION OF TAXI MOVEMENT IN NEW YORK CITY. 2016

- Eric Anderson, Jeff Linderoth. high throughput computing for massive scenarioanalysis and optimization to minimize cascading blackout risk. 2017

- Algoritmos paralelos. Departament d'Informàtica. Escola Tècnica Superior d'Enginyeria. http://informatica.uv.es/iiguia/ALP/• ANDREW S. STEEN Tanenbaum, Maarten Van. Sistemas Distribuidos Principios y Paradigmas. [aut. libro]. 2 ed. Pearson, 2008.

- Cluster:Nociones Generales. http://ibiblio.org/pub/linux/docs/LuCaS/Manuales-LuCAS/doc-manual-openMosix-1.0/doc-manual-openMosix_html-1.0/node16_ct.html

- Comunidad EcuREd. EcuRed. Biblioteca Cubana.Cuba. http://www.ecured.cu/index.php/EcuRed:Enciclopedia_cubana.

- Family, OLCF. OLCF. Oak Ridge Leadership Computing Facility. Cray Inc, 2012. https://www.olcf.ornl.gov/computing-resources/titan-cray-xk7/

- GIMÉNEZ, Domingo. BORATTO, Murilo. COELHO, Leandro. Programación Paralela y Distribuid. [aut. libro] Universidad de Murcia Departamento de Informática y Sistemas. 2009, 1.

- GORDILLO Ruíz, José Luis. GUTIÉRREZ Ramírez, Gustavo A. Internet Cómputo y Telecomunicaciones. ENTER@TE en línea. Disponible http://www.enterate.unam.mx/index.html.

- HOEGER, Herbert. Aceleracion y Eficiencia. [ed.] Universidad de Los Ande. INTRODUCCION A LA COMPUTACION PARALELA. Mérida : Centro Nacional de Cálculo Científico, 2011.

- HTCondor. High Throughput Computing. 2012. http://research.cs.wisc.edu/htcondor/license.html.

- Htcondor. Htcondor, License Information.Htcondor. Http://Research.Cs.Wisc.Edu/Htcondor/License.Html. Consultado: 2014.

- Intel Core2 Quad Processor Q6000 series. http://www.intel.com/support/processor/sb/CS-032818.htm. Consultado 2014