IJCSC International Journal of Computer Science & Communication Vol. 1, No. 2, July-December 2010, pp. 151-155

Design, Implementation and Evaluation of Grid Environment for DV to MPEG4 Video Conversion

Jagpreet Sidhu[1] & Sarbjeet Singh[2]

[1]Information Technology, UIET, Panjab University, Chandigarh, India
[2]Computer Science and Engineering, UIET, Panjab University, Chandigarh, India
Email: jagpreetsidhu@gmail.com[1], sarbjeet@pu.ac.in[2]

ABSTRACT

Grid computing deals with large scale resource sharing and can be exploited to carry out storage and compute intensive tasks which are otherwise not practical to be executed on a single system. Video encoding is a lengthy, CPU intensive task, involving the conversion of video media from one format to another. Video files can be easily broken down into smaller work-units. This factor makes the distribution and execution of video encoding processes viable on grid. In this paper we have made an attempt to design, implement and evaluate a grid environment for DV to mpeg4 video conversion. A grid environment consisting of 10 nodes has been implemented using GT4. Over this environment, a video conversion process (DV to MPEG4) has been implemented and performance of the grid for this conversion process has been evaluated. The performance evaluation has been done with respect to time and it has been observed that grid provide performance benefit with respect to time, but, in small environments, the effect of increasing the number of nodes does not provide as much speed advantage as is expected theoretically.

Keywords: Grid Computing, Video Conversion, Video Encoding, DV, MPEG-4.

1. INTRODUCTION

Grid Computing focuses on large-scale resource sharing in a flexible, secure, and coordinated way [1]. This dynamic, coordinated and secure resource sharing enables the development of innovative applications [2, 3]. Grid computing uses resources of many separate computers connected by a network to solve large-scale computation problems. Grid computing oûers a model for solving massive computational problems by making use of unused resources (CPU cycles and/or disk storage) of large numbers of disparate computer systems, treated as a virtual cluster, embedded in a distributed infrastructure [2].

Grid computing is diverse and heterogeneous in nature, spanning multiple domains whose resources are not owned or managed by a single administrator. This presents grid resource management with many challenges such as site autonomy, heterogeneous substrate and policy extensibility. The Globus [4, 5] middleware toolkit addresses these issues by providing services to assist users in the utilization of grid resources. Users are still exposed to the complexities of grid middleware, however, and there is a substantial burden imposed on them in that they must have extensive knowledge of the various grid middleware components in order to be able to utilize grid resources. Such knowledge ranges from querying information providers, selecting suitable resources for the user's job, forming the appropriate JSDL (Job Submission Description Language), submitting the user's job to the resources and initiating job execution.

Grids primarily oûer a way to solve grand challenge problems like protein folding, ûnancial modeling, earthquake simulation, climate/weather modeling etc. Grids also enable the optimum use of information technology resources inside an organization [4]. It also provides a mean for oûering information technology resources as a utility to clients who pay only for what they use. In short, it involves virtualizing computing resources.

The SETI@home project [6, 7, 8], launched in 1999, is a widely and well-known example of a very simple grid computing project. This project works by running as a screensaver on users' personal computers, which process small pieces of the overall data when the computer is either completely idle or lightly used [8]. Other examples of popular grid computing projects are POEM@Home [9, 10], Climateprediction.net [11, 12], SZTAKI Desktop Grid [13, 14, 15], MilkyWay@Home [16, 17], ZetaGrid [18], Grid.org [19, 20] etc.

2. PROBLEM UNDERTAKEN

The computing environments have evolved from single-user environments to Massively Parallel Processors (MPPs), clusters of workstations, general distributed computing systems and now to grid computing systems. Every transition has been a revolution, allowing scientists and engineers to solve complex problems and sophisticated applications previously incapable of solving. However every transition has brought new challenges and problems in its wake, as well as the need for technical innovation. The evolution of computing systems has led to the current situation in which millions of machines are interconnected via the Internet with various hardware and software configurations, capabilities, connection topologies, access policies and so forth [21]. The formidable mix of hardware and software resources on the Internet has fuelled researchers' interest in investigating novel ways to exploit this abundant pool of resources in an economical and efficient manner, as well as in aggregating these distributed resources so as to benefit a single application [21].

It is clear from the literature that grids are extensively being used to carry out compute and storage intensive tasks, so there is a need to evaluate the performance of grids with respect to different parameters like processing time, resource utilization, memory statistics, network statistics etc., so that user of the grid can make easy decisions regarding the configuration of the environment they have to build or use for the execution of their grid jobs. This Research focuses on evaluating the performance of grid (with respect to time) by executing a compute intensive, storage intensive and network bandwidth intensive job. There exists a variety of compute intensive problems that can be executed and solved using grid computing [22]. E.g. drug discovery, economic forecasting, seismic analysis, and back-office data processing in support for e-commerce and web services, video encoding and streaming, image processing etc.

The exact performance benefit depends upon the problem at hand, configuration of the environment, resources available in the environment, and a lot of other factors. So it is necessary and desirable that a grid should be evaluated against different parameters. The evaluation results with respect to different parameters can be of much importance to various researchers performing similar types of experiments in the related area and can serve as a basis for making many critical decisions. In this paper we have made an attempt to evaluate performance of grid with respect to time for DV to MPEG4 video conversion process. The process is compute and storage intensive. It is network bandwidth intensive also if it is to be executed over a distributed system. The DV video format files are generally very large in size compared to other video formats and require a huge amount of storage. So it is necessary and desirable to compress and convert these videos to some other formats. But this compression and conversion process can be very compute intensive depending upon the size of video and algorithm to be used. Grids can be exploited for the execution of such jobs as a large DV video file can be easily split into different fragments and these fragments can be distributed to different grid nodes for individual processing. The approach is workable but this makes the entire process compute intensive, storage intensive and network bandwidth intensive also. In the present research work, we have evaluated a grid environment for this conversion process.

Rest of the paper is organized as follows: Next section explains the objectives of the research conducted. Section four explains the proposed solution. Section five presents the configuration of the environment. Section six discusses results obtained and section seven talks about conclusion and future scope.

3. OBJECTIVES

The prime objective of the research work which has been carried out is to evaluate the performance of grid (with respect to time) by executing a video conversion (DV to MPEG4) process.

The secondary main objectives intend to study grid computing, its types, its relationship with other computing technologies and to study open source middleware available for its implementation etc. It also involves the analysis, design and implementation of a grid environment. The secondary objectives also intend to study compute, storage and network intensive problem of DV to MPEG4 video conversion and to implement it on the grid environment.

4. PROPOSED SOLUTION

The basic idea of the proposed video conversion process on a single system and a grid computing system has been represented in Figure 4.1 and Figure 4.2. The process consists of following phases:

Initialization/Setup Phase: In this stage the client machine submits media file to media splitter where it is split into different slices and then distributed among different nodes on the grid. The splitting operation performed by media splitter is controlled through media content decoder script which in turn is updated according to the configuration and resources available in the environment.

Client Request: The client sees the grid to schedule the job. The client requests grid for its currents CPU utilization status. Based upon the current status of grid, the client schedules the splitted videos (i.e. different slices) in the environment.

Client Scheduling: The client proxy forwards the request to the remote scheduler who maintains metadata of different nodes participating in the grid.

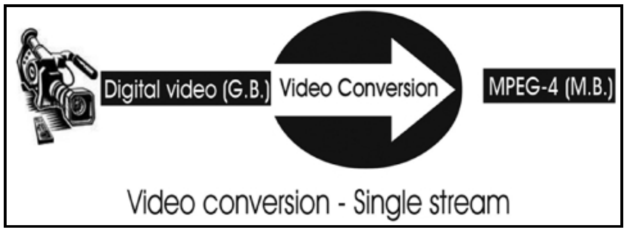

Fig. 4.1: Video Compression by Single Stream

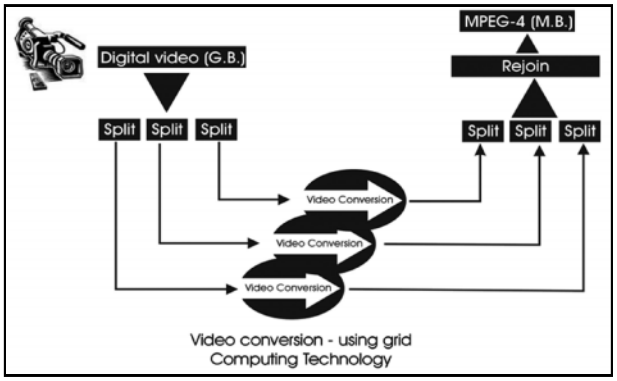

Fig. 4.2: Video Compression in Grid Computing System

The client scheduler submits the splitted videos to remote schedulers of each of the nodes by sending them a stream of the sliced videos and a script to execute the job of conversion effectively.

Remote Scheduling: Each of the remote grid nodes starts scheduling the job. Remote scheduler on nodes makes sure that the job is handled by the remote schedulers as its own job for conversion. Remote scheduler collects the converted streams and redirects them to the client scheduler from where the job was received earlier.

Scheduling Response and Rejoining: In this phase the client scheduler starts counting the responses from the remote schedulers after the conversion process is over. The client scheduler checks whether all the slices came back after conversion or not. If not, the client scheduler requests the remote schedulers to complete the conversion process at top priority and send the completed job back. When all the spitted jobs came back to client after conversion, the rejoining phase starts. During this phase, the client machine starts rejoining the splits, and forms a converted video of full length.

5. CONFIGURATION OF THE ENVIRONMENT

The Grid Computing Environment [23] designed and implemented for the present research work includes 12 Linux PCs which are connected in two different labs. The blueprint of the environment used is shown in Figure 5.1. Following are the details of the nodes:

Node A: It acts as a client and has been used to submit the video conversion job to other nodes on the grid. The software installed on node A includes Fedora Core 4, Globus Toolkit 4, Java Development Kit, Certificate Authority Client and other related software.

Node B: It implements CA (Certificate Authority) which generates certificates for different grid users and hosts. The X.509 certificates generated by this CA have been used to authenticate grid users and hosts on the grid. It also implements NTP (Network Time Protocol) and acts as a time server to synchronize all grid nodes in the environment to a uniform time. The software installed on Node B are Fedora Core 4, Globus Toolkit 4, Java Development Kit, Certificate Authority, Network Time Protocol, PBS and other related software.

Node C to **Node L** are grid processing nodes. The software installed on these nodes are Fedora Core 4, Globus Toolkit 4, Java Development Kit, Certificate Authority Client and other related software.

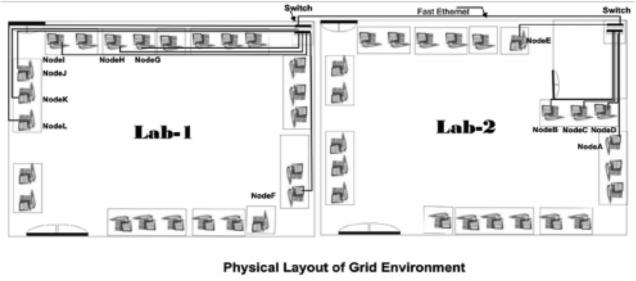

Fig. 5.1: Physical Layout of Grid Environment

As shown in Figure 5.1, Lab-1 and Lab-2 are located at different locations and are connected through fast Ethernet connection. Lab-1 has 5 grid nodes, all having single Intel Pentium 4 (2.0 GHz to 3.0 GHz) processor, 256MB DDRAM, and 3Com 3c9051 and Intel 82566DM-2 Ethernet Interfaces. Lab-2 has 7 grid nodes, all having dual Intel Pentium 4 2.0 GHz processor, 256MB SDRAM, and 3Com 3c9051 Ethernet Interfaces. The details have been diagrammatically represented in Figure 5.1.

6. RESULTS

The results of evaluations have been divided into 3 parts according to the length of input video. Following are the three cases that have been taken for evaluation.

1. Evaluation of grid with 6 minute video job on one to ten nodes.
2. Evaluation of grid with 8 minute video job on one to ten nodes.
3. Evaluation of grid with 10 minute video job on one to ten nodes.

For evaluation we took three digital videos in DV format having length of 6 minutes, 8 minutes and 10 minutes. Three grid jobs have been prepared to convert these videos to mpeg4 format and these jobs have been executed on different number of grid nodes. The nodes have been added from 1 to 10, one by one in each of the above cases and time taken to complete the job has been noted in each case. The graphs below show the test results of job execution on different number of grid nodes.

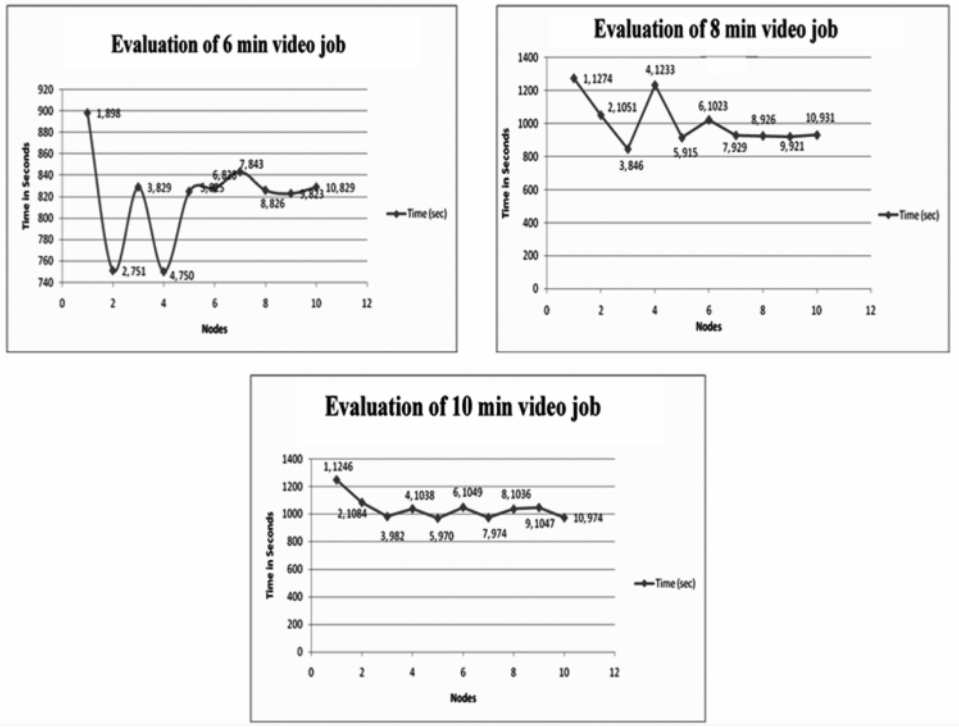

Fig. 6.1: Evaluation of Video Jobs on Different Number of Grid Nodes

It is clear from the results obtained in Figure 6.1 that grid shows benefits over single processing systems but it does not show linear increase in performance by increasing the number of grid processing nodes.

7. CONCLUSION AND FUTURE SCOPE

After conducting different runs for three experiments by taking videos of different lengths on one to ten processing nodes, it is clear that grid shows benefits over single/ centralized systems but it does not show up linear increase in performance by increasing the number of grid processing nodes. Theoretically the experiment was expected to give linear increase in performance by increasing the grid processing nodes but this unexpected behavior is due to following reasons:

- The job chosen was network bandwidth intensive but the network used for current implementation was slow. As network bandwidth plays a considerable role in grid performance, a faster network (Gigabit) is proposed to be used in high performance grid systems.
- The default scheduling strategy used in implementation is random but different and better results are expected if other scheduling strategies (e.g. priority based) are used.
- The performance also depends upon node's local processing status and availability. The nodes used during evaluation were not perfectly idle. So different and better results are expected if the grid nodes to be used are completely idle. Moreover, for better performance the nodes in grid should have high availability of its resources also.

In future, we are planning to conduct the same experiments in fast gigabit network for better performance. We are also planning to change the default scheduling strategy used in current implementation to priority based. Then results of performance evaluation of different scheduling strategies will be examined.

As extensive research is going on better scheduling and better bandwidth networks for grid systems, with the advancements in these fields and technologies, grid systems can one day be proved equivalent to cluster computing systems also.

REFERENCES

[1] J. Joseph, M. Ernest, C. Fellenstein, "Evolution of Grid Computing Architecture and Grid Adoption Models", *IBM Systems Journal*, **43 (4)**, pp. 624-645, 01 Dec. 2004.

[2] Francisco Brasileiro, Patricia Machado, Walfredo Cirne, Alexandre Duarte, "GridUnit: Software Testing on the Grid," *28th International Conference on Software Engineering (ICSE'06)*, pp.779-782, 2006.

[3] I. Foster and C. Kesselman, "Future Generation Computer System", *The Globus Project: a Status Report*, **15**, pp. 607-621, 1999.

[4] C.T Yang, P.C Shih, C.F Lin, S.Y Chen, "A Resource Broker with an Efficient Network Information Model on Grid Environments", *The Journal of Supercomputing*, 2007 Springer, **40 (3)**, pp. 249-267, June 2007.

[5] Sharma Rohit, Chana Inderveer, "Design and Development of Resource Repository for Grid Environment", M.E. Thesis Thapar University Punjab India, 29-Sep-2008.

[6] http://setiathome.berkeley.edu/

[7] SETI@home, Available at http://en.wikipedia.org/wiki/SETI@home

[8] http://setiathome.ssl.berkeley.edu/

[9] POEM@Home, Available at http://boinc.fzk.de/poem/

[10] http://en.wikipedia.org/wiki/POEM@Home

[11] http://climateprediction.net/

[12] http://en.wikipedia.org/wiki/Climateprediction.net

[13] SZTAKI Desktop Grid, Available at http://www.desktopgrid.hu/

[14] http://szdg.lpds.sztaki.hu/szdg/

[15] http://en.wikipedia.org/wiki/SZTAKI_Desktop_Grid

[16] MilkyWay@Home, Available at http://milkyway.cs.rpi.edu/milkyway/

[17] http://milkyway.cs.rpi.edu/milkyway_gpu/

[18] ZetaGrid, Available at http://www.zetagrid.net/

[19] Grid.org. Available at http://en.wikipedia.org/wiki/Grid.org

[20] http://www.grid.org/

[21] Alfawair Mai, "A Framework for Evolving Grid Computing Systems", PhD Thesis De Montfort University U.K. England, Available at https://www.dora.dmu.ac.uk/bitstream/handle/2086/3423/Mai%20Thesis.pdf?sequence=1

[22] Grid Computing, Available at http://en.wikipedia.org/wiki/Grid_computing.

[23] Globus Project Home, Available at http://www.globus.org/