2009 IEEE International Advance Computing Conference (IACC 2009)
Patiala, India, 6–7 March 2009

Performance Analysis of Load Balancing Algorithms for cluster of Video on Demand Servers

Pushpendra Kumar Chandra
Department of CSE
National Institute of Technology Rourkela
Orissa, INDIA -769008
+91-9861286202
pushpendrachandra@gmail.com

Bibhudatta Sahoo
Department of CSE
National Institute of Technology Rourkela
Orissa, INDIA-769008
+91-9861286202
bdsahu@nitrkl.ac.in

Abstract— **In this paper we have proposed an algorithm for a wide variety of workload conditions including I/O intensive and memory intensive loads. However , in our task the CPU requirements of the system is minimum as the tasks which come are mostly video fetch tasks which require negligible system interaction but a lot of I/O consumption. The goal of the proposed algorithm is to balance the requests across the entire cluster of servers basing on its memory, CPU and I/O requirements so that the response time and the completion time for each job is minimum. Here preemptive migrations of tasks are not taken into consideration. A typical transaction in our model can be defined as the duration between the acceptance of task into the system and fulfillment of its requirements by the system. The requirements of the task are video files which the system has to load from a secondary storage device and stream the video continuously to the end user who initiated the request. We have compared our algorithm (IOCMLB) to two other allocation policies and trace driven simulation shows that our algorithm performed better than other two policies.**

Keywords-Video on demand, I/O-intensive task, Load balancing

I. INTRODUCTION

Video On Demand [1] commonly called VOD service has been feasible due to availability of a number of enabling technologies such as MPEG (Moving Picture Expert Group), ATM (Asynchronous Transfer Mode) and ADSL (Asymmetric Digital subscribers Line). A key challenge involved is in providing the service is requirement of storing a large amount of video objects such that each video stream can be accessed and transmitted to the client in real time.

Multimedia databases have become more important since the demand for multimedia information (such as text, audio, image and video) has increased. Currently content-based retrieval of multimedia data is being actively researched. However, content-based retrieval of multimedia data encounters three major difficulties. First, the content is subjective; this needs a powerful set of search facilities including keywords, sounds, color, texture, spatial information and motion. Second, if a method or processing technique is designed and developed for one type of data or feature, it's usually not appropriate for others. For instance, a technique designed for indexing audio data may not be usable for image data or, a technique developed for a color feature may not be useful for a texture feature in image and video data .Third, the usual huge size of multimedia data requires an exhaustive search.

VOD systems can be classified to two types either single server or multi server architecture. But the multi server architecture proves to the better as we see that the multi server architecture caters to the client's requests more efficiently whereas the single server is not able to handle the video traffic.

The VOD system can thus be described as a cluster of homogeneous or heterogeneous computing nodes. In the past, some problems have been addressed in designing a VOD system, such as data placement, resource management, disk scheduling, admission control, synchronization, and fault tolerance, etc. However the problem about dynamic load balancing among the servers was seldom explored. Although the previous researches explored the load balancing problem, most are applied to general tasks, not to video tasks. Furthermore although the data placement strategies mentioned above could be used to achieve load balancing among the servers, they are static and not good enough. One of the data placement strategies is to strip each video object across all the disks/servers on the system and then to avoid the load unbalance. However, the approach suffers from the following drawbacks. First, it results in additional complexity such as some form of synchronization in delivering a single video object from multiple disks/servers. Second, it is not practical to assume that a system must be constructed using homogeneous disks/servers. Third, as client demands and/or data sizes grow, the system requires one or more disks/servers, thereby resulting in restriping of all video objects. Another data placement strategy is to replicate popular video objects among the servers. It also has some problems such as requiring extra storage spaces and deciding the appropriate time to perform de-replication, etc. Besides some researches studied the load balancing in a distributed VOD system. However their models are distributed VOD servers, completely different from ours clusters of VOD servers.

The rest of the paper is organized as follows. In the section 2 that follows, system model and methodology. In section 3, we describe the dynamic load balancing algorithm for I/O intensive tasks. Section 4 shows the simulation and experimental results. Finally section 5 concludes the paper by summarizing the main contribution of this paper.

II. SYATEM MODEL AND METHODOLOGY

The model of VOD system is illustrated in Fig 1. All servers in the system are connected with a high-speed network, such as an ATM switch, a fast Ethernet, or a cross bar switch, etc.

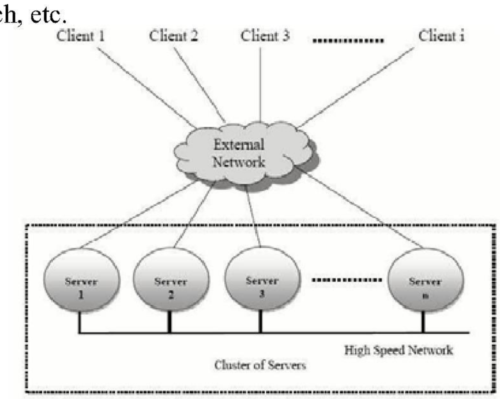

Fig.1. Model of VOD System

Each server has its CPU, memory, and I/O sub-system, but they would not necessarily have the same capabilities. This implies that the system could achieve good scalability. When a client issues a request for a video object, a specified server called the dispatcher will filter the request and transfer the request to the destination server with the requested video object. Then the destination server delivers the video stream to the requested client at a given rate through the external network until the display of the video stream is finished. VOD system is disk-bounded. For networks, the network bandwidth from servers to clients is fast enough to deliver video streams and it is also fault free.

The VOD clusters can be modeled by the mathematical queuing theory showed in Fig 2.

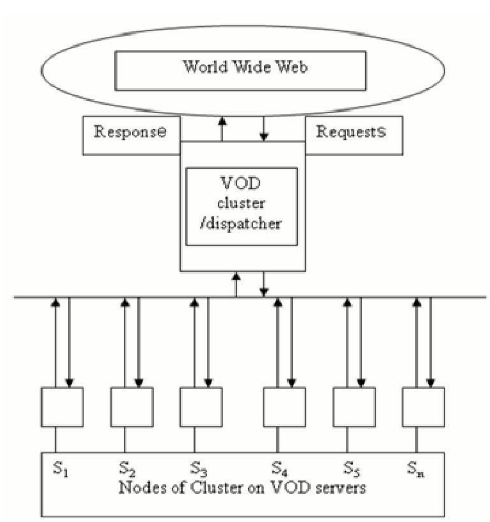

Fig. 2. Queuing Model

Here, VOD cluster is connected to the web by the interconnection network which provides two way traffic for the requests to the cluster and the response from the clusters. We assume a constant arrival rate that is Poisson's distribution. The nodes are heterogeneous in nature, so they have different service rates denoted by $\{S_1, S_2, S_3 \ldots S_n\}$.The requests arrive from the user via the web to the VOD dispatcher. Here the Dispatcher allocated requests to the various nodes which generate the response back to the user via the web.

III. DYNAMIC LOAD DISTRIBUTION ALGORITHM

We proposed an algorithm (IOCMLB) for a wide variety of workload conditions including I/O intensive and memory intensive loads. However , in our task the CPU requirements of the system is minimum as the tasks which come are mostly video fetch tasks which require negligible system interaction but a lot of I/O consumption. The goal of the proposed algorithm is to balance the requests across the entire cluster of servers basing on its memory, CPU and I/O requirements so that the response time and the completion time for each job is minimum. Here preemptive migrations of tasks are not taken into consideration.

A typical transaction in our model can be defined as the duration between the acceptance of task into the system and fulfillment of its requirements by the system. The requirements of the task are video files which the system has to load from a secondary storage device and stream the video continuously to the end user who initiated the request.

Algorithm: Load balancing algorithm (IOCMLB)
Whenever a request is made to the cluster:

1. For each task i do
2. Compute the value of its I/O, CPU and memory requirements.
3. if I/O requirement(i) = max [CPU requirement, memory requirement, I/O requirement] then
4. Choose the set of nodes which has the highest unused I/O capability and is meeting the memory and CPU requirements.
5. task is assigned to that node which has the lowest response time for that particular task.
6. else if memory requirement(i) = max [CPU requirement ,memory requirement ,I/O requirement] then
7. choose the set of nodes which has the highest unused memory capability and is meeting the I/O and CPU requirements
8. task is assigned to that node which has the lowest response time for that particular task.

9. else if CPU requirement(i) = max [CPU requirement, memory requirement, I/O requirement] then
10. choose the set of nodes which has the highest unused CPU capability and is meeting the I/O and memory requirements
11. task is assigned to that node which has the lowest response time for that particular task.
12. update load status for that node.
13. end for

This is the proposed algorithm that we used to schedule tasks across n heterogeneous servers. We introduce the following three load indices with respect to I/O, CPU, memory resources. CPU load of a node is characterized by the length of CPU waiting queue to identify whether node i is CPU overloaded or not. Memory load of a node is the sum of the memory space allocated to all the task running on that node.
I/O load measures two types of I/O accesses, i.e. Implicit I/O request includes by page fault; Explicit I/O request is used from tasks.

Now we describe the load balancing algorithm of which the pseudo code is shown above. Given a set of independent tasks submitted to the load manager. Our algorithm make an effort to balance the load of the cluster resource's by allocating each task to a node such that the expected response time is minimized.

Steps 1 and 2 are responsible for the acceptance of the task into the system and the prediction of its CPU, I/O and memory resources. This is done at the central node known as the dispatcher which is already discussed.

The maximum of all the three requirements is found in step 3 which is also at the dispatcher. The result of this step delivers the control to *(a)* Step 4 –if the I/O requirement of the task is maximum *(b)* Step 7 - if the memory requirement of the task is maximum *(c)* Step 10 - if the CPU requirement of the task is maximum.

Further two more sub-conditions are to be satisfied for the final node to be chosen: *(a)* That the remaining two capacities must be enough for the task to finish at that node. *(b)* The response time for that task should be minimum at that node.

After all the above conditions are satisfied the dispatcher allocates the task to selected node and the task is run on that node.

IV. SIMULATION AND EXPERIMENTAL RESULTS

The VOD system is a group of nodes, where we select a main node whose role is described as follows: the main node also called as head node in the cluster is responsible for load balancing and monitoring available resources of the node. Head node processes all tasks in First Come First Serve manner. The computing nodes in the cluster solely depend on the information available with the head node for allocation decision.

Tasks that are to be executed in the cluster arrive at the head node. We assume that the arrivals rate is constant in nature. After being handled by the head node, the tasks are dispatched to one of the best suited nodes for execution. The nodes each have a local queue which executes tasks in parallel.

A. Performance of the System

The model that we have used for simulation has been validated by the following graph (see Fig. 3), which has been got by plotting the time taken to execute a given set of tasks by carrying the number of servers involved. We can see that the execution time for the set of tasks gradually decreases as we increase the number of servers. But it is observed that after a threshold value the execution time becomes constant which indicates a saturation value for the completion of tasks.

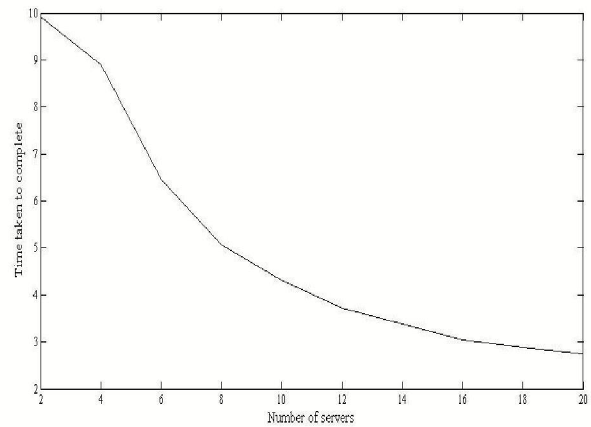

Fig.3. Performance of the model

From the Fig. 3 we can conclude that for a given task set, varying the number of servers from two to twenty we find that our system reaches the saturation point at about 18-19 servers i.e. the execution time doesn't improve any more. Hence we say that our model has been validated.

B. Performance Comparison of IOCMLB

For simulations we have taken two other policies for node allocation into consideration for comparison. They are First come first serve and Random. In this experiment we explore load balancing of servers under constant arrival rate of tasks. To facilitate this observation the standard deviation [1] of the load balancing of servers is shown for various numbers of tasks. The closer to zero, the standard deviation the better the load balancing. The standard deviation σ can be defined for the heterogeneous environments with n servers as follows:

$$\sigma^2 = \frac{1}{n}\left(\left(P_1 - \frac{c_1}{c_1 + c_2 + \cdots + c_n}\right)^2 + \left(P_2 - \frac{c_2}{c_1 + c_2 + \cdots + c_n}\right)^2 \dots + \left(P_n - \frac{c_n}{c_1 + c_2 + \cdots + c_n}\right)^2\right)$$

Where c_i is the capability value and P_i cumulative probability of server *i* in the proportion.

1. *FCFS Allocation:* In this policy the tasks are allocated to the nodes in order of their arrival. For example the task 1 is allocated to node 1, task 2 to node 2 and so on. We got the graph (see Fig. 4).

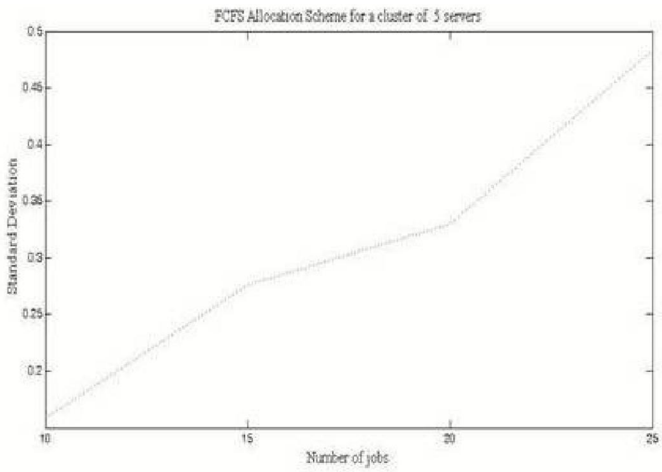

Fig. 4. FCFS policy performance graph

2. *Random Allocation:* In this policy, the tasks can be allocated to any node randomly. This does not use any logic but just allocates the task to any node randomly. We got the graph (see Fig. 5) for random allocation of tasks.

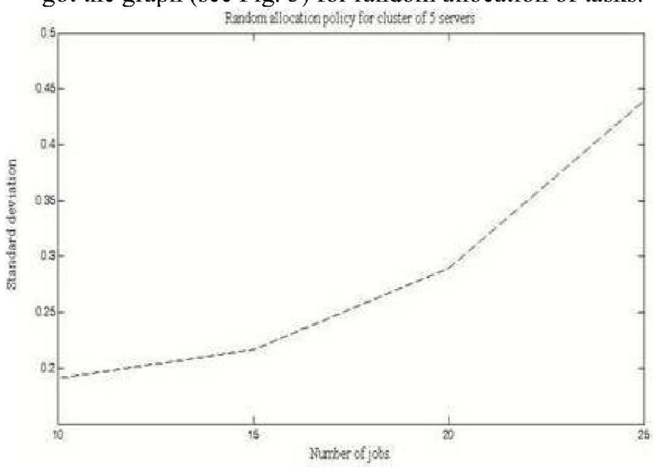

Fig. 5. Random policy performance graph

3. *IOCMLB Algorithm*: In this policy, the tasks were allocated as per the algorithm IOCMLB described earlier and we obtained the graph (see Fig. 6).

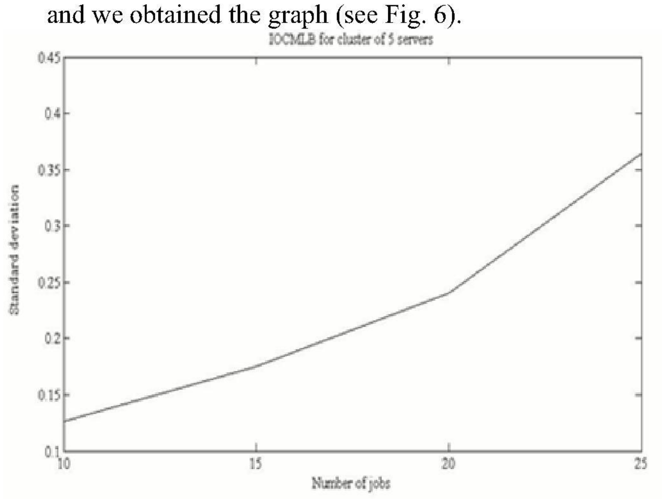

Fig. 6. IOCMLB algorithm performance graph

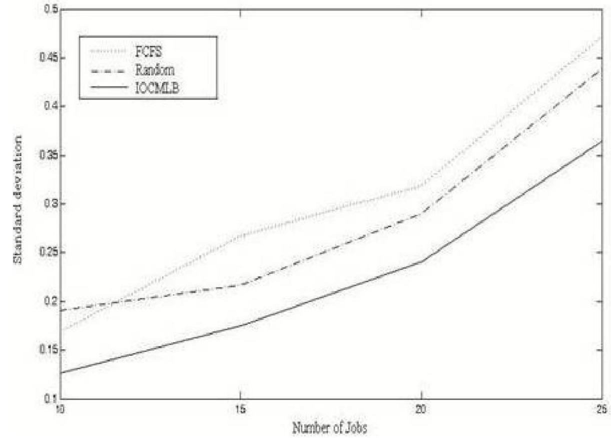

Fig. 7. Performance comparisons of three algorithms

The Fig.7 definitely shows that the nodes experience better load balancing using our algorithm for load balancing of VOD clusters. Further, as shown in all figures the load balancing of servers is graduating worse as the number of tasks getting admitted into the system keep on increasing. Therefore it is should be noted that the algorithm will have threshold limits as to the number of tasks that it can successfully balance keeping the number of servers constant.

V. Conclusion

In a VOD system with clustering servers, how to support more clients and reduce the average response time of requests is a critical topic. In this paper, we focused on dynamic load balancing among the VOD servers with constant arrival rate to reach these goals. Cluster computing has emerged as a result of the convergence of several trends, including the availability of inexpensive high performance microprocessors and high speed networks, the development of standard software tools for high performance distributed computing, and the increasing need of computing power for computational science and commercial applications. Even though there are number of different dynamic load balancing techniques for VOD cluster systems, their efficiency depends on topology of the communication network that connects nodes. This research has developed an efficient load-balancing algorithm for I/O intensive tasks that uses a new procedure for calculating the load at individual node. The proposed load balancing algorithm (IOCMLB) aim to achieve the effective usage of global disk resources in the VOD cluster. This can minimizes the average slow down of all parallel jobs running on the VOD cluster and reduces the average response time of the jobs. We have compared our policy to two other popular strategies namely FCFS and random, it is seen that we get better load balancing results using our algorithm.

Acknowledgments

This research was supported by R&D project grant 2005-2008 of MHRD Government of India with the title as "Fault Tolerant Real Time Dynamic Scheduling Algorithm For Heterogeneous Distributed System" and being carried out at

department of Computer Science and Engineering, NIT Rourkela.

REFERENCES

[1] Yin-Fu Huang ,Chih-Chiang Fang ,Load balancing for clusters of VOD servers ,Information Sciences 164 (2004) ,pp.113–138.

[2] Xiao Qin, Performance comparisons of load balancing algorithms for IO-intensive workloads on clusters, Journal of Network and computer applications(2006), doi:10.1016/j.jnca.2006.07.001.

[3] B. Berson, S. Ghandeharizadeh, R. Muntz, X. Ju, Staggered stripping in multimedia information systems, Proceedings of the ACM International Conference on Management of Data, 1994, pp. 79–90.

[4] D. N. Sujatha K. Girish ,An Integrated Quality-of-Service Model for Video-on-Demand Application , IAENG International Journal of Computer Science.

[5] Xiao Qin ,Dynamic Load Balancing for IO-Intensive Tasks on Heterogeneous Clusters, Proceeding of the 2003 International Conference on High Performance Computing(HiPCO3).

[6] Xiao Qin ,Hong Jiang ,Yifeng Zhu ,David R. Swanson ,A Dynamic Load Balancing Scheme for IO-Intensive Applications in Distributed Systems, Proceeding of 2003 international conference on Parallel processing Workshop(ICPP 2003 Workshop).

[7] Sanan Srakaew et al.,Content-based Multimedia Data Retrieval on Heterogeneous System Environment, George Blankenship,International Conference on Intelligent Systems (ICIS-99) , Denver, Colorado, June 24-26, 1999 ,pp 13-18.

[8] Paul Werstein ,Hailing Situ and Zhiyi Huang , Load balancing in cluster computer, Proceeding of the seventh international conference on Parallel and Distributed Computing, Applications and Technology (PDCAT'06).

[9] Xiao Qin, H.Jiang, Y.Zhu and D.swanson, Toward load balancing support for I/O intensive parallel jobs in a cluster of workstation, Poc. Of the 5th IEEE international conference cluster computing(cluster 2003) ,Hong Kong,Dec.1-4,2003.

[10] Kumar K. Goswami, Murthy Devarakonda and Ravishankar K. Iyer, Prediction–baesd dynamic load-sharing heuristics, IEEE transaction on parallel and distributed systems, VOL.4, No.6, june 1993.

[11] Mohammed Javeed Zaki, Wei Li, Srinivasan Parthasarathy, A Review of Customized Dynamic Load Balancing for a Network of Workstations.

[12] M. Kandaswamy, M.Kandemir, A.Choudhary, D.Benholdt, Performance implication of architectural and software techniques on I/O intensive application, Proceeding International conference parallel processing 1998

[13] Neeraj Nehra, R.B.Patel, V.K. Bhat ,A Framework for Distributed Dynamic Load Balancing in Heterogeneous Cluster,Journal of computer science 3(1):14-24-2007.

[14] Pushpendra Kumar Chandra, Bibhudatta Sahoo, " A Novel Dynamic Load Balancing Algorithm for I/O-intensive task in Heterogeneous Cluster", International Conference on Advance Computing, Feb. 21-22, 2008.

[15] Marc H. Willebeek-LeMair , Strategies for Dynamic Load Balancing on highly parallel computer IEEE Transactions on parallel and distributed systems Vol. 4,No. 9, September 1993.