Volume: 04 Issue: 06 Pages:1786-1792 (2013) ISSN : 0975-0290

ECLB: A Novel Exhaustive Criterion Based Load Balancing Algorithm for E-Learning Platform by Data Grid Technologies

S. Rajarajeswari
Department of Computer Science, Virudhunagar Hindu Nadars'
Senthikumara Nadar College (Autonomous), Virudhunagar - 626001.
Email: rajeshwarisethuraman@gmail.com

----------**ABSTRACT**----------
Virtual learning is a method to access and share study materials for students on web. As it is the easy way of accessing data, millions of student's widely using E-learning. Due to the increase of users day by day there is an issue in scheduling and providing service to plenty of users, thus load balancing among the nodes is a great deal faced by server. This makes many research people to carry out their work in load sharing and balancing between nodes. Though the existing methods provide solution for load sharing and balancing it requires high cost. In this paper, we mainly focus on allocating loads among nodes in a network with minimum cost. K-means algorithm is used to cluster the node. Clustering increases the performance of load balancing by forming tighter clusters. To share load, among inter-clustering and for intra clustering we use the proposed method, Exhaustive Criterion based Load Balancing (ECLB) algorithm. The experimental results show that the load is properly and effectively shared among nodes with minimum cost.

Keywords - **ECLB, Load-balancing, Low-cost, E-learning.**

Date of Submission: April 10, 2013 Date of Acceptance: May 01, 2013

I. INTRODUCTION

An absolute solution for today's competitive and fast moving upper crust educational institutions is E-learning. They can't afford to waste time or money in preparing notes and delivering it to students personally. It requires a solution to use online study materials. E-learning faces a situation to meet the needs of educational institutions. E-learning (electronic learning) is a term that covers a huge set of applications and processes, such as virtual classrooms, Web-based learning, and digital collaboration. It delivers the content via the Internet, intranet, audio and videotape, satellite broadcast and more. Due to the sophistication of using internet, many pupils get attracted by E-learning. E-learning enables a student to effectively find and share study resources worldwide. The count of users using virtual learning is increasing day to day. Obviously, as the number of persons using network upturns then there will be a splendid management problem in server. Sever, struggles to lineup the request given by network users at the same time. It requires much more time to perform all tasks. Since, sever is constructed to response limited number of user. On exceeding the limit server jams to response the user request due to over load. Also all users have to wait for a long time to get service from such a server.

To blend, to above said server disputes, many research works are taken to free server and to handle millions of request from user without changing or upgrading the server. Many researchers found solutions to overcome the drawback of server. Maximum researchers stick on to a solution to uniformly distribute workload among all nodes in the network. Load can be distributed across the network depending on some attributes. There are lots of load balancing algorithm exist. Many load distribution algorithm uses the process capability of nodes as attribute to distribute the loads to many nodes in a network. The process of distributing loads to all nodes in a network is said to be load-balancing. It distributes processing and communication activities evenly across a computer network so that no single device is overwhelmed. As in E-learning method it is more complicated to find number of request to server given by the users at a particular time, in such situations load balancing serves better than other methodology in order to avoid server downs. Thus load balancing method will assist server to provide a quicker responses even when number of users upsurges and helps server to response the users quickly. Scheduling and resource management are closely related topics of load balancing. A raw load distribution will distribute the load only to idle resource, it won't consider the under loaded resources. This way of distributing load to nodes is named as load sharing. It is the unrefined form of load-balancing. Load balancing is the finest of load distribution. Third form is load leveling used to describe the middle ground between load sharing and load balancing. Load leveling avoids congestion on any resource. Many researches are undertaken in load balancing for server management. To carry out the process of load balancing server should check all the nodes in network and have to find the nodes that are under loaded to carry out the given request. Once under loaded node is found then it makes a load distribution in such a way that it will get loads from the

Volume: 04 Issue: 06 Pages:1786-1792 (2013) ISSN : 0975-0290

node which is overloaded and gives some of its task to other under loaded or idle nodes in the network. This load balancing process involves transferring of datum from one system to other. Large amount of communication cost will be required to transfer data from one node to another for processing. Also a network may have nodes with different configuration; a best solution should be given by considering all the difficulties in the network. All the existing work provides solution to load balancing, were as earlier methods back on excessive cost in terms of transferring data between nodes, memory requirement is also high, also requires much time in order to complete the task. In addition to that many research solutions are developed to meet out the requirements only on homogenous environment. There are no much load-balancing methodologies to tackle the heterogeneous environment.

This paper, ways out for load-balancing which desire less cost in terms of data transferring cost and time requirement in sever side and also considers the heterogeneous environment. In order to offer load balancing in heterogeneous environment with minimal cost, we proposed a new load-balancing algorithm named (Exhaustive Criterion based Load Balancing) ECLB. To use this ECLB, the nodes are clustered based on weight and cost to improve the performance. On using load balancing algorithms it is necessary to find the capacity of all nodes to check that the node are either overloaded or under loaded. Checking nodes individually requires much time. So in our work we cluster the nodes depending on weight and cost of nodes. All clusters have their own cluster head. So that the verifying each node of the server is reduced. Since sever will give the checking and allocation work to the cluster head. On receiving request from server, cluster head will do the checking work

In this paper, we uses K-means algorithm to cluster the nodes. Grid computing is used to optimize the infrastructure; it provides capacity for high demand application. An added advantage of using grid environment is that it enables failure and recovery. It has the ability to run large-scale of application. It executes and computes across a large distributed set of resources instead of centralized one. Earlier days computations are carried out in Silos or in SMP boxes. Even today many companies using this SMP boxes, but it is expensive and suites only good for very big companies. Today all such large computations are carried out in grid computing by maximum of companies due to its less cost. K-means algorithm out performs in terms of computation. It is faster and also forms tighter cluster than the hierarchical methods. Especially the clusters are globular while they are designed by means of k-means algorithm. It is simple to use, since it doesn't requires experts to generate the training data. To further refine the work we uses inter and intra clustering. These inter and intra clustering process is carried out by the proposed ECLB Load-balancing Algorithm. Exhaustive Criterion based Load Balancing algorithm is designed by integrating the Cauchy and Normal distribution methods. Normal distribution techniques provides better solutions even in rough approximations, and derive the results genuinely, it works well in intra clustering. Cauchy method works well in inter-clustering, so our proposed ECLB load-balancing algorithm integrates the best parts of both distributions. The architectural diagram in Figure 1 shows how the load-balancing is carried out. The main advantage of using this method is that it works well even in heterogeneous environment with minimal cost. It reduces the cost incurred by transmission of data from an overloaded node to under loaded node in a network. Remaining section of this paper consist of related works presented in section 2. In section 3, our proposed work is explained in detail. Experiment results showing better performance of ECLB is in section 4. Section 5 concludes the paper with feature work.

II. RELATED WORKS

For the past three decays there are plenty of researches works have been under taken in the field of load-balancing. Among the vast research paper here we focus on the overview of relevant ones. Dobber et al. [1] analyzed the impact of the changes in the processing speed of the grid applications. They have also shown the experimental results and highlighted the need of dynamic load balancing for better performance. Authors also demonstrated the increase in processing speed while using simple dynamic load balancing scheme based on forecasts through exponential smoothing. The main drawback of the methodology in [1] was that it suites only for the simple computation-intensive applications not for the complex computational applications with complex structures.

Cao et al. [2], proposed a method for load balancing. AI technique in [2] is used to achieve workload and resource management. They combined intelligent agents and multi-agent approaches local grid resource scheduling and global grid load balancing. The proposed method was initial step were as there was no environment to provide self management. This framework doesn't include the features like coordination, knowledge based reasoning, and ontology based service brokering. The methods above are central in nature; this is prone to single point of failure. An intelligent, autonomous method for load balancing was proposed in [4]. This technique has ant, depending upon the existing condition the act either commit suicide or procreate. The difference between the similar mechanisms was it deploys ant colony optimization. Drawback of the method in [4] was that it won't schedule the jobs well in the scenarios were all the jobs are being sent to one or two nodes in the network.

Azin Moallem and Simone A. Ludwig [5] come up with two distributed artificial life-inspired load-balancing algorithm, it combinations Ant colony optimization and Particle Swarm optimization. This stands out robust against all type of topology changes in the network. It defends against single point of failure. Many load balancing strategies were developed by assuming homogenous set of sites. Heterogeneity, scalability and adaptability are the major issues that should be considered by computational grids. A layered algorithm proposed in

Volume: 04 Issue: 06 Pages:1786-1792 (2013) ISSN : 0975-0290

[3] achieves dynamic load-balancing. This method was independent from any physical architecture of grid. However it was not experimented in real grid application in order to validate the practicality of the model. A Global Load Balancing (GLB) algorithm in [6] was proposed by Belabbas yagoubi and Meriem Meddeber to meet out the requirements of fully distributed load-balancing with the objective of reducing response time and communication cost required during task transformation. A different way was found for load balancing in [7]. Author's use machine learning based load-balancing algorithm. This algorithm uses initial load information at initial level and current load information when load imbalance occurs. That information are taken as raw data and processed further to produce the rules to balance the load. Though it gives good result it suffers a lot to gather information and to take decisions from those collected information.

III. PROPOSED WORK

E-learning systems are the most hunting sites by learners now a day. Day by day users of E-learning sites are tremendously increasing. This may cause the server to response the user quires with delay or sometimes users may wait indefinitely without any response from server. This is due to server capacity may cause the server down as number of users increases. To tackle this server problem even in plenty of user request we proposed a load balancing algorithm named Exhaustive Criterion based Load Balancing (ECLB). It efficiently carries all user requests even in more complex situations by means of load balancing techniques. Our proposed work consists of following steps.

3.1 Cluster Formation
3.2 Workload Estimation
3.3 Decision making
3.4 Response to query

The flow of our proposed work is as shown in figure 2.

3.1 Cluster Formation

To balance work among nodes, we initially cluster the node in a network. To have tighter clusters we uses k-means algorithm to cluster nodes. Each cluster will have its own Cluster Head (CH). The cluster head calculates the capacity of cluster by calculating the capacity of individual nodes. Capacity calculation among cluster is simultaneous. It is evident that capacity calculation among nodes is easy and also reduces time and cost to process and collects information among groups rather than from individual nodes.

3.2 Workload Estimation

All Cluster Head collects the capacity details such as weight, cost, speed, capacity, etc., from each node present under it and estimates the overall capacity of their cluster. Cluster capacity details are sent to Central Manager (CM). Central Manager maintains a database that holds the capacity information of each cluster. It helps the CM to estimate the work load of each cluster. Central Manager updates the capacity of each cluster either by on demand update request from Cluster Head (CH) or periodically. Upon receiving workload information of all nodes from CH, CM performs the Exhaustive Criterion Load Balancing algorithm (ECLB) to estimate and distribute workload among clusters. It follows the following step to estimate the workload of each cluster:

i. Depends on capacity of each cluster it computes speed of each cluster
ii. Evaluate the load and processing time of clusters
iii. Compute standard deviation over processing time of cluster to calculate the workload of each cluster depending on the equation 1.

$$f(x;0,1) = \frac{1}{\pi(1+x^2)} \quad \text{..... (1)}$$

This equation is used to estimate the workload. 'X' in equation (1) represents the capacity of cluster and this equation inherits the concept of Cauchy distribution. This works well during inter clustering process. These information's stored in database of CM, are used during decision making process.

3.3 Decision making

A centralized database is maintained by server, which contains all the details of files such as file size, location, etc. In receipt of user request, CM sends the request to server to find file size. Depending on the file size CM decides the request should be accepted or it should be discarded. If the file size is less than or equal to the capacity of the entire network then it accept and returns the file to the corresponding CH. Otherwise discards the request. There are two cases exist on accepting the request.

i. Load distribution among inter clustering
ii. Load distribution among intra clustering

The above two steps make use of our ECLB algorithm to distribute load among clusters.

3.3.1 Load distribution among inter-clustering

In this event, if CM finds a single cluster to accomplish the task, then it assigns the work to a corresponding cluster head based on file size and processing time. For example, consider a network with three clusters c1, c2, and c3. Let 10MB, 20MB, 15MB be the capacity of the three clusters respectively. A request to handle a file with 20 MB arrives, the CM chooses c2 cluster to execute the task. The cluster c2 may have nodes with identifier n21, n22, n23, n24, n25, n26, and n27 with capacity of 5,0,5,2,1,10, and 3Mb's respectively, these capacities of nodes are found during workload estimation itself using the equation 1. The CH on acceptance from CM it starts sharing the workload among the nodes. In our scenario, CH checks all nodes capacity and then assigns

Volume: 04 Issue: 06 Pages:1786-1792 (2013) ISSN : 0975-0290

the work first to node n26. As it has no required capacity, other node with next maximum capacity is selected, this case node n1 with 5Mb is selected. Still we require 5Mb more to carry out the process the CH continuously quest for nodes to share the work. It chooses one more node n3 with 5Mb. These nodes together will complete the task. In this way the CH will share the workload among nodes in its cluster. Cluster Head keep on looking for nodes still it meet the required amount of capacity to complete task. In this circumstance, it incurs no communication cost. Since all the nodes are connected to the same LAN. So transferring information from one node to the other is will not suffers from transferring datum.

3.3.2 Load distribution among intra clustering

If there is no only one cluster has the capacity to accomplish the task, then it is the duty of CM to share the load among more than two clusters. Take the example explained earlier. If 40 GB is required to finish a task, single cluster will not be able to perform the task then a need arises to finish the task by sharing the workload among the cluster. Sharing the work among clusters is said to be intra clustering. This process should be taken out with extra care. Since, it requires additional cost during task transformation from one cluster to another. This scenario subsists only when there is demand for load by receiver is higher than the load supplier by exists. In this case a threshold value is set by the user depending upon the capacity of clusters. There are three conditions exists as follows

i. if (load supply available on receiver node/ load demand required by source node) < threshold then it is under loaded
ii. if (load supply available on receiver node/ load demand required by source node) = threshold then it is shared
iii. if (load supply available on receiver node/ load demand required by source node) > threshold then it is overloaded

In case (iii), a cluster is overloaded so it is important to share the load to another cluster. As said earlier it is significant to share the task with another cluster which is having low latency. This will reduce the time and cost of transferring task from an overloaded to under loaded cluster. Once the sharing work finishes clusters will execute their process and generate result.

3.4 Response to query

Once the processing are finished, then node report result of the execution to its cluster head. In case of intra clustering the CM should wait until all the CH responses. Only on receiving responses from all CH it can response to the user. Thus on using the clustering and ECLB algorithm we can replay the user as early as possible with low cost.

IV. EXPERIMENTAL RESULTS

We have implemented our proposed strategy on the GridSim V4.0 simulator to appraise the practicability along with performance of our ECLB algorithm. Resource scheduling as well as management in a large scale distributed system is complex. Therefore for analyzing the algorithm we use GridSim simulators before applying them in real system. It is a Java-based discrete-event tool kit, which is a feasible way to analyze algorithms on wide ranging distributed system of heterogeneous, resources, users, applications. Here tasks or jobs are modeled as Gridlet objects, that contains all information related to the job and execution constrains like, task parameter, resource parameters. All experiments have been performed on Pentium(R) Dual-Core processor with 1 GB as memory running on Windows 7.0. The experiments were carried out by varying performance parameters of Grid such as, number of clusters, nodes and tasks. The experimental results are compared against an existing GLB algorithm [6].

4.1 Variation of Average Response time before and after load balancing

We analyze the average response time by varying the number task then the number of nodes to accomplish the task. Number of nodes is varied from 100 to 400 by increasing 100 nodes and the number of tasks varied from 6000 to 10000 by increasing 2000 tasks. Bulk 1, Bulk 2, Bulk 3 in all figures represents the number of task 6000, 8000, 10000 that arrives respectively. The experimental result shows that the gain of average response time of ECLB is lesser than the GLB. Likewise from the analysis it is clear that average response time is better when there are 200 to 300 nodes. Below 200 and above 300 the performance slightly decreases. Figure 3 and 4 shows average response time before balancing by using GLB and ELB respectively. Average response time after balancing is reduced significantly that is shown in figure 5 and 6.

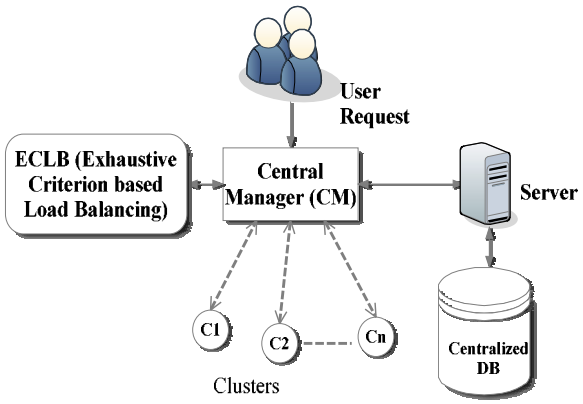

Figure 1 Architecure of proposed work

Volume: 04 Issue: 06 Pages:1786-1792 (2013) ISSN : 0975-0290

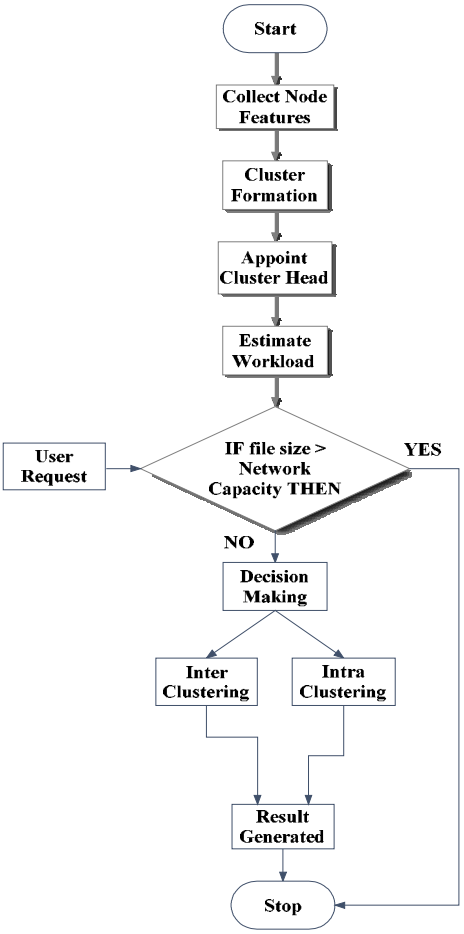

Figure 2 Flow Diagram of ECLB

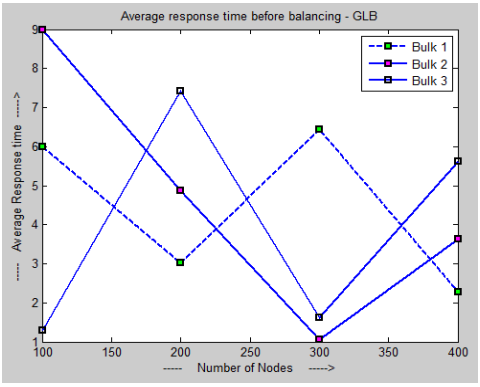

Figure 3 Average response time before balancing using GLB

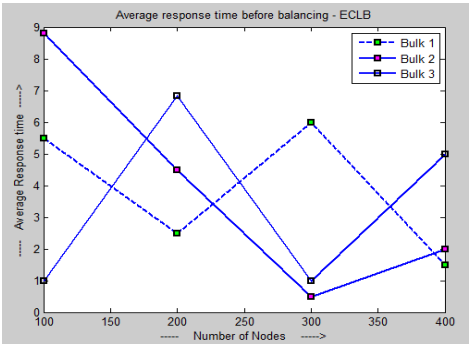

Figure 4 Average response time before balancing using EVLB

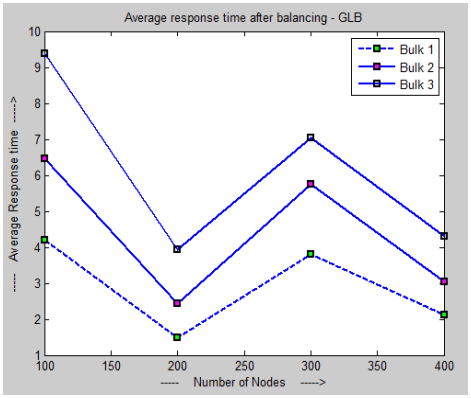

Figure 5 Average response time after balancing GLB

Figure 6 shows that the average response time after balancing using our proposed Exhaustive Load Balancing algorithm is reduced when compared to the response time of existing GLB algorithm shown in Figure 5. Once the response time reduces consequently user's waiting time also decreases. Our algorithm provides response to user effectively than GLB at all situations (Even in different number of tasks 6000, 8000, 10000).

Figure 7 and Figure 8 shows the gain of average response time by the existing (GLB) and our Proposed (ECLB) algorithm. It shows that as the response time decreases then gain increases.

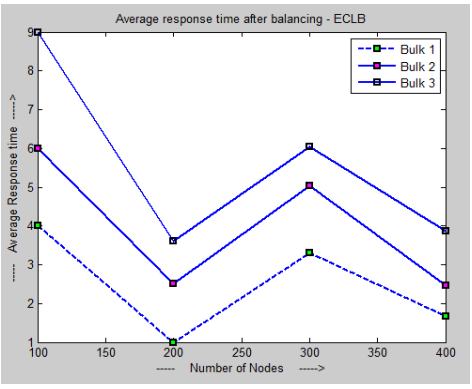

Figure 6 Average response time after balancing ECLB

Volume: 04 Issue: 06 Pages:1786-1792 (2013) ISSN : 0975-0290

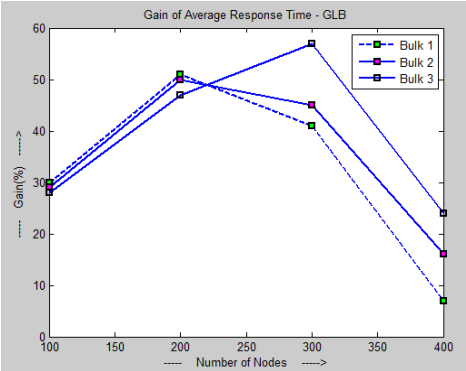

Figure 7 Gain of average response time for GLB

4.2 Number of task transformation

Here we analyze the number of task transformation by varying the number task and number of clusters. Number of cluster is varied from 2 to 16, and step up by multiples of 2 clusters and the number of tasks varied from 6000 to 10000 by increasing 2000 tasks. The Bulk values in figures represent the number of task. Figure 9 and 10, shows number of task transformation required to finish job for the existing (GLB) and our Proposed (ECLB) algorithm.

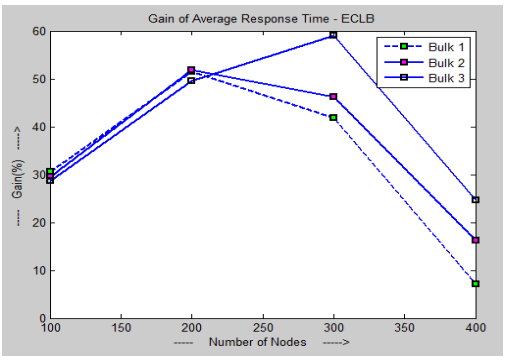

Figure 8 Gain of average response time for ECLB

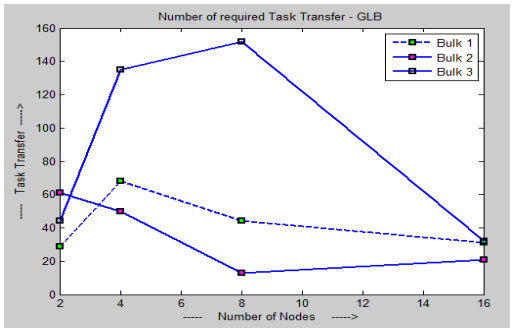

Figure 9 Number of task transfer required on using GLB

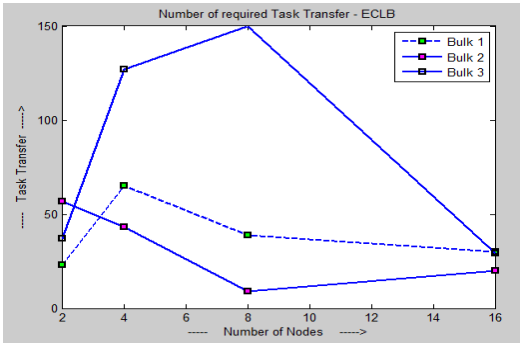

Figure 10 Number of task transfer required on using ECLB

From the simulation outcome, it is vibrant that the number of task transfer required by ECLB is less than the existing (GLB) method. As we clustered the nodes using k-means algorithm, we attain tighter clusters based on weight and cost. This reduces the number of task to be transferred between nodes. Number of task transfer required to accomplish the job is directly proportional to cost. Therefore our ECLB algorithm requires less cost to balance the load. This achieves our objective of sharing and balancing load at less cost. Following figures 11 and 12 portrays the ratio of task transfer using GLB and ECLB.

From the result we can observe that when there are 16 clusters we get marginal benefit. The reason is that there are more under loaded or idle nodes when there are 16 clusters.

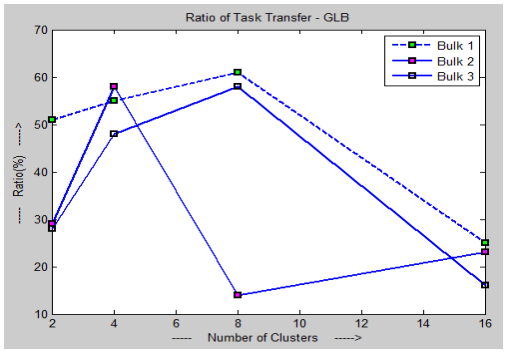

Figure 11 Ratio of Task Transfer Using GLB

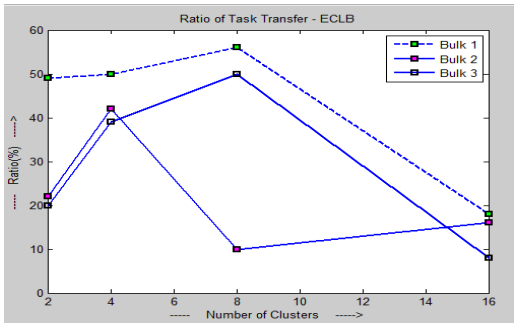

Figure 12 Ratio of Task Transfer using ECLB

Volume: 04 Issue: 06 Pages:1786-1792 (2013) ISSN : 0975-0290

V. CONCLUSION

A fruitful solution for educational institution for sharing their study material is E-learning. Internet makes E-learning so popular this good sine leads to a server management problem on handling huge amount of request. This problem of server is successfully overcome by our proposed ECLB algorithm. This algorithm out performs the existing methods in terms of time, memory, and cost. Our experimental results in section 4, shows that throughput of our ECLB algorithm higher than the existing work. Also it improves the response time of the user by assuring maximum utilization of available resources. ECBL requires less number of task transfers which minimizes cost incurred to finish a task. In future this work will be extended to implement in real grid scenarios to analyze their performance in more realistic environment. Also enhance our ECLB algorithm for the development of intelligent E-learning framework.

REFERENCES

[1] Menno Dobber, Ger Koole and Rob van der Mei, "*Dynamic Load Balancing for a Grid Application*," 11th *International Conference on High performance Computing (HiPC)*, (Bangalore, India, pp. 342-352), Dec 19-22, 2004.

[2] Junwei Cao, Daniel P. Spooner, Stephen A. Jarvis, and Graham R. Nudd, "*Grid Load Balancing Using Intelligent Agents," in Future Generation Computer System*, Elsevier, (vol. 21, issue 1, pp. 135-149), Jan 2005.

[3] Belabbas Yagoubi and Yahya Slimani, "*Dynamic Load Balancing Strategy for Grid Computing," in World Academy of Science, Engineering and Technology*, 2006.

[4] Mohsen Amini Salehi and Hossain Deldari, "*Grid Load Balancing Using An ECHO System of Intelligents," in the proceedings of the 24th IASTED International Multi-Conference on Parallel and Distributed Computing and Networks*, (pp. 47-52, Innbruck, Austria), Feb 14-16, 2006.

[5] Azin Moallem and Simone A. Ludwig, "*Using Artificial Life Techniques for Distributed Grid Job Scheduling," in the proceedings of ACM symposium on Applied Computing SAC '09*, (pp. 1091-1097, New York, USA), 2009.

[6] Belabbs Yagoubi and Meriem Meddeber, "*Distributed Load Balancing Model for Grid Computing*," in ARIMA journal, (vol. 12, pp. 43-60), 2010.

[7] Ashish Revar, Malay Andhariya, Dharmendra Sutariya, and Madhuri Bhavsar, "*Load Balancing in Grid Environment Using Machine Learning – Innovative Approach," in the proceedings of International Journal of Computer Applications*, (vol. 8, no. 10, pp. 31-34), Oct 2010.

[8] Cardellini V, Colajanni M and Yu P. S, "*Dynamic load balancing on Web-server Systems," in the journal of Internet Computing*, (vol.3, issue 3, pp. 28-39), May/Jun 1999.

[9] Mor Harchol-Balter and Allen B. Downy, "*Exploiting process lifetime distribution for dynamic load balancing," in the journal of ACM Transactions on Computer System*, (vol. 15, issue 3, pp. 253-285), Aug 1997.

[10] Willebeek-LeMair, M.H, and Thomas J. Waston, "*Strategies for dynamic load balancing on highly parallel computers," in IEEE transactions on Parallel and Distributed System*, (vol. 4, issue 9, pp. 979-993), Sep 1993.

[11] John Byers, Jeffrey Considine and Michael Mitzenmacher, "*Simple Load Balancing for Distributed Hash Tables," in the Lecture Notes of Computer Science*, (vol. 2735, pp. 80-87), 2003.

[12] Amis, A.D and Prakash, R., "*Load-balancing clusters in wireless ad hoc networks," in the conference publications of Application-Specific System and Software Engineering Technology*, (pp. 25-32), 2000.

[13] Zaki, M, J, Wei Li and Parthasarathy S, "*Customized dynamic load balancing for a network of workstations," in the proceeding of 5th IEEE International Symposium on High Performance Distributed Computing*,(pp. 282-291, 6-9) Aug 1996.

[14] James Aspnes, Yossi Azar, Amos Fiat, Serge Plotkin and Orli Waarts, "*On-line load balancing with applications to machine scheduling and virtual circuit routing," in the proceedings of 25th annual ACM symposium on Theory of Computation*,(ISBN. 0-89791-591-7, pp. 623-631), 1993.

[15] Schloegel, K, Karypis, G and Kumar, V, "*A Unified Algorithm for Load-balancing Adaptive Scientific Simulations," in the conference publications of ACM/IEEE on Supercomputing*, (pp.59, 04-10) Nov 2000.

[16] Watts J, "*A practical approach to dynamic load balancing," int the IEEE transactions on Parallel and Distributed System*, vol. 9, issue 3, (pp. 235-248), Mar 1998.

Authors Biography

Rajarajeswari.S is an Assistant Professor of the Department of Computer Science, Virudhunagar Hindu Nadars' Senthikumara Nadar College, and Faculty of Computer Applications. She has been involved in several international and national research conferences and presented various papers related to intrusion tolerance and network security, data mining. Her main research interests are: intrusion tolerance, network security, distributed systems, data mining.