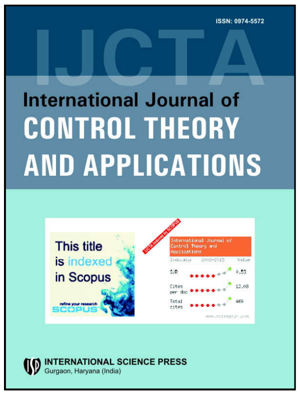

International Journal of Control Theory and Applications

ISSN : 0974-5572

© International Science Press

Volume 10 • Number 15 • 2017

Centralized approach of load balancing in Homogenous Grid Computing Environment

Priyank Sirohi[1], Arpit Chhabra[2] and Manav Bansal[3]

[1] *Email: priyanksirohi@gmail.com*
[2] *Email: arpit.0121@gmail.com*
[3] *MANAVBANSAL82@GMAIL.COM, SCRIET, CCS UNIVERSITY*

Abstract: The two major challenges in Grid Computing are, managing the workload information and resource management to better facilitate to the Grid users. It depends on many factors like computing nodes, nature of the computing nodes, nature of the resources, processing capabilities of the nodes, the various load conditions on the different computing nodes, overloading and no utilization of the nodes, non-availability of the resources.

The above challenges can be covered by an efficient method of load balancing where load can be managed properly. It means if a node is overloaded then shifts the job to the other underutilized node or ideal node, this process is called the job migration. Increase the throughput and reduce the communication cost during job migration are major aspects of load balancing in grid computing environment. Another major point that we have to address is load should be equally distributed among the nodes connected in the gird system.

In this paper we proposed a centralized approach of load balancing for homogenous grid computing environment. The proposed algorithm works in three stages, first the monitored node to get the workload information from all connected computing nodes in the system, second the resource allocation node, responsible for allocating the resources followed by load balancing node responsible for job migration.

Keywords: Load balancing, Static Load Balancing, Application Programming interface, Job Migration, Dynamic Load Balancing, Grid Computing, Load Sharing, Distributed Computing, Cluster Computing

1. INTRODUCTION

With rapid growth of internet and other technologies, it seems more challenging to develop the high speed networks with powerful capabilities in lower computational cost [15]. In order to address this; grid computing has emerged rapidly. Using Grid Computing technologies the distributed resources can be accessed or used without any geographical limitations. Various services are used to fulfill the user requests and the client is not aware about the functioning of the server. Services are the abstraction of the resources connected to the system for fulfilling the user request and server is delivering the responses to the users. Client is only sending the requests through application programming interface (API's) and server has to respond as per the requests receives from the client.

Client sends the requests and server respond to the Clients; this type of communication is called inter process communication. To make communication between the client and server we need a communication language and set of rules and dependencies that are suitable to both client and server. The communication language and dependencies are properly defined and mentioned as a communication protocol and all the protocols should operate at application layer. For exchange data between the client and server we need to implement API like web service to facilitate services. API is the abstraction between the client and server that can be achieved by specialized computer programs. Server is receiving the request from clients in a different period of time and processing the request. Server will perform the limited number of requests as per the availabilities at server; it depends on the various factors like scheduling policies or the priorities of the incoming request from the clients. In order to prevent from denial of service and maximize the response time the server has to limit the number of request from number of clients. A server should have the better scheduling policies and having adequate application programs that can save the server activities from bombarding of request from the clients. This is the ability of the server to restrict requests from unauthorized clients. Load Balancing is the process where the tasks are equally distributed to the processors or computing elements connected to the grid environment to achieve the higher level of performance [16]. The load balancing can be done either at compile time called static load balancing or run time known as dynamic load balancing. This chapter clearly focuses on the various issues to initiate a load balancing algorithms and comparative analysis between existing load balancing algorithms.

The proposed model have a monitoring node, it can manage the workload information of the computing elements connected to the system. The monitored node sends the workload and resource information to the resource allocation node for the assignment and creating pool of the resources. The resource allocation node sends the workload information for load balancing and migration of tasks from one computing element to another due to failure or higher load at the node

2. LITERATURE REVIEW

The *terms Grid computing* used in the early 1990s as a representation for making computer system more powerful and easy to access like power grids used in electric system [20]. According to The Grid it is Blueprint for a new computing Infrastructure [1]. Sharing of resources and processing speed and volunteer computing was famous in the 1997 by distributed computing method and then in 1999 by SETI to connect the power of networked computers worldwide, to solve problems requires computational power.

A number of load balancing algorithms already proposed, those deals with both homogenous and heterogeneous computing environment on varying work load conditions. The load balancing criteria depend on various factors like topology, bandwidth of the network, latency, throughput and arrival of tasks at the node [2][3].

Balance the load in distributed computing environment is bigger challenge and the same studied for a longer interval of time. Characterized the behavior of decision-making policies by various load sharing and load balancing algorithms and different structures; those are the performance criteria and efficiency of the proposed methods [4]. A dynamic load balancing approach where migration of jobs done from a working node to its nearest neighboring working node [5]. A detailed study of various load sharing algorithms having different design structure is proposed in the literature [6]. A load balancing algorithm which deals with heterogeneous nature of resources and performance of the working nodes is proposed [7]. Various load balancing algorithms are analyzed for web server based on cluster computing approach has been presented [8]. An analysis for distributed computing environment is proposed to take care of performance and efficiency of the system by considering the load conditions and timely submission of the jobs in the grid computing environment [9]. Allocation of tasks is the bigger problem in the distributed/cluster computing environment has been proposed as a binary (0-1) problem by the algorithm [10]. A hybrid particle swarm optimization algorithm is proposed to optimally allocate the tasks in timely manner. A linear programming using multiple integer method is proposed to minimize the cost of setting grid computing environment [11]. Load balancing approach for multi-server, multi-class environment with Poisson distribution is proposed having transfer in heterogeneous centralized grid computing environment and decentralized non-cooperative or distributed computing environment. The load balancing can be done in optimal basis and the condition of load imbalance is discussed in the algorithm [16]. An appropriate method is proposed to minimize the load imbalance probability in the grid computing system. Queueing model is also discussed to check the performance and optimization of the working nodes in distributed and parallel computing environment [12]. The optimal load balancing of task using computing elements discussed in queuing theory [13].This is the model of load balancing in dynamic grid computing environment that works in G/L/M fashion where load is balanced in multiple levels having various parameters like load of the working node, overloaded node, transfer of load and the number of nodes [13][20]. Queuing approach for distributed and parallel computing systems are also discussed in literature [12]. Adaptive load sharing method was proposed to get the higher efficiency and optimal resource utilization by queuing model [14].

3. ELEMENTS FOR INITIATING LOAD BALANCING ALGORITHM

There are various factors required for initiating load balancing like arrival of job, completion time of job, resources in and out that affects decision making while starting load balancing algorithm [17]. Some other major points are failure of nodes and overloading at different nodes that can affect the performance of grid environment. These factors are summarized below:

- Arrival of new job: While any new job enters into the system, it requests for the resources that initiates the load balancing algorithms then after it assigns the resources to the job.
- Completion of job: After getting the resources, process them and sends for execution.
- Execution of job: Once the job complete, it releases the current occupied resources so that it can be assigned to any other waiting jobs in the system.
- New resource join Grid: Whenever any new resource wants to join the system it can join and then other waiting job gets the resources as per the requirement.
- Leaving of existing resource in the System: Any time any resource can leave the system but it can increase the overhead to the remaining resources in the system.
- Overloaded or failure of node: Due to the overload or failure of node the other nodes face the challenges of additional load and in this situation we have to start load balancing. In this scenario we have to start load balancing by avoiding transferring the jobs to such nodes those are utilized by approximately 80 percent.

4. LOAD BALANCING ALGORITHM

Load can be calculated based on the number of jobs are waiting in the queue or may have the higher load at the computing node/ non performing node due to failure. We can improve the performance of the computational grid by transferring the load from one node which is highly loaded to the lightly loaded or allotting the computing resources to the jobs those are in the waiting queue. Load balancing is the process by we can improve the performance of the ideal/unutilized resources to ensure the maximum utilization of the resources. Designing a load balancing algorithm that can improve performance, throughput, and resource utilization is a challenging task. Proposed load balancing model is based on client server approach that effectively manages all the challenges.

4.1. Proposed Load Balancing Model

This section describes the model for load balancing for client server architecture. Figure 1 is the flow of information and resources in the system. Server is responsible for collecting the workload information of the computing nodes like used and unused status. This server handles workload and resource information of the connected nodes and the same updated in the database of the server. The information stored in the server can be accessed by the users by requesting with specified application and it can be retrieved from any computing node in the grid system.

Server keeps all the information and states of the nodes by monitoring tool and updates the current status in the database. For accessing the server we have created web pages shown in figure 2 and deployed on local area network that access the information of the resources by using Apache HTTP server. The grid environment on local area network running on Apache HTTP server by which we can access the web pages depicted in following figure:

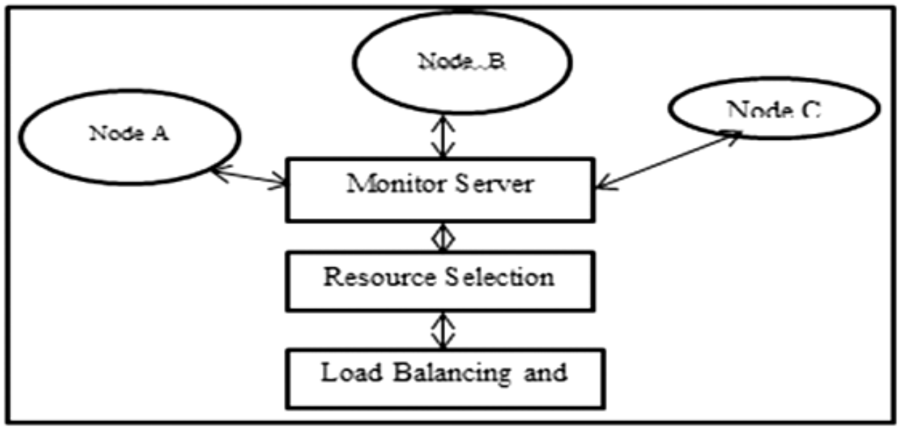

Figure 1: System Overall Architecture

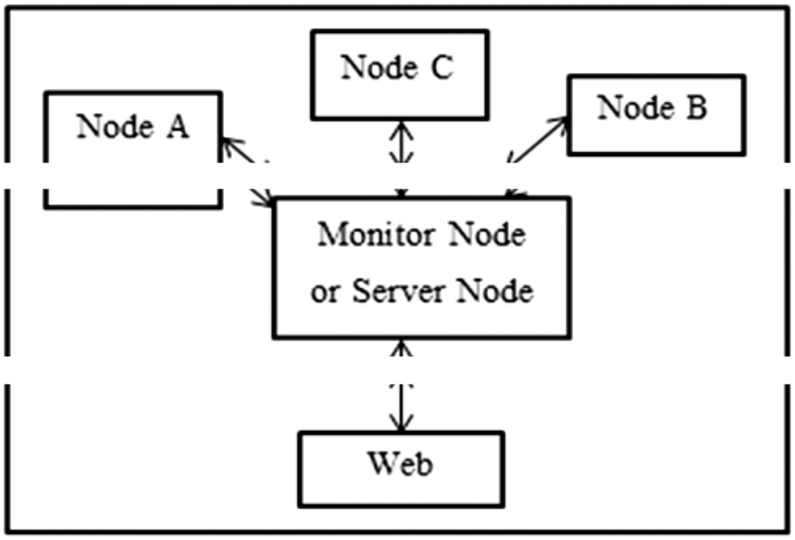

Figure 2: Implementation View of Algorithm

4.2. Load Balancing Algorithm

The proposed model is explained by the following algorithm:

Active (Active the system for execution)

(Start)

```
Monitoring information & job request (n jobs, m resources)
Form job waiting queue
If (job request get the resources)
Assigned job to resource
Else
Return to main queue
If (failure of node)
Return to main queue with its status marked
Else
Job finished
End
```

4.3. Complexity Analysis of Algorithm

Complexity is the suitable way to analyze the performance of the algorithm by determining the CPU time taken and use of memory. The complexity of proposed algorithm is computed by the following notation.

Max (F3, F2, F1, F, C) where C is a constant

As per the definition of the above notations the complexity is

$f(n) = \text{Max}(O(F3), O(F2), O(F1), O(F), O(C))$

$f(n) = O(F3)$

So the complexity of the algorithm is

$F(n) = O(F3)$.

There are some other methods available to find the complexity of the above algorithm:

Complexity = Total number of closed loop in the algorithm = 2 (in the proposed algorithm)

Complexity = Number of decision making conditions + 1; =(2+1)=3

4.4. Implementation of Algorithm

The web pages are used to monitor the nodes after submitting jobs to find out overloading. Three web pages have been designed.

5. JOB MIGRATION TECHNIQUE

Migration represents the transfer of jobs from one computing node to another computing node using selection criteria. Job migration can be done in runtime due to overload or failure of the node. While load balancing it is important to transfer the jobs from one node to another node efficiently to achieve higher optimization. It guarantees that the submitted jobs will execute as per user requirements. In such cases when any one of the node is highly loaded the job will be dynamically transferred to the node having lightly loaded with the load balancing algorithm. We can use the Gridftp service for the job migration that transfers the large amounts of data securely, reliable and faster more than normal FTP. GridFTP may be used either stand alone or as with Globus Toolkit [19].

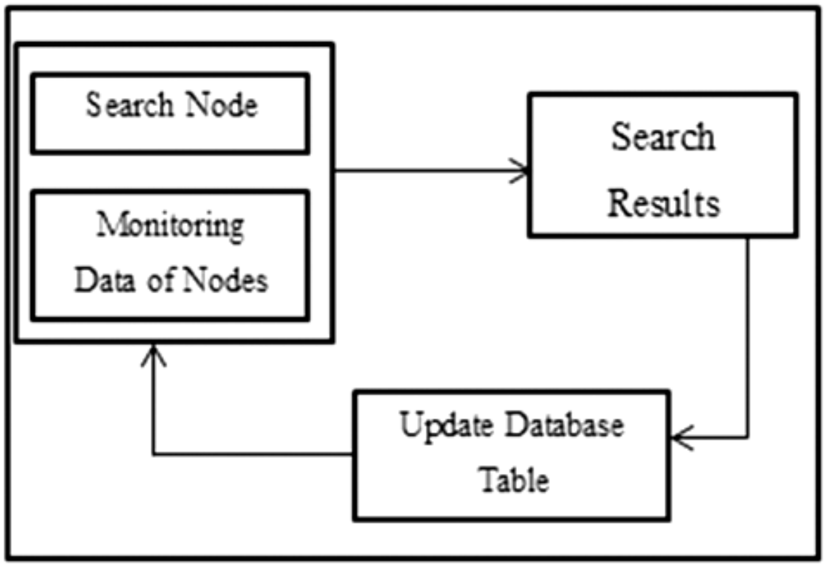

Figure 3: Design of Web Pages

6. EXPERIMENTAL STUDY

The proposed algorithm is executed by sending the requests from the computing nodes to the monitor node and checking the availability of the resources at the resource allocation node followed by load balancing and processing of the jobs at individual nodes. Table 1 show the main page of the algorithm, where users can search nodes according to CPU speed, idle CPU and free memory required by job. This also provides monitoring data of all the connected nodes. The monitoring is shown per job and reflects, if any overloading found there. If the current idle CPU and free memory is less than that required by the job, it shows overloading as depicted in Table 4.

The following parameters are used during experimental analysis:

CPU Speed- 2000 KHz

Idle CPU-75%

Free Memory - 2500 MB

Table 1: Main Page of Proposed Algorithm

Monitored Nodes								
IP Address	Host Name	CPU Speed	CPU Idle	Total Memory	Free Memory	Average Load	Running Status	Job Status
172.18.10.105	node x.grid	2500 KHz	100 %	8132 MB	6015 MB	0.96, 0.91, 0.65	Global	Running
172.18.10.106	Node y.grid	2500 KHz	90%	8132 MB	2613 MB	0.25, 0.14, 0.05	Local	Running

Table 2: Status of Jobs

Available Nodes								
IP Address	Host Name	CPU Speed	CPU Idle	Total Memory	Free Memory	Average Load	Running Status	Job Status
172.18. 10.105	Nodex .grid	2500 KHz	100%	8132 MB	6512 MB	0.97, 0.92, 0.67	Global	Job Submitted
172.18. 10.106	Nodey .grid	2500 KHz	90 %	8132 MB	5102 MB	0.49, 0.57, 0.51	Global	Job Submitted
172.18. 10.123	Nodez .grid	2500 KHz	85 %	8132 MB	3345 MB	0.32, 0.19, 0.08	Local	Job Submitted
172.18. 10.135	Nodeu .grid	2500 KHz	80 %	8132 MB	3456 MB	0.29, 0.21, 0.13	Local	Job Submitted

The results of Table 2 shows all matched available nodes. It shows the IP address, host name, CPU speed, CPU idle, total memory, free memory and load average. It also shows whether the Globus toolkit is running or not. It also provides the hyperlink to jobSubmit.

If no node is available with respect to search criteria than an error message will be display that is shown in Table 3. jobSubmit gets the required free memory ,idle CPU required by job and the IP address of the selected node from the search page then it updates the jobSubmit table. This table is required to monitor the jobs status against nodes. This page redirects to the main page. Table 4 depicts a scenario where the available idle CPU and/ or free memory is less than required by the job and the main page shows it by an overloading error. In this case

Table 3: Error Messages
Error: - No Matched node found

Available Nodes								
IP Address	Host Name	CPU Speed	CPU Idle	Total Memory	Free Memory	Average Load	Running Status	Job Status

Table 4: Overloading Message

Monitored Nodes								
IP Address	Host Name	CPU Speed	CPU Idle	Total Memory	Free Memory	Average Load	Running Status	Job Status
172.18.1 0.105	Nodex grid	2500 KHz	30 %	8132 MB	6512 MB	0.97, 0.92, 0.78	Global	Job Running
172.18.1 0.105	Nodex grid	2500 KHz	30 %	8132 MB	5102 MB	0.97, 0.92, 0.78	Global	Overloading

two jobs are running at nodex. First is running the job and the second one is showing overloading because one job using resources according to requirement but another one having the higher load so it is showing overloading.

7. CONCLUSION AND FUTURE SCOPE

A comparative study among the existing load balancing algorithms is performed, which will aid in exploring an efficient algorithm. A load balancing algorithm based on client server model of a grid is then proposed which works for homogenous grid environment.

The proposed algorithm reduces the complexity on transferring the jobs from one node to another that can be seen by the complexity analysis. The complexity of the proposed algorithm is only 2 or 3 so the algorithm is best suited for local communication within a local area network. The algorithm is implemented in real grid environment using Globus Toolkit. The experimental results shows how the monitored node get the workload information from the various nodes connected to the system and forward the information to the resource allocation node followed by load balancing. The results show the states of nodes in each level, either the nodes are properly utilized, having overload problem or error problem. All the results best fit as per the user requirement in a grid computing environment.

The following are the suggestions for future scope of investigations:

- The reliability optimization of the proposed models can be studied.
- Possibilities of designing proposed algorithms using soft computing techniques like ANN, genetic algorithms can be investigated.
- The proposed algorithms and models can be extended for fault tolerance.

REFERENCES

[1] Foster, I., Kesselman, C. and Tuecke, S. "The Anatomy of the Grid: Enabling Scalable Virtual Organizations" International Journal of High Performance Computing applications, Vol. 15, issue 3, pp. 200-222, 2001.

[2] Zeng Z. , Veeravalli B.," Design and performance evaluation of queue-and-rate- adjustment dynamic load balancing" IEEE Transactions on policies for distributed networks Computers, volume 55, issue 11, pp.1410–1422, 2006.

[3] Wu J., "Distributed System Design", CRC press, 1999.

[4] Casavant T.L., Kuhl J.G. ," A taxonomy of scheduling in general-purpose distributed computing systems", IEEE Transactions on Software Engineering, volume 14, issue 2, pp. 141–154, 1988.

[5] Xu C., Lau F., "Iterative dynamic load balancing in multicomputer", Journal of the Operational Research Society, pp. 786–796, 1994.

[6] Shivaratri N.G., Krueger P., Singhal M., "Load distributing for locally distributed systems. IEEE Transactions on Computers, volume 25, issue 12, pp.33–44, 1992.

[7] Boyer M., Skadron K., Che S., Jayasena N., "Load balancing in a changing world: dealing with heterogeneity and performance variability", Computing Frontiers an international conference, pp. 21-25, 2013.

[8] Zou S., "Analysis of load balancing strategy of the web server cluster system", Communications and Information Processing by springer, pp. 699–706, 2012.

[9] Kremien O. , Kramer J. , " Methodical analysis of adaptive load sharing algorithms in Parallel and Distributed Systems", IEEE Transactions on Parallel and Distributed Systems, volume 3, Issue 6, pp.747–760, 1992.

[10] Yin W., Yu S., Wang P., Wang T. , "Task allocation for maximizing reliability of a distributed system using hybrid particle swarm optimization", Journal of Systems and Software, volume 80, issue 5, pp.724–735, 2007.

[11] Aubrey A., Rossi A., Espinouse M. , Jacomino M.," Minimizing setup costs for parallel multi-purpose machines under load-balancing constraint", European Journal of Operational Research, volume 187, issue 3, pp.1115–1125, 2008.

[12] Boxma O.J. , Koole G., Liu Z. , "Queuing-theoretic solution methods for models of parallel and distributed systems" Centre for Information, Department of Operations Research Statistics and System Theory, 1994.

[13] Spies F., "Modeling of optimal load balancing strategy using queuing theory", Micro processing and microprogramming, volume 41, issue 8, pp.555–570, 1996.

[14] Kabalan K.Y.,Smari W., Hakimian J., " Adaptive load sharing in heterogeneous systems: Policies, modiûcations, and simulation", International Journal of Simulation Systems Science and Technology, volume 3, issue 1-2, pp.89–100, 2002.

[15] Patni J.C., Aswal M.S., Gupta A., Pal O.P., "Load balancing strategies in Grid Computing" 3rd International Conference on, Volume:3, Kanyakumari, India, pp. 239 – 243, 2011.

[16] Patni J.C., Trivedi N., Sharma A., Agarwal A. "An optimal approach of load Balancing for grid computing", International conference on information Technology, Pune, India, pp. 567-571, 2012.

[17] Patni J. C., Aswal M. S.,. "A dynamic and optimal approach of load balancing in heterogeneous Grid Computing environment" Emerging ICT for bridging the future by Springer volume 2, pp. 447-456, 12-14 December, 2014.

[19] Smith P., Hutchinson N., "Heterogeneous Process Migration", The Software Practice and Experience", Volume 28, Issue 6, pages 611–639, May 1998.

[20] Patni J. C., Aswal M. S., "Dynamic Load Balancing Model for layered Grid Architecture" 1st International Conference on Next Generation Computing Technologies (NGCT), IEEE, pp. 119-122, 4-5 September, 2015.

[21] Patni J. C., Aswal M. S., "Distributed Load Balancing Model for Grid Computing Environment" 1st International Conference on Next Generation Computing Technologies (NGCT), IEEE pp. 123-126, 4-5 September, 2015.