*Proceedings of **BITCON-2015** Innovations For National Development*
National Conference on : Information Technology Empowering Digital India

Research Paper

A SURVEY REPORT ON LOAD BALANCING ALGORITHM IN GRID COMPUTING ENVIRONMENT

Anuradha Sharma[1], Seema Verma[2]

Address for Correspondence

[1]Asst. Prof. (IT Deptt.), Bhilai Institute of Technology, Durg(CG),
[2]PG Fellow (Comp. Network), SMT Kashibai Navale College of Engg,Pune (MH),

ABSTRACT
Grid computing is extension of distributed computing that incorporates coordinating and sharing of computational power, data storage and network resources across dynamic and geographically dispersed organizations. One motivation of Grid computing is to unite the power of widely distributed resources, and provide non-trivial services to users. To achieve this goal, an efficient Load Balancing system is an essential part of the Grid. The main goal of Load Balancing algorithm is to distribute the jobs among processors to maximize throughput, minimize total turnaround of jobs, to match the application need with the available computing resources, maintain stability, and resource utilization. Motivation of the survey is to encourage the researcher in the field of grid computing, so that they can easily understand the concept of Load Balancing and can contribute in developing more efficient Load Balancing algorithm.

KEY WORDS: Grid Computing, Distributed Computing, Load Balancing.

1. INTRODUCTION

Rapid growth in use of computer has increased the number of applications which uses the shared hardware and software resources (e.g. memory, processor, files etc.) and ultimately increased the amount of submitted jobs across internet. Problem can be solved if we distribute the applications across different computer, in such a manner that it reduces the job response time and the overhead on a single computer. Proper distribution of applications across different available resources is termed as Load Balancing.

Load balancing algorithms can be classified into two categories, *static and dynamic.* In static scheduling, the assignment of the tasks to the nodes is done before the execution of the program. Information regarding task execution time and processing resources is assumed to be known at compile time. A task is always executed on the node to which it is assigned. Dynamic load balancing algorithm is based on the re-distribution of processes among the processors during execution time. Dynamic load balancing algorithm assumes no a priori knowledge about job behavior or the global state of the system (Dynamic load balancing algorithm don't have prior knowledge about job behavior or the global state of the system), i.e., load balancing decisions is solely based on the current status of the system. The development of an effective dynamic load balancing algorithm involves many important issues: load estimation, load level comparison, system stability, amount of information exchanged among nodes, job resource requirements estimation, job's selection for transfer, and more.

Advantage of dynamic load balancing is that the run-time behavior of the system does not need to be known in advance. The major drawback of dynamic load balancing schemes is the run-time overhead due to the load information transfer among processors and time consumed for selection of processes and processor for job transfers, and the communication delay due to task relocation itself.

Section 2 presents a survey of several existing Load Balancing algorithms in grid environment. Section 3 provides an analysis and a parameter wise comparison among all the surveyed papers. Section 4 presents conclusion of this paper and lastly the references.

2. EXISTING LOAD BALANCING ALGORITHM IN GRID ENVIRONMENT

Many papers have been published to address the problem of load balancing in different environments such as Grid computing, peer-to-peer, distributed etc. Some of the proposed Grid computing load balancing policies is modifications or extensions to the traditional distributed systems load balancing policies. Some of them are summarized here:

Tal Maoz et.al. [11], presented distributed model of dynamic load balancing algorithm with reduced migration time and down-times. This algorithm's focuses on the topology and the physical parameters of the links. This proposed model is compared with MOSIX process migration and Jobrun's VM migration. Michael Schmitz et.al. [12], presented decentralize approach of LB, which Focuses on minimization of communication delays and communication costs, Avoidance of unproductive migration, and avoidance of oscillations.

In paper Distributed Route Control Schemes [13] authors proposed Centralized LB model that Handles traffic fluctuations. Main feature of this Centralized routing techniques is to distribute the incoming traffic of a multi-homed stub network among its various egress links. Md. Abdur Razzaqu et. al. [14], formulated a distributed LB technique which try to reduce the amount of message transfer between two nodes so as to decrease scheduling decision time to improve the system performance. The major contribution of this paper includes workload migration technique and dynamic and stable technique to schedule the jobs that requires only 2(K-1) messages to decide whether to execute a process locally or remotely.

In paper [15], authors proposed a distributed Biased Random Sampling (BRS) technique, in which network structure can be changed dynamically to efficiently distribute the load. This technique will not require any monitoring mechanism since it is encoded in the network structure. Load-balancing is achieved without the need to monitor the nodes for

their resources availability. Authors K. Saruladha and G. Santhi in paper [16], presents distributed agent based approach that improves the response time of the user submitted jobs, overall execution time required for the completion of the submitted jobs is found to decrease and also explore and find the under loaded nodes is done more quickly.

In paper [17], author's present decentralized LB algorithm, the goal of this algorithm is to allocate the available channels to the agents in a balanced way such that the overall system achieves the best possible performance. Many other channel allocation algorithms require a priori information on whether the channel allocation is done with overlapping or non-overlapping channels, in this approach there is no such issue, as the method is not based on explicit interference models. But this algorithm is strictly locally executed algorithm.

2.1. Dynamic Load Balancing with Multiple Supporting Nodes (MSN)

This paper describes a centralized model for dynamic load balancing with Multiple Supporting Nodes. Objective of this algorithm is to reduce the communication delay and traffic up to some extent, and this is obtain because load is transfer within cluster itself (from primary node to supporting node) [6]. Further detail of this algorithm is given below:

Three approaches are mentioned in this algorithm, primary, centralized, and modified approach. In primary approach, initially processes are stored in queue or process can be allotted as they arrive. If these are placed in queue, processes are allotted one by one to primary nodes. Processes are migrated from heavily loaded node to light weighted node. First a light weighted node is checked in the same cluster, if suitable node not found then after nearby cluster is searched and after getting a required node transfer takes place if a protocol is satisfied for load transfer. Whereas in centralized approach one centralized node is provided in each cluster. Whenever a primary node is over loaded, first it search the other light weighted primary nodes, if such primary node is available, load transfer take place between these two node and load is balanced, otherwise if such light weight node is not available, one centralized node is available to accommodate the overload of a primary node. In Centralized approach there is single node, so process the load at high speed by using switching but still a limitation is there. An approach is there i.e. modified approach to remove the limitation is to split the centralized node into small nodes called supporting nodes (SNs). Suppose a process Pi is currently executed by SNi and a Primary node Ni is overloaded so that it finds a supporting node SNi suitable for transferring its overload [5].

2.2. Recent Neighbor Load Balancing Algorithm (RNLBA)

The problem of load balancing in grid architecture is addressed by assigning loads in a grid without neglecting the communication overhead in collecting the load information. An efficient dynamic load balancing algorithm named as 'Recent Neighbor' (RN) has been presented to tackle the above challenges. RN performs intra-cluster and inter cluster (grid) load balancing [4].

In the work [4], the authors logically divided the grid architecture into three levels: Grid-Level, Cluster-Level and the Leaf-Nodes. The clusters in the grid are fully interconnected. Each cluster may contain multiple computing nodes called as leaf nodes. The computing nodes in the cluster are heterogeneous in nature. The processing power of the grid cluster is measured by the average CPU speed across all computing nodes within the grid cluster. The tasks are assumed to be computationally intensive, mutually independent and can be executed at any cluster. No deterministic or priori information about the task is available.

It performs two level load balancing algorithms, in Cluster-Level load balancing, depending on the current workload of its associated cluster, estimated from its own neighbors, each Cluster-Level manager (CM) decides whether to start or not a load balancing operation. If it decides to start a load balancing operation, then it tries to load balance the workload among its under-loaded neighbors.

In Grid-Level load balancing, the load balancing is performed only if CM fails to load balance their workload among their associated neighbors. In this case, tasks of overloaded clusters are transferred to under loaded ones regarding the communication cost and according to the selection criteria. The chosen under loaded clusters are those which need minimal communication cost for transferring tasks from overloaded clusters.

2.3. A Load Balancing Policy for Heterogeneous Computational Grids (LBPHCG)

The proposed policy tends to improve grid resources utilization and hence maximizes throughput [8]. This paper focuses on the steady-state mode, where the number of jobs submitted to the grid is sufficiently large and the arrival rate of jobs does not exceed the grid overall processing capacity. The class of problems addressed by the proposed load balancing policy is the computation-intensive and totally independent jobs with no communication between them. This paper describes a two-level load balancing policy for the multi-cluster grid environment i.e. Local Grid Manager and Site Manager Load Balancing.

Local Grid Manager (LGM) Load Balancing Level

The total processing capacity of a LGM is Local grid manager Processing Capacity (LPC) which is the sum of all the Site Processing Capacity (SPC) for all the sites managed by that LGM. Based on the total processing capacity of every site SPC, the LGM scheduler distributes the workload among his sites group members (SMs). The ith site workload (Si WL) which is the number of jobs to be allocated to ith site manager is obtained as follows:

$$S_i WL = N (SPC_i / LPC)$$

Where, N is number of jobs arrived at a LGM, SPC_i is number of jobs that can be executed by the i^{th} site/sec, and LPC is number of jobs that can be executed under the responsibility of the LGM/sec.

Site Manager Load Balancing Level

The total site processing capacity SPC is obtained by summing all the Processing Element Capacities (PECs) of all the processing elements in that site. The i^{th} PE workload (PE_i WL) which is the number of jobs to be allocated to i^{th} PE is obtained as follows:

$$PEi\ WL = M\ (PECi / LPC)$$

Where M is the number of jobs arrived at a SM, (PEC_{ij}): Number of jobs that can be executed by the j^{th} PE at full load in the i^{th} site per second.

2.4. Grid Load Balancing using Intelligent Agents (IA)

This paper focuses on Grid load balancing with intelligent agents and multi-agent approaches. These approaches are used to schedule local Grid resources and do global Grid load balancing [10].

A Grid resource can be a multiprocessor or a cluster of workstations. An agent is at the Grid level a presentation for a Grid resource offering services and a high performance computing power. Agents are the high-level abstraction of a Grid resource. Each agent consists of 3 main layers, from bottom to top: communication, coordination and local management layer. The latter performs functions of an agent for local Grid load balancing. The coordination layer treats requests and organizes the local knowledge. The communication layer enables to interact with other agents. The agents use the PACE performance prediction engine [10]. PACE is a tool set for performance prediction in Parallel and Distributed Systems. The algorithms are developed in two different scopes:

- Local Grid Load Balancing
- Global Grid Load Balancing

In the first, a local Grid resource is considered to be a cluster of workstations or a multiprocessor. The authors show 2 different algorithms for local Grid load balancing; first algorithm is first-come-first-served, in their second algorithm authors uses a genetic algorithm where the goal is to minimize the latest completion time when all tasks are considered together. The second deals with the Grid load balancing. The problem that is addressed in this algorithm is how the discovery of available Grid resources that provides the optimum execution performance for a globally submitted task.

2.5. Decentralized Genetic Algorithm (DGA)

In this, authors directed their research towards speeding up the convergence of genetic algorithms by using multiple agents and different populations to schedule sets of tasks. The use of multiple initial search points in the problem space favors a high probability to converge towards a global optimum. Combined with the lookup services, this approach offers a solution to high scalability and reliability demands [2]. DGA is summarized below:

Authors uses SAGA Model, in this model, users submit Scheduling requests. A near-optimal schedule is computed by the Scheduler based on the Scheduling requests and the Monitoring data provided by the Grid Monitoring Service (MonALISA) [3]. The schedule is then sent as a Request for task execution to the Execution Service. The user receives feedback related to the solution determined by the scheduler, as well as to the status of the executed jobs in the form of the Schedule and task information. Furthermore, the system can easily integrate new hosts in the scheduling process, or overcome failure situations by means of the Discovery Service.

2.6. Prediction Based Technique (PBT)

In this paper authors present Load Balancing technique that can deal with applications with heterogeneous cluster that reduce the average response time [6]. They are considering three type of load I/O, CPU, MEMORY. In this paper authors considered a cluster computing platform of heterogeneous system in which, a load manger or master node is responsible for load balancing and monitoring available resources of the node. Load manger is composed of three modules: (1) predictor; (2) selector; (3) scheduler; Predictor is used to predict the file I/O, CPU and memory requirements of a task, for this author uses a statistical pattern-recognition method.

The prediction is a weighted mean calculation of resource requirements using the program's current state-transition model and the actual resource usage in its most recent execution. Then predicted value is fed to the selector that is used to select the best node among all nodes where the task will execute. Scheduler is responsible to dispatch the task to the node selected by the selector. Then task will send to that node and task will execute there. Load manager update the load status table. Preemptive migrations of tasks are not supported by this algorithm.

3. ANALYSIS OF VARIOUS LOAD BALANCING ALGORITHM

Algorithm described in section II A, is reduces the communication cost, because load is transferred locally. However drawback of this algorithm is at very initial phases utilization of supporting nodes is decreases. In recent neighbors (described in section II B), is provides shorter response time, enhances the resource utilization and balances the load in an effective manner. In future, more complex models such as nesting of clusters need to be investigated.

Algorithm LBPHCG described in section II C, is attempted to minimize the overall job mean response time and maximize the system utilization and throughput. Communication is needed only if a processing element joins or leaves its site. Intelligent Agent based approach (described in section II D) has an advantages of the evolutionary algorithm is that it is adaptive to changes in the system. It absorbs changes such as addition or deletion of tasks or changes in the number of hosts. However, this algorithm cannot be employed for a large scale, since complexity increases exponentially with the number of hosts.

Prediction Based technique (PBT described in section II F) aim to achieve the effective usage of global disk resources in cluster. This can minimizes the average slow down of all parallel jobs running on a cluster and reduce the average response time of the jobs. All these algorithms are summarized in TABLE 1 given below. Column 1 of this table representing the abbreviation of algorithms described in section 2 e.g. RNLBA for recent neighbors etc.

Table 1 Comparative Analysis of Existing Load Balancing Algorithms in Grid Environment

S.N.	Model	Compared Model	Strength	Contribution/ Research focus/ Features	Drawback
MSN	Centralized	Primary and centralized approach.	Minimum traffic due to attached central node at each cluster.	Load is transfer within the cluster itself (from Primary node to Supporting node) so communication delay and traffic is reduced up to some extent.	Initially process utilization at each supporting node (SN) is less.
RNLBA	Distributed	Tested by taking of 3 clusters each containing 2 nodes.	The parameters measured are average response time, system load and communication delay.	It provides shorter response time, enhances the resource utilization.	Need to make this algorithm for complex model nested of cluster.
IA	Decentralized (grid computing environment : two-level load balancing policy)	1) Compared with the Random distribution LB & Uniform distribution LB policy.	It considers load index as a decision factor for scheduling of jobs in a cluster and among clusters.	Communication is needed only if a processing element joins or leaves its site.	This method: 1) Not suited for dependent jobs. 2) Does not consider data intensive jobs.
LBPHCG	Distributed (grid environment)	Tested by taking 12 nodes in hierarchy. Performance total application execution time, resource utilization are measured.	1) Uses a combination of both intelligent agents and multi-agent approaches. 2)The algorithm is based on an evolutionary process therefore able to absorb system changes	Dynamically minimize task makespan and host idle time, while meeting the deadline requirements for each task.	Can do further extension of the agent framework with new features, e.g. automatic QoS negotiation, self-organizing coordination etc.
DGA	Decentralized:(Heterogeneous environment)	1) The experimental results show that the number of generations necessary for the algorithm to converge is significantly reduced.	1) It focused on classes of independent tasks, which avoids communication costs due to dependencies. 2) It presents SAGA (”Scheduling Application using Genetic Algorithms”).	1) Accomplishment of this research is the migration towards a decentralized scheduler by means of lookup services. 2) It also speeding up the convergence of genetic algorithms by using multiple agents and different populations to schedule sets of tasks.	Cannot handle data intensive program.
PBT	Centralized		Better resource utilization and reduce the average job response time.	1) The objective this algorithm is to balance the load of three types i.e. I/O-intensive, CPU-intensive and memory intensive load.	1) Not suitable for inter-dependent task. 2) It assumed that the network communication cost is negligible.

3. CONCLUSION

In this survey paper we analyzed different solutions and compared them to each other by considering their drawback. The researchers can use these facts to develop better algorithms. In the above study it is found that some algorithms does not specified memory requirement of the jobs while submitting the jobs to the selected resources and some of algorithm does not considered communication cost, which we cannot neglect. Memory requirement of a job is vital in completing the execution of jobs at the selected resources within a time bound in realizing a real grid system.

REFERENCES

1. Ali M. Alakeel, “A Guide to Dynamic Load Balancing in Distributed Computer Systems”, IJCSNS International Journal of Computer Science and Network Security, VOL.10 No.6, June, 2010.
2. George V. Iordache, Marcela S. Boboila, Florin Pop, Corina Stratan, and Valentin Cristea , “A Decentralized Strategy for Genetic Scheduling in Heterogeneous Environments”, Springer-Verlag Berlin Heidelberg, 2010.
3. I.C.Legrand, H.B.Newman (2003), “Monalisa: An Agent Based, Dynamic Service System to Monitor,Control and Optimize Grid Based Applications”, European Center for Nuclear Research – CERN.
4. Jasma Balasangameshwara, Nedunchezhian Raju, “A Decentralized Recent Neighbour Load Balancing Algorithm for Computational Grid”,The International Journal of ACM Jordan (ISSN 2078-7952), Vol. 1, No. 3, September, 2010.
5. M. Kamarunisha, S.Ranichandra, T.K.P.Rajagopal , **“**Recitation of Load Balancing Algorithms In Grid Computing Environment Using Policies And Strategies - An Approach**”** International Journal of Scientific & Engineering Research, 2011.
6. Parveen Jain, Daya Gupta, “An Algorithm for Dynamic Load Balancing in Distributed Systems with Multiple Supporting Nodes by Exploiting the Interrupt Service”, ACEEE, ACADEMY PUBLISHER, Delhi College of Engineering, New Delhi, 2009.
7. Pushpendra Kumar Chandra, Bibhudatta Sahoo, “ Prediction Based Dynamic Load Balancing Techniques in Heterogeneous Clusters”, National Institute of Technology Rourkela, Orissa, 2008.
8. Sandeep Singh Waraich, “Classification of Dynamic Load Balancing Strategies in a Network of Workstations”, Fifth International Conference on Information Technology: New Generations, 2008.
9. Stephen A. Jarvis, Daniel P. Spooner, Junwei Caoa, Graham R. Nudd, “Grid load balancing using intelligent agents”, Elsevier B.V., 2004.
10. Stephen A. Jarvis, Daniel P. Spooner, Helene N. Lim Choi Keung, Graham R. Nudd, “Performance Prediction and its use in Parallel and Distributed Computing Systems” in Proceedings of the International Parallel and Distributed Processing Symposium, 2003.
11. Tal Maoz, Amnon Barak, Lior Amar, “Combining Virtual Machine Migration with Process Migration for HPC on Multi-Clusters and Grids” in IEEE International Conference on Cluster Computing, The Hebrew University of Jerusalem, Israel, 2008 .
12. Hans-Ulrich Heiss, Michael Schmitz, “Decentralized Dynamic Load Balancing: The Particles Approach”, Department of Informatics, University of Karlsruhe, Germany, 1999.
13. Ashok Singh Saira, Gautam Barua, “Distributed Route Control Schemes to Load Balance Incoming Traffic in Multihomed Stub Networks”, IEEE ,Dept. of Computer

Science and Engineering Indian Institute of Technology Patna & Guwahati, 2010.
14. Md. Abdur Razzaqu, Choong Seon Hong, "Dynamic Load Balancing in Distributed System: An Efficient Approach", Korea, 1991.
15. M. Randles , Abu-Rahmeh , P. Johnson, A. Taleb-Bendiab, "Biased random walks on resource network graphs for load balancing", Springer,dec. 2009.
16. K. Saruladha, G. Santhi, "Behavior of Agent Based Dynamic Load Balancing Algorithm for Heterogeneous P2P Systems", CSE Dept., Pondicherry Engineering College, Pondicherry, International Conference on Computational Intelligence and Multimedia Applications, 2007.
17. Marina Petrova, Natalia Olano and Petri Mahonen, "Balls and Bins Distributed Load Balancing Algorithm for Channel Allocation" in IEEE, RWTH Aachen University Aachen, Germany, 2010.
18. V. Mani, D. Ghose, and L. Anand, "ELISA: An Estimated Load Information Scheduling Algorithm for Distributed Computing Systems" in Int'l J. Computers and Math. with Applications, vol. 37, no. 8, pp. 57-85, Apr. 1999.

Note: This Paper/Article is scrutinised and reviewed by Scientific Committee, BITCON-2015, BIT, Durg, CG, India