A Study of Load Balancing in Cloud Computing using Soft Computing Techniques

Akhil Goyal
Centre for Development of Advanced Computing
(C-DAC), Mohali, Punjab, India.

Bharti
Mohali,
Punjab, India.

ABSTRACT
Cloud computing is a business oriented approach which involves collaboration of multiple computing technologies via internet. With the rapid increase in cloud usage, it becomes a challenge to deliver the services effectively and efficiently as per client's demand. In this concern Load Balancing has become one of the major key areas for research. There are a number of soft computing techniques available to optimize the load. In this paper, those techniques are investigated. We will discuss and compare these soft computing algorithms to provide an overview of the proposed approach i.e. Particle Swarm Optimization (PSO).

General Terms
Load balancing techniques, Cloud computing, Algorithms, Virtual machines, Green Computing.

Keywords
Cloud Computing, Load Balancing, Soft Computing Techniques, PSO.

1. INTRODUCTION
Recently Cloud Computing has become one of the popular research field adopted by both academic world and industry. Prime focus of this technology is to distribute computing resources and services online. Here, store and computing services are purchased on demand by the user. Cloud resources are not only shared but also reallocated at run time. End user knowledge about the configuration of service delivering system and resource management is not required as this aspect is to be handled by cloud system automatically. So, a number of distributed host machines are grouped in a cloud. Cloud computing system constitutes several servers, virtual machines, datacenters and storage devices etc as resources; interconnected in a reliable approach. Whenever there is any demand from any user of the cloud then cloud system creates a virtual machine inside any host machine from that cloud to fulfill the clients demand in the form of resources on pay per use criteria. Due to this reason each host machine has variable load as virtual machines are created randomly on client's demand. Some host machines may get overloaded and some remain light-weighted. This load may be the CPU load, memory load, storage load or network related load. Now Load balancing ensures distribution of cloud resources efficiently and effectively among running cloud services. Load from over-weighted hosts is migrated to light-weighted host using any one among many types of soft computing algorithms.

In this paper, a summarized overview of various soft computing techniques is presented; that have been previously used for load balancing in cloud computing environment. These techniques are then compared and evaluated on the basis of some fixed parameters. A single optimization technique would be preferred over the others and will be proposed for the Load Balancing scenario.

Rest of the Paper is organized as follows. Previously used soft computing techniques for Load Balancing are reviewed in Section II. Then, these are compared on the basis of fixed parameters in section III. A preferred technique is proposed in section IV. This section will also give a brief overview of the proposed technique. Finally the conclusion and future work is given in Section V.

2. REVIEW OF PREVIOUSLY USED LOAD BALANCING TECHNIQUES
In this section, discussion is focused on the most preferred researches in the literature for load balancing in cloud computing. We are going to discuss these techniques year wise to evaluate the fixed parameters. This will help us to compare these techniques and conclude an optimized one.

Shridhar G. Damanal et al. introduced a modified throttled algorithm for load balancing in cloud computing [1]. This algorithm concerns with the fact that how incoming jobs are assigned to the available virtual machines effectively and efficiently. This algorithm works on the grounds of throttled algorithm by maintaining an index table of virtual machines and their states. In this modified algorithm an attempt is made to improve the response time and achieve efficient usage of available virtual machines. VM is initially selected according to the state of VM. If VM is available request is approved else -1 is returned to datacenter. At the next request, VM at index next to already assigned VM is chosen. The two algorithms are different as rather than returning the id or -1 the index table is parsed from first index every time in case of basic throttled algorithm.

Seyed Mohssen Ghafari et al. proposed a load balancing algorithm for power consumption management in cloud computing and named this algorithm Bee-MMT (artificial bee colony algorithm-Minimal Migration Time) [2]. This algorithm use Artificial Bee Colony algorithm (ABC) to detect over-weighted hosts. Then it use MMT algorithm to transfer one or more virtual machines from those over weighted hosts to decrease their load. In the meantime it can detect under-weighted hosts and if possible transfer all virtual machines allocated to these hosts and then toggle them to the sleep mode.

Yatendra Sahu et al. proposed a dynamic compare and balance algorithm to optimize cloud data center in order to balance the host machine [3]. This algorithm primarily focuses on load balancing of cloud datacenters to increase efficiency of host machine. It adds a new concept known as green computing concept by reducing number of active host machines. For load balancing of entire data center, virtual machines of overloaded hosts are migrated to light weighted hosts using migration techniques. DCABA, in support with green computing concept, reduces the number of host machines to be activated, for reducing the cost of cloud services. Two concepts of cloud optimization are used. One,

optimize the cloud system at host machine level and the other, to optimize the cloud system using dynamic threshold values. Threshold values are calculated at runtime using total capacity of the server multiplied by their weight coefficients respectively. Three parameters for threshold values are used: Host_Limit, Upper_Threshold_Value_Of_Host (H_UTD), Lower_Threshold_Value_Of_Host (H_LTD). When load of host is more than the H_UTD host is considered as being overloaded. When load is below H_LTD host is considered as under loaded.

Hunkai Chen et al. proposed a User-Priority Guided Min-Min scheduling algorithm for load balancing in cloud computing. Authors modified the basic Min-Min scheduling algorithm by improving the load imbalance of the Min-Min to reduce the execution time of each resource effectively. They named this improved algorithm as Load Balanced Improved Min-Min (LBIMM) scheduling algorithm [4]. LBIMM has the capability to obtain a schedule which improves a load balancing and in addition it also reduces the overall completion time. Authors also extended LBIMM to User Priority Aware-LBIMM (PA-LBIMM) algorithm by considering the user priority between the tasks and resources based on LBIMM.

Kumar Nishant et al. proposed an efficient algorithm by updating actual Ant Colony Optimization (ACO) algorithm in their own way for load balancing of nodes in cloud environment [5]. The standard ACO algorithm is modified in the way that ants continue to update a single result set rather than updating their own result set. In this algorithm a Regional Load Balancing Node (RLBN) is elected to act as a head node. Choice of head node is also critical. It might be elected in such a way that it links with maximum number of nodes as immediate neighbors. This will provide other ants maximum possible routes to traverse. Other ants consider head node as the root; which means they will update the single result set obtained from the head node. In this approach, the ants never reach a dead end. Adding to this, once head node is elected; doesn't mean that now it is permanent. Selection can be reset if the past selection stops functioning efficiently due to some inappropriate conditions.

Al-Jaroodi et al. proposed a Dual Direction Downloading algorithm from FTP servers (DDFTP) for cloud computing load balancing [6]. DDFTP works by dividing a file of size m into m/2 partitions. Now each server node can work independently on these two partitions one in the incremental order while other in a decrement order. Along with load balancing this algorithm minimizes the extent of network communication needed between the clients and nodes resulting in reduced network overhead. It also works on other parameters such as network load, node load, network speed etc.

S-C Wang et al. proposed an algorithm called Load Balancing Min-Min (LBMM) to balance the load of nodes in a cloud [7]. This algorithm works on the grounds of Opportunistic Load Balancing (OLB) algorithm. The difference lies in the fact that OLB has nothing to do with the execution time of the node; results in the tasks to be processed in a slower manner and requests might be pending waiting for nodes to be free. LBMM is an improved version with a three level load balancing framework. First level is request manager for receiving the tasks and allocating it to second level i.e. service manager. Service manager divides the task into sub tasks to speed up the processing. Service manager assign the sub task to service node sitting at third level actually responsible for executing the task.

3. DISCUSSION AND COMPARISON

In this section we discuss and compare various algorithms that were reviewed in Section II. Algorithms in previous section are discussed in descending order on yearly basis. So, obviously the first algorithm facilitates some advantages as compared to the last one. Different approaches offer specific solutions and also suits some particular situations but other don't.

Table 1 shows a comparison among the different load balancing techniques. The comparison shows the pros and cons of each soft computing technique. For example, Modified throttled algorithm evaluates the results in two aspects. One is load balancing of virtual machines and other response time for the execution of the client's request. Authors have successfully implemented the algorithm using five virtual machines.

Bee-MMT technique is evaluated on the basis of energy consumption or CPU utilization, PDM (Performance Degradation due to Migrations), and number of VM migrations. DCABA primarily focuses on minimizing the cost of cloud services but appropriate threshold values should be selected to obtain optimized results. Here, number of active host machines is reduced to great extent in order to maintain efficient consumption of resources and conservation of energy. This criterion strictly supports the green computing concept.

Using LBIMM and PA-LBIMM authors concluded Makespan, Average resource utilization ratio, Average VIP task completion time, and average ordinary task completion time. In every case both the algorithms proved better than basic Min-Min algorithm but average ordinary task completion time results degraded in later algorithms. In ACO technique, a single node is chosen as head node and all the ants traverse the path from this head node. If the head node fails due to any inevitable circumstances, it can be exchanged with some other node having the matching capability to be the head node. So this algorithm provides fault tolerance but also increases the overhead. Selection of head node is also a critical scenario.

DDFTP algorithm is easy to handle as no run time monitoring is required. This algorithm is efficient approach for load balancing but still needs some improvements for better utilization of resources. The major drawback of DDFTP is that it introduces high overhead on the network. LBMM is improved version of OLB which provides reliability during task assignment, maintains uniformity in distributing load. The major drawback is that this algorithm is much slower than other algorithms.

Table 2 demonstrates a comparison of reviewed algorithms on basis of fixed parameters. For example the only algorithm that incorporate network overhead is the ACO algorithm due to large number of ants. Among seven algorithms reviewed, the only algorithm having average resource utilization is DDFTP due to requirement of high storage capacity in all nodes.

4. PROPOSED APPROACH: (PSO)

This section includes a review of several algorithms which focus on load balancing in cloud computing. These reviewed algorithms have also been used in other research areas such task scheduling, grid Computing, distributed computing etc. It is already known that task assignment, load balancing has been found to be NP-complete problems. In earlier days genetic algorithm has been considered as the best method to solve these NP-complete problems but now it has been proved

that the particle swarm optimization algorithm is able to get the better schedule than genetic algorithm. L. Zhang [8] has applied PSO algorithm in grid computing and has got the better results. PSO algorithm has also been proved better than ACO in distributed system [9]. Here, this algorithm not only improves the quality but also run faster than ACO. So, a method called Particle Swarm Optimization is suggested to optimize the load balancing problem in cloud computing.

This section includes the introduction of PSO in brief. The PSO is inspired by the behavior of bird flocking or fish flocking which randomly search food through the search space. During their search the swarm population changes direction, scatters, regroups, till they achieve the target. The single entity either bird or fish is considered as a particle having a velocity vector and a position vector. Each particle randomly move and change direction according to the velocity and position in the search space at a particular instant. Each particle has a fitness value, to be evaluated by a fitness function.Two parameters are considered in evaluation of PSO. One is Pbest i.e. the best position of the particle which it has gained so far and other Gbest i.e. best value obtained by a particle in the population. During their searching each particle adjusts their direction and speed on the basis of following parameters such as current position, velocity, Pbest and Gbest. Performance is measured using fitness function.

An attempt for dynamic load balancing to a cloud based call centre using swarm intelligence algorithm has also been tried [10]. They created a pattern, called SILBA (Self Initiative Load Balancing Agents) for intelligent load balancing policies. Basically, SILBA is a framework that comprises different types of intelligent and unintelligent algorithms.

5. CONCLUSION AND FUTURE WORK

In this paper we studied various algorithms for load balancing in cloud computing. This paper also discusses the pros and cons of these algorithms. Then, a comparison of these algorithms on the basis of fixed parameters is done. A study about the working of PSO algorithm in different research areas is also accomplished. Every time PSO gave better results. This study focus our vision on the aspect that PSO can be used to optimize load balancing in cloud computing. Therefore, In future work, we are planning to optimize PSO to make it suitable for cloud environments and more efficient in terms of load balancing. Additionally, this research work can also be exaggerated by implementing the optimization of PSO on various cloud simulators and compare the proposed approach with previously tested soft computing techniques based on some fixed parameters.

Table 1. Pros and Cons of Reviewed Load Balancing Algorithms

	Pros.	**Cons.**
Modified throttled	Response time has improved considerably as compared to Round Robin and basic throttled algorithms.	Distributes load nearly uniform among VM's but it can be further optimized by using paradigms of parallel computing.
Bee-MMT	It can be a green solution as operational cost is reduced to high extent.	It has SLA violations and also performance degradation occur due to migrations.
DCABA	Presents better results for cloud server optimization based on both load balancing and server consolidation	Selection of appropriate thresh hold values is required. Better results are obtained in the lower and upper value [0.15-0.85 respectively]
PA-LBIMM	Decreases the total completion time, the Makesspan of tasks, and increases average resource utilization ratio by 20 %. Average completion time is reduced by 4.38s compared with LBMM.	Results in tradeoff between VIP tasks and ordinary tasks. Average ordinary completion time is increased by 1.83s compared with LBMM.
ACO	Under loaded node is found at the beginning. New head node can be elected if previous nodes stop functioning properly.	Network over head due to large number of ants. Choice of head node is crucial. Nodes status change after ants visit to them is not considered.
DDFTP	Fast and reliable downloading of files is possible.	Complete replication of data files is there therefore; require high storage capacity in all nodes.
LBMM	Effective and efficient task assignment to nodes.	Slower than LBIMM because tasks pass through three level architecture.

Table 2. Comparison of reviewed algorithms on basis of fixed parameters

	Network overhead	Replication of Files	Resource Utilization	Implementation Complexity	Response Time	Fault Tolerance
Modified Throttled	NO	FULL	HIGH	LOW	FAST	YES
Bee-MMT	NO	FULL	HIGH	MODERATE	FAST	YES
DCABA	NO	FULL	HIGH	LOW	FAST	YES
PA-LBIMM	NO	FULL	HIGH	LOW	MODERATE	YES
ACO	YES	FULL	HIGH	LOW	FAST	YES
DDFTP	NO	FULL	AVERAGE	LOW	FAST	YES
LBMM	NO	FULL	HIGH	LOW	SLOW	YES

6. REFERENCES

[1] Shridhar G. Domanal, G. Ram Mohana Reddy, "Load Balancing in Cloud Computing Using Modified Throttled Algorithm," in proc. International Conference on Cloud Computing in Emerging Markets (CCEM), IEEE, pp. 1-7, October 2013.

[2] Seyed Mohssen Ghafari, Mahdi Fazeli, Ahmad Patooghy, Leila Rikhtechi, "Bee-MMT: A Load Balancing Method for Power Consumption Management in Cloud Computing," in proc. 6th International Conference on Contemporary Computing (IC3), IEEE, pp. 76-80, August 2013

[3] Yatendra Sahu, R.K. Pateriya, Rajeev Kumar Gupta, "Cloud Server Optimization with Load Balancing and Green Computing Techniques Using Dynamic Compare and Balance Algorithm," in proc. 5th international Conference on Computational Intelligence and Communication Networks, IEEE, pp. 527-531, 2013.

[4] Huankai Chen, Proff. Frank Wang, Dr. Nahelian, Gbola Akanmu, "User-Priority Guided Min-Min Scheduling algorithm for Load Balancing in Cloud Computing," in proc. National conference on Parallel Computing Technologies (PARCOMPTECH), IEEE, pp. 1-8, February 2013

[5] Kumar Nishant, Pratik Sharma, Vishal Krishna, Chavvi Gupta Kuwar Pratap Singh, Nitin and Ravi Rastogi, "Load Balancing of Nodes in Cloud Using Ant Colony Optimization," in proc. 14th International Conference on Modelling and Simulation, IEEE, pp. 3-8, July 2012.

[6] Al-Jaroodi, J. and N. Mohamad, "DDFTP: Dual-Direction FTP," In proc. 11th IEEE/ACM International symposium on Cluster, Cloud and Grid Computing (CCGrid), IEEE, pp. 504-513, May 2011.

[7] Wang, S-C., K-Q. Yan, W-P. Liao and S-S. Wang, "Towards a load balancing in a three-level cloud computing network," in proc. 3rd International conference on Computer Science and Information Technology (ICCSIT), IEEE, Vol. 1, pp. 108-113, July 2010.

[8] L. Zhang, Y.H. Chen, R.Y. Sun, S. Jing, B. Yang, "A Task Scheduling Algorithm Based on PSO for Grid Computing," International Journal of Computational Intelligence Research, pp. 37-43, 2008.

[9] A. Salman, "Particle Swarm Optimization for Task Assignment Problem," Microprocessors and Microsystems, pp. 363-371, Nov 2002.

[10] Vesna Sesum-Cavic, Eva Kuhn, "Applying Swarm Intelligence Algorithms for Dynamic Load Balancing to a Cloud Based Call Center," in proc. Fourth International Conference on Self-Adaptive and Self-Organizing Systems, IEEE, pp. 255-256, 2010.