A Survey on Load Balancing Techniques in Cloud Computing

Ekram M. Rewehel
Master student in Arab Academy for Science,
Technology & Maritime Transport
Cairo – Egypt

Mostafa-Sami M. Mostafa
Professor of Computer Science,
Helwan University
Cairo – Egypt

***Abstract*-Nowadays, cloud computing has become a key technology for online allotment of computing resources and online storage of user's data in a lower cost, where computing resources are available all the time, over the internet with pay per use concept. Cloud computing is business oriented concept where computing resources are outsourced by cloud provider to their client, load balancing is required to distribute the dynamic local workload evenly across all the nodes. Load balancing is the process of distributing the load among various nodes of a distributed system to improve both resource utilization and job response time while also avoiding a situation where some of the nodes are heavily loaded while other nodes are idle or doing very little work. The load can be CPU load, memory capacity, or network load. This paper discusses the entire existing load balancing algorithms in cloud computing and also their challenges. Our objective is to explain the concept of load balancing, types of load balancing algorithms, general idea about dynamic load balancing algorithms and metrics that can be used in it. An overall description of various distributed load balancing algorithms that can be used in cloud computing is also presented**

Keywords – Cloud Computing, Load Balancing, Distributed Systems, Load balancing algorithms.

1. INTRODUCTION

In the field of information technology, cloud computing is a recent trending that moves computing and data away from desktop and portable computers into large data centers [1]. Cloud computing allows everyone to use software and computing services on-demand at anytime, anywhere and anyplace through the internet. Cloud computing mainly deals with computation, software, data access and storage services that may not require end-user knowledge of the physical location and configuration of the system that is delivering the services [2]. The definition of cloud computing provided by National Institute of Standards and Technology (NIST) says that: "Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, data storage, software applications and other computing services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [3]". By sharing of resource the overall cost reduces.

Load balancing can help in reducing energy consumption by evenly distributing the load and minimizing the resource consumption. This paper focuses on the prevalent load balancing techniques in cloud computing environment. Load balancing is one of the central issues in cloud computing [5]. It is a mechanism that distributes the dynamic local workload evenly across all the nodes in the whole cloud to avoid a situation where some nodes are heavily loaded while others are idle or doing little work. It helps to achieve a high user satisfaction and resource utilization ratio, hence improving the overall performance and resource utility of the system. It also ensures that every computing resource is distributed efficiently and fairly [13]. It further prevents bottlenecks of the system which may occur due to load imbalance. When one or more components of any service fail, load balancing helps in continuation of the service by implementing fair-over, i.e. in provisioning and re-provisioning of instances of applications without fail. It also ensures that every computing resource is distributed efficiently and fairly [5] [9].

Consumption of resources and conservation of energy are not always a prime focus of discussion in cloud computing. However, resource consumption can be kept to a minimum with proper load balancing which not only helps in reducing costs but making enterprises greener [6] [13]. Scalability which is one of the very important features of cloud computing is also enabled by load balancing. Hence, improving resource utility and the performance of a distributed system in such a way will reduce the energy consumption and carbon footprints to achieve Green computing [6] [10] [12]. The motivation of the survey of existing load balancing techniques in cloud computing is to encourage the amateur researcher to contribute in developing more efficient load balancing algorithms. This will benefit interested researchers to carry out further work in this research area.

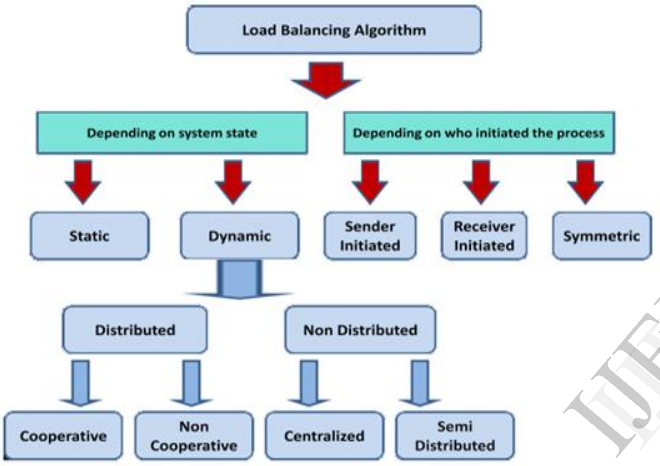
This paper is organized as follows: Section II present the need of load balancing in clouds, Section III discusses types of load balancing algorithms, Section IV shows the study and analysis of the existing load balancing techniques in cloud computing, Section V identifies the metrics considered in the existing load balancing techniques and carries out the comparison between them based on those identified metrics and Section VI concludes the paper. To the best of our knowledge, none of the techniques has focused on energy consumption and carbon emission factors that are a dire need of cloud computing.

2. LOAD BALANCING IN CLOUD

Load balancing in clouds is a mechanism that distributes the excess dynamic local workload evenly across all the nodes. It is used to achieve a high user satisfaction and resource utilization ratio [8], making sure that no single node is overwhelmed, hence improving the overall performance of the system. Proper load balancing can help in utilizing the available resources optimally, thereby minimizing the resource consumption. It also helps in implementing fail-over, enabling scalability, avoiding bottlenecks and over-provisioning, reducing response time etc.

Figure (1) classification of load balancing algorithm [4]

Load balancing is the process of reassigning the total load to the individual nodes of the collective system to make effective resource utilization and to improve the response time of the jobs, simultaneously removing a condition in which some of the nodes are over loaded while some others are under loaded. Depending on system state, load balancing algorithms divided into two types as static and dynamic. A load balancing algorithm which is dynamic in nature, does not consider previous state or behavior of the system, that is, it depends on the present behavior of the system. Depending on who initiated the process, load balancing algorithms can be divided into three types as sender Initiated, receiver Initiated and symmetric.



The important things to consider while developing such algorithm are : estimation of load, comparison of load, stability of different system, performance of system, interaction between the nodes, nature of work to be transferred, selecting of nodes and many other ones. This load considered can be in terms of CPU load, amount of memory used, delay or Network load.

3. LOAD BALANCING ALGORITHM

3.1 Static

This approach is generally defined in the design or implementation of the system. Static algorithms divide the traffic equivalently between servers. It doesn't depend on the current state of the system. Prior knowledge of the system is needed. By this approach the traffic on the servers will be disdained easily and consequently it will make the situation more imperfectly.

3.1.1 Static Load Balancing Algorithm

- Round Robin Algorithm: In this algorithm [11], the processes are divided between all processors. Each process is assigned to the processor in a round robin order. The process allocation order is maintained locally independent of the allocations from remote processors. Though the work load distributions between processors are equal but the job processing time for different processes are not same. So at any point of time some nodes may be heavily loaded and others remain idle. This algorithm is mostly used in web servers where Http requests are of similar nature and distributed equally.

- Randomized Algorithm: Randomized algorithm is of type static in nature. In this algorithm [11] a process can be handled by a particular node n with a probability p. The process allocation order is maintained for each processor independent of allocation from remote processor. This algorithm works well in case of processes are of equal loaded. However, problem arises when loads are of different computational complexities. Randomized algorithm does not maintain deterministic approach. It works well when Round Robin algorithm generates overhead for process queue.

3.2 Dynamic

This approach takes into account the current state of the system during load balancing decisions. No prior knowledge is needed. This approach is more suitable for widely distributed systems such as cloud computing. On servers and by searching in whole network a lightest server preferred to balance the traffic. However, selecting an appropriate server needed real time communication with the networks, which will lead to extra traffic added on system. So it is better than static approach. Here we will discuss on various dynamic load balancing algorithms for the clouds of different sizes.

3.2.1 Dynamic Load Balancing Algorithm

In a distributed system, dynamic load balancing can be done in two different ways: distributed and non-distributed.

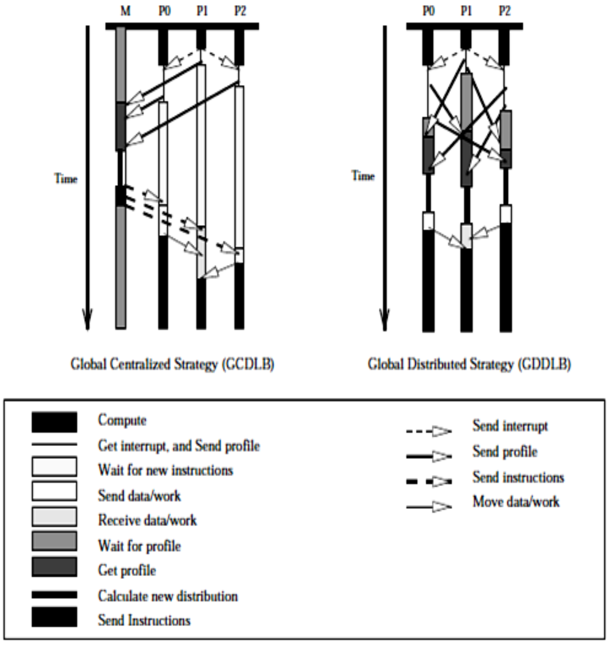


Figure 2.Centralized Vs Distributed Strategies

In the distributed one, the dynamic load balancing algorithm is executed by all nodes present in the system and the task of load balancing is shared among them. The interaction among nodes to achieve load balancing can take two forms: cooperative and non-cooperative [4].

In the first one, the nodes work side-by-side to achieve a common objective, for example, to improve the overall response time, etc. In the second form, each node works independently toward a goal local to it, for example, to improve the response time of a local task. Dynamic load balancing algorithms of distributed nature, usually generate more messages than the non-distributed ones because, each of the nodes in the system needs to interact with every other node. A benefit, of this is that even if one or more nodes in the system fail, it will not cause the total load balancing process to halt; it instead would affect the system performance to some extent.

Distributed dynamic load balancing can introduce immense stress on a system in which each node needs to interchange status information with every other node in the system. It is more advantageous when most of the nodes act individually with very few interactions with others. In non-distributed type, either one node or a group of nodes do the task of load balancing figure (2).

Non-distributed dynamic load balancing algorithms can take two forms: centralized and semi-distributed. In the first form, the load balancing algorithm is executed only by a single node in the whole system: the central node. This node is solely responsible for load balancing of the whole system. The other nodes interact only with the central node. In semi-distributed form, nodes of the system are partitioned into clusters, where the load balancing in each cluster is of centralized form. A central node is elected in each cluster by appropriate election technique which takes care of load balancing within that cluster. Hence, the load balancing of the whole system is done via the central nodes of each cluster [4].

Centralized dynamic load balancing takes fewer messages to reach a decision, as the number of overall interactions in the system decreases drastically as compared to the semi distributed case. However, centralized algorithms can cause a bottleneck in the system at the central node and also the load balancing process is rendered useless once the central node crashes. Therefore, this algorithm is most suited for networks with small size.

4. EXISTING LOAD BALANCING TECHNIQUES IN CLOUD COMPUTING

In complex and large systems, there is a tremendous need for load balancing. For simplifying load balancing globally (e.g. in a cloud), one thing which can be done is, employing techniques would act at the components of the clouds in such a way that the load of the whole cloud is balanced. For this purpose, we are discussing the existing Load Balancing Techniques which can be applied to cloud [7]:

• *Honeybee Foraging Behavior* – M. Randles et al. [14] investigated a decentralized honeybee-based load balancing technique that is a nature-inspired algorithm for self-organization. It achieves global load balancing through local server actions. Performance of the system is enhanced with increased system diversity but throughput is not increased with an increase in system size. It is best suited for the conditions where the diverse population of service types is required.

• *Biased Random Sampling* – M. Randles et al. [14] investigated a distributed and scalable load balancing approach that uses random sampling of the system domain to achieve self-organization thus balancing the load across all nodes of the system. The performance of the system is improved with high and similar population of resources thus resulting in an increased throughput by effectively utilizing the increased system resources. It is degraded with an increase in population diversity.

• *Active Clustering* – M. Randles et al. [14] investigated a self-aggregation load balancing technique that is a self aggregation algorithm to optimize job assignments by connecting similar services using local re-wiring. The performance of the system is enhanced with high resources thereby increasing the throughput by using these resources effectively. It is degraded with an increase in system diversity.

• *ACCLB* (Load Balancing mechanism based on ant colony and complex network theory) – Z. Zhang et al. [15] proposed a load balancing mechanism based on ant colony and complex network theory in an open cloud computing federation. It uses small-world and scale-free characteristics of a complex network to achieve better load balancing. This technique overcomes heterogeneity, is adaptive to dynamic environments, is excellent in fault tolerance and has good scalability hence helps in improving the performance of the system.

• *COMPARE AND BALANCE* – Y. Zhao et al. [16] addressed the problem of intra-cloud load balancing amongst physical hosts by adaptive live migration of virtual machines. A load balancing model is designed and implemented to reduce virtual machines' migration time by shared storage, to balance load amongst servers according to their processor or IO usage, etc. and to keep virtual machines' zero-downtime in the process. A distributed load balancing algorithm COMPARE AND BALANCE is also proposed that is based on sampling and reaches equilibrium very fast. This algorithm assures that the migration of VMs is always from high cost physical hosts to low-cost host but assumes that each physical host has enough memory which is a weak assumption.

• *CARTON* – R. Stanojevic et al. [17] proposed a mechanism for cloud control named as CARTON that unifies the use of LB (Load Balancing) and DRL (Distributed Rate Limiting). LB is used to equally distribute the jobs to different servers so that the associated costs can be minimized and DRL is used to make sure that the resources are distributed in a way to keep a fair resource allocation. DRL also adapts to server capacities for the dynamic workloads so that performance levels at all servers are equal. With very low computation and communication overhead, this algorithm is simple and easy to implement.

• *Two-phase load balancing algorithm (OLB + LBMM)* - S.-C. Wang et al. [18] proposed a two- phase scheduling algorithm that combines OLB (Opportunistic Load Balancing) and LBMM (Load Balance Min-Min) scheduling algorithms to utilize better executing efficiency and maintain the load balancing of the system. OLB scheduling algorithm, keeps every node in working state to achieve the goal of load balance and LBMM scheduling algorithm is utilized to minimize the execution time of each task on the node thereby minimizing the overall completion time. This combined approach hence helps in an efficient utilization of resources and enhances the work efficiency.

• *Event-driven* – V. Nae et al. [20] presented an event driven load balancing algorithm for real-time Massively Multiplayer Online Games (MMOG). This algorithm after receiving capacity events as input, analyzes its components in context of the resources and the global state of the game session, thereby generating the game session load balancing actions. It is capable of scaling up and down a game session on multiple resources according to the variable user load but has occasional QoS breaches.

• *Decentralized content aware load balancing* – H.Mehta et al. [21] proposed a new content aware load balancing policy named as workload and client aware policy (WCAP). It applies a technique to specify the unique and special property (USP) of the requests as well as computing nodes. USP helps the scheduler to decide the best suitable node for the processing the requests. This strategy is implemented in a decentralized manner with low overhead. By using the content information to narrow down the search, this technique improves the searching performance and hence overall performance of the system. It also helps in reducing the idle time of the computing nodes hence improving their utilization.

• *Server-based load balancing for Internet distributed services* – A. M. Nakai et al. [22] proposed a new server based load balancing policy for web servers which are distributed all over the world. It helps in reducing the service response times by using a protocol that limits the redirection of requests to the closest remote servers without overloading them. A middleware is described to implement this protocol. It also uses a heuristic to help web servers to endure overloads.

• *Join-Idle-Queue* – Y. Lua et al. [23] proposed a Join- Idle-Queue load balancing algorithm for dynamically scalable web services. This algorithm provides large-scale load balancing with distributed dispatchers by, first load balancing idle processors across dispatchers for the availability of idle processors at each dispatcher and then, assigning jobs to processors to reduce average queue length at each processor. By removing the load balancing work from the critical path of request processing, it effectively reduces the system load, incurs no communication overhead at job arrivals and does not increase actual response time.

• *A Lock-free multiprocessing solution for LB* – X. Liu et al. [24] proposed a lock-free multiprocessing load balancing solution that avoids the use of shared memory in contrast to other multiprocessing load balancing solutions which use shared memory and lock to maintain a user session. It is achieved by modifying Linux kernel. This solution helps in improving the overall performance of load balancer in a multi-core environment by running multiple load-balancing processes in one load balancer.

• *Scheduling strategy on load balancing of virtual machine resources* – J. Hu et al. [25] proposed a scheduling strategy on load balancing of VM resources that uses historical data and current state of the system. This strategy achieves the best load balancing and reduces dynamic migration by using a genetic algorithm. It helps in resolving the issue of load-imbalance and high cost of migration thus achieving better resource utilization.

• *Central load balancing policy for virtual machines* –A. Bhadani et al. [26] proposed a Central Load Balancing Policy for Virtual Machines (CLBVM) that balances the load evenly in a distributed virtual machine/cloud computing environment. This policy improves the overall performance of the system but does not consider the systems that are fault-tolerant.

• *LBVS:* Load Balancing strategy for Virtual Storage -H. Liu et al. [27] proposed a load balancing virtual storage strategy (LBVS) that provides a large scale net data storage model and Storage as a Service model based on Cloud Storage. Storage virtualization is achieved using an architecture that is three-layered and load balancing is achieved using two load balancing modules. It helps in improving the efficiency of concurrent access by using replica balancing further reducing the response time and enhancing the capacity of disaster recovery. This strategy also helps in improving the use rate of storage resource, flexibility and robustness of the system.

• *A Task Scheduling Algorithm Based on Load Balancing* – Y. Fang et al. [28] discussed a two-level task scheduling mechanism based on load balancing to meet dynamic requirements of users and obtain high resource utilization. It achieves load balancing by first mapping tasks to virtual machines and then virtual machines to host resources thereby improving the task response time, resource utilization and overall performance of the cloud computing environment.

• *VectorDot* – A. Singh et al. [29] proposed a novel load balancing algorithm called VectorDot. It handles the hierarchical complexity of the data-center and multidimensionality of resource loads across servers, network switches, and storage in an agile data center that has integrated server and storage virtualization technologies. VectorDot uses dot product to distinguish nodes based on the item requirements and helps in removing overloads on servers, switches and storage nodes.

Table (1) in the last page, shows the existing load balancing techniques in cloud computing. This review identifies the techniques, their environment, description and findings according to what has been stated in previous works.

5. METRICS FOR LOAD BALANCING IN CLOUDS

The existing load balancing techniques in clouds, consider various parameters “metrics” like performance, response time, scalability, throughput, resource utilization, fault tolerance, migration time and associated overhead. Those are discussed below

- *Performance* – is used to check the efficiency of the system. It has to be improved at a reasonable cost e.g. reduce response time while keeping acceptable delays.
- *Throughput* - is used to calculate the no. of tasks whose execution has been completed. It should be high to improve the performance of the system.
- *Overhead Associated* - determines the amount of overhead involved while implementing a load-balancing algorithm. It is composed of overhead due to movement of tasks, inter-processor and inter-process communication. This should be minimized so that a load balancing technique can work efficiently.
- *Resource Utilization* - is used to check the utilization of resources. It should be optimized for an efficient load balancing.
- *Scalability* - is the ability of an algorithm to perform load balancing for a system with any finite number of nodes. This metric should be improved.
- *Response Time* - is the amount of time taken to respond by a particular load balancing algorithm in a distributed system. This parameter should be minimized.
- *Fault Tolerance* - is the ability of an algorithm to perform uniform load balancing in spite of arbitrary node or link failure. The load balancing should be a good fault-tolerant technique.
- *Migration time* - is the time to migrate the jobs or resources from one node to other. It should be minimized in order to enhance the performance of the system.
- *Carbon Emission (CE)* - calculates the carbon emission of all the resources in the system. As energy consumption and carbon emission go hand in hand, the more the energy consumed, higher is the carbon footprint. So, for an energy-efficient load balancing solution, it should be reduced.

Based on these metrics, the existing load balancing techniques have been compared in Figure (3). In this figure each technique is referred by symbol Ti (T1,T2,T3,……Tn) and the followed index refers to one of these techniques sequentially ordered in table (1). Each metric is assigned a different color. The y-axis is number of metrics which appear in every technique

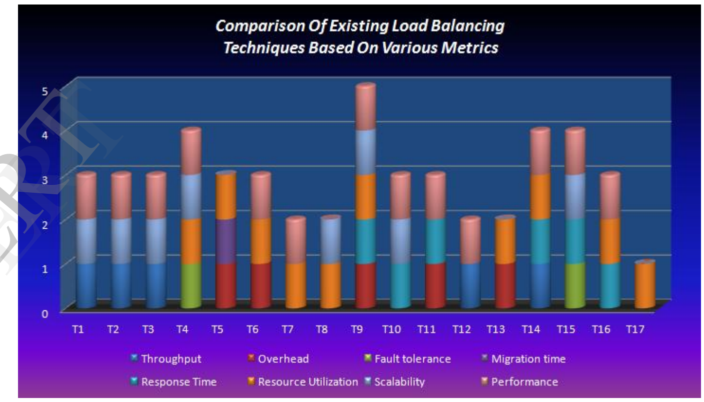


Figure (3) Metrics in Load balancing techniques

6. CONCLUSION

Load balancing is one of the main challenges in cloud computing. It is required to distribute the dynamic local workload evenly across all the nodes to achieve a high user satisfaction and resource utilization ratio by ensuring that every computing resource is distributed efficiently and fairly. With proper load balancing, resource consumption can be kept to a minimum which will further reduce energy consumption and carbon emission rate which is one of main objectives of cloud computing. Existing load balancing techniques that have been discussed mainly focus on reducing associated overhead, service response time and improving performance etc. but none of the techniques has considered the energy consumption and carbon emission factors. Therefore, there is a need to develop an energy-efficient load balancing technique that can improve the performance of cloud computing by balancing the workload across all the nodes in the cloud along with maximum resource utilization, in turn reducing energy consumption and carbon emission to an extent which will help to achieve Green computing. This paper explains the concept of load balancing, types of load balancing algorithms, general idea about dynamic load balancing algorithms and the different policies that can be used in it and gives an overall description of various distributed load balancing algorithms that can be used in case of clouds.

7. REFERENCES

1. Marios D. Dikaiakos, George Pall is, Dimitrios Katsaros, Pankaj Mehra, Athena Vakali, "Cloud computing : Distributed Internet Computing for IT and Scientific Research" IEEE Internet Computing, 2009. Published by the IEEE Computer Society.
2. Panagiotis Kalagiakos, Panagiotis Karampelas, "Cloud Computing Learning" IEEE, Hellenic American University Manchester, N.H. 2011 - U.S.A, pp: 7/11.
3. Peter Mell,Timothy Grance, The NIST Definition of "Cloud Computing" National Institute of Standards and Technology - Computer Security Resource Center-www.csrc.nist.gov.
4. 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.
5. B. P. Rima, E. Choi, and I. Lumb, "A Taxonomy and Survey of Cloud Computing Systems", Proceedings of 5th IEEE International Joint Conference on INC, IMS and IDC, Seoul, Korea, August 2009, pages 44-51.
6. R. Mata-Toledo, and P. Gupta, "Green data center: how green can we perform", Journal of Technology Research, Academic and Business Research Institute, Vol. 2, No. 1,May 2010, pages 1-8.
7. Martin Randles, Enas Odat, David Lamb, Osama Abu-Rahmeh and A. Taleb-Bendiab, A Comparative Experiment in Distributed Load Balancing, 2009 Second International Conference on Developments in eSystems Engineering.
8. Z. Zhang, and X. Zhang, "A Load Balancing Mechanism Based on Ant Colony and Complex Network Theory in Open Cloud Computing Federation", Proceedings of 2nd International Conference on Industrial Mechatronics and Automation (ICIMA), Wuhan, China, May 2010, pages 240- 243.
9. B. P. Rimal, E. Choi, and I. Lumb, "A Taxonomy, Survey, Issues of Cloud Computing Ecosystems, Cloud Computing: Principles, Systems and Applications", Computer Communications and Networks, Chapter 2 , pages 21-46, DOI 10.1007/978-1-84996-241-42, Springer – Verlag London Limited, 2010.
10. S. Kabiraj, V. Topka, and R. C. Walke, "Going Green: A Holistic Approach to Transform Business", International Journal of Managing Information Technology (IJMIT), Vol. 2, No. 3, August 2010, pages 22-31.
11. Zhong Xu, Rong Huang, "Performance Study of Load Balancing Algorithms in Distributed Web Server Systems", CS213 Parallel and Distributed Processing Project Report, 2009.
12. J. Baliga, R. W. A. Ayre, K. Hinton, and R. S. Tucker, "Green Cloud Computing: Balancing Energy in Processing, Storage, and Transport", Proceedings of the IEEE, Vol. 99, No. 1, January 2011, pages 149-167.
13. A. M. Alakeel, "A Guide to dynamic Load balancing in Distributed Computer Systems", International Journal of Computer Science and Network Security (IJCSNS), Vol. 10, No. 6, June 2010, pages 153-160.
14. M. Randles, D. Lamb, and A. Taleb-Bendiab, "A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing", Proceedings of 24th IEEE International Conference on Advanced Information Networking and Applications Workshops, Perth, Australia, April 2010, pages 551-556.
15. Z. Zhang, and X. Zhang, "A Load Balancing Mechanism Based on Ant Colony and Complex Network Theory in Open Cloud Computing Federation", Proceedings of 2nd International Conference on Industrial Mechatronics and Automation (ICIMA), Wuhan, China, May 2010, pages 240-243.
16. Y. Zhao, and W. Huang, "Adaptive Distributed Load Balancing Algorithm based on Live Migration of Virtual Machines in Cloud", Proceedings of 5th IEEE International Joint Conference on INC, IMS and IDC, Seoul, Republic of Korea, August 2009, pages 170-175.
17. R. Stanojevic, and R. Shorten, "Load balancing vs. distributed rate limiting: a unifying framework for cloud control", Proceedings of IEEE ICC, Dresden, Germany, August 2009, pages 1-6.
18. S. Wang, K. Yan, W. Liao, and S. Wang, "Towards a Load Balancing in a Three-level Cloud Computing Network", Proceedings of the 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), Chengdu, China, September 2010, pages 108-113.
19. Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges", Journal of Internet Services and Applications, Vol. 1, No. 1, April 2010, pages 7-18.
20. V. Nae, R. Prodan, and T. Fahringer, "Cost-Efficient Hosting and Load Balancing of Massively Multiplayer Online Games", Proceedings of the 11th IEEE/ACM International Conference on Grid Computing (Grid), IEEE Computer Society, October 2010, pages 9-17.
21. H. Mehta, P. Kanungo, and M. Chandwani, "Decentralized content aware load balancing algorithm for distributed computing environments", Proceedings of the International Conference Workshop on Emerging Trends in Technology (ICWET), February 2011, pages 370-375.
22. A. M. Nakai, E. Madeira, and L. E. Buzato, "Load Balancing for Internet Distributed Services Using Limited Redirection Rates", 5th IEEE Latin-American Symposium on Dependable Computing (LADC), 2011, pages 156-165.
23. Y. Lua, Q. Xiea, G. Kliotb, A. Gellerb, J. R. Larusb, and A. Greenber, "Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services", An international Journal on Performance evaluation, In Press, Accepted Manuscript, Available online 3 August 2011.
24. Xi. Liu, Lei. Pan, Chong-Jun. Wang, and Jun-Yuan. Xie, "A Lock-Free Solution for Load Balancing in Multi-Core Environment", 3rd IEEE International Workshop on Intelligent Systems and Applications (ISA), 2011, pages 1-4.
25. J. Hu, J. Gu, G. Sun, and T. Zhao, "A Scheduling Strategy on Load Balancing of Virtual Machine Resources in Cloud Computing Environment", Third International Symposium on Parallel Architectures, Algorithms and Programming (PAAP), 2010, pages 89-96.
26. A. Bhadani, and S. Chaudhary, "Performance evaluation of web servers using central load balancing policy over virtual machines on cloud", Proceedings of the Third Annual ACM Bangalore Conference (COMPUTE), January 2010.
27. [27] H. Liu, S. Liu, X. Meng, C. Yang, and Y. Zhang, "LBVS: A Load Balancing Strategy for Virtual Storage", International Conference on Service Sciences (ICSS), IEEE, 2010, pages 257-262.
28. Y. Fang, F. Wang, and J. Ge, "A Task Scheduling Algorithm Based on Load Balancing in Cloud Computing", Web Information Systems and Mining, Lecture Notes in Computer Science, Vol. 6318, 2010, pages 271-277.
29. A. Singh, M. Korupolu, and D. Mohapatra, "Server-storage virtualization: integration and load balancing in data centers", Proceedings of the ACM/IEEE conference on Supercomputing (SC), November 2008.
30. Nidhi Jain Kansal, Inderveer Chana "Cloud Load Balancing Techniques: A Step Towards Green Computing", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 1, No 1, January 2012.

TABLE 1: Existing Load Balancing Technique

No./Techniques	***Author/ Year***	***Environment***	***Description***	***Findings***
T1- Honeybee Foraging Behavior [14]	M. Randles et al. / 2010	Large scale Cloud systems	1. Achieves global load balancing through local server action	1. Performs well as system diversity increases 2. Does not increase throughput as system size increases
T2- Biased Random Sampling [14]	M. Randles et al. / 2010	Large scale Cloud systems	1. Achieves load balancing across all system nodes using random sampling of the system domain	1. Performs better with high and similar population of resources 2. Degrades as population diversity increases
T3- Active Clustering [14]	M. Randles et al. / 2010	Large scale Cloud systems	1. Optimizes job assignment by connecting similar services by local re-wiring	1. Performs better with high resources 2. Utilizes the increased system resources to increase throughput 3. Degrades as system diversity increases
T4- ACCLB (Ant Colony and Complex Network Theory) [15]	Z. Zhang et al. / 2010	Open Cloud Computing Federation	1. Uses small-world and scale-free characteristics of complex network to achieve better load balancing	1. Overcomes heterogeneity 2. Adaptive to dynamic environments 3. Excellent in fault tolerance 4. Good scalability
T5- Compare and Balance [16]	Y. Zhao et al. / 2009	Intra-Cloud	1. Based on sampling 2. Uses adaptive live migration of virtual machines	1. Balances load amongst servers 2. Reaches equilibrium fast 3. Assures migration of VMs from high-cost physical hosts to low-cost host 4. Assumption of having enough memory with each physical host
T6- Carton(LB + DRL) [17]	R. Stanojevic et al. / 2009	Unifying framework for cloud control	1. Uses Load Balancing to minimize the associated cost and uses Distributed Rate Limiting for fair allocation of resources	1. Simple 2. Easy to implement 3.Very low computation and communication overhead
T7- Two-phase scheduling(OLB +LBMM) [18]	S.-C. Wang et al. / 2010	Three-level Cloud Computing Network	1. Uses OLB (Opportunistic Load Balancing) to keep each node busy and uses LBMM (Load Balance Min-Min) to achieve the minimum execution time of each task	1. Efficient utilization of resources 2. Enhances work efficiency
T8- Event-driven [20]	V. Nae et al. / 2010	Massively Multiplayer Online Games	1. Uses complete capacity event as input, analyzes its components and generates the game session load balancing actions	1. Capable of scaling up and down a game session on multiple resources according to the variable user load 2. Occasional QoS breaches as low as 0.66%
T9- Decentralized content aware [21]	H. Mehta et al. / 2011	Distributed computing	1. Uses a unique and special property (USP) of requests and computing nodes to help scheduler to decide the best node for processing the requests 2. Uses the content information to narrow down the search	1. Improves the searching performance hence increasing overall performance 2. Reduces idle time of the nodes
T10- LB for Internet distributed services [22]	A. M. Nakai et al. / 2011	Distributed web servers	1. Uses a protocol to limit redirection rates to avoid remote servers overloading 2. Uses a middleware to support this protocol 3. Uses a heuristic to tolerate abrupt load changes	1. Reduces service response times by redirecting requests to the closest servers without overloading them 2. Mean response time is 29% smaller than RR(Round Robin) and 31% smaller than SL(Smallest Latency)
T11- Join-Idle-Queue [23]	Y. Lua et al. / 2011	Cloud data centers	1. First assigns idle processors to dispatchers for the availability of the idle processors at each dispatcher 2. Then assigns jobs to processors to reduce average queue length of jobs at each processor	1. Effectively reduces the system load 2. Incurs no communication overhead at job arrivals 3. Does not increase actual response times
T12- Lock-free multi-processing [24]	X. Liu et al. /2011	Multi-core	1. Runs multiple load-balancing processes in one load balancer	1. Improves overall performance of load balancer
T13- Scheduling strategy on LB of VM resources [25]	J. Hu et al. / 2010	Cloud Computing	1. Uses Genetic algorithm, historical data and current state of system to achieve best load balancing and to reduce dynamic migration	1. Solves the problems of load imbalance and high migration cost
T14- Central LB policy for VMs [26]	A. Bhadani et al. / 2010	Cloud Computing	1. Uses global state information to make load balancing decisions	1. Balances the load evenly to improve overall performance 2. Up to 20% improvement in performance 3. Does not consider fault tolerance
T15- LBVS: LB strategy for Virtual Storage [27]	H. Liu et al. / 2010	Cloud Storage	1. Uses Fair-Share Replication strategy to achieve Replica Load balancing module which in turn controls the access load balancing 2. Uses writing balancing algorithm to control data writing load balancing	1. Enhances flexibility and robustness 2. Provides large scale net data storage and storage as a service
T16- Task Scheduling Based on LB [28]	Y. Fang et al. / 2010	Cloud Computing	1. First maps tasks to virtual machines and then virtual machines to host resource	1. Improves task response time 2. Improves resource utilization
T17- VectorDot [29]	A. Singh et al. / 2008	Datacenters with integrated server and storage virtualization	1. Uses dot product to distinguish node based on the item requirement	1. Handles hierarchical and multidimensional resource constraints 2. Removes overloads on server, switch and storage