International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 07 | July 2021 www.irjet.net p-ISSN: 2395-0072

Survey on Load Balancing Techniques in Cloud

Vaibhav V Athani[1], Sandhya S[2]

[1]Student, Department of Computer Science and Engineering, R V College of Engineering, Karnataka, India
[2]Assistant Professor, Department of Computer Science and Engineering, R V College of Engineering, Karnataka, India

---

Abstract - *Today, there are different load balancing algorithms used in distributed, centralized and semi-distributed systems. Increase in the heterogeneity in the cloud environment has led to increased incoming requests to the servers which would eventually overwhelm the servers. There are various load balancing algorithms that can be used for efficient load distribution. There is also increase in the demand for fast processing capability to support distributed architecture. This has led to increase in the trend for using distributed systems. This involves load balancing which requires distributing workload among all the available nodes. In this paper different load balancing algorithms are discussed and analyzed based on the performance.*

Key Words: **virtual machine, proxy, workload distribution, SLA, distributed system, load balancing, resource utilization**

1. INTRODUCTION

As the incoming requests to the servers are heterogenous in nature and the number of incoming requests will be high only during peak hours, it is important to load balance the requests. During the peak hours, to handle the user requests in least amount of time, it is necessary to manage the incoming load properly. Load balancing can be done using done at the hardware or software level. An autonomous approach is required to optimally redirect incoming requests to appropriate servers leading proper load distribution.

As there is huge rise in distributed computing systems, they should be used as efficiently as possible. The mapping of tasks to the resources should be optimized for effective resource utilization and reducing the response time. Hence, load balancing for these systems which are highly heterogenous is necessary.

When a new request is generated, the load balancing algorithm should be able to decide on the node for workload distribution computations. This also involves transfer of processing tasks from over loaded node to under loaded / normal node for effective load distribution

The challenges involved in distributed system load balancing are requirement of different compute resources for different tasks in the system, difference in the computation capacity between the nodes and non-uniform inter-process communication.

The main objective of this paper is to study the load balancing techniques of distributed, non-distributed, semi-distributed systems. This paper covers classification of different load balancing techniques. This also follows performance analysis of these algorithms using various evaluation metrics and studying their advantages and disadvantages.

2. LOAD BALANCING IN CLOUD

Load balancing can be defined as redirecting the requests to appropriate nodes in the system for effective resource utilization. The main goal of any load balancing algorithm is to provide services to the user without any significant delay. The users and cloud service providers will maintain a contract in the form of SLAs. The main objective of load balancing is to handle multiple requests from the users without overwhelming and degrading the performance of the servers. Some of the advantages of load balancing techniques include system adaptability, decreased idle time of resources, reliability, decreased starvation, decreased cost and response time, fault tolerant [1]. The main aim of load balancing algorithms is to prevent the servers from being overwhelmed with requests which would eventually degrade their performance along with other disadvantages such as high response time and server crashes. The main advantage of having a load balancer is scalability and high availability. It is scalable because the user requests can be redirected to many servers based on a load balancing policy and highly available because, even if any server goes down for some reason, other servers can service user requests flawlessly without any interruption. Load balancing at the software level can be achieved by implementing it at the application layer of the network stack. In this case the software program acts as an orchestrator for redirecting requests to different servers for load distribution. Hardware load balancing can be achieved by setting up multiple nodes for handling the workload distribution. The hardware load balancing technique is more efficient and provides good performance when compared to software load balancing. It is also more suitable for applications which require more scalability. Some of the challenges that must be addressed in the cloud environment are privacy, interference in performance and noisy neighbours. As the cloud platform supports multitenancy, a condition of noisy neighbours can arise wherein an unusual activity on the neighbouring VMs will lead to performance degradation of the VMs which are hosted on top of the same physical host. Another challenge is

with respect to load balancing which involves distributing workload across multiple compute resource nodes. Load balancing aims to optimize throughput, resource utilization, response time, fault tolerance and reduce any overheads on the compute nodes.

3. TYPES OF LOAD BALANCING ALGORITHMS

Load balancing techniques are classified based on current state of the system and process initiation

3.1 Based on the current state of the system

Static load balancing*:* Static load balancing technique makes use of prior information about resources and nodes. This method is useful in a homogenous environment and is not scalable.

Dynamic load balancing*:* Dynamic load balancing technique [2] collects current state information about all the available resources in the system and makes the decision. This is suitable for heterogenous environment. Dynamic load balancing algorithm can be divided into two types: distributed, semi-distributed and centralized. In distributed algorithm, all computing nodes are responsible for balancing the load. The responsibility of load balancing is distributed among all the available resources. In centralized algorithm, a single centralized node is responsible for management of load based on the available resources. The remaining nodes handle the requests assigned to them by the centralized node and do not perform any load balancing tasks. Hence, it is not fault tolerant. In semi-distributed algorithms, resource nodes are grouped into clusters based on a predefined criterion. A centralized node is elected in all the clusters responsible for handling load balancing tasks [3]. Distributed load balancing algorithms are further divided into cooperative and non-cooperative. In case of non-cooperative, all nodes act as independent units making individual decisions. Non-cooperative nodes operate independently to achieve their respective local tasks. In case of cooperative, all nodes work together with a common goal. This involves nodes working to achieve a common goal. Dynamic load balancing algorithms which are distributed usually involve more interaction between the nodes. The advantage of using distributed algorithms is that even if some of the nodes are unavailable, then the load balancing will not be affected. Only the performance is affected due to shift in the load to other available nodes. Dynamic load balancing involves policies such as selection, transfer, location, and information policy for implementing the load balancing algorithm [4]. Transfer policy is used for transferring job from local to remote node. Selection policy is used for selecting the resources for processing. Location policy is used for determining a destination node for the task. Information policy is used for gathering information about all the nodes involved in the system for load distribution.

3.2 Based on the process initiation

Sender initiated*:* In this type, load distribution is initiated by the heavily loaded node or sender node.
Receiver initiated*:* In this type, load distribution is initiated by lightly loaded node or receiver node.
Symmetric*:* In this type, load distribution is initiated by both sender and receiver or peers.

4. LITERATUE REVIEW

The load balancing algorithm is responsible for determining the most suitable resource for handling the request. There are some challenges associated while distributing workload such as mapping of tasks and resources and network traffic. The tasks received must be mapped to suitable resources for execution. This involves mapping resources to the task, execution of tasks and monitoring of resources. The resources can be classified into over loaded, under loaded and normal loaded. The tasks can then be assigned to resources which are either under loaded or normal loaded. An algorithm for determining the resource for a particular task was implemented by [5]. The function uses ratio of completion and past information of the resources for determining the most suitable resource for a task. Traditional round robin scheduling technique works well only if the resource nodes are of same capacity. It is not suitable for heterogenous environment, this method fails to provide appropriate result. A weighted round robin method was proposed by [6]. This involves assigning weights to each node based on the resources available on each node. This algorithm performs the workload distribution more efficiently when compared to traditional round robin algorithm. Another challenge to be resolved is network traffic. As the number of requests increase on the server, the response time of the server decreases due to the load. When the load on the server crosses a given threshold, the server eventually crashes. An immediate solution is to increase the number of resources on the server. With this, the server would be able to handle much more requests, but this approach isn't scalable and also it requires more maintenance. The most suitable solution is to use a load balancer which load balances the request among all the available servers intelligently. On receiving a request, the load balancer can redirect the request to a server using different routing strategies such as round robin, random and shortest queue. A load balancing algorithm based on bipartite match and priority-based scheduling was implemented by [7]. A load balancing algorithm based on clustering was proposed by [8]. The virtual machines in the environment were divided into clusters using k-means clustering algorithm. The tasks were assigned to suitable virtual machines for balancing the workload. An in-depth analysis for min-min and max-min algorithms was provided by [9]. A heuristic based approach for load balancing was provided by [10]. This involved assigning tasks to clusters using the Bayes theorem. An in-depth analysis of various

factors affecting load balancing algorithms was done by [11]. This included parameters such as time required for migration, central node failure and performance. The authors provided an in-depth analysis about some well-known load balancing algorithms using performance metrics such as throughput, resource utilization, response time, migration time, fault tolerance and network congestion.

4.1 Static load balancing techniques

In these algorithms, load distribution is done based on a predefined strategy prior to execution.

Min-Min and Max-Min algorithm [4]: In case of Min-Min, a task with lesser execution time is assigned to a node with more resources. It is only useful when tasks require considerably lesser load that can be executed quickly, and it can also lead to starvation of tasks with large execution time. In case of Max-Min, tasks with large execution time are executed first. However, tasks with lesser execution time must wait for a long time before they are picked up for processing.

Round robin and weighted round robin [12][13]: In this case, the requests are handled in a circular manner. A major advantage of using this method is that it doesn't require any inter-process communication. In case of weighted round robin, the requests are assigned based on the weights assigned to the nodes. The weights for the nodes can be determined based on various factors such as resource availability.

Threshold algorithm: In this case, three levels are defined for all the nodes – Underloaded, Normal, Overloaded. All nodes exist in any one of these three states at any point of time. A threshold value is defined which determines the current state of each node. When the load on a node is less than the threshold then it will be in underloaded state. If the load is more than the threshold then it will be in overloaded state and will not accept any further requests for processing.

4.2 Dynamic load balancing techniques

In these algorithms, workload distribution happens at runtime unlike static load balancing algorithms

Ant colony optimization [13]: This algorithm is based upon the behaviour of ants. The idea here is based on the shortest path taken by the ants in search of food. The ants come together at the place which has more food. In this way, other ants will also follow the same path to the food resource. This results in all ants following a single path. The same approach can be used for finding an optimal solution for load distribution

Central queue algorithm [14]: In this technique, there will be a central node which carries out decision making tasks. It maintains a queue for tracking all the requests. When a node has resources available, the local load manager sends a request to the central node for processing the requests.

Local queue algorithm [15]: This algorithm is based on process migration between the nodes. In this algorithm, the processes from one node are migrated to another node which initiated the request and has load below the threshold level. When a node has processes ready for providing service it sends requests randomly to remote load managers. On receiving the request, the remote load manager compares the local number of processes ready for providing service with that in the arrived request. If the local processes are greater, then some of the running processes are migrated to the requested node.

Active clustering*:* This algorithm involves grouping nodes of similar type together. A node is selected initially. This node selects any of its neighbouring node which is of different type. This selected node explores its neighbours which are of same type as that of the first node and establishes a connection between these neighbours. The selected node then detaches from the first initial node and this process continues.

Adaptive contracting with neighbour*:* This algorithm involves dynamic load balancing based on a predefined threshold load. When a request arrives to the load balancer it redirects the request to the nearest node with least amount of load. When the request arrives, the node compares the requested load with that of its threshold load. If it's less than threshold load than it will execute the requested task. If not, then it will identify the nearest node which has load lesser than the threshold load and forwards the request.

4.3 Performance metrics for load balancing in cloud

Throughput*:* It indicates the number of tasks which have got completed.
Resource utilization*:* It indicates the utilization of all resources important to ensure effective load balancing.
Response time: It indicates the time taken to respond by a given load balancing algorithm.
Scalability: It indicates the ability to scale the nodes for load balancing.
Fault tolerance: It indicates the resistance of load balancing algorithm towards failures such as node crash.
Migration time: It indicates the time taken to transfer the resources from one node to another.
Overhead*:* It indicates the extra load on the nodes during workload distribution which can be due to inter-process communication which has to minimized to achieve efficient load balance.

4.4 Performance analysis of state-of-the-art algorithms

Generally, load balancing algorithms try to improve the overall resource utilization, throughput, performance, and energy consumption. There will be overheads associated with any algorithm while trying to improve one of these factors. This involves trade-offs to be made between these metrics to achieve overall efficiency. Centralized load balancing algorithms provide good performance but can lead to problems. In this case, there will be more load on the central node and eventually would lead to single point of failure. This section explains different load balancing algorithms along with their benefits.

Ant colony optimization algorithm [16] involves live migration of VMs from underloaded nodes to some other nodes which are in normal or underloaded state so as to save energy consumption on the nodes. This introduces significant computational overhead on the nodes. This approach is also scalable. Artificial bee colony optimization algorithm [17] involves process migration to underloaded nodes from overloaded nodes for efficient load distribution. This approach is not scalable and may lead to starvation of low priority processes. This algorithm is good in terms of response time and resource utilization. Agent based load balancing algorithm [18] redirects requests to closest nodes in nearby datacentres thereby reducing response time. Dynamic clustering-based algorithm [19] involves grouping nodes for load distribution to improve the overall throughput. An adaptive load balancing algorithm [20] involves process migration with high throughput. This algorithm can incur considerable computational overhead on the nodes. Adaptive load balancing algorithm proposed by [21] improved resource utilization and throughput. The round robin algorithm involves redirecting requests in a circular manner and this would help in scalability. Min-Min algorithm can provide high throughput, response time, resource utilization and performance.

5. CONCLUSIONS

This paper presents an in-depth survey and analysis on different load balancing techniques with major focus on distributed, semi-distributed and centralized dynamic algorithms. It also presents a brief overview over different static load balancing algorithms. Due to increase in the heterogeneity in the systems today, dynamic load balancing algorithms play a major role in efficient load distribution. This survey can also provide the information for implementing efficient load balancing algorithms. This paper also lists advantages as well as disadvantages of different algorithms along with their challenges which would help in developing more efficient algorithms.

ACKNOWLEDGEMENT

This work was supported by Department of Computer Science and Engineering, R V College of Engineering, Bangalore.

REFERENCES

[1] Alakeel AM. A guide to dynamic load balancing in distributed computer systems. International Journal of Computer Science and Information Security. 2010;10(6):153-160
[2] Deepa T, Cheelu D. A comparative study of static and dynamic load balancing algorithms in cloud computing. In: 2017 International conference on energy, communication, data analytics and soft computing (ICECDS), Chennai; 2017, pp. 3375–3378.
[3] 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, June2010.
[4] O. Elzeki, M. Reshad, M. Elsoud, "Improved max-min algorithm in cloud computing," International Journal of Computer Applications, vol. 50 (12). 2012, pp. 22–27.
[5] Naik KJ, Jagan A, Narayana NS. "A novel algorithm for fault tolerant job scheduling and load balancing in grid computing environment". In: International Conference on Green Computing and Internet of Things. 2015. pp. 1113-1118
[6] Raman K, Subramanyam A, Rao AA. Comparative analysis of distributed web server system load balancing algorithms using qualitative parameters. VSRD International Journal of Computer Science and Information Technology. 2011;8:592-600
[7] Pradeep Naik, Surbhi Agrawal, Srikanta Murthy, A Survey on Various Task Scheduling Algorithms Toward Load Balancing in Public Cloud, American Journal of Applied Mathematics, Vol. 3, No. 1-2, pp.14-17, 2015
[8] Surbhi Kapoor, Dr. Chetna Dabas, Cluster Based Load Balancing in Cloud Computing, IEEE, pp.1- 6, 2015.
[9] Geethu Gopinath, Shriram K Vasudevan, An in-depth analysis and study of Load balancing techniques in the cloud computing environment, Procedia Computer Science, pp .427–432, 2010.
[10] Jia Zhao, Kun Yang, Senior Member, IEEE, Xiaohui Wei, Member, IEEE, Yan Ding, Liang Hu, and Gaochao Xu, A Heuristic Clustering-Based Task Deployment Approach for Load Balancing Using Bayes Theorem in Cloud Environment, VOL. 27, NO. 2, pp.305-316, February 2016.
[11] V RaviTeja Kanakala,V.Krishna Reddy and K.Karthik, Performance Analysis of Load Balancing Techniques in Cloud Computing Environment, IEEE, pp.1-6, 2015.
[12] A.khiyait, H.Bakkali, M.Zbakh, D.Kettani, Load Balancing Cloud Computing: state of art", University Mohammed V Souissi Rabat Morocco, 2012.

[13] Qi Zhang, Lu Cheng, Raouf Boutaba; Cloud computing: sate-of-art and research challenges; Published online: 20th April 2010, Copyright : The Brazillian Computer Society 2010.
[14] William Leinberger, George Karypis, Vipin Kumar, "Load Balancing Across Near-Homogeneous Multi-Resource Servers", 0-7695-0556-2/00, 2000 IEEE
[15] H.Mahalle, P.Kaveri, V.Chavan, "Load Balancing on Cloud Data Centers", international Journal of Advance Research in computer Science and Software Engineering, vol. 3, Jan. 2013.
[16] Farahnakian, F., Ashraf, A., Pahikkala, T., Liljeberg, P., Plosila, J., Porres, I., Tenhunen,H., 2014. Using Ant Colony System to consolidate VMs for Green Cloud Computing.IEEE Trans. Serv. Comput. 8 (2), 187–198.
[17] Dhinesh Babu, L.D., Krishna, P.V., 2013. Honey bee behavior inspired load balancing of tasks in cloud computing environments. Appl. Soft Comput. 13 (5), 2292–2303.
[18] Singh, A., Juneja, D., Malhotra, M., 2015. Autonomous Agent based load balancing algorithm in cloud computing. International Conference Adv. Comput. Technol. Appl. 45, Mumbai, India. pp. 832–841.
[19] Zhao, J., Yang, K., Member, S., Wei, X., 2016. A heuristic clustering-based task deployment approach for load balancing using Bayes theorem in cloud environment. IEEE Trans. Parallel Distrib. Syst. 27 (2), 305–316.
[20] Liu, Y., Zhang, C., Li, B., Niu, J., 2017. DeMS: a hybrid scheme of task scheduling and load balancing in computing clusters. J. Netw. Comput. Appl. 83, 213–220.
[21] Patil, S.S., Gopal, A.N., 2017. Dynamic Load Balancing Using Periodically Load Collection with Past Experience Policy on Linux Cluster System. Am. J. Math.
[22] Nitin R Chopde, Pritish A Tijare. A Survey – Review on Load Balancing Algorithms. International Journal of Advanced Research in Computer Science. Volume 3, No. 1, 2012. ISSN No. 0976-5697
[23] Divyashree Nath, Uday Chourasia, Shikha Agarwal. A Survey on various Load Balancing Algorithms in Cloud Computing. International Journal of Advanced Research in Computer Science. Volume 8, No. 5, May – June 2017. ISSN No. 0976-5697
[24] Y. Ranjith Kumar, M. Madhu Priya , K. Shahu Chatrapati. Effective Distributed Dynamic Load Balancing For The Clouds. International Journal of Engineering Research & Technology (IJERT). Vol. 2 Issue 2, February- 2013. ISSN: 2278-0181
[25] Hariprakash mishra, Sreelakshmi Nair. SURVEY PAPER ON CLOUD COMPUTING. International Journal of Scientific & Engineering Research Volume 12, Issue 3, March-2021. ISSN 2229-5518 193
[26] Vinay Darji, Jayna Shah, Rutvik Mehta.Survey paper on various load balancing algorithms in cloud computing. International Journal of Scientific & Engineering Research, Volume 5, Issue 5, May-2014 583. ISSN 2229-5518