IRACST - International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555
Vol.7, No.2, Mar-April 2017

A Survey: A Comprehensive Study of Static, Dynamic and Hybrid Load Balancing Algorithms

Siham Hamadah
Al-Zaytoonah University of Jordan
Faculty of Science and Information Technology
Computer Information System Department
Amman-Jordan. Email: siham@zuj.edu.jo

***Abstract*—Load balancing is a policy of distributing the load among various nodes or servers to improve both resource utilization and job response time. It also maximizes the throughput and avoids a situation where some of the nodes are heavily loaded, while other nodes are idle, or are performing very little work. There are many algorithms for load balancing. In this paper; comparative studies for static, dynamic, and hybrid algorithms have been conducted.**
The result shows that dynamic and hybrid algorithms in general give better results than static algorithms.

Keywords-component; load balancing algorithms; static load balancing; load balancing and hybrid load balancing.

I. INTRODUCTION

Load balancing is a very important topic in our life, since the Internet traffic increases day by day, and rapid development of technologies have emerged. Therefore, there is a high need for availability and rapid response. Thus, a load balancer mediates client access requests to many servers, and should decide which server is suitable for each request as shown in Figure 1. Decisions of a load balancer are either static, dynamic, or hybrid. A static decision is independent of the current system state, simple, and easy to be implemented by using a queuing process. On the other hand, a dynamic decision depends on the system state of the decision time, and the dynamic, which is more complex because each node should know about other nodes [1], while hybrid algorithms inherit the benefits of static and dynamic algorithms. There are lots of researches and comparative studies of static, dynamic, and hybrid algorithms. Nonetheless, in this paper, the author compares extra techniques and algorithms with their merits and demerits.

Several benefits of load balancing [2]:

Scalability: is the ability of the algorithm to give optimized results with any finite number of nodes, and it is the capability of a system, network, or process to handle a growing amount of work. Accordingly, a load balancer uses different algorithms to distribute the clients' requests among all real servers.

Availability: is the probability for which a system is operational at a given time, and is the ability of a client to access the information or resources in a specified location, and in the correct format. The availability features allow the system to stay operational even when faults do occur. Thus, a load balancer continuously monitors the health of the real server sand application running on them (health check). If the server fails, then it will send the request to another one.

Manageability: is the ability of a system to be controlled easily either by means of a self-control, or by providing certain techniques to ease external controls, and how efficiently and easily a software system can be monitored. Consequently, a load balancer will assist in the manageability process, such as stopping to send requests or graceful shutdowns during maintenance.

Security: Because load balancers are the front end to the server farm, they can protect the servers from bad users. Load balancers have security features that can stop attacking.

The outlines of the paper are highlighted as follows. In section II, the static algorithms are discussed and are compared with their merits and demerits. In section III, the dynamic algorithms are discussed. In section IV, the hybrid algorithms are elaborated. Finally, conclusions are drawn in Section V.

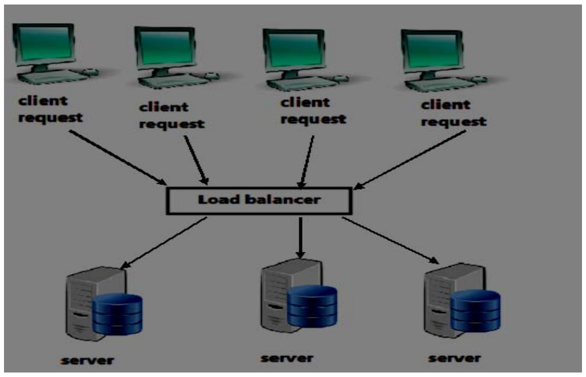

Figure 1. Load Balancer mediates client requests to many servers

II. Static Load Blancing Algoritms

In this approach, the load balancing is performed by previous information with regard to the system. Next, depending on the performance workload is distributed without considering the current state of the node. Once the load is allocated to the node, it cannot be transferred to another one. The nodes perform their work, and send back the results to their clients.

A. *The Features of static Load Balancing Algorithims*

- It needs less communication in order to minimize the communication delays, where this reduces the execution time.
- It does not take the current state of the system while producing the allocation distribution.
- Weighted algorithms achieve a better response time and processing time.
- Load balancing algorithms load the distribution depending on the load at the time of selecting the node before the execution starts.
- Static algorithms are mostly suitable for the constant work application, and for homogeneous and stable environments that can produce better results in within these environments [10].
- It is easy to be implemented.
- It is easy to predict the behavior of the static algorithms.
- It is quite impossible to make predictions of arrival times of loads and processing times that are required for future loads.

B. *Static Load Balancing Algorithms.*

Many static algorithms do exist, where five of them have been taken in this survey, and which comprise:

a) The Round Robin algorithm: is a simple method for distributing clients' requests across set of listed servers, where an equal load is assigned to each node in a circular order without any priority. When it reaches the end of the list, the load balancer loops again, where it sends the next request to the first listed server, and then, to the second one, and so forth.

b) The Weighted Round Robin algorithm: A weight is assigned to each server according to the server's traffic-handling capacity. It is defined to improve the critical challenges associated with the round robin algorithm [5]. The server with a higher weight receives more tasks. For instance, if a Server S1 has a weight 4, and a Server S2 has a weight 1, then, Server S1 will receive 4 tasks, while Server S2 will receive 1 task.

c) The Randomized algorithm: in this algorithm, a process can be handled by a particular node n with a probability [8]. The node is selected based on a random selection, without having any information about the current, or the previous load over the node.

d) The Central Manager algorithm: is a predefined table that is saved in the central node. The table contains the pairs of nodes for load cooperation purpose. When any node finds itself overloaded, it sends a request for the central node, where the central node replies with a node ID, which will share the load with the overloaded node.

e) The Hash IP address algorithm [2] [3]: This algorithm takes the source and destination of the IP address for the clients and the server in order to create a unique hash key by using a particular mathematical calculation. In particular, this key is used to assign the client with the server. If a session is broken, the client is redirected with same server that was being used previously. Further, the client generates each request by using different source port numbers, where the entire request from the same client can be distributed among multiple servers.

Table (1) shows merits and demerits of each algorithm.

III. Dynamic Load Balancing Algorithms

These algorithms monitor changes on the system workload, and redistribute the entire works. This algorithm is usually composed of three strategies, which comprise: the transfer strategy, the location strategy, and the information strategy. The transfer strategy decides which tasks are eligible to be transferred to other nodes for processing. The location strategy nominates a remote node in order to execute a transferred task. The information strategy represents the information center for the Load Balancing algorithm [9]. In fact, it is responsible for providing the location and the transferred strategies to each node. The Dynamic Load Balancing algorithm can be achieved based on three ways: non-distributed, distributed, or semi-distribute methods. In the non-distributed method, there is one node (centralized) that receives all requests and distributes them to the servers. In the distributed method, all nodes are shared with the distribution of the requests. As for the semi-distributed method, the nodes are divided up into a group of clusters, where each cluster works as a central node in order to distribute the requests, and all clusters are responsible for the load balancing distribution.

A. *The features of Dyanic Load Balancing Algoritms.*

- It selects the suitable node that requires real time communication with the networks, which will lead to an extra traffic to be added to a system [5].

- Dynamic algorithms provide better performance.
- It is difficult to be implemented.
- Dynamic algorithm are suitable for adaptive applications where the workload is unpredictable, or keeps changing during an execution [9].
- Dynamic algorithms are also mostly suitable for heterogeneous and distributed systems.
- These algorithms require that each node must know the states of other nodes.
- Processes may migrate from one node to another even in the middle of the execution to ensure providing equal loads [11].

B. Dynamic Load Balancing Algorithms:

Many static algorithms do exist, where five of them have been taken in this survey, and which comprise:

a) The Dynamic Round Robin algorithm: It is similar to the Weighted Round Robin algorithm, however, a weight is assigned for each server dynamically, based on real-time data of the server's current load and capacity. The weights are continuously changing [7].

b) The Least Connection and Weighted Least Connection algorithm [2] [3]. In this method, the load balancer sends a new request to the server with the least number of concurrent connections. The least connection method takes up the current server load into consideration. Thus, the load balancer needs to keep track of the total number of the concurrent active connection. In the Weighted Least Connection algorithm, each server has a numerical value to allocate the requests to the servers. If two servers have the same number of active connections, then the load balancer sends the request to the server with a higher weight.

c) The Response Time and Weighted Response Time algorithm [2] [3]. This method is based on the response time of the entire servers, and is based on how fast they respond and serve. The load balancer sends requests to the server by providing the fastest response time. Nonetheless, the response time should be first measured by health checks. The load balancer uses either an in-band monitoring, or an out-band monitoring. In the in-band monitoring, the load balancer uses a natural traffic flow between the client and the server in order to measure the response time, but in the out-band monitoring, the load balancer explicitly generates a request to the server in order to measure the response time. In particular, the response time must be measured over the time in order to ensure providing an effective load balancing. It must give more weight to the most recent response time. In the Weighted Response Time algorithm, the information is taken from the server health checks.

d) The Throttled Load Balancer algorithm [14]: the load balancer have an indexed table for all Virtual Machines VMs. The index table contains two parameters. One is to identify the ID of the VM, and one is to provide the status of that VM in the form of 'Available' or 'Busy'. Initially, all states are in the form 'Available'. Any request should be sent to the Data Center Controller (DCC) in order to find a suitable VM. The DCC asks the load balancer to find the next allocation by parsing the index table from the beginning until it finds an available VM. If the VM is found, the DCC assigns the task to the VM by an ID, and the load balancer updates its state. On the other hand, if the VM is not found, the load balancer returns -1 to the DCC. Then, the DCC will put this job in a queue. When the VM ends the job, and the DCC receives the response, it will notify the load balancer as a request in order to de-allocate the same VM whose id is already communicated. The DCC checks if there are any waiting requests in the queue, and so forth.

e) The Greedy algorithm [12]: is an algorithm that uses many iterations in order to compute the best results at the moment that makes an optimal choice and solutions. In fact, it always selects the best site to execute the job according to the least workload, and to the least queuing time. In this algorithm, an indexed table is maintained for nodes with current allocated jobs. In addition, it makes one pass the jobs in any order, and assigns Job j to the computing node n that has the minimum load. This allocation is based on an expected time of job completion over the entire nodes. Moreover, it is suitable for heterogeneous systems.

Table (2) shows merits and demerits of each algorithm.

IV. HYBRID LOAD BALANCING ALGORITHMS

These algorithms are achieved and are proposed to eliminate the drawbacks of dynamic and static load balancing methods, and they are being used to aggregate the benefits and merits of static and dynamic algorithms in order to design a new one[6]. In fact, this implies that combinations the benefits of two or more existed algorithms either dynamic or static algorithms are able to present a new one.

A. The features of the Hybrid Load Balancing Algorithms.[6][13]

- Hybrid methods reduce the response time.
- They are more scalable.

TABLE I. COMPARSIONS OF THE STATIC ALGORITHMS

Algorithm	**Merits**	**Demerits**
Round Robin algorithm	1- Is simple to be implemented by using a circular queuing process. 2- It does not require an inter process communication. 3- It provides the best performance only for a special purpose application.	1- It cannot provide good results in the general case, and when jobs are unequal. 2- It assumes that all servers are similar in their storages, and in their response times 3- It does not give the priority for more important required tasks.
Weighted Round Robin algorithm	1- Is simple to be implemented by using multiple entries of a circular queuing process. 2- It achieves a better response time and processing time in comparison with the Round Robin algorithm.	1- It does not have an information of states of nodes.
Randomized algorithm	1- Is simple to be implemented by using an array of severs that are being load balanced, and it uses a random number generator. 2- This algorithm works well when the processes are equally loaded. 3- It provides the best performance only for a special purpose application.	1- It may sometimes cause a single overloaded node, while the other is under loaded. 2- There is a problem when loads are of different computational complexities. 3- It does not achieve a good performance in the general case.
Central Manager algorithm	1- Is of a low cost communication. 2- It is only suitable for a special purpose application.	1- Is of a single point failure.
Hash IP address algorithm	1- It is simple and easy to be implemented. 2- It ensures that connections within existing user sessions are consistently routed to the same back-end servers. 3- It ensures an improved performance in situations where a single virtual machine communicates with multiple virtual machines. A virtual machine can use more bandwidth in comparison with other machines. 4- Hashing presents some issues when a server goes down, and when it redistributes.	1- Many clients hide a single IP address, or has multiple IP addresses. 2- Some servers may receive more clients request in comparison with others that have light loads.

TABLE II. COMPARISON OF THE DYNAMIC ALGORITHMS

Algorithm	**Merits**	**Demerits**
The Dynamic Round Robin algorithm	1- It minimizes the response time. 2- It has the maximum throughput. 3- It always uses the recent load information.	1- It needs an extra traffic.
The Least Connection and Weighted Least Connection algorithm	1- It is an effective method for many applications. 2- It is simple and easy to be understood. 3- It is even a load distribution. 4- The Weighted Distribution algorithm achieves a better processing time, and controls different capabilities of servers.	1- It provides no persistence between the client and the server. If the same client sends a simple web page, it might then have its first request to Server 1, its second request to Server 2, and so forth.
The Response Time and Weighted Response Time.	1- It is useful in environments where servers are distributed across different logical networks.	1- It has more complexity. 2- It includes software on each server to feedback the information about the ongoing performance.
The Throttled Load Balancer algorithm	1- It works more efficiently in terms of the cost for load balancing on cloud data centers in comparison with other algorithms.	1- It works properly only when the hardware configuration of the entire VMs of the data center have similar hardware configurations.
The Greedy algorithm	1- It is suitable in heterogeneous systems. 2- It can be applied in different greedy heuristic task allocations.	1- The Greedy algorithm does not always yield to an optimal solution.

- They provide an efficient usage of a resource.
- The major drawback is the inability to provide non-complex techniques.
- Hybrid methods inherit the properties from both static and dynamic load balancing techniques, and attempts at overcoming the limitation of both algorithms.

B. Hybrid Load Balancing Algorithims.

Many static algorithms do exist, where three of them have been taken in this survey, and which comprise:

a) The Least Load and Round Robin algorithm [13]: In this algorithm, functionalities from both the Least Load and Round Robin algorithms are inherited, where this inheritance makes this algorithm simple, fast, and efficient. This method reports on the server current load to the load balancer before sending the request to the corresponding server, and monitors the previously selected server. In fact, this assists the selected server to avoid participating in next server load decision. The sparse requests can be easily distributed among the *n* servers more evenly.

b) The Throttled and Round Robin algorithm [6]: In this algorithm, the authors propose combination load balancing algorithms and brokering algorithms (intermediary agent). They merge the Throttled and the Round Robin VM load balancing techniques with an optimized performance and the Brokering algorithm. It is experimented in a cloud environment. The nearest data center brokering policy chooses the data center that is closest to the user's region. This policy chooses the nearest available data center that is based on the network latency. The Round Robin algorithm distributes the user's requests as the Round Robin's fashion. On the contrary, the Throttled Load Balancing algorithm policies maintain a table for the entire available VMs. It is inferred from the results that the combination of the closest data center brokering policy and the Throttled Load Balancing Policy algorithm require the minimum processing time, and the combination of the Brokering policy and the Throttled Load-Balancing Policy leads to encounter the lowest response time. However, the execution time for algorithms is high.

c) The Random and Greedy algorithm [4]: In this algorithm, the authors propose a combination of the random and greedy methods. It is experimented in heterogeneous cloud environment. The authors inherit the advantages of the random and greedy algorithms in order to design an efficient load balancing. This algorithm considers the current resource factor and the CPU capacity. First, the virtual machines VMs are distributed over the hosts according to the hosts of the CPU capacity. The most qualified host has the largest number of VMs. Second, the algorithm uses a new indexed table in order to record the current loads of each VM, and to check the current load for each iteration. When any request is arrived to the data center, it sends to the hybrid load balancer an instruction to select *k* nodes (VM) with their current loads in a random manner. Then, the load balancer selects a VM with the least current loads, and returns a VM id to the data center. The data center updates the indexed table of the current nodes. This algorithm improves the average response time, and the average processing time in comparison with other non-hybrid algorithms.

V. Conclusion and Future Work

In this paper, the most used algorithms for Load balancing are introduced. A comprehensive study is carried out, where the comparisons are conducted among static, dynamic, and hybrid algorithms. The results show that the hybrid algorithms give efficient results since they inherit the benefits of other algorithms, and avoid drawbacks. Finally this work could be of great help for future researchers who might be interested in this field.

Acknowledgment

The author would like to thank Al-Zaytoonah University of Jordan for its support this survey.

References

[1] A. Al-Dahoud, M. A. Belal, and M. B. Al-Zoubi, "Load Balancing of Distributed Systems Based on Multiple Ant Colonies Optimization", American Journal of Applied Sciences Vol.7 (3), 2010, ISSN 1546-9239, pp. 428-433.

[2] C. Kopparapu, Load balancing Servers, Firewalls, and Caches, Published by John Willy and Sons Inc.,2002, pp.23-35..

[3] M. Syme, and P. Goldie, , Optimizing Network Performance with Content Switching Server, Firewall, and Cache Load Balancing, Publishing as Prentice Hall Professional Technical References, 2004, pp.96-102.

[4] H. J. Younis, A. Halees, and M. Radi, "Hybrid Load Balancing Algorithm in Heterogeneous Cloud Environment", International Journal of Soft Computing and Engineering (IJSCE) ISSN: 2231-2307, Volume-5 Issue-3, July 2015, pp.61-65.

[5] D.Probhuling L., "Load Balancing Algorithms in Cloud Computing", International Journal of Advanced Computer and Mathematical Sciences ISSN 2230-9624. Vol4, Issue3, 2013, pp.229-233

[6] S. Milani, N. J. Navimipour, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends", Journal of Network and Computer Applications Vol.71, 2016, pp.86–98.

[7] A. Gulati, and R. K. Chopra, "Dynamic Round Robin for Load Balancing in a Cloud Computing", International Journal of Computer Science and Mobile Computing, A Monthly Journal of Computer Science and Information Technology ISSN 2320–088X IJCSMC, Vol. 2, Issue. 6, June 2013, pp.274-278.

[8] [8] P. Gautam1 and R. Bansal, "Extended Round Robin Load Balancing in Cloud Computing", International Journal Of Engineering And Computer Science ISSN: 2319-7242, Vol.3 Issue 8 August, 2014 pp. 7926-7931.

[9] P. Beniwal and A. Garg, "A comparative study of static and dynamic Load Balancing Algorithms", International Journal of Advance

Research in Computer Science and Management Studies, Vol.2, Issue 12, December 2014, ISSN: 2327782, pp.386-392.

[10] S. khan, and N. Sharma, "Ant Colony Optimization for Effective Load Balancing In Cloud Computing", International Journal of Emerging Trends & Technology in Computer Science (IJETTCS), Vol. 2, Issue 6, November – December 2013, ISSN 2278-6856, pp.77-82.

[11] S. Sharma, S. Singh, and M. Sharma, "Performance Analysis of Load Balancing Algorithms", World Academy of Science, Engineering and Technology ,International Journal of Computer, Electrical, Automation, Control and Information Engineering Vol.2, No.2, 2008, pp.367-370.

[12] . Sahoo , D. Kumar, and S. K.Jena, "Observing the Performance of Greedy algorithms for dynamic load balancing in Heterogeneous Distributed Computing System", 1st Int. Conf. On Computing, Communication and Sensor Networks-CCSN'2012.

[13] R.Saini, and A. Bisht, "A Hybrid Algorithm for Load Balancing", International Journal of Advanced Research in Computer Science and Software Engineering, Vol.5, Issue 7, July 2015 ISSN: 2277 128X. pp.1-6.

[14] H. Singh, and R. C. Gangwar," Comparative Study of Load Balancing Algorithms in Cloud Environment", International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169 Vol.2 Issue.10. pp. 3195- 3199.

AUTHORS PROFILE

Siham Hamadah has received her M.Sc degree in the Computer Science field from the University of Jordan in 1998. She is currently working as an instructor at Al-Zaytoonah University of Jordan. Her research interest involves databases and operating systems fields. Email : siham@zuj.edu.jo.