Asian Journal of Research in Computer Science

10(1): 47-62, 2021; Article no.AJRCOS.70184
ISSN: 2581-8260

Web Server Performance Improvement Using Dynamic Load Balancing Techniques: A Review

Ibrahim Mahmood Ibrahim[1*], Siddeeq Y. Ameen[1], Hajar Maseeh Yasin[1], Naaman Omar [1], Shakir Fattah Kak[1], Zryan Najat Rashid [2], Azar Abid Salih[1], Nareen O. M. Salim[1] and Dindar Mikaeel Ahmed[1]

[1]Duhok Polytechnic University, Duhok, Kurdistan Region, Iraq.
[2]Sulaimani Polytechnic University, Sulaimani, Kurdistan Region, Iraq.

Authors' contributions

This work was carried out in collaboration among all authors. All authors read and approved the final manuscript.

Article Information

DOI: 10.9734/AJRCOS/2021/v10i130234
Editor(s):
(1) Dr. Dariusz Jacek Jakóbczak, Koszalin University of Technology, Poland.
Reviewers:
(1) Bikramaditya Das, VSSUT, India.
(2) N. K Choudhari, Priyadarshini Bhagwati College of Engineering, India.
Complete Peer review History: http://www.sdiarticle4.com/review-history/70184

Review Article

Received 18 April 2021
Accepted 22 June 2021
Published 22 June 2021

ABSTRACT

Today, web services rapidly increased and are accessed by many users, leading to massive traffic on the Internet. Hence, the web server suffers from this problem, and it becomes challenging to manage the total traffic with growing users. It will be overloaded and show response time and bottleneck, so this massive traffic must be shared among several servers. Therefore, the load balancing technologies and server clusters are potent methods for dealing with server bottlenecks. Load balancing techniques distribute the load among servers in the cluster so that it balances all web servers. The motivation of this paper is to give an overview of the several load balancing techniques used to enhance the efficiency of web servers in terms of response time, throughput, and resource utilization. Different algorithms are addressed by researchers and get good results like the pending job, and IP hash algorithms achieve better performance.

Keywords: Web server; load balance; cluster; NginX server; apache sXerver.

**Corresponding author: E-mail: mohammed.abdulrazaq@dpu.edu.krd;*

1. INTRODUCTION

With the increasing development of computer technologies with the Internet, Internet service is an exciting difference in individual and community relationships and a revolution for all forms of industries to accelerate their transformation and modernization [1, 2]. The extent of Internet use as an information exchange has been noted to be rising, leading several organizations to allocate website construction to provide reliable web server services [3, 4]. The majority of websites are updated from static pages to dynamic interactive pages. Higher specifications are required to perform different software systems, widespread social networks, e-commerce, and video sites [5, 6]. Most internet users depend on web services to share information and accomplish much of their daily routine [7, 8]. Therefore a load of web servers radically increased as they are basic architectures for hosting and designing web pages and applications [9, 10]. A web server can host many websites for education, business, or technology uses. The number of users is rising dramatically due to intensive growth in the website domain [11, 12]. As a result, the number of site requests on the web servers has been reached unsustainable and gets overloaded and slowly responds [13, 14]. Hence, a solution is essential to handle simultaneous requests from users using cluster technology on a web server [15, 16] and using several servers to build a network-providing cluster framework user program [17, 18]. All servers share a max number of requests from users within the cluster, thus preventing the bottleneck on the server [10]. Servers in the collection can be seen as a single unit by users. Thus increasing or decreasing the number of servers in the cluster will alter the distribution of workload among available servers without affecting the interaction of users with the system [19-21].

To distribute equal workload to all web servers in the cluster and avoiding bottleneck, need to use a load balancing technique. Load balancing refers to a situation in which the load on each server is roughly equal [22]. First, distribute the load on the servers and then check the current limitation for some form of more distribution. If shipments are evenly distributed, then they are balanced [23-25]. To improve the performance of distributed web servers, the load balancing method is required. To achieve the best version of the distributed Web servers, client requests should be distributed through the DNS between the web servers in the cluster according to a load balancing strategy to serve the client as a static load balancing device within the best time, and the load of the overloaded web server must be allocated to another under loaded webserver to improve the performance of the cluster and maximize resource utilization [26, 27].

Nowadays, the Internet is flooded with massive traffic, and many applications have millions of users. Because a single server can't handle many such clients, many application providers will group several servers as a computing unit to support a single application [28]. Most people will use distributed computing and load balancing technology to complete the task. The employment of a dedicated load balancer to redirect client requests to multiple servers is a standard load balancing strategy [29]; however, this technique necessitates specific hardware support, which is costly, inflexible, and prone to becoming a single point of failure [30].

The rapid popularization and growth of the World Wide Web have resulted in an increase in users and the production of various services in recent years [31]. As a result, the Web server's workload has increased. Due to increased traffic, the Web server is unable to provide adequate service regularly [32]. When this happens, the user complains, and the service provider loses their reputation. In most cases, a server load balancing strategy is used to solve the problem of server overload. However, it is impossible to provide adequate computer resources to eliminate customer complaints due to operating costs [33].

Client-trafficked websites cannot simply rely on mirrored servers or a single server to balance the load created by client requests [34]. The benefits of DNS load balancing solutions in dealing with high Web traffic have been demonstrated [35]. The time-to-live (TTL) value associated with a name-to-address translation is used in several procedures [36]. Unfortunately, name-to-address translations are cached for a period determined by the TTL in intermediate name servers. For the duration of the TTL period, all requests will be directed to the same Web server [37].

The Web's scalability and availability can be improved by dispersing Web servers (Web Clusters), with client requests being balanced among these Web servers to improve performance [38]. The DNS-based load balancing system has been one of the most

challenging problems to solve in recent years [39].

As the amount and hours of movies available within a company grows, so does demand, necessitating rapid video streaming services [40]. Cloud-based services are not cost-effective and are not a viable solution for storing the growing amount of video data often saved and used only within a single organization, such as a university [41].

For clustered web applications, dynamic application placement has a significant impact on system performance and user experience [42]. Existing approaches claim to maximize throughput, balance resource use among servers, and reduce the cost of starting and stopping application instances [43]. However, they fail to reduce server utilization in the worst-case scenario; load balancing performance is not optimal. Furthermore, some apps, which we refer to as dependent applications, must connect; the network cost of these applications must also be considered [44].

Distributed key-value systems (e.g., Memcached) are essential tools for caching popular content in memory, saving time and money by avoiding complex and expensive database searches and file system visits [45]. It's critical to balance the load across a cluster of cache servers if you want to make the most of your cache resources [46]. Current methods use a proxy at each client to redirect requests across the collection, but this necessitates client modification and makes essential dynamic replication problematic. A centralized representative can be utilized. However, this hasn't always proven scalable [47].

The Distributed Web Caching method allows for quick retrieval of web pages, although server delays still restrict it [48]. Due to highly congested servers, these systems have a low level of robustness. Servers are frequently disconnected, just as they are in real-time, resulting in a service tradeoff. Robust Distributed Web Caching (RDWC) takes care of the robustness but not the frequent disconnections [49].

Cluster administration is a time-consuming task. Allocating cluster resources by hand, in particular, can soon become unmanageable in a dynamic environment like the Internet, where processing requirements can vary rapidly [50]. The use of a dynamic architecture for cluster self-reconfiguration is one solution to this challenge [51].

By replicating material, data-sharing scientific communities use storage systems as distributed data stores [52]. The same dataset can exist in numerous locations in such highly duplicated settings and can thus be retrieved from any of them [53]. Because the datasets of interest are typically massive, increasing download speeds through server selection or co-allocation can provide significant benefits [54].

Massive Multiuser Virtual Environments (MMVEs) are currently attracting a lot of interest. Beyond the commotion, some exciting notions and potential for a supposedly future Web dubbed the 3D Web [55]. The global distribution of such a vision is considerably more than just another evolutionary step in content visualization; instead, future Internet infrastructure and protocols must address severe technical hurdles [56]. Due to their reliance on centralized solutions, existing commercial systems have limited scalability. When considering global-scale scenarios such as the 3D Web, distributed and decentralized approaches become essential.

Cloud computing results from the convergence of some technologies that have changed the way businesses create IT infrastructure. Grid computing, cluster computing, software-as-a-service, utility computing, autonomic computing, and other computing approaches are incorporated. It introduces an entirely new deployment mechanism for enterprise web apps [57]. When opposed to employing an internal IT infrastructure, the cloud offers significant cost savings. Cloud computing's "pay as you go" concept is substantially less expensive for a firm than internal IT's "pay for everything upfront" methodology [58]. Hardware Many cloud infrastructure vendors' products rely on virtualization as an enabling technology. A physical server can be partitioned into many virtual servers, each with its operating system and memory, CPU, and disk footprints, thanks to virtualization [59].

The rapid rise of the Internet and the confluence of computers and telecommunications have reshaped business regulations and the way people communicate information. Users have assured timeliness and reliability expectations for information service use in a varied and developing Internet environment due to the rise

of e-commerce [60]. These consumers' needs cannot be met by information systems based on traditional client/server database technology [61]. This necessitates creating an information service system that can meet users' diverse service expectations while also ensuring adaptability to deal with ever-changing events to give high assurance [62].

Nowadays, the most common distributed computing environments are Jini and.NET. The Jini architecture provides a framework for defining, promoting, and discovering services in a network, while.NET remoting allows developers to create Internet-based, distributed applications. Both frameworks are capable of facilitating the implementation of distributed systems and are incredibly comparable in many ways [63]. They are, however, incompatible. Jini clients cannot request remote services from.NET systems, and.NET clients are unable to locate Jini services. Furthermore, no load-balancing methods are available in any of these situations [64].

Routing is a crucial function that occurs at the network layer in any IP communication network. Anycasting is a new mechanism for IP packet delivery from a sending node to any of a group of receiving nodes with the same IP addresses introduced lately [65]. The sender will receive a response from any one of the group's nodes. The introduction of IPv6 made it possible to locate services using the anycasting approach, which was not possible in IPv4. In most cases, packets are routed using either a static or dynamic routing technique [66]. Single path routing (SPR), multiple-path routing (MPR), and integrated routing (SPR&MPR) approaches can be used to route packets [67].

In large-scale distributed virtual environments, scalability and consistency are critical concerns (DVEs). A VE is usually partitioned into several regions and handled by a series of dedicated servers to make it scalable [68]. Because static arrangements can't handle dynamic loads, dynamic methods are being researched and tested extensively [69]. A multitude of solutions addressing either scalability or consistency may be discovered in the literature, and systems requiring both demanded a new infrastructure [70].

Satellites all across the world generate enormous amounts of data regularly. The administration of such massive amounts of data presents some challenges, including data storage, retrieval, and manipulation [71]. The existing situation for managing the Indian Space Research Organisation's (ISRO) ASTROSAT-CZTI satellite data involves an elementary and unsophisticated technique, which results in significant data retrieval delays and data portability concerns [72]. The complication, as mentioned above's evaluation and research paved the path for us to develop a new architectural solution to handle this complex optimization challenge [70].

The remainder of this article is structured as follows: Section II Web server, section III Load balancing technology, section IV literature review, section V discussion and comparison, and section VI is the conclusion.

2. WEB SERVER

A web server is a machine that can reply to client requests. The response is regarding websites. When software is installed and connected to the Internet, a computer can become a web server [73, 74]. The web server is responsible for accepting and serving HTTP requests from web clients, usually in web pages containing static content (text, images, etc.) and dynamic content (scripts). Therefore, the leading role of the webserver is to store and process files and provide clients with web page requests via a communication protocol among clients and the server that uses the Hypertext Transfer Protocol (HTTP). The page submitted is a Hypertext Markup Language (HTML) document. Many web servers depended on the Apache web server, which is the most commonly used web server, is one of the most popular since Apache can be used. It is an HTTP web server and has access control, PHP, and SSL support [75-78]. The most popular web servers are Apache Web Server, Nginx, and NodeJS. The Apache web server is a Unix-based, accessible, and open-source web server built by the Apache Software Foundation. Apache is lightweight, wholly featured, and more powerful than other web servers based on Unix. The architecture of Apache is thread-based, where the primary process (Multi-Processing Modules-MPM) is named at startup and performs child processes/threads (modules) to handle requests simultaneously [79, 80]. Nginx, developed by Igor Sysoev and then by NginX Inc. in 2004, is a free, open-source, and high-performance web server. To benefit effectively from the hardware resources available, NginX often uses multi-processes.

Together with the cache loader, a master process is called at startup to load disk-based cache in memory and cache manager to monitor memory utilization. This design seeks to reduce the impact of context switching in the architecture of multi-processes. NodeJS is a JavaScript single-threaded server-side environment created in 2009 by Ryan Dahl. With a scalable architecture based on events, NodeJS can achieve a high concurrency level. However, for non-blocking I/O operations, NodeJS also uses several threads. Many of NodeJS's main modules are written in Java script and run on the Google V8 Java script Engine. Thus, NodeJS's entire code path is asynchronous and non-blocking [76, 79, 81, 82].

3. LOAD BALANCING TECHNOLOGY

Web traffic has generally affected the performance of any web server, and it makes a slow response of web server due to get overloaded. Each web server faces the problem of overloading and needs an optimal solution. So to overcome this, using load balancing techniques [83, 84]. It acts as a transmitter between the requests of the client and the web servers by distributing the incoming requests to all the nodes within the cluster to improve the use of speed and resources and ensure cost-effective and scalable [85, 86]. It is also used to provide high performance by preventing each server from being overloaded inside the cluster. In addition, the load balancer is a means of software or hardware designed to balance traffic across various servers and brilliantly share client load across several nodes to one real IP [87, 88]. Load balancing extends the bandwidth of network devices, and servers improve throughput and improve network data processing capabilities [89]. To ensure high efficiency and high-performance computing of the system, some reasonable resource allocation is based on the load pressure [90, 91]. In addition, server load balancing provides applications, cloud services, and websites with scalability and high availability by monitoring server health, distributing loads equally among servers, and maintaining session persistence if one or more servers are overloaded or unresponsive. Several load balancing algorithms are used to balance the workload, such as round robin, Least Connections, and Weighted Least Connections [92, 93]. The Load balance can be divided into software and hardware balancing methods. The first technology (Software) is done by installing on one or many nodes (servers) the corresponding additional software and performing the complementary software load balancing technique. The benefits of the first technology are easy to configure, flexible use, no external device needed, and low cost.

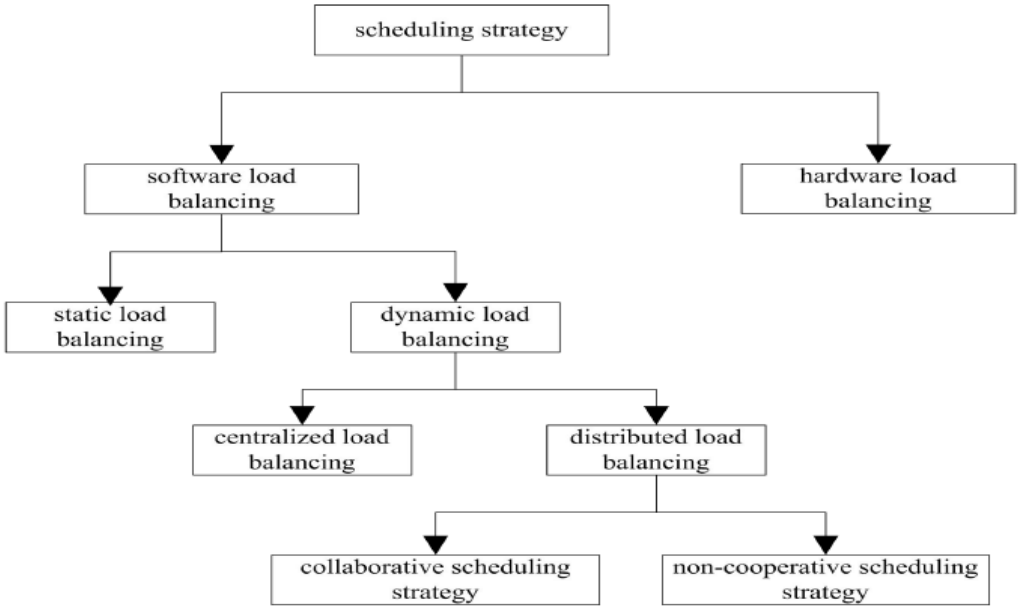

Fig. 1. Classification of load balancing [20]

The drawback of the software load balancer is the program itself can consume some machine resources. The problem of the resource usage of the program itself will also become a bottleneck of the load capacity of the device when the number of requests rises to a certain degree. In general, loading in the form of hardware installs the load balancer directly to the server and the output connected to the external network. It is independent of the OS and uses special installation facilities, which dramatically improves enhancement. The drawback is that the cost of hardware upgrades is costly [19, 94]. The classification of load balancing illustrates in Fig. 1.

4. LITERATURE REVIEW

In 2017, Pan Zhu and Jiangxing Zhang [95], Improved Load Balancing approach for Web Server Cluster Enhanced Weighted Minimum Join. Firstly, the algorithm considers the usage of the server, memory use, and mutilation of network bandwidth. Other variables are combined with the web server's properties to dynamically change the weight's size according to the server's real-time load. Secondly, assigns the new connection request according to the weighted minimum connection number algorithm. The algorithm reduces server latency, decreases HTTP response time, and can effectively increase load balancing performance using OPNET simulation tools.

In 2017, Peirui JIANG and et al. [96], a novel load balancing algorithm was suggested that considers the proximity between client and server. The Client Proximity Based Load Balancing Algorithm (PBLB) is compared with the Modified Random Allocation (MRA) method and the Round Robin (RR) scheme. During the simulation, the PBLB algorithm has a higher performance as compared to MRA and RR algorithms.

In 2017, Sanjaya Kumar Panda and et al. [97] proposed (SIS) algorithm to distribute and balance the load among servers, using a three steps process. The SIS algorithm is compared with the client/server approach, and the performance is evaluated according to the number of load comparisons and the load factor. The obtained result shows the performance of the algorithm proposed.

In 2018, Jie Li and et al. [98] A DCH algorithm based on consistency hash were suggested to improve the load balancing capability of the webserver. First, for all the performance indicators of each server in the cluster, the quantitative values are specified. The initial virtual node set is measured according to the quantization value, optimizing the unequal load distribution caused by differences in the server's performance. Then the load is refined, and the virtual node is dynamically calculated based on the measured server's performance value and load value so that the cluster load is more balanced. The result shows that the proposed method minimizes the average response time and maximizes throughput.

In 2018, Putu Adhi Suwandika and et al. [99] proposed Least Connection and IP Hash load balancing algorithm to be applied on web servers over SDN networks. Both algorithms will be evaluated with some basic parameters for a web server. Response Time, Throughput, and Resource Utilization are these parameters. The parameter test results show that the IP Hash algorithm provides 17% more optimal performance response time, 10% more optimal throughput, and 8% more effective memory use than the Least Connection algorithm.

In 2018, Guoqing Liu and et al. [100], A new load balancing algorithm was proposed that considers the nodes' load and performance information and the type of request to solve the problem of unequal load distribution in cluster servers. The discussed approach achieves a relatively optimum load balance compared to the traditional Round-Robin algorithm.

In 2018, Gurasis Singh and Kamalpreet Kaur [101] enhanced the weighted least connection algorithm, which manages the load among different servers by preventing requests to the new server. The proposed algorithm excludes and disables a new server from its scheduling list when HTTP requests are continuously allocated to only a new server above the maximum continuous assigned number (C). Finally, after C-1 allocation round times, the new server is activated and included in the server scheduling list. The suggested algorithm balances the load between real servers by preventing overloads on the latest real server.

In 2018, Iqsyahiro Kresna A and Yusep Rosmansyah [102] implemented a web server cluster and load balancer as a high availability framework, which is expected to overload the server problem. The results of the load balancer

test showed that the LC algorithm was outperforming the RR algorithm. In High Availability or failover experimenting, the average downtime is 6,587 seconds.

In 2018, Zepeng Wen and et al. [103] presented a dynamic feedback load balancing algorithm based on Nginx. The proposed algorithm is realized based on queuing theory and through the plug-in mechanism of Nginx by developing an algorithm. A test environment was then designed. Loadrunner software is used to perform a comparison test on the efficiency of the Weighted Round-Robin Scheduling, IP Hash algorithm, and the dynamic feedback algorithm built-in by Nginx. The results showed that the load balance could be better with the dynamic algorithm used in this work.

In 2018, Deepti Sharma [104] presented a load balancing approach that allows HTTP requests to be distributed dynamically among all servers in the cluster depend on the response time. In this work, the load balancer performs on a lightweight server that is easily integrated with java. Requests come from two sources, either from the browser or from another tool called SoapUII. A track of HTTP requests and responses on all servers is kept in the hash data structure. Due to the response time and pending workloads for that server, a new request is distributed on web servers. Different experimental results show the efficiency of the proposed algorithm.

In 2018, Hsien-Yi Liu and et al. [105] suggested an Open Flow-based Least-Loading load-balancing technique to make full use of hardware resources. The proposed algorithm allocates requests to the server and every node in the server pool's real-time status. Furthermore, minimizing the overhead of choosing the request managing server and essentially increasing the entire system throughput, the least loading strategy prevents the controller from being a system bottleneck. Experimental findings show that the Lest Loading strategy is 17.2% and 21.4% better than RR and WRS algorithms in the Floodlight controller.

In 2018, Anji Yu and Shimin Yang [106], a new approach that distributes a load on the server's algorithm, is suggested to minimize the execution time. The algorithm's strategy is to divide the degree of effect on the request client's server and assign the corresponding weight. The performance and load of the server are combined at the same time to transfer the request to the lightest pack on the webserver. Obtained results illustrated that the algorithm increases software and hardware resource usage and balances the load among servers.

In 2018, Ruoyu Li and et al. [107], Analyses the container-based WEB application deployment architecture, measures the host load parameter combined with the performance of the host indicates and operating container status, including Processor use, memory use, network use, and the amount of unused memory assigned to the containers, and suggests a Dynamic Weighted Least-Connection Algo (DWLC). The results show that the DWLC algorithm is 52.6 % and 46.4 % faster than the ordinary Round-Robin and Least-Connection algorithm for the WEB application response.

In 2018, Fatma Mbarek and Volodymyr Mosorov [108], load balancing was dedicated to handling the web cluster for distributed and parallel systems. Using several scheduling algorithms, it distributes the load across web servers. This study discusses a detailed comparison of several heterogeneous web cluster HAProxy algorithms and the metaheuristic methods inspired by the behavior of the insect colonies, such as the Ant Colony Optimization algorithm and the Honey Bee algorithm. Solving combinatorial optimization problems is the fundamental concept of meta-heuristic methods.

In 2018, Minato Omori and Hiroaki Nishi [109] proposed a load balancing algorithm that distributes the request based on processing time estimates, which prevents mismatches between server characteristics and the content of the request. Based on the requested content, the processing time for submissions is estimated by online machine learning. A technique to cover the latency of machine learning is proposed and partially carried out. They developed a model of multiple database servers to test the algorithm and performed an experiment using actual log data for database requests. The simulation results indicate that, compared to round-robin, the proposed algorithm decreased the average processing time for requests by 94.5 % and by 28.3 % to the least connections.

In 2018, Mochamad Rexa Mei Bella a dit al [110], A method was proposed to track each host machine's memory utilization and distribute web traffic based on the memory utilization of each host machine. Promising is the outcome of this experiment. Each of the worker nodes receives

Table 1. Summary related to the previous study (webserver with load balancing technique)

Ref	Year	Server Name	No. of server	Algorithm	Parameter	OS	Tools	Finding
[95]	2017	N/A	4	ADWLS algorithm	Responding time, Delay, CPU utilization.	N/A	OPNET	Minimize HTTP responding time, deducing the server delay, balance each server's CPU utilization.
[96]	2017	N/A	4	PBLB algorithm	Response time and load balance	N/A	MATLAB	PBLB has better load balancing, shorter response, and performance as compared to RRLB, MRA algorithm
[97]	2017	N/A	4	SIS Algorithm	Load balance	Microsoft Windows 10, 64-bit	MATLAB	The proposed algorithm outperforms NOC and LF in the term number of comparisons and loads factor.
[98]	2018	N/A	3	DCH Algorithm	response time, throughput	Linux	JMeter	Improve the throughput of the system and improve the performance of the cluster system as a whole.
[99]	2018	N/A	3	Least Connection and IP Hash algorithm	latency, Throughput, Resource Utilization	Linux	Iperf, Htop, wget	Hash IP algorithm is more potent than the Least Connection algorithm in terms of resource usage, latency, and throughput.
[100]	2018	N/A	4	MRRBC Algorithm	CPU, Memory, Disk I/O, and bandwidth	Linux	Httperf	MRRBC achieves a relatively optimum load balancing than result compared to the standard Round-Robin algorithm.
[101]	2018	Nginx	3	improved WLC algorithm	Throughput, execution time	N/A	Docker	the proposed algorithm maintains load balance among real servers by preventing overloads on the new real server
[102]	2018	Apache2	3	RR algorithm and Least Connection algorithm	throughput, response time, request loss, and most extended transactions	Ubuntu 64 bit	Docker	The server computer's best transaction rate and response time result from its much greater resources than the Desktop PC.
[103]	2018	Nginx	3	dynamic feedback load balancing algorithm	Response time, throughput	Windows	LoadRunner	The results showed that the dynamic feedback algorithm is highly efficient and feasible, enhancing the server cluster efficiency to some extent.
[104]	2018	Tomcat	3	Proposed algorithm	Response time, total processing	N/A	SoapUI	Transactions handled per sec of the proposed approach outperform with 23% as

Ref	Year	Server Name	No. of server	Algorithm	Parameter	OS	Tools	Finding
				(pending jobs algorithm)	time, throughput transactions, and server utilization.			RR Algorithm. And total processing time is 18.5% better as compared to RR Algorithm.
[105]	2018	Apache2.3	3	Lest Loading Algorithm	CPU, memory, and disk utilization	Ubuntu	real environment and JMeter	In the homogeneous environment, there is no difference in the throughput of LL, RR, and WRS. Whereas, in a heterogeneous environment, the throughput of LL is 17.2% better than that of RR and 21.4% better than that of WRS.
[106]	2018	Nginx	4	dynamic weighted polling algorithm	User request response time, response per second, and resource utilization	Linux	Httperf	Compared to the static weighted polling algorithm, the dynamic weighted polling algorithm increases the actual number of concurrent peaks.
[107]	2018	Nginx	5	DWLC algorithm.	Response time.	Ubuntu14.04	Iperf	The proposed algorithm outperformed than RR and LCA in the term of response speed in WEB cluster
[111]	2019	Apache	3	WSQ algorithm	throughput, drop rate, response time, CPU, Memory, and Disk I/O.	N/A	N/A	The proposed algorithm has high throughput, less drop rate, and minimized mean response time in a heterogeneous environment.
[112]	2020	Nginx	3	Improved Dynamic WRR Algorithm	Response Time, the number of connection, throughput	Centos 7 64 bit	Siege.	The proposed algorithm outperforms the fair and dynlb in the term of response time, the number of connection and throughput

fair load distribution. This technique can minimize the probability of a single point of failure in the web server cluster.

In 2019, Kadiyala Ramana and M. Ponnavaikko [111], An approximate web server queuing technique was introduced for web server clusters and an analytical model for web server load calculation. To achieve better efficiency, the requests are categorized according to the service time and track the number of outstanding requests on each webserver. The estimated load of each web server is used for load balancing. The research findings demonstrate the efficacy of the suggested approach by enhancing the server cluster's mean response time, throughput, and drop rate.

In 2020, E Qin and et al. [112]. Improved Nginx-based dynamic weighted round-robin algorithm, which collects real-time server load information and dynamically adjusted weight. The proposed algorithm is compared to weighted round-robin approaches in terms of throughput and response time. The obtained result demonstrates that the proposed method performs better than the weighted round-robin algorithm.

In 2020, WEI Chunlei and et al. [113] proposed an approach based on the socket buffer's feedback mechanism, which takes the server as the load indicator, collects the load indicator via the load agent, and calculates the discrete server socket buffer coefficient. The dispatch controller determines the weight of each server, and the higher the weight, the lighter the server load is. The tests illustrate that the CMTS feedback on the server load status is more realistic than the LC method. The average cluster response time is reduced as the number of cluster node connections and message sending rates to maximize.

In 2020, Mei-Ling Chian and et al. [114], Dynamic weighted random selection was proposed as a novel load balancing algorithm (DWRS). DWRS considers the real-time server loads while delivering requests to servers. Underutilized servers are given larger weights, increasing their chances of being chosen to process requests. Modify the packet handling flow in the Floodlight controller to improve system performance. When selecting the target server, a multi-threaded technique is employed to effectively exploit the parallel processing power of numerous cores, which speeds up the processing of packet-in messages.

The summary of the work performed by researchers is explained in the above section after applying load balancing algorithms to improve web server performance illustrated in Table 1.

5. ASSESSMENT AND DISCUSSION

This research aims to overview the dynamic load balance techniques to improve web server performance. The dynamic load balancing algorithm adopted will allocate the vast data request content equally to each server and enhance the server's concurrent processing capacity to shorten the average server response time and increase the best degree of adaptation. According to the previous studies related to improving the performance of web servers by using a dynamic load balance technique mentioned in the last section, many algorithms are used to enhance the work of web servers, and most of them achieve good performance.

The proposed algorithm (pending job algorithm) in [104] has been validated using a different number of requests ranging from 100 to 10000. The algorithm sets the server priority with the lowest response time and the lowest pending request. So each server will handle a different number of requests. This algorithm is analyzed based on two parameters. First, the proposed algorithm manages transactions per second, handling more transactions that achieve better efficiency. The second parameter is the total processing time, and the proposed algorithm takes less time to complete all jobs than other algorithms. The author in [99] depended on using two algorithms, IP hash and the least connection algorithms. The two algorithms are evaluated using three different parameters. The first parameter is response time, and the response time testing is done by using wget tools when 20 clients send a request to the server. With the IP hash technique, the average response time indicates the desired result. The second parameter is throughput. Perf tools tease the throughput of the IP hash algorithm. When implementing the load balancing approach with a predetermined algorithm, throughput testing is performed to evaluate the bandwidth efficiency on the webserver. The IP Hash algorithm shows stable performance on all servers. The third parameter is resource utilization. Htop tools do this parameter testing to retrieve current memory usage. Testing is done with ten clients up to 100 clients. According to the obtained result, the IP hash algorithm is more suitable than the

least connection algorithm for resource utilization.

6. CONCLUSION

Load balancing techniques are used with the web server cluster to enhance the performance of the webserver. This paper reviewed nineteen previous studies, and many algorithms get a better understanding and superior to other algorithms. It can be concluded that the Pending job and IP hash algorithms achieve better performance. The pending job algorithm is evaluated under 200 to 10000 dynamic requests. Total time processing is 18.5% outperforms the round-robin. In the term of transactions handled, the job pending is 23% better than round-robin. Also, the IP hash algorithm outperforms the minor connection in three different periods: First, the IP hash has less latency and has an average of 61 ms, while the Least Connection has an average of 73 ms. The second term is throughput; the IP hash gets an average of 1.23 Mbps, and the Least Connection receives an average of 1.11 Mbps. Third, in resource utilization, the IP hash uses resource memory with an average of 189.2 MB, while the Least Connection uses resource memory with an average of 203.8 MB.

7. FUTURE SCOPE

In load balancing, resource allocation efficiency strategy is critical. It can impact a variety of variables like cost, waiting time, response time, throughput, and other parameters. There has been a lot of work done in this area, but more research is still needed to improve the overall system's effectiveness.

DISCLAIMER

The products used for this research are commonly and predominantly use products in our area of research and country. There is absolutely no conflict of interest between the authors and producers of the products because we do not intend to use these products as an avenue for any litigation but for the advancement of knowledge. Also, the research was not funded by the producing company rather it was funded by personal efforts of the authors.

COMPETING INTERESTS

Authors have declared that no competing interests exist.

REFERENCES

1. Shang W, Liu D, Zhu L, Feng D. An improved dynamic load-balancing model. International Journal of Software Innovation (IJSI). 2017;5:33-48.
2. Haji SH, Ameen SY. Attack and anomaly detection in iot networks using machine learning techniques: A review. Asian Journal of Research in Computer Science. 2021;30-46.
3. Yasin HM, Zeebaree SR, Zebari IM. Arduino based automatic irrigation system: Monitoring and SMS controlling. in 2019 4th Scientific International Conference Najaf (SICN). 2019;109-114.
4. Yahia HS, Zeebaree SR, Sadeeq MA, Salim NO, Kak SF, Adel AZ, et al. Comprehensive survey for cloud computing based nature-inspired algorithms optimization scheduling. Asian Journal of Research in Computer Science. 2021;1-16.
5. Li S, Jiang H, Shi M. Redis-based web server cluster session maintaining technology. in 2017 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD). 2017;3065-3069.
6. Izadeen GY, Ameen SY. Smart android graphical password strategy: A review. Asian Journal of Research in Computer Science. 2021;59-69.
7. Zebari IM, Zeebaree SR, Yasin HM. Real time video streaming from multi-source using client-server for video distribution. in 2019 4th Scientific International Conference Najaf (SICN). 2019; 109-114.
8. Ageed ZS, Zeebaree SR, M. M. Sadeeq, S. F. Kak, Z. N. Rashid, A. A. Salih, et al., "A survey of data mining implementation in smart city applications," Qubahan Academic Journal, vol. 1, pp. 91-99, 2021.
9. Zebari RR, Zeebaree SR, Jacksi K. Impact analysis of HTTP and SYN flood DDoS attacks on apache 2 and IIS 10.0 web servers. in 2018 International Conference on Advanced Science and Engineering (ICOASE). 2018;156-161.
10. Haji SH, Zeebaree SR, Saeed RH, Ameen SY, Shukur HM, Omar N, et al. Comparison of Software Defined Networking with Traditional networking. Asian Journal of Research in Computer Science. 2021;1-18.
11. Abdulqadir HR, Zeebaree SR, Shukur HM, Sadeeq MM, Salim BW, Salih AA, et al. A

study of moving from cloud computing to fog computing. Qubahan Academic Journal. 2021;1:60-70.
12. Abdulrahman LM, Zeebaree SR, Kak SF, Sadeeq MA, Adel AZ, Salim BW. et al. A state of art for smart gateways issues and modification. Asian Journal of Research in Computer Science. 2021;1-13.
13. Ramana K, Ponnavaikko M. A Multi-Class Load Balancing Algorithm (MCLB) for heterogeneous web cluster. Stud. Informat. Control. 2018;27:443-452.
14. Hassan RJ, Zeebaree SR, Ameen SY, Kak SF, Sadeeq MA, Ageed ZS, et al. State of art survey for iot effects on smart city technology: challenges, opportunities, and solutions. Asian Journal of Research in Computer Science. 2021;32-48.
15. Yusuf R, Nuha RSPU. Comparative analysis of HAProxy& Nginx in round robin algorithm to deal with multiple web request; 2018.
16. Yasin HM, Zeebaree SR, Sadeeq MA, Ameen SY, Ibrahim IM, Zebari RR, et al. IoT and ICT based smart water management, monitoring and controlling system: A review. Asian Journal of Research in Computer Science. 2021;42-56.
17. Malallah H, Zeebaree SR, Zebari RR, Sadeeq MA, Ageed ZS, Ibrahim IM, et al. A comprehensive study of kernel (Issues and Concepts) in different operating systems. Asian Journal of Research in Computer Science. 2021;16-31.
18. Ageed ZS, Zeebaree SR, Sadeeq MA, Abdulrazzaq MB, Salim BW, Salih AA, et al. A state of art survey for intelligent energy monitoring systems. Asian Journal of Research in Computer Science. 2021;46-61.
19. Li B, Shang J, Dong M, He Y. Research and application of server cluster load balancing technology. in 2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC). 2020;2622-2625.
20. Gebrehiwot ME, Aalto S, Lassila P. Energy efficient load balancing in web server clusters," in 2017 29th International Teletraffic Congress (ITC 29). 2017;13-18.
21. Abdullah SMSA, Ameen SYA, Sadeeq MA, Zeebaree S. Multimodal emotion recognition using deep learning. Journal of Applied Science and Technology Trends. 2021;2:52-58.
22. Zebari SR, Yaseen NO. Effects of Parallel processing implementation on balanced load-division depending on distributed memory systems. J. Univ. Anbar Pure Sci. 2011;5:50-56.
23. Shukla P, Kumar A. CLUE based load balancing in replicated web server. in 2018 8th International Conference on Communication Systems and Network Technologies (CSNT). 2018;104-107.
24. Aziz ZAA, Ameen SYA. Air pollution monitoring using wireless sensor Networks. Journal of Information Technology and Informatics. 2021;1:20-25.
25. Yazdeen AA, Zeebaree SR, Sadeeq MM, Kak SF, Ahmed OM, Zebari RR. FPGA implementations for data encryption and decryption via concurrent and parallel computation: A review. Qubahan Academic Journal. 2021;1:8-16.
26. Abdulmohsin HA. A load balancing scheme for a server cluster using history results. Iraqi Journal of Science. 2016;57:2121-2130.
27. Amanuel SVA, Ameen SYA. Device-to-device communication for 5G security: A review. Journal of Information Technology and Informatics. 2021;1:26-31.
28. Elzanati WM, Ameen SY. Cost effective air-conditioning for bahrain domestic applications. in 2013 7th IEEE GCC Conference and Exhibition (GCC). 2013;535-540.
29. Ameen SY, Al-Badrany MR. Optimal image steganography content destruction techniques. in International Conference on Systems, Control, Signal Processing and Informatics. 2013;453-457.
30. Fawzi LM, Alqarawi SM, Ameen SY, Dawood SA. Two levels alert verification technique for Smart Oil Pipeline Surveillance System (SOPSS),. International Journal of Computing and Digital Systems. 2019;8:115-124.
31. Ameen SY, Ahmed IM. Design and implementation of e-laboratory for information security training. in 2013 Fourth International Conference on e-Learning. Best Practices in Management, Design and Development of e-Courses: Standards of Excellence and Creativity. 2013;310-317.
32. Ameen SY, Al-Jammas MH, Alenezi AS. FPGA implementation of modified architecture for adaptive Viterbi decoder. in 2011 Saudi International Electronics,

Communications and Photonics Conference (SIECPC). 2011;1-9.
33. Al-Sultan MR, Ameen SY, Abduallah WM. Real time implementation of stegofirewall system. International Journal of Computing and Digital Systems. 2019;8:498-504.
34. Ameen SY, Mahdi AH. AES cryptosystem development using neural networks. International Journal of Computer and Electrical Engineering. 2011;3:309.
35. Ameen SY, Ibrahimi IA. MANET routing protocols performance evaluation with TCP Taho, Reno and new-Reno. International Journal of u-and e-Service, Science and Technology. 2011;4:37-49.
36. Ameen SY, Saud LJ. Computing nodes and links appearances on geodesics in network topologies using graph theory. in Full Papers Conference Proceeding. 2011;235.
37. Al Janaby AO, Al-Omary A, Ameen SY, Al-Rizzo HM. Tracking high-speed users using SNR-CQI mapping in LTE-A networks. in 2018 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT). 2018;1-7.
38. Taher KI, Saeed RH, Ibrahim RK, Rashid ZN, Haji LM, Omar N, et al. Efficiency of semantic web implementation on cloud computing: A review. Qubahan Academic Journal. 2021;1:1-9.
39. Othman A, Ameen SY, Al-Rizzo H. Dynamic switching of scheduling algorithm for. International Journal of Computing and Network Technology. 2018;6.
40. Sadeeq MM, Abdulkareem NM, Zeebaree SR, Ahmed DM, Sami AS, Zebari RR. IoT and Cloud computing issues, challenges and opportunities: A review. Qubahan Academic Journal. 2021;1: 1-7.
41. Ameen SY, Ali ALSH. A comparative study for new aspects to quantum key distribution. Journal of Engineering and Sustainable Development. 2018;11:45-57.
42. Kareem FQ, Zeebaree SR, Dino HI, Sadeeq MA, Rashid ZN, Hasan DA, et al. A survey of optical fiber communications: challenges and processing time influences. Asian Journal of Research in Computer Science. 2021;48-58.
43. Omer MA, Zeebaree SR, Sadeeq MA, Salim BW, x Mohsin S, Rashid ZN, et al. Efficiency of malware detection in android system: A survey. Asian Journal of Research in Computer Science. 2021;59-69.
44. Fawzi LM, Ameen SY, Alqaraawi SM, Dawwd SA. Embedded real-time video surveillance system based on multi-sensor and visual tracking. Appl. Math. Infor. Sci. 2018;12:345-359.
45. Rashid ZN, Zeebaree SR, Shengul A. Design and analysis of proposed remote controlling distributed parallel computing system over the cloud. in 2019 International Conference on Advanced Science and Engineering (ICOASE). 2019;118-123.
46. Rashid ZN, Zeebaree SR, Sengur A. Novel remote parallel processing code-breaker system via cloud computing."
47. Ali ZA, Ameen SY. Detection and prevention cyber-attacks for smart buildings via private cloud environment. International Journal of Computing and Network Technology. 2018;6:27-33.
48. Rashid ZN, Zebari SR, Sharif KH, Jacksi K. Distributed cloud computing and distributed parallel computing: A review. in 2018 International Conference on Advanced Science and Engineering (ICOASE). 2018;167-172.
49. Farhan FY, Ameen SY. Improved hybrid variable and fixed step size least mean square adaptive filter algorithm with application to time varying system identification. in 2015 10th System of Systems Engineering Conference (SoSE). 2015;94-98.
50. Rashid ZN, Sharif KH, Zeebaree S. Client/servers clustering effects on CPU execution-time, CPU usage and CPU Idle depending on activities of parallel-processing-technique operations. Int. J. Sci. Technol. Res. 2018;7:106-111.
51. Fawzi LM, Ameen SY, Dawwd SA, Alqaraawi SM. Comparative study of ad-hoc routing protocol for oil and gas pipelines surveillance systems. International Journal of Computing and Network Technology. 2016;4.
52. Jijo BT, Zeebaree SR, Zebari RR, Sadeeq MA, Sallow AB, Mohsin S, et al. A comprehensive survey of 5G mm-wave technology design challenges. Asian Journal of Research in Computer Science. 2021;1-20.
53. Sadeeq MA, Zeebaree S. Energy management for internet of things via distributed systems. Journal of Applied Science and Technology Trends. 2021;2:59-71.

54. Othman A, Ameen SY, Al-Rizzo H. A new channel quality indicator mapping scheme for high mobility applications in LTE systems. Journal of Modeling and Simulation of Antennas and Propagation. 2015;1:38-43.
55. Maulud DH, Zeebaree SR, Jacksi K, Sadeeq MAM, Sharif KH. State of art for semantic analysis of natural language processing. Qubahan Academic Journal. 2021;1:21-28.
56. Shukur H, Zeebaree SR, Ahmed AJ, Zebari RR, Ahmed O, Tahir BSA, et al. A state of art survey for concurrent computation and clustering of parallel computing for distributed systems. Journal of Applied Science and Technology Trends. 2020;1:148-154.
57. Jacksi K, Ibrahim RK, Zeebaree SR, Zebari RR, Sadeeq MA. Clustering documents based on semantic similarity using HAC and K-Mean algorithms. in 2020 International Conference on Advanced Science and Engineering (ICOASE). 2020;205-210.
58. Sadeeq MA, Abdulazeez AM. Neural networks architectures design, and applications: A review. in 2020 International Conference on Advanced Science and Engineering (ICOASE). 2020;199-204.
59. Othman A, Othman SY, Al-Omary A, Al-Rizzo H. Comparative performance of subcarrier schedulers in uplink LTE-A under high users' mobility. International Journal of Computing and Digital Systems. 2015;4.
60. Ageed ZS, Ibrahim RK, Sadeeq MA. Unified ontology implementation of cloud computing for distributed systems. Current Journal of Applied Science and Technology. 2020;82-97.
61. Sallow AB, Sadeeq M, Zebari RR, Abdulrazzaq MB, Mahmood MR, Shukur HM, et al. An investigation for mobile malware behavioral and detection techniques based on android platform. IOSR Journal of Computer Engineering (IOSR-JCE). 22:14-20.
62. Othman A, Ameen SY, Al-Rizzo H. An energy-efficient MIMO-based 4G LTE-A adaptive modulation and coding scheme for high mobility scenarios. International Journal of Computing and Network Technology. 2015;3.
63. Sulaiman MA, Sadeeq M, Abdulraheem AS, Abdulla AI. Analyzation Study for gamification examination fields. Technol. Rep. Kansai Univ. 2020;62:2319-2328.
64. Ameen SY, Nourildean SW. Firewall and VPN investigation on cloud computing performance. International Journal of Computer Science and Engineering Survey. 2014;5:15.
65. Sadeeq M, Abdulla AI, Abdulraheem AS, Ageed ZS. Impact of electronic commerce on enterprise business. Technol. Rep. Kansai Univ. 2020;62:2365-2378.
66. Alzakholi O, Shukur H, Zebari R, Abas S, Sadeeq M. Comparison among cloud technologies and cloud performance. Journal of Applied Science and Technology Trends. 2020;1:40-47.
67. Al-Khayat ON, Ameen SY, Abdallah MN. WSNs power consumption reduction using clustering and multiple access techniques. International Journal of Computer Applications. 2014;87.
68. Ageed Z, Mahmood MR, Sadeeq M, Abdulrazzaq MB, Dino H. Cloud computing resources impacts on heavy-load parallel processing approaches. IOSR Journal of Computer Engineering (IOSR-JCE). 2020;22:30-41.
69. Sallow A, Zeebaree S, Zebari R, Mahmood M, Abdulrazzaq M, Sadeeq M. Vaccine tracker," SMS reminder system: Design and implementation; 2020.
70. Ameen SY, Yousif MK. Decode and forward cooperative protocol enhancement using interference cancellation. Int. J. Elect., Comput., Electron. Commun. Eng. 2014;8:273-277.
71. Sadeeq MA, Zeebaree SR, Qashi R, Ahmed SH, Jacksi K. Internet of things security: A survey. in 2018 International Conference on Advanced Science and Engineering (ICOASE). 2018;162-166.
72. Abdulazeez AM, Zeebaree SR, Sadeeq MA. Design and implementation of electronic student affairs system. Academic Journal of Nawroz University. 2018;7:66-73.
73. Prabu U, Malarvizhi N, Amudhavel J, Sriram R, Ravisasthiri P. Load balancing policies of web servers: research analysis, classification and perspectives. EAI Endorsed Transactions on Scalable Information Systems. 2019;6.
74. Abdullah DM, Ameen SY. Enhanced Mobile Broadband (EMBB): A review. Journal of Information Technology and Informatics. 2021;1:13-19.

75. Nurwarsito H, Sejahtera VB. Implementation of dynamic web server based on operating system-level virtualization using docker stack. in 2020 12th International Conference on Information Technology and Electrical Engineering (ICITEE). 2020;33-38.
76. Kunda D, Chihana S, Sinyinda M. Web server performance of apache and nginx: A systematic; 2017.
77. Khalid LF, Ameen SY. Secure Iot integration in daily lives: A review. Journal of Information Technology and Informatics. 2021;1: 6-12.
78. Abdulla AI, Abdulraheem AS, Salih AA, Sadeeq MA, Ahmed AJ, Ferzor BM, et al. Internet of things and smart home security. Technol. Rep. Kansai Univ. 2020;62:2465-2476.
79. Prakash P, Biju R, Kamath M. Performance analysis of process driven and event driven web servers. in 2015 IEEE 9th International Conference on Intelligent Systems and Control (ISCO). 2015;1-7.
80. Sharif KH, Ameen SY. A Review of security awareness approaches with special emphasis on gamification. in 2020 International Conference on Advanced Science and Engineering (ICOASE). 2020;151-156.
81. Nguyen VN. A comparative performance evaluation of web servers; 2017.
82. Al Janaby AO, Al-Omary A, Ameen SY, Al-Rizzo H. Tracking and controlling high-speed vehicles via CQI in LTE-A systems. International Journal of Computing and Digital Systems. 2020;9:1109-1119.
83. Shukla A, Kumar S, Singh H. Load balancing approaches for web servers: A survey of recent trends. International Journal of Engineering. 2018;31:263-269.
84. Zeebaree S, Ameen S, Sadeeq M. Social media networks security threats, risks and recommendation: A case study in the kurdistan region. International Journal of Innovation, Creativity and Change. 2020;13:349-365.
85. Abdulraheem AS, Salih AA, Abdulla AI, Sadeeq MA, Salim NO, Abdullah H, et al. Home automation system based on IoT; 2020.
86. Salih AA, Zeebaree SR, Abdulraheem AS, Zebari RR, Sadeeq MA, Ahmed OM. Evolution of mobile wireless communication to 5G revolution. Technology Reports of Kansai University. 2020;62:2139-2151.
87. Jader OH, Zeebaree S, Zebari RR. A state of art survey for web server performance measurement and load balancing mechanisms. International Journal of Scientific & Technology Research. 2019;8:535-543.
88. Hamed ZA, Ahmed IM, Ameen SY. Protecting Windows OS Against local threats without using antivirus. Relation. 2020;29:64-70.
89. Dino HI, Zeebaree SR, Salih AA, Zebari RR, Ageed ZS, Shukur HM, et al. Impact of process execution and physical memory-spaces on OS performance.
90. ZH. a. e. al. Web load balance strategy with energy consumption constrained based on DNS collaboration. presented at the 4th IEEE International Conference on Big Data Security on Cloud; 2018.
91. Mohammed K, Ameen S. Performance investigation of distributed orthogonal space-time block coding based on relay selection in wireless cooperative systems; 2020.
92. ST. a. K. Kungumaraj. Survey on a content based dynamic load balancing algorithm for heterogeneous web server cluster. International Journal of Computer Science Trends and Technology (IJCST). 2016;4.
93. Ageed ZS, Zeebaree SR, Sadeeq MM, Kak SF, Yahia HS, Mahmood MR, et al. Comprehensive survey of big data mining approaches in cloud systems. Qubahan Academic Journal. 2021;1: 29-38.
94. Ibrahim IM. Task scheduling algorithms in cloud computing: A review. Turkish Journal of Computer and Mathematics Education (TURCOMAT). 2021;12:1041-1053.
95. PZ. a. J. Zhang. Load balancing algorithm for web server based on weighted minimal connections. Journal of Web Systems and Applications. 2017;1.
96. P.-r. J. a. e. al. A client proximity based load balance algorithm in web sever cluster. presented at the 2nd International Conference on Wireless Communication and Network Engineering; 2017.
97. SKP. a. e. al. An efficient intra-server and inter- server load balancing algorithm for internet distributed systems. International Journal of Rough Sets and Data Analysis. 2017;4.
98. J. L. a. e. al. A dynamic load balancing algorithm based on consistent hash. presented at the 2nd IEEE Advanced

Information Management Communicates, Electronic and Automation Control Conference, IEEE; 2018.
99. P. A. S. a. e. al. Increasing SDN network performance using load balancing scheme on web server. presented at the 6th International Conference on Information and Communication Technology, IEEE, 2018.
100. GL. a. X. Wang. A modified round-robin load balancing algorithm based on content of request. presented at the 5th International Conference on Information Science and Control Engineering, IEEE, 2018.
101. GS. a. K. Kaur, "An Improved Weighted Least Connection Scheduling Algorithm for Load Balancing in Web Cluster Systems," International Research Journal of Engineering and Technology, vol. 5, 2018.
102. Rosmansyah IKAaY. Web server farm design using personal computer (PC) desktop. presented at the 10 th International Conference on Information Technology and Electrical Engineering, IEEE; 2018.
103. ZW. a. e. al. Research and realization of nginx-based dynamic feedback load balancing algorithm. presented at the 3rd Advanced Information Technology, Electronic and Automation Control Conference. IEEE; 2018.
104. Sharma D. Response time based balancing of load in web server clusters. IEEE; 2018.
105. HY. L. a. e. al. Open flow-based server cluster with dynamic load balancing. IEEE; 2018.
106. AY. a. e. al. Research on web server cluster load balancing algorithm in web education system. Springer Science+Business Media, LLC, part of Springer Nature; 2018.
107. RL. a. e. al. An integrated load-balancing scheduling algorithm for nginx-based web application clusters. Journal of Physics: Conf. Series 1060; 2018.
108. FM. a. V. Mosorov. Load balancing algorithms in heterogeneous web cluster. IEEE; 2018.
109. Nishi MO.a.H. Request distribution for heterogeneous database server clusters with processing time estimation. IEEE; 2018.
110. MRM. B. a. e. al. Web server load balancing based on memory utilization using docker swarm. IEEE; 2018.
111. KR. a. M. Ponnavaikko. AWSQ: an approximated web server queuing algorithm for heterogeneous web server cluster. International Journal of Electrical and Computer Engineering. 2019;9:2083~2093.
112. E. Q. a. e. al. Research on nginx dynamic load balancing algorithm. presented at the 12th International Conference on Measuring Technology and Mechatronics Automation, IEEE; 2020.
113. WC. a. e. al. Design and implementation of a TCP long connection load balancing algorithm based on negative feedback mechanism. presented at the Journal of Physics: Conference Series1659; 2020.
114. Chiang ML, Cheng HS, Liu HY, Chiang CY. SDN-based server clusters with dynamic load balancing and performance improvement. Cluster Computing. 2021;24:537-558.

Peer-review history:
The peer review history for this paper can be accessed here:
http://www.sdiarticle4.com/review-history/70184