DOI: http://dx.doi.org/10.26483/ijarcs.v8i8.4561

ISSN No. 0976-

Volume 8, No. 8, September-October 2017

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

DCWSLBQ: DYNAMIC CONTENT BASED WEB SERVER LOAD BALANCING QUEUE ALGORITHM FOR HETEROGENEOUS WEB CLUSTER

S.Tamilarasi
Research Scholar of
Mother Teresa Women's University,
Kodaikanal. India

Dr. K.Kungumaraj
Head and Assistant Professor,
Department of Computer Applications
Arulmigu Palaniandavar College for Women Palani, India

Abstract: Load balancing is the way toward circulating workload crosswise over different gadgets on a system in networking. It expects to limit reaction time, expand throughput, upgrade asset and maintain a strategic distance from over-burden of any single asset. To deal with secure substance over Web trade a protected channel gave by Secure Socket Layer convention is proposed. The selected internet service for the online training and independent request over server can avoid the overload, the performance can be increased if the services are designed so that the methodology scalable.

Keywords: Load balancing, Scalability, SSL Response time, Throughput, Content

1. INTRODUCTION

The web server load balancing innovation is regularly utilized as a part of present day endeavor system to offer high caliber and dependable administration. Unsurprising web server load balancing technology is regularly achieved by particular equipment that is commonly extremely costly and needs adequate adaptability. Incidentally, it is anything but difficult to change over a specific purpose of dissatisfaction and would be constrained in virtualization areas. To suggest a heap adjusting calculation in view of server running state, which can evaluate comprehensive load balancing permitting to the CPU, memory, and system use movement of the servers. Furthermore, programming characterized systems (SDN) innovation is connected a heap adjusting arrangement, and it is considered and acknowledged in Open Flow organize. To trust arrange administration and server state screen in this framework, in which the open Flow switches forward the demand to the littlest finish stacking server by moving the packet.
Presently, activity on the system is extremely immense and is rising rapidly. System sticking and server over-burden are getting to be plainly extreme issues confronted by undertakings. Specifically, the advances and ideas, for example, distributed computing, virtualization, and enormous information make the issue especially critical. Run of the mill activity systems are extremely unpredictable and with the development of business, endeavors need to buy greater
hardware, fabricate more refined systems, and control more movement.

2. OPEN FLOW METHOD:

Unmistakable load balancing plans have a few deficiencies in present day conditions. The critical intention is that the customary outline of the Internet has some glitches. Affected by the new demands, the bottleneck of conventional system engineering has remained recreated in many parts. Individuals are looking for new receptions to meet evolving business. Between many plans, SDN is the most extreme compelling and recognized one, as the delegate of SDN from the earliest starting point, Open Flow to got wide consideration from specialists. The objective of Open Flow is to change the method for controlling customary system gadgets. In customary systems, organize gadgets forward information as per appropriated information administration and, during the time spent sending information from source to goal, singular hardware decides how to forward its information freely.

An Open Flow parts the controller module from the gadgets and sets it into an outside server with a switch program running on it; the server can send orders to the Open Flow controls to switch the quickening approach, and the control program can likewise give an outer application programming interface for organize overseers to control the switch automatically. This does not just decrease the requirement for manual setups on switch, it can likewise give more prominent adaptability in arrange administration. Expending load adjusting systems in system can all around overawed a portion of the restrictions of customary load adjusting and gives a straightforward and viable arrangement with high tractability.

Because of the alteration among the conventional Internet and Open Flow organize, they unavoidably shift from each other in utilizing load adjusting strategies in customary system and Open Flow arrange. The customary load adjusting innovation is not by any means identified with the Open Flow organize. New issues have happened, for example, stack adjusting module outline, the server working status checking, and how to affirm the adaptability of load adjusting.

Numerous dealings in the activity organize are exchanging to virtual situations since virtualization technology can

bolster organizations to spare cash, join servers other than attempted the act of in-entire properties. In any case, now that load balancing innovation is not developed adequate in virtual condition, the utilization of customary load balancing items is under impediments in server farm virtualization areas, which kept up battle to activity server farm virtualization advancement.

Many dealings in the initiative network are transferring to virtual environments because virtualization technology can support companies to save money, combine servers besides undertaking the practice of in-complete properties. But now that load balancing technology is not matured sufficient in virtual environment, the application of traditional load balancing products is under limitations in data centre virtualization locations, which maintained struggle to initiative data centre virtualization development. The future design and execution of Open Flow-based server clusters dynamic load balancing in a virtualized location. The architecture not only is low-cost but also offers the flexibility to write modules in the controller for realizing the customizable policy set. Internet requests can attain real-time monitoring of weight and timely access the suitable resources by the flexible arrangement capabilities of this architecture. An Open Flow switch can connect to several controllers in Open Flow to use several servers as controller connecting to Open Flow Switch, so as to progress the strength of the system. Tack vectors that challenge the defense with high probability. This has stimulated research on trust supervision model.

3.ALGORITHMS USED FOR LOAD BALANCING

Round Robin Algorithm (RRA)

In the [1] RRA procedures are separated equally among all processors. Every new process is allocated to new processor in round robin instruction. The procedure distribution order is continued on every processor locally independent of distributions from remote processors [2]. Through equivalent web server workload round robin algorithm is predictable to work well [3]. Correspondingly when the works are of imbalanced processing time this process suffers as the particular nodes can become strictly loaded while others keep on idle. Round Robin is mostly utilized in web servers where commonly HTTP requests are of interrelated nature and thereby be distributed similarly.

3.1 Randomized Algorithm

The [4] randomized algorithm is a static algorithm relates a probabilistic methodology as conflicting to the deterministic methodology shadowed in Round Robin Scheduling. In this technique a procedure can be controlled by a specific node n with a possibility p. This also works very well in case every process loads is equivalent but fails if lots are of several computational complexities. It is utilized in modification to round robin algorithm when there are huge numbers of nodes as preserving the queue of nodes for round robin become an overhead.

3.2 Central Manager Algorithm

The [5] central manager algorithm has an essential processor which chooses the host for novel development. The slightly loaded processor depending on the complete load is designated when process is generated. Load manager chooses hosts for novel developments so that the processor load approves to similar level as much as potential. The essential load manager creates the load balancing decision from the on hand evidence on the system load state. This evidence is modernized by remote processors, which send a message every time the load on them modifications [6]. The load manager creates load balancing resolutions based on the system load evidence, permitting the best resolution when of the process generated. High degree of IPC (inter-process communication) could create the bottleneck state. This algorithm is estimated to perform better than the parallel requests, particularly when dynamic actions are generated by heterogeneous hosts [7].

3.3 Threshold Algorithm

The processes are allocated closely upon formation to hosts. Hosts for novel developments are nominated locally without sending remote messages. Every processor recollects a reserved copy of the system's load. The load of a processor can illustrate by one of the three levels: under loaded, medium and overloaded. Two threshold restrictions wstunder and wstupper can be utilized to define these stages.

Under loaded - load <wstunder

Medium -wstunder ≤ load ≤ twsupper

Overloaded - load >wstupper

Firstly, all the processors are measured to be under loaded. When the load state of a processor overdoes a load level boundary, and then it directs messages regarding the new load state to all remote processors, regularly modernizing them as to the real load state of the whole system. If the local state is not overloaded then the procedure is owed locally. Else, a remote under loaded processor is selected on web server, and if no such host occurs, the process is also owed locally. Thresholds algorithm have small IPC (inter-process communication) and a massive number of native development allocations .The later losses the overhead of remote development distributions and the overhead of remote memory accesses, which leads to enhancement in performance. A weakness of the algorithm is that all procedures are allocated locally when all remote processors are overloaded. A web server load on one overloaded processor can be greatly advanced than other overloaded processors, causing important disorder in load balancing, and increasing the performance time of an application [8].

3.4 Central Queue Algorithm (CQA)

Central Queue Algorithm (CQA) [9] involves on the standard of dynamic delivery. It supplies novel actions and unsatisfied requests as a cyclic FIFO queue on the chief host. Every novel action in ward at the queue manager is interleaved into the queue. Then, whenever a request for an activity is established by the queue manager, it eliminates the first activity from the queue and sends it to the requester. If there are no organized actions in the queue, the request is protected, until a new action is available. If a new action arrives at the queue manager while there are unreturned requests in the queue, the first such request is detached from the queue and the new action is allocated to it. When a processor load drops below the threshold, the local load manager sends a request for a new action to the central load manager. The central load manager answers the request immediately if a ready action is found in the process-request queue, or queues the request until a new action arrives [10].

3.5 Local Queue Algorithm (LQA)

LQA[11] is dynamic development relocation maintenance. The simple indication of the local queue algorithm is static

distribution of all new developments with process migration started by a host when its load drops below threshold limit, is a customized constraint of the algorithm. The parameter describes the marginal number of prepared developments the load manager tries to offer on every processor. Firstly, new processes generated on the chief host are distributed on all below loaded hosts. The number of parallel activities generated by the first parallel construct on the main host is typically necessary for allocation on all remote hosts. From then on, all the processes generated on the main host and all other hosts are distributed locally [12]. When the host gets below loaded, the local load manager tries to get several processes from remote hosts. It randomly directs requests with the number of local prepared processes to remote load managers. When a load manager receives such a request, it relates the local number of prepared processes with the received number. If the former is better than the latter, then several of the running processes are relocated to the client and a favorable confirmation with the number of processes relocated is returned [13].

3.6 Least Connection Algorithm

The least-connection scheduling algorithm aims network connections to the server with the tiniest number of proven connections. This is one of the dynamic scheduling algorithms; since it requests to count the amount of connections for each server dynamically to evaluation its load. The load balancer records the connection number of each server, raises the connection number of a server when a new connection is transmitted to it, and decrease the connection number of a server when a connection finishes or timeouts. The formal procedure of round robin scheduling is as follows [14]:

Supposing that there is a web server set WS = {$WS_{0,}$ WS_1, ..., WS_{n-1}},

$W_{(i)}$ isWS_i the weight of web server WS_i,

$C(WS_i)$ is the current connection number of web server S_i,

```
for (m = 0; m < n; m++)
{
if (W(WSm) > 0)
{
for (i = m+1; i ¡ n; i++)
{
if (W(WSi) ≤ 0)
continue;
if (C(WSi) < C(WSm))
m = i;
}
return WSm;
}
 return NULL;
}
```

3.7 Weighed Least Connection Algorithm

The weighted least-connection is a superset resolution of the least-connection scheduling in which you can able to allocate a performance weight to every real web server. The real web servers with a greater weight value will obtain a higher percentage of active networks at any one time [15]. The default web server weight is one, and the IPVS Administrator or monitoring program can allocate any weight to real web server. In the weighted least-connections scheduling, new network connection is allocated to a web server which has the least ratio of the recent active connection number to its weight factor of web server. The suitable technique of is as follows [16]:

Supposing there is a web server set WS = {WS0, WS1, ..., WSn-1},

$W(WS_i)$ is the weight of web server WS_i;

$C(WS_i)$ is the current connection number of web server WS_i;

$CSUM = \Sigma\ C(WS_i)$ (i=0, 1, .. , n-1) is the sum of current connection numbers;

```
for (m = 0; m < n; m++)
{
if (W(WSm) > 0)
{
for (i = m+1; i < n; i++)
{
if (C(WSm*W(WSi) > C(WSi)*W(WSm)) m = i;
 }
return WSm;
}
}
 return NULL;
```

4. OBSERVATIONS FROM THE STUDY OF VARIOUS ALGORITHMS

1. If more than one factor is utilized for load estimation the estimation is more reliable than cases where only one parameter is used.
2. If individual nodes calculate their load status, the network load of running the algorithm is reduced.
3. Threshold values should be 2 or more in order to allow smooth transition from under-loaded to overloaded conditions. If only one threshold is used the fluctuations are high and the performance may degrade.
4. The condition that many nodes simultaneously select a certain node for running a process must be avoided.
5. In case of heterogeneous nodes, weights can be assigned so as to ensure that the differences in amount of resources can be accounted for.

Clusters are tried to be kept as homogeneous as possible, to maintain the Single System Image (SSI) Thus under the usual circumstances, a heterogeneous cluster seems undesirable. However, it is sometimes deliberately needed to include heterogeneous nodes to take advantage of the performance of certain architectures and other advanced components. Moreover, the improved hardware in newer machines must be taken advantage of else there is no use investing in such hardware. But these essential be used in aggregation with the other accessible machines. Thus, the concept of heterogeneous clusters in the real world is indeed a very important one.

For the present study, the term

"Heterogeneous Clusters have been characterized as takes after:- The individual hubs may have diverse figuring power as CPU speed may change, distinctive measure of Memory accessible and in addition diverse seller particular structures. On the OS front, it has been expected that every one of the hubs are running a Linux based OS for the equipment related data is put away in them in a comparative organized record framework. The heterogeneity if reached out to Windows based OS will expand the intricacy of the calculation in view of the distinction of data structure on the Linux and Windows OS. The apparently limitation decision OS is not extremely huge as the Cluster middleware are on the GNU permit, which viably port the SSI picture to

Windows OS. Likewise the utilization of Virtual Machines in the framework will empower the Linux Kernel to keep running over the Windows based OS.
"Heterogeneous Clusters have been defined as follows:- The individual nodes may have different computing power as CPU speed may vary, different amount of Memory available as well as different vendor specific architectures. On the OS front, it has been assumed that all the nodes are running a Linux based OS for the hardware related information is stored in them in a similar structured file system. The heterogeneity if extended to Windows based OS will increase the complexity of the algorithm because of the difference of information structure on the Linux and Windows OS. The seemingly constraint choice OS is not very significant as the Cluster middleware are on the GNU license, which effectively port the SSI image to Windows OS. Also the use of Virtual Machines in the system will enable the Linux Kernel to run on top of the Windows based OS.

5. DEALING WITH HETEROGENEITY OF CLUSTERS

In order to manage the heterogeneity in the groups, we have a four stage worldview as clarified beneath :

1. State Measurement: This stage lies with the load estimation.

2. Data Exchange: It is where the correspondence overheads should be considered with a specific end goal to guarantee which strategy for trading load states would be helpful to the calculation.

3. Start Rule: The load adjusting calculation ought to be started just when the advantages of load adjusting calculation exceed the cost of running the calculation.

4. Load Balancing Operation: This progression is additionally subdivided into 3 more strides the area, appropriation and determination rules. They manage the way the calculation disseminates the heap and on the hubs the load goes.

The condition of-craftsmanship web demands interconnect and deal with number of simply secluded data assets giving data to tremendous number of clients. Comparable server bunches are not capable of fulfilling the rising solicitation of such applications containing constant sound, continuous video, PHP, JSP, and ASP and so forth. Heterogeneity contains treatment of low level interoperability issues e.g. programming dialects, crisscross of equipment, database pattern, working stages, topology and so forth. Versatile server group licenses expansion of new servers as the load increments without disturbing the administrations. Additionally, it likewise offers better unwavering quality by richly exchanging the load from server which is blocked off because of disappointment or for defensive protection. Heterogeneity with adaptability makes the framework more troublesome. The Dynamic Load Balancing-DLB existing calculations are not straightforwardly interrelated for conveyed planning for such circumstances. To propose another calculation for versatile different server group utilizing content mindfulness web server. The calculation mirrors server's line length, usage proportion, handling ability and so on as load records. As the bunch systems of support a few administrations, at the essential level, utilized substance mindfulness sending calculation and at the auxiliary level, held up round robin calculation has been used heterogeneous load adjusting web server.

6. OBJECTIVES OF PROPOSED WORK

As indicated by what is normal from the improvement of this examination, solid goals are the accompanying:

I. To acquire a wide framework of the importance of substance based load balancing web server grouping and the distinctive divisions that can be made by it.

II. To deliver a more profound examination on acknowledged web server bunching typologies to discover their pecularaties and subtle elements.

III. To do a market way to deal with get a genuine attention to the watched typologies dissemination.

IV. To pick one of the market choices coordinating a solid bunch sort as a contender for more profound examination. Bunch typology of web server is to enhance one recognized as Load Balancing. The market decision must permit a free usage in out of form PCs.

V. To understand the adaptability that Windows/Linux Operating System offers, in its execution over old equipment, as well as in the practicality of web server grouping on it.

VI. To execute picked answer for set in motion the hypothetical learning acquired beforehand.

VII. To test if the usage over old PCs can create satisfactory outcomes.

VIII. To test to which degree is genuine that a bunched configuration utilizing unobtrusive assets and free programming can be attainable.

IX. At last, to make an archive that could be utilized by an organization required of a monstrous web server administration to choose if Windows/Linux Web Server Clustering are acquire commendable determination.

7. IMPLEMENTED METHODOLOGY

In the proposed algorithm, three parameters (CPU load, memory load and queue length) of the server have been considered which is used for the selection of the least loaded web server in the cluster. The proposed load balancing architecture has presented in the Figure1. along with steps of load balancing operations.

'n' number of web servers and unique load balancer in a load balancing system. Whenever any users access a specific web page, the DNS server is accountable for assigning respective IP address of the web server. In this technique, the DNS server decides to resolves the IP address of the load

balancer only and it is not accountable for the accessing web pages.

Advance, the client directs an HTTP request to the load balancer and while the load balancer obtains the request, a redirection page sends back to the client. The redirection page encloses the IP address of the web server, which is the best one to serve this request. This redistribution depends on the load balancing algorithm utilized by the load balancer. The client concerns over HTTP request to the real web server. The web server support the client requests and relocation the desired pages.

A new module in the web server has been presented as "status monitor". This status monitor module has observed the performance of the web server constantly. Status monitor constantly directs back the status of the web server to the load balancer.

Load balancer also has a module "load collection", which gathers the recent load evidence of every web server and occasionally calculated the actual load on the web server. The status monitor sends the report of the current value of outstanding memory, outstanding CPU usage and queue-length of the web server queue constantly and the load balancer uses this status information for the selection of least loaded web server.

The proposed algorithm has been designed based for the proposed load balancing architecture of the web server. Three performance parameters have been considered for the proposed load balancing algorithms as the remaining capacity of memory, number of active connections to the web server and the server queue-length. Based on these parameters, the least loaded web server has been selected and assigned the respective web requests.

A web server has shown the current load information as {m, q, r} parameters, in which 'm' represents the remaining capacity of the CPU, 'q' represents the remaining capacity of the memory and the remaining queue-length of the server has represented by 'r'. Self-styled code of the proposed load balancing algorithm is given below.

WSLBQ Algorithm (Reporting Phase, Decision Phase, Queue Processing Phase)

Step-1 (Reporting Phase): Takes place periodically

(i) Every web server calculates the three parameters such as m, q and r. Where,

m= percentage of remaining CPU capacity

m= percentage of remaining memory capacity

r= percentage of remaining queue capacity

(ii) Report the above parameters to load balancer

Step-2 (Decision Phase): For each client request.

(i) For each web server, the load balancer calculates,

Serving Load Capacity, WSL= (X× m) + (Y× q) + (Z× r),

Where X, Y, and Z are weighting factors such that (X+ Y+ Z) = 1.

(ii) Maximum serving load capacity, $WSL_{max} = \max(WSL_1, WSL_2, WSL_3, \ldots, WSL_n)$

, where n represents the number of available web servers

(iii) The i^{th}server such that $WSL_i = WSL_{max}$is considered as the least loaded server to practise the recent request.

(iv) In i^{th}server:

if ($m = m_{min}$or $n = 0$), where $m_{min}$is minimum required memory to process a request

then

if (Number of requests in FCFS request queue of i^{th}server < queue capacity)

then

Add current request into FCFS request queue of i^{th}server.

else

Drop the current request

else the i^{th}server processes the current request.

Step-3 (Queue Processing Phase): For each awaiting request in queue, whenever server has no new requests to process, and the number of requests in its FCFS Request Queue>0.Then it processes a new request from the Queue.

The proposed algorithm has processed each request of queue in First Come First Serve (FCFS)manner and is assigned to the server based on the web server load capacity. Based on the processing capability of the WSLBQ algorithm has been achieved in O (n) time complexity.

8. CONCLUSION

The load balancing calculation on the web server framework has been used to enhance the accessibility and to diminish the over-burdening of the web servers. A productive load adjusting calculation, WSLBQ has been proposed to deal with the load of the web servers utilizing some new modules as "status screen" and "load gathering" to figure the current over-burdening state of the web servers. To compute the execution parameters, for example, remaining CPU utilization, remaining memory use, and remaining line length to ascertain the present serving limit of the web servers. New asks for have been exchanged to the web servers which have the most extreme residual administration limit.

9.REFERENCES

[1] Tinghuai Ma, Ya Chu, Licheng Zhao & Otgonbayar Ankhbayar, "Resource Allocation and Scheduling in Cloud Computing: Policy and Algorithm"IETE Technical review Volume 31, Issue 1, pp.4-16,January 2014.

[2] Nidhi Jain Kansal, Inderveer Chana, "Cloud Load Balancing Techniques: A Step Towards Green Computing", IJCSI, Vol. 9, Issue 1, January 2012.

[3] Hitesh Bheda, Hiren Bhatt," An Overview of Load balancing Techniques in Cloud Computing Environments",International journal of Engineering and Computer Science Volume 4, pp.9874- 9881,JANUARY 2015.

[4] Sukhvir Kaur, Supriya Kinger "Review on Load Balancing Techniques in Cloud Computing Environment", International Journal of Science and Research (IJSR) Volume 3,Issue 6, June 2014.

[5] Nusrat Pasha, Dr. Amit Agarwal and Dr.Ravi Rastogi,"Round Robin Approach for VM Load Balancing Algorithm in Cloud Computing Environment" International Journal of Advanced Research in Computer Science and Software Engineering Volume 4, Issue 5, May 2014.

[6] A. Bhadani and S. Chaudhary, " performance evaluation of web servers using central load balancing policy over virtual machine on cloud", proceedings of third Annual ACM.

[7] J. M. Galloway, K. L. Smith, and S. S. Vrbsky, "Power aware load balancing for cloud computing," in Proceedings of the World Congress on Engineering and Computer Science, vol. 1, pp.19–21, 2011.

[8] S. Sethi, A. Sahu, and S. K. Jena, "Efficient load balancing in cloud computing using fuzzy logic," IOSR Journal of Engineering, vol. 2, no. 7, pp.65–71, 2012.

[9] Z. Nine, M. SQ, M. Azad, A. Kalam, S. Abdullah and R. M. Rahman, "fuzzy logic based dynamic load balancing in virtualized data centers" In fuzzy system (FUZZ), IEEE International conference on, pp. 1-7, 2013.

[10] Ms.Nitika, Ms.Shaveta, Mr. Gaurav Raj; "Comparative Analysis of Load Balancing Algorithms in Cloud Computing", International Journal of Advanced Research in Computer Engineering & Technology Volume 1, Issue 3, May 2012.

[11] M. Randles, D. Lamb, and A. Taleb-Bendiab, "A Comparative Study into Distributed Load Balancing Algorithms for Cloud Computing", Proceedings IEEE International Conference on Advanced Information Networking and Applications Workshops, Perth, Australia, pp.551-556, April 2010.

[12] Dr. Amit Agarwal, Saloni Jain "Efficient optimal algorithm of task scheduling in cloud computing environment" International Journal of computer Trends and Technology (IJCTT). G. Kliotb, Y. Lua, Q. Xiea, A. Gellerb, J. R. Larusb, and A. Greenber, "Join-Idle-Queue: A novel load balancing algorithm for dynamically scalable web services", An international Journal on computer Performance and evaluation, In Press, Accepted Manuscript, Available online 3 August 201l.

[13] Kousik Dasgupta, Brototi Mandal, Paramartha Dutta,Jyotsna Kumar Mandal,Santanu Dam , "A Genetic Algorithm (GA) based Load Balancing Strategy for Cloud Computing", Elsevier (CIMTA) 2013.

[14] Anamika Jain, Ravinder Singh, " Review of Peer to Peer Grid Load Balancing Model Based on Ant Colony Optimization with Resource Management" Volume 3, Issue 4, April 2013 IJARCSSE.

[15] Kousik Dasgupta, Brototi Mandal, Paramartha Dutta, "Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft Computing Approach",Elsevier (C3IT) 2012.

[16] H. Mehta, P. Kanungo, and M. Chandwani,"Decentralized content aware load balancing algorithm for distributed computing environments", Proceedings of the International Conference Workshop on Emerging Trends in Technology (lCWET),pp.370-375, February 2011.

[17] A. M. Nakai, E. Madeira, and L. E. Buzato,"Load Balancing for Internet Distributed Services Using Limited Redirection Rates", 5th IEEE Latin-American Symposium on Dependable Computing (LADC),pp.156- 165 2011.