© 2019 IJSRST | Volume 6 | Issue 2 | Print ISSN: 2395-6011 | Online ISSN: 2395-602X
Themed Section: Science and Technology
DOI : 10.32628/IJSRST119624

Implementation of Online Test Engine Based on Load Balancing

Diksha Ninave[1], Abhishek Aphale[1], Ashlesha Wath[1], Ankita Watkar[1], Tushar Likhar[1], Prof. Shraddha Karale[2]
[1]BE Students, Computer Science & Engineering, Rajiv Gandhi College of Engineering and Research, Nagpur, Maharashtra, India
[2]Assistant Professor, Computer Science & Engineering, Rajiv Gandhi College of Engineering and Research, Nagpur, Maharashtra, India

ABSTRACT

Numerous expansive web locales get a huge number of hits each day. They require a versatile web server framework that can give better execution to every one of the customers that might be in various topographical locales. The regular way to deal with enhancing execution is to have completely repeated web server bunches in various topographical areas with duplicated servers. In such a domain, a standout amongst the most critical issues is choosing a server for adjusting a demand. Customer solicitations ought to be coordinated to a server with the end goal that the time taken for adjusting the demand can be limited. Diverse approaches are feasible for server determination and it is hard to decide the effect of various arrangements. A distributed framework comprises of a few self-sufficient nodes, where a portion of the nodes might be overloaded because of a substantial number of job entries while others nodes are inactive with no preparing. Load Balancing is utilized for viably dispersing the load among the nodes. Centralized load balancing plans are not adaptable as the load balancing choice relies upon a focal server. Interestingly completely distributed plans are versatile however, they do not deliver a reasonable load circulation as they utilize nearby data. In this examination, we propose a clustered load balancing arrangement for a heterogeneous distributed registering framework.

Keywords : Test Engine, Clustered Load Balancing, Distributed Systems, Workload, Response Time, Node Utilization

I. INTRODUCTION

The number of clients getting to the Internet is expanding quickly and usually to have in excess of 100 million hits every day for famous web locales. For instance, netscape.com website gets in excess of 120 million hits per day. The quantity of clients is relied upon to keep expanding at a quick rate and consequently, any website that is mainstream faces the test of serving a substantial number of customers with great execution. Full reflecting of web servers or replication of web locales is one approach to manage the expanding number of solicitations. Numerous methods exist for the determination of the closest web server from the customer's perspective. In a perfect world, the determination of best server ought to be done straightforwardly without the mediation of the client.

Huge numbers of the current plans do just load-balancing. These plans expect that the reproduced webpage has all the web servers in a single group. This is okay for medium measured destinations, however past a specific measure of traffic, the

network to this one bunch turns into a bottleneck. So expansive web locales have numerous bunches, and it is best to have these groups geologically distributed. These progressions the issue to initially choose the closest group and after that do load balancing inside the servers of that bunch. Obviously, in the event that all servers in a group are intensely loaded, another bunch ought to have been picked. Therefore, the issue is increasingly overwhelming in such a domain.

Planning such a framework includes settling on choices about how as well as can be expected to be chosen for demand with the end goal that the client gets a response in the least time and how this demand is coordinated to that server. In a few procedures, a server is chosen without considering any framework state data, e.g. irregular, round robin and so on [6]. A few approaches utilize weighted limit calculations to guide more level of solicitations to increasingly competent servers [7]. A few procedures select a server depending on the server state [7] and some others consider customer state data [7]. There is dependably a trade-off between the overhead because of the gathering of framework state data and execution gain by utilization of accessible state data. In the event that an excessive amount of state data (of server or customers) is gathered, it might result in high overheads for accumulation of data and execution gain may not be practically identical to overheads. The execution of any load balancing approach relies upon a large group of highlights like system delays, parcel misfortunes, transmission mistakes, and a rate of solicitations, server load and so forth. It is generally difficult to systematically decide the execution of an approach given a few conditions. Re-enactments additionally have constraints in that they can just consider restricted factors and with the complexities associated with this case impact of the considerable number of factors cannot be systematically decided and reproduced. Thus testing for execution by setting up a testbed is a sensible method to assess diverse methodologies.

This testing will when all is said in done require a devoted testbed on which execution studies should be possible. Such a testbed ought to be configurable for various techniques and system qualities. In that capacity a testbed is probably going to be a devoted set up with an engaged reason, it will be helpful if a web service can be made for assessing an approach. In this, all parameters will be set in the web service, which will at that point drive the testbed to acquire the outcomes. Such a service will make a devoted testbed available over the world. Such a service ought to satisfy various prerequisites. It ought to be conceivable to consider the effect of various parameters on the execution of different methodologies and to look at them. It ought to be conceivable to submit new techniques and contrast their execution and existing ones or to distinguish conditions where the arrangement performs best.

In this investigation, we portray a web service for assessing load-balancing procedures for distributed web server systems. The web service cooperates with a testbed to mechanize the procedure of accommodation of parameters, testing and result from age to enable the client to test and think about load balancing approaches through in an exceedingly configurable way. We have pre-characterized some well-known techniques, which can be assessed for various parameter settings. The service additionally enables clients to test new arrangements for load balancing and contrast their execution and that of existing ones on an assortment of parameters and under different conditions and settings.

II. REVIEW OF LITERATURE

Load balancing for distributed figuring framework is profoundly examined, for quite a while. For the most

part, there are two strategies for load balancing in a distributed situation:

i) A static load balancing approach [1,2,3] relies upon static data, for example, CPU limit, memory space and so forth in settling on load balancing choices.
ii) Dynamic load balancing arrangements [2, 4, and 5] settles on load balancing choices dependent on the present condition of the framework and consequently can additionally enhance the framework execution.

Various load-balancing approaches utilize chart calculations [12] for load conveyance. For load balancing procedure of a distributed system, the calculations require two rounds of message trades to disseminate the workload data, therefore, expanding the correspondence intricacy of the framework. In spite of the fact that the conventional goal of load circulation is to limit the general execution time called make span when managing substantial systems, the correspondence cost is likewise a basic issue. Arora et al. [13] proposed a decentralized load balancing calculation for heterogeneous Grid condition.

In spite of the fact that they endeavoured to incorporate correspondence inactivity amid activating procedure yet the genuine expense of job exchange was not considered. Zaki et al. [7] consider diverse processor speeds and appropriate the load sufficiently. Hendrickson and Devine [8] considered the distinctive measure of processor power and memory limit with respect to heterogeneous systems. They likewise stressed that organize associations with various paces must be considered for distributed load balancing technique. Kielmann et al. [9] considered an accumulation of groups as a various leveled framework and utilized a tree topology to do load balancing. Willebeek and Reeves exhibited a various leveled balancing strategy (HBM) which sorts out the framework into a progressive system of subsystems. The base of the HBM technique incorporates the worldwide data about the framework and can be a bottleneck.

Our methodology for load balancing thinks about the system as a two-dimension progression by evacuating the base of the tree. This model enhances normal response time of the framework. We have utilized the occasional status trade idea of ELISA [6].

In ELISA, load-balancing choice is taken dependent on line length as it were. Our model settles on load balancing choice by considering every single influencing factor, which are current load at processor and memory, line length and preparing limit of a node.

III. RELATED WORK

Online examination is one of the vital parts for online training framework. It is proficient, quick enough and decreases the substantial measure of the material asset. An examination framework is produced dependent on the web. Online examination will reduce the hurried control of assessing the appropriate responses given by the candidates physically. Being an organized Online examination structure it will diminish paperwork.

The primary objective of this online examination framework is to adequately assess the understudy completely through a completely computerized framework that decreases the required time as well as acquires quick and exact outcomes. Utilizing an open source dialect gives us greater adaptability, and yet it required more opportunity to be modified. The proposed Online Examination System (OES) can be effectively embraced by colleges and establishments so as to make the test progressively secure and increasingly adaptable. The framework is subdivided into two principal subsystems (understudy and head) that are intended to give the framework most

extreme advantage by exhibiting cautiously every subsystem service.

The formal paper-based examination framework is a long technique to lead an examination. The formal framework makes everything in paper-based. It gives advantageous time to go to for understudies. So the procedure winds up wild additionally the beforehand online examination process utilizes a single server because of this the test procedure is moderate or just a couple of understudies can go to the online examination at the same time.

IV. PROPOSED ARCHITECTURE

The entire system consists of three main components that interact to provide the previously described services.

The Test bed at the back end is a physical set up which can be configured to simulate a variety of network conditions, load conditions, server architectures and load balancing policies.

The web service at the front end is where the interaction with the users takes place and which automates the process from submission of test parameters by the user, configuring the test bed to follow those parameters, running tests and display of results to the users. It includes the web server, application logic, database server and the interface to the testbed.

Also an API has been designed and implemented using which new policies can be submitted for testing on the testbed. It includes standard libraries (API's) using which a set of interfaces have to be implemented by the user according to the policy he is submitting.

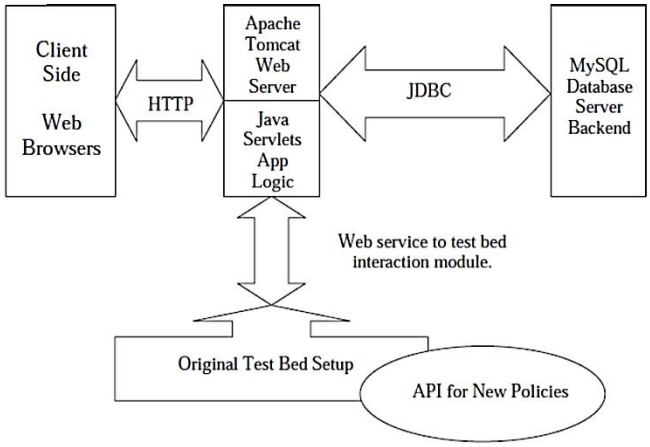

Figure 1. System Architecture

V. IMPLEMENTATION

Load Balancing: In computing, load balancing improves the distribution of workloads across multiple computing resources, such as computers,

- computer cluster,
- network links,
- central processing units,
- Disk drives.

Load balancing aims to optimize resource use, maximize throughput, minimize response time, and avoid overload of any single resource. Using multiple components with load balancing instead of a single component may increase reliability and availability through redundancy. Ally involves dedicated software or hardware, such as a multilayer switch or a Domain Name.

In our system we utilizes multiple server (server_1, server_2... server_n) to balance the workload and to improve the efficiency. For Load balancing Round Robin Algorithm is used.

In this approach, time quantum is taken as the range of the CPU burst time of all the processes. The range of the processes is the difference between the largest (maximum) and smallest (minimum) values.

A. Uniqueness of Our Approach

Let's assume that the data are sorted in increasing numerical order. It gives better turnaround time and waiting time. Generally, the performance of RR algorithm depends upon the size of static Time Quantum (TQ). If the TQ is extremely large, the algorithm approximate to First-Come First-Served (FCFS). If the TQ is extremely small, the algorithm causes too many context switches. So, our approach solves this problem by taking a dynamic TQ where the TQ is the difference between maximum and minimum CPU burst time as shown in equation (1).

TQ = MAXBT – MINBT (1)
Where MAXBT = MAXimum Burst Time
MINBT = MINimum Burst Time

B. Proposed Algorithm

In our algorithm, processes are already present in the Ready Queue (RQ). By default, Arrival Time (AT) is assigned to zero. The number of processes 'n' and CPU Burst Time (BT) are accepted as input and Average Turnaround Time (ATT), Average Waiting Time (AWT) and number of Context Switch (CS) are produced as output. Let TQ and TQnew be the time quantum and new time quantum respectively. The pseudocode for the algorithm is presented as follows and the flowchart of the algorithm is presented in Figure 2.

1. All the processes present in the ready queue are sorted in ascending order.
//n = number of processes, i = loop variable
2. while (RQ != NULL)
//RQ = Ready Queue
TQ = MAXBT – MINBT
//TQ = Time Quantum
//MAXBT = MAXimum Burst Time
//MINBT = MINimum Burst Time
(Remaining burst time of the processes)
// If one process is there then TQ is equal to BT of itself
3. if (TQ < 25)
set TQnew = 25
else
set TQnew = TQ
end if
4. //Assign TQ to (1 to n) process
for i = 1 to n
{
Pi → TQnew
}
end for
// Assign TQnew to all the available processes.
5. Calculate the remaining burst time of the processes.
6. if (new process is arrived and BT != 0)
go to step 1
else if (new process is not arrived and BT != 0)
go to step 2
else if (new process is arrived and BT == 0)
go to step 1
else
go to step 7
end if
end while
7. Calculate ATT, AWT and CS.
//ATT = Average Turnaround Time
//AWT = Average Waiting Time
//CS = number of Context Switches
8. End

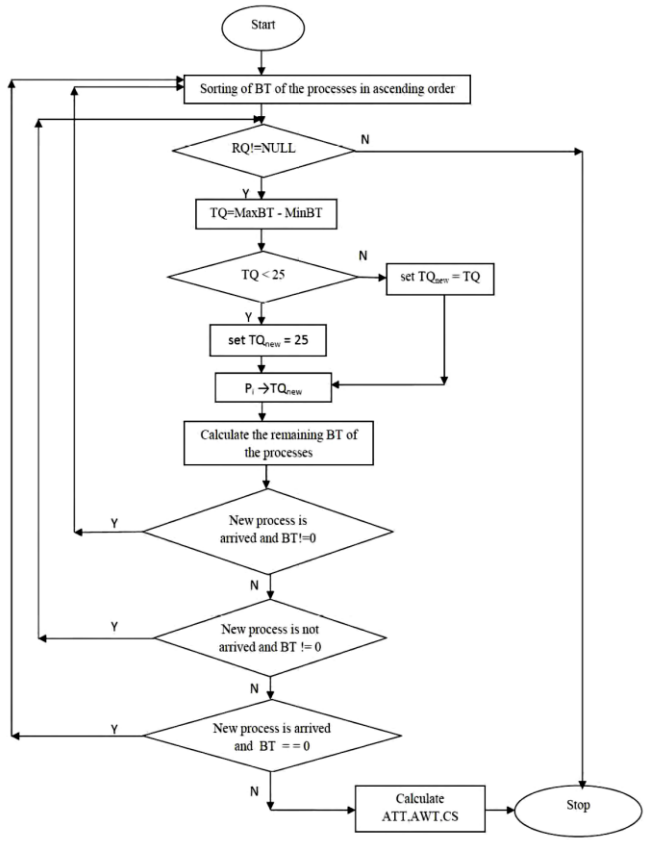

Figure 2. Flowchart of Min-Max Round Robin (MMRR) algorithm

VI. CONCLUSION

Load balancing based Online Examination System is a web application. The key idea is utilized numerous server for load balancing reason and to limit the measure of paper and convert all types of documentation to advanced shape. It can see that the data required can be gotten effortlessly and exactness in the mechanized framework. The client with least learning about PC can be capable work the framework effortlessly. The framework additionally delivers brief outcome required by the administration.

VII. REFERENCES

[1] S. F. El-Zoghdy, H. Kameda, J. Li, "Comparison of dynamic vs. static load balancing policies in a mainframe-personal computer network model", INFORMATION, vol. 5, no. 4, pp. 431-446, 2002.

[2] K.Y. Kabalan, W.W. Smari, J.Y. Hakimian, "Adaptive load Sharing in heterogeneous system: Policies Modifications and Simulation", CiteSeerx, 2008.

[3] H. C. Lin, C.S. Raghavendra, "A Dynamic Load Balancing Policy with a Central Job Dispatcher (LBC)" in , 1992.

[4] Ioannis Konstantinou, Dimitrios Tsoumakos, "Fast and Cost-Effective Online Load-Balancing in Distributed Range-Queriable Systems", Parallel and Distributed Systems IEEE Transactions on, vol. 22, no. 8, pp. 1350-1364, 2011.

[5] I. Ahmad, A. Ghafoor, K. Mehrotra, "Performance Prediction of Distributed Load Balancing on Multicomputer Systems", ACM, pp. 830-839, 1991.

[6] L. Anand, D. Ghose, V. Mani, "ELISA: An Estimated Load Information Scheduling Algorithm for distributed computing systems", International Journal on Computers and Mathematics with Applications, vol. 37, no. 8, pp. 57-85, April 1999.

[7] Mohammad Javeed Zaki, We Li, Srinivasan Parthasarathy, "Customized Dymnamic Load Balancing for a network of workstations", Journal of Parallel and Distributed Computing, vol. 43, no. 2, pp. 156-162.

[8] K Devine, B Hendrickson, E. Boman, M St John, C Vaghan, "Design of dynamic load balancing tools for parallel applications", Proceedings of the 14th International Conference on Super Computing, pp. 110-118, 2000.

[9] Kielmann Niewpoort, Bal, "Efficient Load Balancing for wide area divide and conquer applications", proceedings of the eighth ACM SIGPLAN symposium on Principles and Practises of Paralle programming, pp. 34-43.

[10] Belabbas Yagoubi, Meriem Meddeber, "Distributed Load Balancing Model for Grid Computing", Revue ARIMA, vol. 12, pp. 43-60, 2010.

[11] P. Neelakantan, "International Journal of Research in Computer Science ISSN 2249–8265", Anadaptive load sharing algorithm for Heterogeneous distributed system, vol. 3, no. 3, pp. 9-15, 2013.

[12] L.K. Dey, D. Ghosh, Satya Bagchi, J. Zhang, "Efficient Load Balancing Algorithm Using Complete Graph" in ICAIC 2011 Part V CCIS, vol. 228, pp. 643-646, 2011.

[13] M. Arora, S.K. Das, R. Biswas, "A De-centralized scheduling and load balancing algorithm for heterogeneous grid environments", Proceedings of the International Conference on Parallel Processing Workshops (ICPPW.2002), pp. 499-505, 2002.

[14] A. Menendez LC, H. Benitez-Perez, "Node Availability for Distributed Systems considering processor and RAM utilization for Load Balancing", Int. J. of Computers Communications & Control, vol. V, no. 3, pp. 336-350, 2010, ISSN 1841-9836, E-ISSN 1841-9844.

[15] Orly Kremen, Jeff Kramer, "Methodical Analysis of Adaptive Load Sharing Algorithms", IEEE Transactions on Parallel and Distributed Systems, vol. 3, no. 6, November 1992.

[16] Marc. H. Willebeek, Le Mair, Anthony P. Reeves, "Strategies for Dynamic Load Balancing on Highly Parallel Computers", IEEE Transactions on Parallel and Distributed Systems, vol. 4, no. 9, September 1993.

Cite this article as :

Diksha Ninave, Abhishek Aphale, Ashlesha Wath, Ankita Watkar, Tushar Likhar, Prof. Shraddha Karale, " Implementation of Online Test Engine Based on Load Balancing", International Journal of Scientific Research in Science and Technology(IJSRST), Print ISSN : 2395-6011, Online ISSN : 2395-602X, Volume 6, Issue 2, pp.69-75, March-April-2019. Journal URL : http://ijsrst.com/IJSRST119624