International Journal of Innovative Research and Advanced Studies (IJIRAS)
Volume 5 Issue 7, July 2018

ISSN: 2394-4404

Improved Intelligent Load Balancing Model For A Multiserver Environment

Ekong A. P.

Nwachukwu E.O.

Onyejegbu L. N.

Department of Computer Science, University of Port Harcourt,
Rivers State, Nigeria

Abstract: ***The random assignment of jobs in a multi-server network environment can give rise to a situation where some servers are heavily over-utilized while others are under-utilized. Existing systems on load balancing has several shortcomings such as the narrowness of parameter in consideration, the time spent by processes in the system, the failure to reassigned executing processes on failed servers to active and suitable ones. In this article, An Improved Intelligent Load Balancing (ILB) Model for a Mutliserver Environment has been proposed with the aim of developing an improved system. A software to be used as a tool for evaluating the performance of the model has been developed and the model using the tool developed has been implemented and evaluated using server cumulative total process time as the parameter and the result compared with existing static and intelligent algorithms. The developed algorithm uses Ram Capacity, CPU and storage utilization in determining threshold values for each server. It has been assumed that all the servers in the network have different or similar computational capacity. The result obtained showed that there 75% less time was spent by processes using Improved Intelligent Load Balancing than using Intelligent and static load balancing alternatives. The improved model shows that process spends less time in the system as the improved ILB algorithm distributes the load evenly to all the virtual machines based on the learnt server status and reassign processes on failed servers to active and less busy ones hence solving the problem of load imbalance and high cost incurred by traditional algorithms.***

I. INTRODUCTION

Load balancing is defined to be the distribution of workloads across many computing resources. The resources could be computers, central processing unit or other sharable resources. In a multiserver environment, it is the distribution of workloads across different servers in a network. Load balancing in a multiserver environment main goal is to ensure that servers are optimally used, enhance throughput and to reduce the response time, hence avoiding the over-utilization of any particular server. It mostly involves agent, hardware and/or software, to handle it. Intelligent server load balancing is the automatic selection of a server based on the server's configuration and status and assigning the processes or jobs on queue to the most appropriate one. In consideration of the competition in Information Technology, inefficient network systems can cause a considerable loss. The nature of queues and the ways they are managed are of paramount importance to good network design. For this reason, queuing theory and scheduling to maintain a competitive advantage is often applied. The knowledge of probabilities in addition to some improvements mechanisms, and artificial intelligence, design and implementation of excellent scheduling systems and models makes it feasible to balance the load in a multiserver scenario.

II. LITERATURE REVIEW

Load Balancing being the distribution of workload among different computing resources in order to achieve optimal utilization of resources hence increasing system throughput

and minimizing total system response time is used to avoid overload on the resources and for sharing traffic among different servers *Nusrat et al. (2014*).Using Load balancing, Data communication can take place with minimum delay. It is used to minimize waiting time. In the clouds, load balancing is used for balancing load on virtual cloud machines and other resources resident there (*Nusrat et al., 2014*). Intelligent load balancing is the automatic selection of a system based on the server configuration and status and assigning the job on queue to the most appropriate server.

APPROACHES TO INTELLIGENT LOAD BALANCING IN A MULTI-SERVER ENVIRONMENT

A. RAM BASED APPROACH ON LOAD BALANCING

(Amritpal S. et al., 2014) discussed an algorithm for load balancing that is dynamic. He noted that in the cloud, VMs are of varying configuration with respect to the user requirement and ordered based on Random Access Memory (RAM). When the balancer receives a request, the request is given to the machine with the highest available RAM. The balancer also has a database of the priorities and the request or process count currently running or allocated to the each virtual machine.

He concluded that good load balancing provide performance benefits and that by introducing a new model for load balancing, server conditions can easily and dynamically be adapted to , hence solving most of the traffic related issues. By testing the model using the metrics of time of allocation, and responsiveness, the Load balancing Model works very efficiency.

a. LIMITATIONS OF AMRITPAL S. ET AL. ALGORITHM

The test parameters are so limited and he used RAM capacity of servers only and hence cannot be said to be comprehensive since other parameters like the CPU speed and storage capacity, etc are also very important in systems performances and has effect on how load balancing is done. Furthermore, there is no feedback mechanism by the server to the balancer on process status and processes are not reassigned if server fails in action

B. LOAD BALANCING IN THE CLOUD (DODDINI P., 2013)

(Doddini P., 2013) discussed the requirements for cloud computing with regards to factors such as access control, migration, security, etc and also reviewed the models employed in the clouds for balancing loads; such as weighted active monitoring, dynamic and static load balancing algorithms, ant colony optimization models, and distributed systems load balancing. He attempted to identify the problems in the existing models using some metrics and noticed that they were not performing optimally as a result of improper implementation of load balancing models

a. LIMITATION OF DODDINI ALGORITHM

Doddini concentrated on issues that research are ongoing in the area of load balancing and did not bring any major algorithm to help to improve the existing system.

C. LOAD BALANCING IN THE CLOUD DATA CENTERS BY HEMONT ET AL., (2013)

(Hemont et al., 2013) researched on Load Balancing on Cloud Data Centres. He analysed the load balancing models and their respective implications using a cloud simulating tool, cloudsim. In his research, different user communicated with the data centres and the output was produced. He calculated the maximum and minimum time together with data migration cost. He concluded that in Equally Spread, Round Robin and Throttled Load balancing, the request time were similar. Moreso, the calculated cost for Virtual Machine usage is same for Equally Spread and Round Robin but Throttled Load balancing algorithm showed a reduction in the usage cost implying that the Throttled Load balancing algorithm works more efficiently in the cost metrics for cloud load balancing.

a. LIMITATIONS HEMONT RESEARCH

He simply studied the existing algorithm and did not proposed an improved form.

D. CPU BASED LOAD BALANCING ALGORITHM BY NAYANDEEP SRAN

(Sran et al., 2013) proposed a non-dynamic scheduling model with overhead due VM selection and considered only one factor. He further discussed the two types of policies for VMs: consolidation and migration. He gave priority to Virtual Machine migration policy using resources at its disposal. At first, the CPU utilization of the machine is checked 80% here is an arbitrary value chosen the threshold for the machine being overloaded.

If utilization of the machine is greater than or equal to 80%, that machine was considered to be over utilized and underutilized if else.

Finally, the algorithm allocates priority to virtual machines and the migrated to such machines are thereafter done according to the highest-priority- first method. He noticed from analysis that there is a reduction of the average response time. He concluded that the proposed algorithm can balance the load much more than the existing algorithms. His Load Balancing algorithm was targeted at providing dynamic balance of resources on demand to accomplish any required task. This algorithm also guarantees better security, by focusing on hiding personal details of the cloud users from the cloud provider known as identity based security.

a. LIMITATIONS OF NAYANDEEP SRAN

It is clear that his algorithm works on various parameters such as CPU utilization, data centers in different regions, response time and processing time. However, his Algorithm does not handle certain cases like the failure of nodes.

Another limitation is that the load factor above which the highest priority node is selected is kept constant at 80%).

E. ROUND ROBIN (RR) LOAD BALANCING ALGORITHM (NUSRAT PASHA ET AL., 2014)

Nusrat analysed Round Robin approach to Load Balancing Algorithm in the Clouds and proposed an improved version of the algorithm. In this, he assigned the minimum time and maximum time to different Virtual Machines (VM). From there, he noted that the RR performed well when overall response is considered, the request is small and with number of Virtual Machine set and perform badly when there is a large VM set. Also, by the application of service broker policy, the issue of deadlock and server overflow was greatly minimized. He concluded that the algorithm showed that there was an improvement in the overall performance with respect to time consumption in scheduling.

a. LIMITATION OF NUSRAT PASHA ROUND ROBIN APPROACH

The limitation of the research is that it only concentrates on one algorithm, the round robin algorithm which is a static algorithm and hence highly limited in its applicability.

III. THE PROPOSED SYSTEM

We propose an intelligent model which will introduce intelligence into balancing in multi-server environment: dynamically learning the server state and configuration and sending the process to the less busy server, which will offer a remarkable improvement over the previous or existing models or algorithms that either used an entirely static approach or a dynamic approach that covers only few server parameters. See figure 1 for illustration of the proposed model.

Here, the necessary input are processes and is sent to the appropriate server by the balancer based on the learnt servers' status. At the end, the total time spent by processes in the system is computed and placed side by side with the time spent using the existing load balancing models. Based on this, appropriate decisions on improving the service delivery like increasing the number of servers systems could be made. For example, it will be clear if a process waits longer than necessary in the queue based on the arrival time and the leaving time. Figure 2 shows the input/output interface.

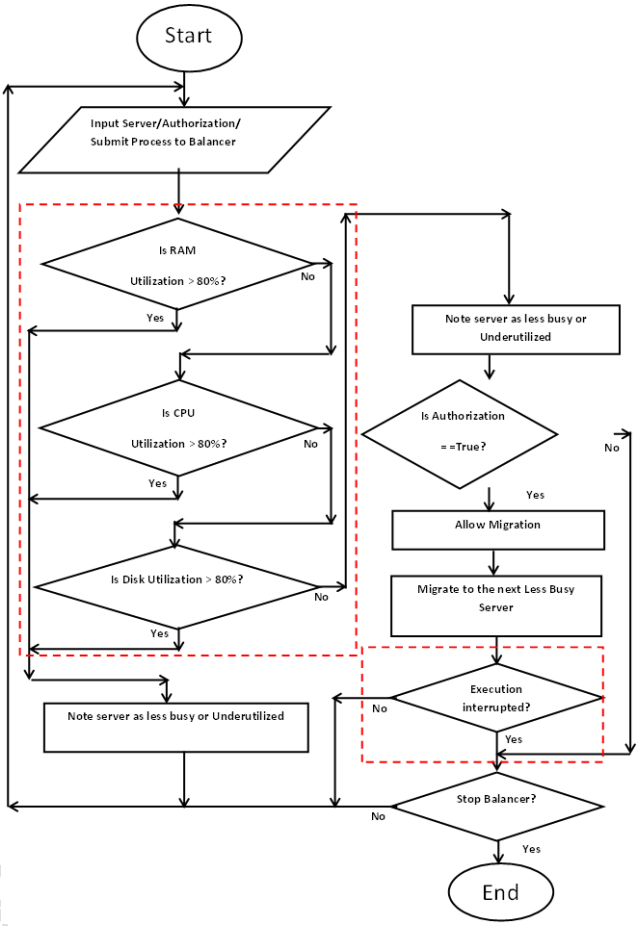

Figure 1: Illustration of the Proposed Model

A. PSEUDOCODE AND ALGORITHM OF THE PROPOSED SYSTEM

Pseudocode
Load balancer_on_best server ()
{
SET all the Machine allocation status to AVAILABLE in the Machine status list;
SET balancer database with no entries;
While(not end of request)
Do {
queue the requests;
If(hash map contain any entry of a Machine specification && Machine all status == AVAILABLE)
{
If (machine cpu utilization>=threshold in percent)
{
add machine to overload Machine map
}
Else
add machine to underutilized Machine map
Check process system requirement
if (process requirement<=machine system specification && VM priority=highest)
{
Migrate process to server }
}

Remove process from queue
Update process table with process executing
Start process timer
If execution time>maximum timer
{
Return server dead
Reassign process to next server meeting the requirement
Update relevant databases
}
Server update the balancer process table with process complete status
Reinitialize server status && return server available
Return proceed assigned Update the entryies and the machine in the hash map and the machine status list;
}
Else
{ Leave the process on queue; }
}
While process queue<>empty

ALGORITHM

Step 1: Initialize the clients
Step 2: Initialize the balancer
Step 3: SET all the Machine allocation status to LESS BUSY in the Machine state list;
Step 4: Accept client's entries to process queues
Step 5: Check the Server RAM Utilization
Step 5: Check the Server CPU Utilization
Step 5: Check the Server DISK Utilization
Step 6: Set the Server Status to Busy if any of RAM, CPU or DISK Utilization is greater than 80%
Step 7: Allocate process to server if server status is less busy
Reallocate process to another server if the server becomes unavailable while executing
Step4: remove process from queue

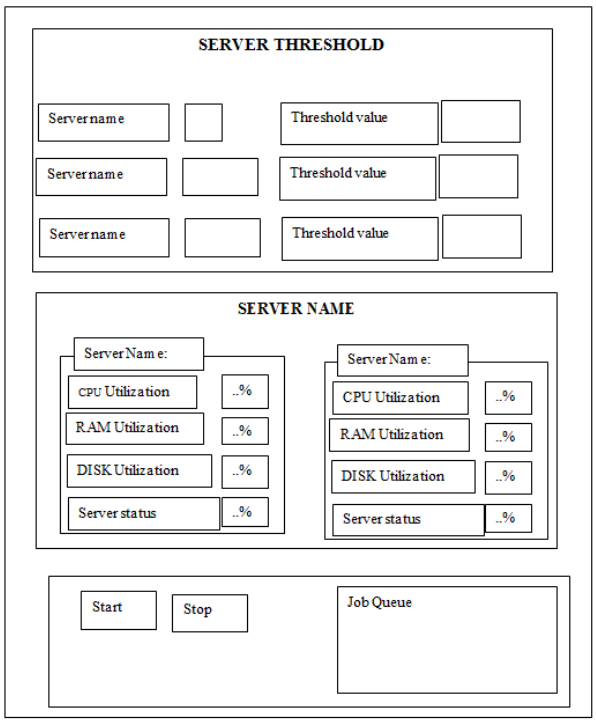

Figure 2: Input /Output Interface

B. MAIN SYSTEM COMPONENT

The system consist of a system that gets the process submission time, and the process and check the servers to determine which of them is best suited to handle the job or process at that instance of time and then transfer the process to that particular server. It also gets the job completed notification from the server when the job is done, in case the process is aborted halfway, the balancer assigns it to another best suited server. We use simulation and setup a virtual network with some servers, and connect them to the systems with the intelligent balancer, then the virtual clients are also connected to the balancer as well, as shown in figure 3

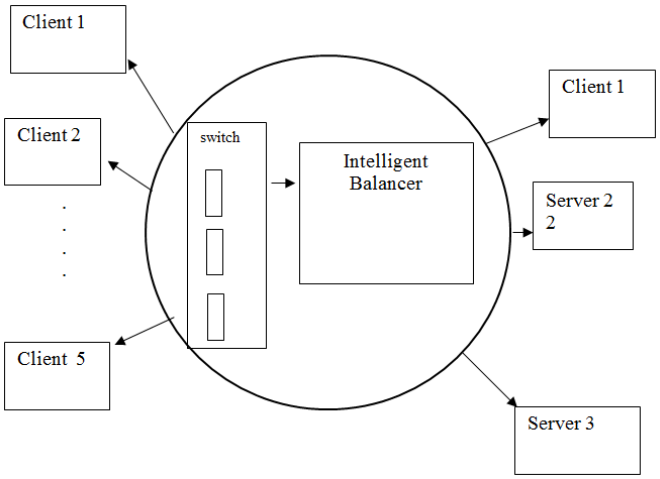

Figure 3: Illustration of Component Connection

C. ADVANTAGES OF THE NEW SYSTEMS

- ✓ It is intelligent and hence can learn the state of the servers
- ✓ It leads to optimal server utilization
- ✓ It reduces process waiting time
- ✓ It prevents process starvation
- ✓ It reduces the service time
- ✓ It reduces the time a process spends in the system
- ✓ It prevents deadlock

IV. RESULT ANALYSIS

We had proposed a load balancing model that demonstrates an improved intelligence over the existing intelligent and static models and which considered several factors in selecting a server in order to achieve good service rate to waiting processes and balanced server load. I worked on an algorithm that migrates jobs or processes to the highest priority server based on such server status at any instance as dictated by the usable resources with respect to the considered metric (RAM, CPU and disk utilization) that is available in such servers.

Here, the algorithm checked the current situation of the RAM, CPU, disk utilization against the set threshold, which

can be changed, to suite any administrative or technical objective.

If the utilization of server is below the set threshold, then that machine is noted as "Less busy" and no process or job can be allocated to such server and "busy" otherwise. The next process in queue is then allocated to the first less busy server. Moreso, If the server doesn't respond for some time, it is then declared unavailable and the process assigned to another server available. In the proposed model, there has been a significant reduction in the total time used by a server to attend to all assigned processes and he model is able to intelligently balance workload across the servers in a multiserver scenario.

We established a prototype system to simulate the multiserver load balancing system. This system consisted of three(3) homogenous virtual servers five (5) clients systems. In our simulation, we abstracted the physical network of computing. We supposed that all the servers in the network have the same computational capacity but with different initial working loads. When the servers working load overweighs its computational capacity as set in the threshold, the server is noted by the balancer as busy.

In the experiment, we submitted processes from the virtual clients to the process queue in the balancer. At this time the server status, based on memory, processor and Disk utilization is being learnt by the balancer and designated as busy or not busy depending on whether the metrics under consideration is greater that the set threshold and the status is being updated frequently by the balancer. Based on the status of the server, the balancer migrates the process from the process submitted queue to the appropriate server and add it to executing process queue.

As a control, at the same time the server is also assessed by the balancer based on a single metrics, the CPU, where the process is assigned to the serve whose CPU utilization is less that the threshold for CPU as set in line with existing with intelligent load balancing model which uses a single parameter. Simultaneously the process are being assigned to the servers in a round robin fashion, which is a static approach to load balancing.

We had to test the three algorithms simultaneously as a control experiment must be tested with the same conditions as the main experiment and since the systems variables are continuously changing, the algorithms has to be tested at the same environmental instance. The generated values are being upload into a database and the results shown as graphs in figure 4.

The results shows that whereas an average of 3 seconds was used using Improved Intelligent Load Balancing Algorithm, 12 seconds each using Intelligent and static load balancing alternatives, this clearly shows a great reduction in the time spent by processes in the system even when the process load is uneven due to the intelligence applied in allocating such process based on the server status at the instance of allocation attempt instead of just dumping the processes on the server in a round robin fashion without consideration of the server status.

Hence, we can deduce that the improved intelligent balancing algorithm is better in overall by dynamically allocating processes to the most suitable server hence marking a remarkable improvement in the time it takes for the process queue to be exhausted.

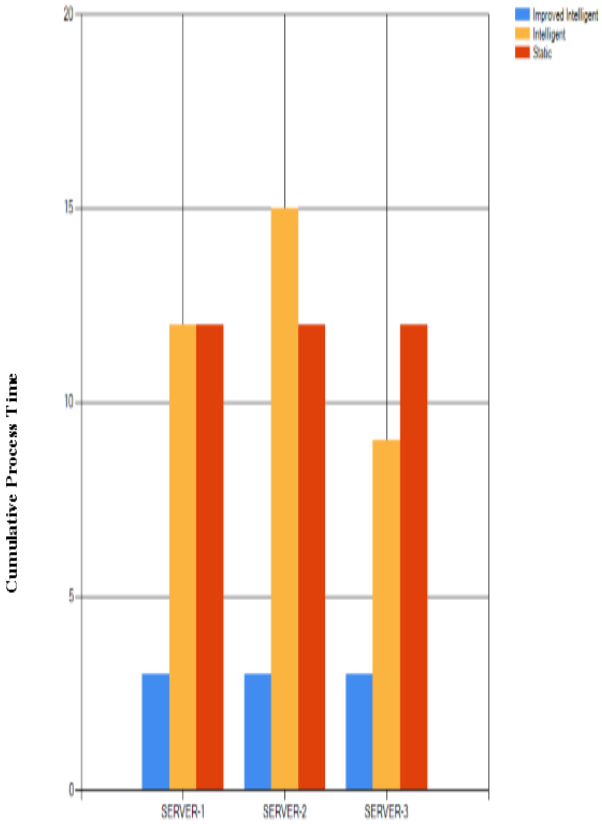

Figure 4: chart showing cumulative process time per serve using improved intelligent, intelligent and static algorithms

V. CONTRIBUTION TO KNOWLEDGE

This research has contributed to knowledge in the following ways:

- ✓ An improved model for intelligent load balancing System for a multi-server environment has been developed
- ✓ focus has been shifted from just a single parameter to the many other issues that can affect a working server
- ✓ it has solved the problem of process dying in action by ensuring process reassignment to other servers
- ✓ Jobs are assigned appropriately instead of blind job assignment as in a non-intelligent scenario.

VI. CONCLUSION

Intelligent load balancing provides useful methods for controlling the utilization of servers in a cluster server environment. An in depth literature review was carried out and a critical analysis of such literature identified major shortcomings and challenges in static and non-intelligent load balancing such as over utilization of servers, congestion of link to over utilized servers, non consideration of the server status before jobs are assigned. Due to these problems, the

service time for jobs are higher than supposed, which reduces the overall performance of the whole network system. Moreso, due to increase in service times, more servers are added with the hope of increasing the throughput which adds to undesirable and non-essential cost. Further, to search the efficient and relevant server is also a big challenge. The diagnostic approach to above challenges directs this research towards the design of an efficient model, Intelligent load balancing model which it is desired to ensure optimal system use and hence distribute load evenly among servers in a multiserver environment The experiments were performed using three(3) servers connected in a wired network and two(2) client with one (1) system as the balancer. Differences in the service times of different already existing algorithms and the proposed approach were compared. A major improvement in the service time was observed in contrast to those traditional or existing methods. Also, an attempt was implicitly made to reduce server load and network traffic congestion and it actually resulted into reduction of response time and hence an improved overall performance could be observed. The aim of this work was to find the less busy server and assign the next available job on queue and hence balancing of the servers could be achieved with the help of collective intelligence based framework for cluster server setup load balancing. The proposed concept is an extension of, rather than a replacement for, traditional and existing load balancing models. To this end it has been established that the use of an improved intelligent load balancing model is indispensable if meaningful network throughput in a cluster server setup is to be achieved.

REFERENCES

[1] Amanpreet C. & Navtej S. (2014). A review study on cloud computing and Load Balancing algorithms, International Journal of Advanced Research in Computer Science and Software Engineering, 4(4).
[2] Amritpal S. (2014). Comparative analysis of proposed algorithm with existing load balancing scheduling algorithms in Cloud Computing, International Journal of Advanced Research in Computer Science and Software Engineering: (3)(1). ISSN 2278-6856
[3] Doddini P. (2013). Load Balancing Algorithms in Cloud Computing. International Journal of Advanced Computer and Mathematical Sciences ISSN 2230-9624. 4(3), 229-233.
[4] Nusrat P., Amit A. & Ravi R. (2014), Round Robin approach to VM load balancing algorithm in clouds computing environment. International Journal of Advanced Research in Computer Science and Software Engineering 4(5),34-39
[5] Hemont S., Parag R. & Vinay C. (2013) Load Balancing On Cloud Data Centres International Journal of Advanced Research in Computer Science and Software Engineering (1),January - 2013, 1-4, 3(1), ISSN: 2277 128X
[6] Sran N., Navdeep K. (2013) Zero Proof Authentication and Efficient Load Balancing Algorithm for Dynamic Cloud Environment International Journal of Advanced Research in Computer Science and Software Engineering 3(7):1327-1332.