International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)
Web Site: www.ijettcs.org Email: editor@ijettcs.org
Volume 3, Issue 6, November-December 2014 **ISSN 2278-6856**

Periodic Cargo balancing for multifarious entity network

FarzeenBasith

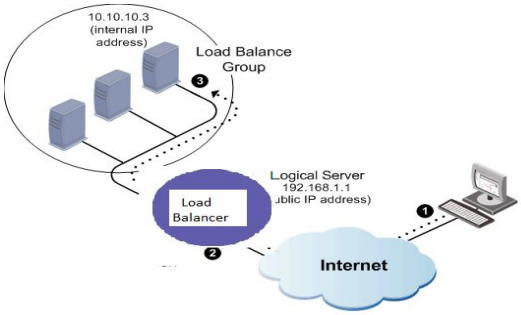
Assistant Professor, Department of MCA, Acharya Institute of Technology, Bangalore, Karnataka, India,

Abstract

Load balancing is a vital networking solution responsible for sharing new arrival of jobs efficiently so that individual servers are not over strike by sudden incoming of tasks. Many provisions have been proposed to handle the load balancing drawbacks. Still all these solutions either ignore the heterogeneity creation of the system or reassign the loads on the servers. The proposed technique gives a fruitful way to overcome the load balancing problem by estimating the application requests across collective servers and also a load balancer prevents any application server from becoming a single point of failure, thus by upgrading the overall application availability and responsiveness.

1. INTRODUCTION

Load balancing is dividing the amount of work that a computer has to do between two or more computers so that more work gets done in the same amount of time and, in general, all users get served faster. Load balancing can be implemented with hardware, software, or a combination of both. Typically, load balancing is the main reason for computer server clustering. Load balancers are used to increase capacity (concurrent users) and reliability of applications. They improve the overall performance of applications by decreasing the burden on servers associated with managing and maintaining application and network sessions, as well as by performing application-specific tasks.



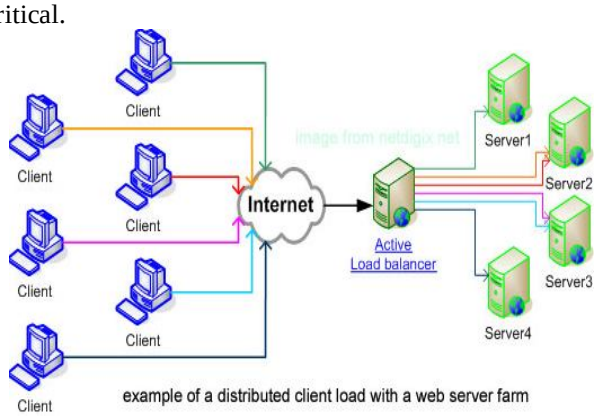
Load balancers are generally grouped into two categories: Layer 4 and Layer 7. Layer 4 load balancers act upon data found in network and transport layer protocols (IP, TCP, FTP, UDP). Layer 7 load balancers distribute requests based upon data found in application layer protocols such as HTTP. Requests are received by both types of load balancers and they are distributed to a particular server based on a configured algorithm.

Some algorithms are:

- Server load balancing
- Global server load balancing
- Firewall load balancing
- Transparent cache switching

1.1. Server load balancing

Server Load balancing is the process by which inbound Internet protocol (IP) traffic can be distributed across multiple servers. This enhances the performance of the servers, leads to their optimal use, and ensures that no single server is overwhelmed. The practice is particularly important for busy networks, where it is difficult to predict the number of requests that will be issued to a server. Server load balancing deals with distributing the load across multiple servers to scale beyond the capacity of one server, and to tolerate a server failure. A server load balancer, on the other hand, distributes traffic among server resources rather than network resources. Load balancers can perform sophisticated health checks on servers, applications, and content to improve availability and manageability. Because load balancers are deployed as the front end of a server farm, they also protect the servers from malicious users, and enhance security. There are two dimensions that drive the need for load balancing: servers and networks. With the advent of the Internet and intranet, networks connecting the servers to computers of employees, customers, or suppliers have become mission critical.



example of a distributed client load with a web server farm

The proliferation of servers for various applications has created data centers full of server farms. One must ensure scalability and high availability for all components, starting from the edge routers that connect to the Internet, all the way to the database servers in the back end. Load balancers have emerged as a powerful new weapon to solve many of these issues.

2. LITERATURE REVIEW

Definition of the problem

Dynamic load balancing is a recent technique that protects ISP networks from sudden congestion caused by load spikes or link failure and is essential for the use of highly efficient parallel system solving non-uniform problems with unpredictable load estimates and it will minimize the execution time of single application running in a multicomputer. Here we are finding the dynamic load balancing algorithm which would provide faster response time for each of the client.

2.1A fast adaptive load balancing method

D. Zhang et al.[1] proposed a binary tree structure that is used to partition the simulation region into sub-domains . The characteristics of this fast adaptive balancing method are to be adjusted the workload between the processors from local areas to global areas. According to the difference of workload, the arrangements of the cells are obtained. But the main workload concentrates on certain cells so that the procedure of adjusting the vertices of the grid can be very long because of the local workload can be considered. This problem can be avoided by the fast load balancing adaptive method. Here the region should be partitioned by using the binary tree mode, so that it contains leaf nodes, child nodes, parent nodes etc. There were partition line between the binary tree and the indexes of the cells on the left are smaller that of right and the indexes on the top are smaller than the bottom. Calculate the workload based on the balancing algorithm. This algorithm has a faster balancing speed, less elapsed time and less communication time cost of the simulation procedure. Advantages are Relative smaller communication overhead relative smaller communication overhead, faster balancing speed, and high efficiency and the disadvantage is it cannot maintain the topology that is neighboring cells cannot be maintained.

2.2 Honey Bee Behavior Inspired Load Balancing

Dhinesh et al. [2] proposed an algorithm named honeybee behavior inspired load balancing algorithm. Here in this session well load balance across the virtual machines for maximizing the throughput. The load balancing cloud computing can be achieved by modeling the foraging behavior of honey bees. This algorithm is derived from the behavior of honey bees that uses the method to find and reap food. In bee hives, there is a class of bees called the scout bees and the another type was forager bees .The scout bee which forage for food sources, when they find the food, they come back to the beehive to advertise this news by using a dance called waggle/tremble/vibration dance. The purpose of this dance, gives the idea of the quality and/or quantity of food and also its distance from the beehive. Forager bees then follow the Scout Bees to the location that they found food and then begin to reap it. After that they return to the beehive and do a tremble or vibration dance to other bees in the hive giving an idea of how much food is left. The tasks removed from the overloaded VMs act as Honey Bees. Upon submission to the under load VM, it will update the number of various priority tasks and load of tasks assigned to that VM. This information will be helpful for other tasks , i.e., whenever a high priority has to be submitted to VMs, it should consider the VM that has a minimum number of high priority tasks so that the particular task will be executed earlier. Since all VMs are sorted in an ascending order, the task removed will be submitted to under loaded VMs. Current workload of all available VMs can be calculated based on the information received from the data center. Advantages are maximizing the throughput; waiting time on task is minimum and overhead become minimum. The disadvantage is if more priority based queues are there then the lower priority load can be stay continuously in the queue.

2.3 A Dynamic and Adaptive Load Balancing Strategy For Parallel File System

B. Dong et al.[3] proposed a dynamic file migration load balancing algorithm based on distributed architecture. Considered the large file system there were various problems like dynamic file migration, algorithm based only on centralized system etc.So these problems are to be avoided by the introduction of the algorithm called self acting load balancing algorithm (SALB).In the parallel file system the data are transferred between the memory and the storage devices so that the data management is an important role of the parallel file system. There were various challenges that are faced during load balancing in the parallel file system. They are scalability and the availability of the system, network transmission and the load migration. Considered the dynamic load balancing algorithms, the load in each I/O servers are different because the workload becomes varies continuously. So there were somedecision making algorithms are needed. In this decision making system, there were firstly central decision maker, by which the central node is the decision maker so that if the central node becomes fail, then the whole system performance becomes down and the reliability becomes less. Secondly group decision maker in which the total system should be divided in to groups so that the communication cost becomes reduced. But taking decision without considered the whole system load so that global optimization explored a major problem. Finally the distributed decision maker in which each I/O server can take their own decision so that they provide better scalability and availability. This proposed SALB addressed the load prediction algorithm, efficient load collection mechanism, effective distributed decision maker, migration selection model and dynamic file migration algorithm for a better load balancing. The disadvantage is degradation of the whole system due to the migration side effect.

2.4 Heat Diffusion Based Dynamic Load Balancing

Yunhua.et al.[4] proposed an efficient cell selection scheme and two heat diffusion based algorithm called global and local diffusion. Considered the distributed virtual environments there were various numbers of users and the load accessing by the concurrent users can cause problem. This can be avoided by this algorithm. According to the heat diffusion algorithm, the virtual environment is divided in to large number of square cells

and each square cell having objects. The working of the heat diffusion algorithm is in such a way that every nodes in the cell sends load to its neighboring nodes in every iteration and the transfer was the difference between the current node to that of neighboring node. So it was related to heat diffusion process. That is the transfer of heat from high to low object, when they were placed adjacently In local diffusion algorithm, there were local decision making and efficient cell selection schemes are used. Here they simply compared the neighboring node loads to the adjacent node loads. If load is small then the transfer of load becomes possible. When global diffusion algorithm considered, becomes possible. When global diffusion algorithm considered, it has two stages that is global scheduling stage and local load migration stage. From various experimental results the global diffusion algorithm becomes the better one. Advantages are communication overhead is less, high speed and require little amount of calculations. Disadvantages are network delay is high and several iterations are taken so there was a waste of time.

2.5 Decentralized Scale-Free Network Construction and Load Balancing in Massive Multiuser Virtual Environments

Markus et al.[5] addressed the concept of overlay networks for the interconnection of machines that makes the backbone of an online environment. Virtual online world that makes the opportunities to the world for better technological advancements and developments. So the proposed network that makes better feasibility and load balancing to the dynamic virtual environments. This proposed system developed Hyper verse architecture, that can be responsible for the proper hosting of the virtual world. There were self organized load balancing method by which the world surface is subdivided in to small cells, and it is managed by a public server. In this cells various hotspots so that the absolute mass of the object in the cell can be calculated by the public server. Hotspot accuracy is better when increasing the network load. The proposed algorithm cannot avoid the overloaded nodes but find out the number of links that assigned to each node while joining the network. The advantages are the network becomes reliable, the network becomes resilience, efficient routing, and fault tolerant. The disadvantage is the overload ratio at the beginning is higher so that public servers are initially placed randomly so some time is used for balancing the load.

3.METHODOLOGIES

3.1 How to generate or collect input data

The user/web application sends a request for data from the server. Internally, the Load Balancer handles the request.

3.2 How to solve the problem

3.2.1 Algorithm Design

The algorithm proposed would consider the following factors in order to solve the Problem.

- Load
- Response Time

In order to calculate the Load on the server we would consider the number of requests that are taken care of by a particular server. The Response time is calculated by making use of the 'Internet Control Message Protocol (ICMP)' protocol. The Combination of the above factors helps us select an optimal server with least number of connections and a good response time. We calculate the product of the load and the response time by ELASTIC SERVER of all the active servers and then compare the values. Once the comparison is done the Load balancer would redirect the request to the server with the least value.

4.IMPLEMENTATION

Typically, two or more web servers are employed in a load balancing scheme. In case one of the servers begins to get overloaded, the requests are forwarded to another server. This process brings down the service time by allowing multiple servers to handle the requests. Service time is reduced by using a load balancer to identify which server has the appropriate availability to receive the traffic.

4.1 Algorithm

Start

Step 1: Select no of jobs.

Step 2: Find the server Si.

Step 3: allocate the job on the server based on Request.

Step 4: find the Least Server size in ES.

Step 5: If Server are not full then go to Step 2.

Step 6: If Server are full then find Least Empty Server.

Step 7: Update the Server.

End

Suppose we have N nodes. Let the weight of the ith node be xi and define X=∑xi. Partition the line segment from 0 to 1 at the following N−1 places:

pi= $\sum_{n=0}^{i} xi / X$, i=1,2,…,N−1

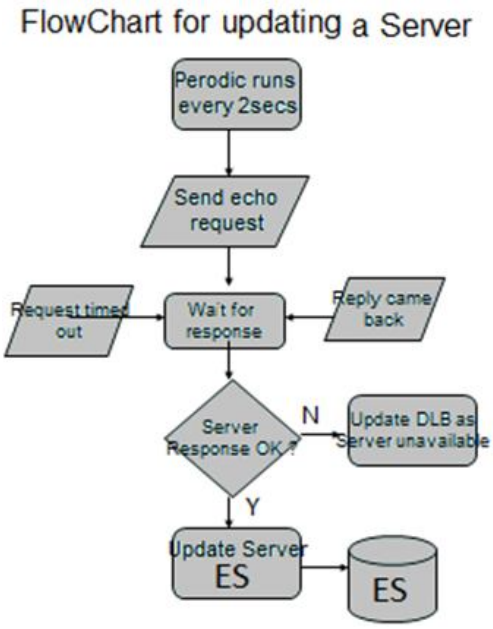
Thus, partition will look like {0,p1,p2,…,pN−1,1}. Then choose a uniformly distributed random number between 0 and 1 and select the node based on the partition in which it lies. That is, if the random number y∈ (0,p1), then choose node 1. If y∈ (p1,p2), then choose node 2, and so on.

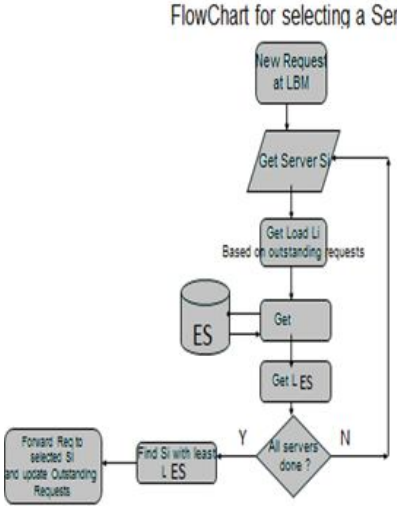
M=S1+S2+S3
LB=(J1−S1)+(J2−S2)+(J3−S3)
Consider Server S1=40, S2=40 and S3=40
Substitute S1,S2,S3 in M
Then we have M=120
Let there are number of jobs like J1=15, J2=20 and J3=40 then
LB=(15-40)+(20-40)+(40-40)
LB=45
Now we have 45 remaining capacity in server S1, S2 (ie. S1=25, S2=20 remaining capacity).

And that capacity can be fulfill by taking the new job value say, J4 and so on, And repeat the process.
By this we will reach LB=0.

5.RESULT

5.1 Design and Flowchart





5.2. Data Analysis

5.2.1 Output Generation

Input

- Varies server load based on requests-10k requests per server.
- Load each server with data.
- Simulate network congestion thereby varying theES.
- Delay Variation: Run iperf to create network traffic between two nodes.

Output

- Each client prints the server used and the response time for each request.
- Compare the worst/average response times for

1. Single iterative server, without multithreading.
2. Concurrent server with multi-threading
3. Concurrent server with multi-threading and dynamicLBM

- Checking Fault tolerance when there is no ES calculation.
- Getting larger ES by increasing congestion.
- Comparing response time when servers are Mild, Optimally and heavyloaded.

REFERENCE

[1] Dongliang Zhang, ChangjunJiang,Shu Li, "A fast adaptive load balancing method for parallel particle-based simulations",SimulationModelling Practice and Theory 17 (2009) 1032–1042.

[2] DhineshBabu L.D, P. VenkataKrishna, "Honey bee behavior inspired load balancing of tasks in cloud computing environments", Applied Soft Computing 13 (2013) 2292–2303.

[3] Bin Dong, Xiuqiao Li, Qimeng Wu, Limin Xiao, Li Ruan, "A dynamic and adaptive load balancing strategy for parallel file system with large-scale I/O servers", J. Parallel Distribution Computing. 72 (2012) 1254–1268.

[4] Yunhua Deng, Rynson W.H. Lau, "Heat diffusion based dynamic load balancing for distributed virtual environments", in: Proceedings of the17th ACM Symposium on Virtual Reality Software and technology, ACM, 2010, pp. 203–210.

[5] Markus Esch, Eric Tobias, "Decentralized scale-free network construction and load balancing in Massive Multiuser Virtual Environments", in: Collaborative Computing: Networking, Applications and Worksharing, Collaborate Com, 2010, 6th International Conference on, IEEE, 2010, pp. 1–10.

AUTHOR

Assistant Professor, Department of MCA, Acharya Institute of Technology, Karnataka, India. Teaching Experience :- 9 year