ADAPTIVE LOAD BALANCING FOR CLUSTER USING CONTENT AWARENESS WITH TRAFFIC MONITORING

Archana Nigam, Tejprakash Singh, Anuj Tiwari, Ankita Singhal

Abstract

With the rapid growth of both information and users on the Internet, how to effectively improve the quality of network service becomes an urgent problem to be addressed. Load balancing is a solution to this problem in an effective way. Different adaptive load balancing methods have been developed to estimate servers load performance. However, they suffer from either increased processing load on the servers, or additional traffic on the network and the servers, or are impractical in real time scenario.

In this paper, we used the concept of content based queues, and have used RTT passive measurement technique to get an adaptive load balancing algorithm inside the cluster and used the round robin load balancing algorithm outside the cluster. Using the same queue for each type of request results into overload on server, so we use the concept of different queue for different type of request. The whole load balancing task is performed by a webproxy, a proxy that has access to all servers so it also removes the drawback of dynamic algorithm in which most of the load balancing task is performed by server itself.

Keywords: **- Round Trip Time (RTT), Adaptive Load Balancing, Content awareness, Webproxy , Distributed Network.**

I. INTRODUCTION

A distributed system consists of a collection of autonomous computers, connected through a network and distribution middleware, which enables computers to coordinate their activities and to share the resources of the system, so that users perceive the system as a single, integrated computing facility [1]. The users of a distributed system have different goals, objectives and strategies and their behaviour is difficult to characterize. In such systems the management of resources and applications is a very challenging task.

When the demand for computing power increases the load balancing problem becomes important. The purpose of load balancing is to improve the performance of a distributed system through an appropriate distribution of the application load.

A general formulation of this problem is as follows: given a large number of requests, find the allocation of requests to servers while optimizing a given objective function (e.g. total execution time) [2]. It is to distribute a large number of requests to different servers, to ease the burden of a single server. Load-balancing technology can balance conflicting factors such as cost, performance, and scalability through a relatively low total cost of the computer cluster to achieve a strong performance that cannot be achieved by stand-alone system. Server load balancing is highly significant for network research and has broad market prospects.

Different load balancing methods have been developed to transfer load among servers. Some of them are impractical in real time scenario while others increase processing load on the server or on the network.

In this paper ,we use a new adaptive load balancing algorithm inside the cluster where the concept of rtt passive measurement technique and content awareness has been used By content awareness we mean having different queue for different request types rather than having the same queue for different requests, which improves the total execution time. Even if the adaptive load balancing method doesn't work then rtt passive measurement will do all load balancing task for selecting the appropriate cluster which increases the reliability of the system.

The Server Selection policy used in our paper is in compliance with the policy described in [3] where application layer RTT between the router and the server is a key parameter to monitor load/performance of server. RTT calculation is done through a passive measurement policy to remove any burden on the network.

II. RELATED WORK

Load balancing mechanisms can be broadly categorized as centralized or decentralized, dynamic or static [4].
In a centralized algorithm, there is a central scheduler which gathers all load information from the nodes and makes appropriate decisions. However, this approach is not scalable for a vast environment and also central scheduler is a bottleneck. In decentralized models, there is usually no specific node known as a server or collector. Instead, all nodes have information about some or all other nodes. This leads to a huge overhead in communication.

Static algorithms are not affected by the system state, as their behaviour is predetermined. On the other hand, dynamic algorithms make decisions according to the system state. The state refers to certain types of information, such as the number of jobs waiting in the ready queue, the current job arrival rate, etc. Dynamic algorithms tend to have better performance than static ones. Some dynamic load balancing algorithms are adaptive; in other words, dynamic policies are modifiable as the system state changes.

In a distributed network, content based adaptive routing algorithm is used where router directs user request to appropriate server but the key parameter for selecting the best server is difficult to choose. Many balancing techniques use different parameters to measure system performance [5]. One of the best way is to use "ICMP echo request and reply" between router and servers. But this is an active approach and creates lot of communication overhead which can be avoided by applying a passive admission monitoring method where no additional traffic is generated in network. Another problem with such a technique is that if all the requests processed by server are enqueued in a single queue it will slow down the overall response time.

The algorithm introduced here does the load balancing inside the cluster uses the concept of content awareness and passive measurement to decrease the overhead on the server significantly..

III. PROPOSED SOLUTION

Architecture

The concept of cluster is used to provide higher availability,reliability and scalability than can be obtained by using a single system. Benefit of having cluster architecture is, it provides high availability by making application software and data available on several servers linked together in a cluster configuration. If one server stops functioning, a process called failover automatically shifts the workload of the failed server to another server in the cluster.
Also, as described earlier, Load balancing algorithms can be classified as either static or dynamic. Unfortunately these methods generates additional processing load on the server, deteriorating its performance. The proposed architecture to handle this is as shown in Figure 1. The function of a web server is to service HTTP requests made by client. Typically the server receives a request asking for a specific resource, and it returns the resource as a response. A client might reference in its request a file, then that file is returned or, for example, a directory, then the content of that directory (codified in some suitable form) is returned. A client might also request a program, and it is the web server task to launch that program (CGI script) [7] and to return the output of that program to the client. Various other type of resources might be referenced in client's request. Different request have different sizes thus required server time is different for them. Client request is transferred to the web server via router, and the router distinguishes the request on the basis of content. As shown in Figure 1, there are clusters of server; each back end server inside the cluster keeps queues for handling each type of request. This database is maintained by the router for each server and router transfers the request to a particular queue of a server after rtt passive measurement. There is a common module for all the servers which we call as web proxy. This is the main improvement over other adaptive load balancing algorithms. It performs the load balancing task by running the algorithm (explained in the next section) thus freeing the servers from performing load balancing technique and thus improves efficiency.

.

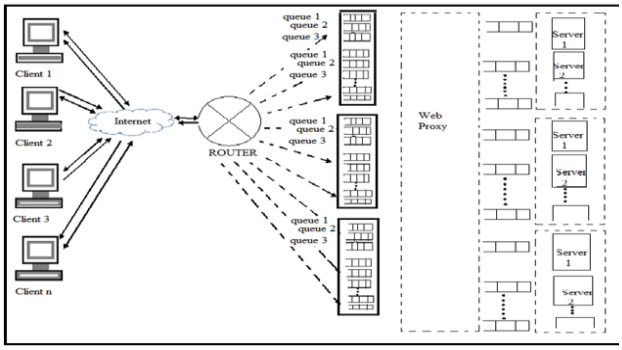


Figure 1 Architecture

RTT Passive Measurement Technique

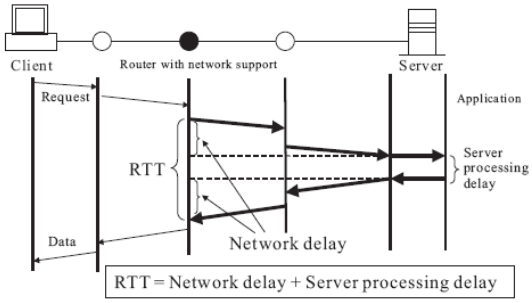


Figure 2 RTT Passive Measurement

The application layer RTT includes the network delay and the server processing delay [3] as shown in figure 2. In some situations, server latency may be dominant due to the load on the particular server. In this case, user response time may be improved by selecting a server that is lightly loaded. In another situation, network delay may be dominant due to congestion. In this case, network delay should be used for server selection. To realize good selection of servers, we believe that both server processing delay and network delay should be taken into account. To do so, we use the application layer RTT between the router and the server as information for server selection. This application layer RTT includes the network delay and the server processing delay. When a router in a network selects a server which has small application layer RTT, total response time can be improved. Now the question arises if router selects a server based on its application layer RTT , client request may have a tendency to concentrate at a particular server. To avoid this situation, we apply probabilistic server selection policy [3] at the router. Selection probability of the server whose RTT is large should be small and the server whose RTT is small should be selected with large probability. We apply the following simple method for calculation of the server selection probability. A router *i* calculates P_{ij}, a probability of selecting server *j*, as follows [3].

$$P_{ij} = \frac{\frac{1}{RTT_j}}{\sum_{m=1}^{n} \frac{1}{RTT_m}}$$

Where n is total number of servers serving the same service and RTT_m is the RTT between router *i* and server *m*.

Before describing the algorithm, first let us understand what is exactly mean by content awareness and its need. Suppose we have three servers in the system and each can handle three different type of request i.e. video, audio, and image. The current load of each of the servers is as follows.

Server 1 : 3 video requests.

Server 2 : 3 audio requests.

Server 3: 3 image requests.

If we consider just the length of the queues then all have same length but if the request is forwarded to server 1 then it will result in higher response time. So the technique that we introduced here is to have different queue for different kind of request and whenever a server gets overloaded due to the requests on a particular queue then transfer the request from this server to the same queue of other server thus balancing the load.

Algorithm

This is the adaptive load balancing algorithm run by webproxy, common for all the servers. So it has access to the entire server's queue and can update them as needed.
Let there be *n* Servers and each server has *m* queues, where each queues is for a different types of request. Each type of request is of different size e.g. Videos are of size say x units whereas audio are of size say y units whereas images are of z units (much smaller than x and y) and so on. We define the initial size of the queue which will change according to the current state of the system.
Define $load_i$ where $1 \le i \le n$, load on i_{th} server.
Max_j where $1 \le j \le m$, maximum load in j_{th} queue.
Min_j where $1 \le j \le m$, minimum load in j_{th} queue.
Initialize limit value by taking the average of the length of all the m queues on n server and divide it into the half. If value of $load_i$ is greater than limit value then transfer job from queue of that server to same type of queue of other server. We could have taken the full average value and compared load with

it, but it will be the situation where server is already overloaded and then measures have to be taken to transfer its load and make it under loaded or lightly loaded. In our case such a situation is not allowed to occur because we take precautionary measures i.e., divide the average into half and transfer the load well before any overloaded condition.

Algo for webproxy

Steps

1.For each i and j initialize $load_i$, $load_c$, Max_j, Min_j to 0.

2.Select cluster on the basis of round robin technique.

3. Repeat step 4 to 7, till server is on

4. If ($load_i$<limit)

Converts the m queues into the single queue on the basis of time and server serves the request in FCFS form.

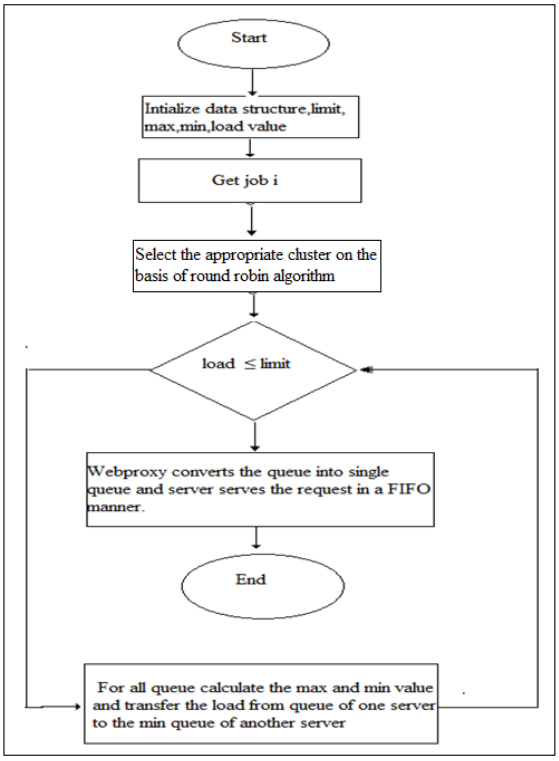
5. If ($load_i \geq$limit) Repeat Step 6 until ($load_i$<limit)

6. For each queue (from 1 to m) calculate max_j and min_j and transfer the request from queue j of server x having maximum load to queue j of server y having minimum load.

7.) Go to step 3.

8.) Exit

The above adaptive load balancing algorithm would be implemented inside the cluster for load balancing and for selecting the appropriate cluster static round robin load balancing algorithm is used. By round robin we mean selecting the cluster from 1 to k, where k is the number of cluster, and after reaching to kth server repeat the same procedure. Flowchart of adaptive algorithm is shown below.



IV. Experiment and Results

A. Simulation

We have implemented the algorithm in netbeans 7.1 using multithreading. Each thread created corresponds to a particular server. We designed a client module that generates requests from the web and handed over to the router. The generated requests are of different types. Cluster are selected on the basis of round robin algorithm and on the basis of type of request, router puts them in an appropriate queue. Router transfers the request to the webproxy where the entire relevant load balancing task is performed. We have also implemented the static algorithm based on round robin technique [8] and dynamic algorithm based on load balancing by content based routing technique [5] for making necessary comparison.

B. Simulation Results

For simulation, we have used the request from web and these requests are of different sizes and types. Algorithm 1 is static algorithm, Algorithm 2 is dynamic algorithm and Algorithm 3 is our proposed adaptive load balancing algorithm. We compare these algorithms on the basis of load distribution among servers and total response time.

C. Analysis of Results

From Fig 3, 4, 5 it may be concluded that the load distribution is fairer in Algorithm 3 i.e. in adaptive algorithm. In Figure 3 it is clear that for static algorithm or Algorithm 1 (server 1 gets overloaded because load is transferred in a round robin fashion but it may happen video or much larger size request are processed by any particular server, so that server will be overloaded) whereas in Figure 4, dynamic algorithm also distribute load to server 1 and 3 while server 2 remains unutilized.

Figure 6 shows the total response time and it is also clear from the graph that on an average total response time of Algorithm 3 is less than that of Algorithm 1 and 2.

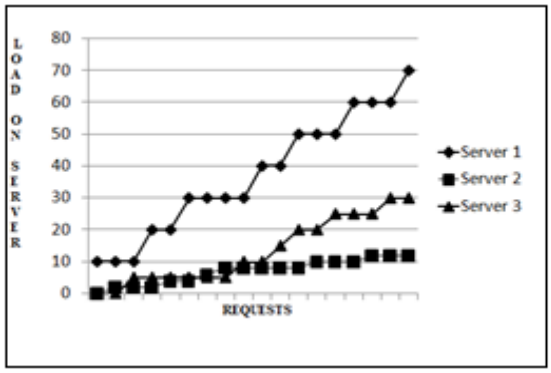


Figure 3 Static Algorithm

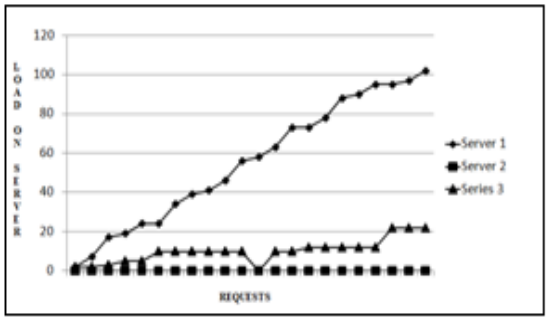


Figure 4 Dynamic load balancing algorithm

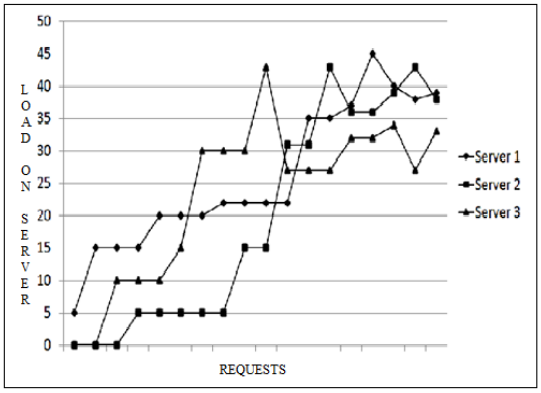


Figure 5 Proposed Adaptive load balancing algorithm

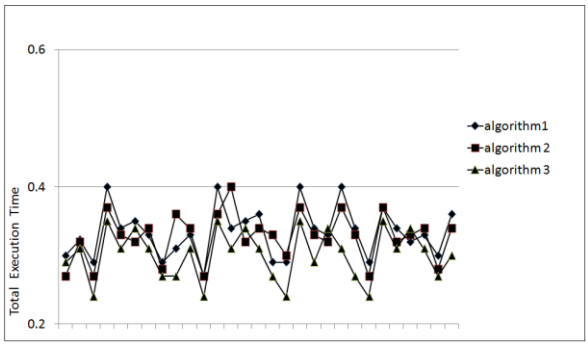


Figure 6 Comparison of three algorithms on the basis of Total Execution Time

V. Conclusion

An adaptive load balancing algorithm has been developed where the concept of rtt passive measurement and content awareness by having different queues for different type of request has been used. We have compared the proposed algorithm with the static algorithm and a dynamic algorithm. The result shows that proposed algorithm gives better result as the load on servers' increases. For lightly loaded server, static algorithm can be used which is less complex than the other two algorithms and hence reduces complexity. However, for huge workload, the proposed algorithm can be used effectively. Also the best part of algorithm is even if webproxy stops working the load balancing can be done on the basis of rtt passive measurement and in this case it is similar to dynamic algorithm, thus the proposed algorithm is more reliable than the dynamic algorithm.

In future we would proposed an adaptive load balancing algorithm for balancing the load among

the cluster and inside the cluster the proposed algorithm can be used. It will improve the network as well as server's performance.

REFERNCES

[1] Andrew S. Tanenbaum, Maarten Van Steen, "*Distributed System Principles and Paradigms"*.

[2] Satoru Ohta, Ryuichi Andou, "*WWW Server Load Balancing Technique Employing Passive Measurement of Server Performance*", ECTI Transactions on Electrical ENG, Electronics and communications vol.8, no.1 February 2010.

[3] Yagoubi, B. and M. Meriem (2010). "Distributed Load Balancing Model for Grid Computing." Revue ARIMA **12**: 43-60.

[4]Chronopoulos, A. T., R. Boppana, et al. "LOAD BALANCING IN DISTRIBUTED SYSTEMS: A GAME THEORETIC APPROACH."

[5] H. Miura and M. Yamamoto, "*Content routing with network support using passive measurement in content distribution networks*," IEICE Trans. on Communs. E86-B, pp.1805-1811, June 2003.

[6] Anuj Tiwari, Ankita Singhal and Archana Nigam, (2012), "ADAPTIVE LOAD BALANCING TECHNIQUE WITH PASSIVE MEASUREMENT AND TRAFFIC MONITORING", International Conference on Computer and Communication (ICCC-2012), 27-28 Jan 2012, Bhopal, Madhya Pradesh, India.

[7] Dragoi, O. A. (1999). "The conceptual architecture of the Apache web server." Dept. of Computer Science, University of Waterloo, http://www. math. uwaterloo. ca/{oadragoi/CS746G/al/apache—conceptual—arch. html.

[8] Zhong Xu, Rong Huang, "Performance Study of Load Balancing Algorithms in Distributed Web Server Systems", CS213 Parallel and Distributed Processing Project Report.

Archana Nigam, pursuing M.tech from IIT Roorkee in Computer Science and Engineering having specialization in Information technology. My area of interest is distributed and parallel computing, designing and analysis of decentralized algorithm.

Tejprakash Singh, pursuing M.tech form IIT Roorkee in Computer Science and Engineering having specialization in Information technology. My Area of interest is Cloud Computing, distributed and parallel computing.

Anuj Tiwari, pursuing M.Tech in Spatial Information Technology from DAVV, Indore.Working as an intern for one year atCentre for Artificial Intelligence and Robotics, DRDO Bangalore. My technical interests are in the area of Computer Networking, Distributed and parallel Computing.

Ankita Singhal, pursuing M.tech from IIT Roorkee in Computer Science and Engineering having specialization in Information technology. My area of interest is distributed and parallel computing, designing and analysis of decentralized algorithm.