Server Load – Balancing using Resources of Clustering Technique

Garima Midya
BTech. (CSE)
SRM University
Chennai

Jayant Parashar
BTech. (CSE)
SRM University
Chennai

Swetabh Suman
BTech. (CSE)
SRM University
Chennai

Rahul Mishra
BTech. (CSE)
SRM University
Chennai

ABSTRACT
Load Balancing is one of the most important issues for clustered servers. Load Balancing is done on the basis of server traffic. The proposed algorithm uses user-session history as the other constraint which is very useful for resource management over the server and easy access for the client. Collection of the data of a particular client from the last user-session, groups the applications that are likely to be opened in a specific time frame. This will provide easy access to the client and will reduce resource wastage on the server by using cluster technology. The primary clusters will be based on the bandwidth of the client's internet bandwidth, which will map to different servers. Whereas secondary clusters will be based on the similarity of application-usage in the past user-session. This approach can reduce the unnecessary searching time. Cluster management will provide better approachability towards the server also the bandwidth based load balancing will lead to the minimum bandwidth wastage.

Keywords
Clustering, Load Balancing, Scalability, Client-Server, Resource Management, Cluster Creation, Cluster allocation, Cluster Reallocation, Cluster Management and Cloud Computing

1. INTRODUCTION
The inspiration behind any Client-Server technology, arguably is its capacity to handle *n* no. of transactions in a given span of Time. Today, in the much credited dynamic environment "*Scalability"* is the concern for modern day Computer Scientist and Algorithm Pundits. You have only few ways left to deal with performance issues.

1. Either you are forced to upgrade hardware resources or
2. A load balancing[1] algorithm to find smart cost-efficient ways to fetch the user their desired resources without compromising Quality of Service.

Coming directly to Load balancing on the Server, various methods have been used in past to balance the same, yet some haven't got desired outcome. Clustering technique had been used in the near past for better fetch and efficient server load-management

Clustering Technique[2] is exploratory technique used in many areas to analyze large data sets. Given a proper notion of similarity, they find groups of similar variables/objects by partitioning the data set in "similar subsets". Typically, several metrics over which a distance measure can be defined are associated with points (named samples) in the data set. There usually are various Clustering techniques available. Most used amongst them are I) Hierarchical and II) Partitioning clustering techniques. User Session Deduction is one of most important criteria for Load-Balancing, given the technique to be used is Server-Clusters.

User-Session Identification has been non-trivial task to be performed on-the-go real time. Applications such as *telnet* or *ssh* typically generate a single TCP connection per single user-session, whereas application layer protocols such as HTTP, IMAP/SMTP and X11 usually generate multiple TCP connections per user-session. User-session characterization genuinely allows researchers to build realistic scenarios when assessing the performance of a complex network via simulation.

Earlier Problems included I) Missed user Sessions, as threshold value had to be statically stated to tell the server that it is running over capacity. Vague, as it may sound, any user-session could be missed once there is bottleneck of connections asking permission for a connection establishment. II) Possibility of a huge data lose once there is a system failure since no backup image available, only to add to the chaos, a main server did not have any backup server available to route traffic towards that server. III) Every returning User had to start-over every time he/she terminates the session, it led to huge delay in page requested and page Fetched. As every user was forced to visit the pre- specified pages and not the user-desired page. Our Approach will ensure a proper user session deduction takes place. The Algorithms will make sure to time stamp every user- session so that in case of any missing sessions the timestamp will auto-correct it's reading. Apart from these a smart Artificial Intelligence Engine will predict user behavior with respect to their usage at different point of times. This will help user to directly visit his/ her required page rather than logging in again and getting diverted once again to Homepage. Also a smart cluster allocation Engine and cluster management algorithm will work hand in hand and deliver real- time cluster allocation to users without any hassles whatsoever. Even Cluster Allocation depends upon various parameters viz. Bandwidth, Predicted user-behavior.

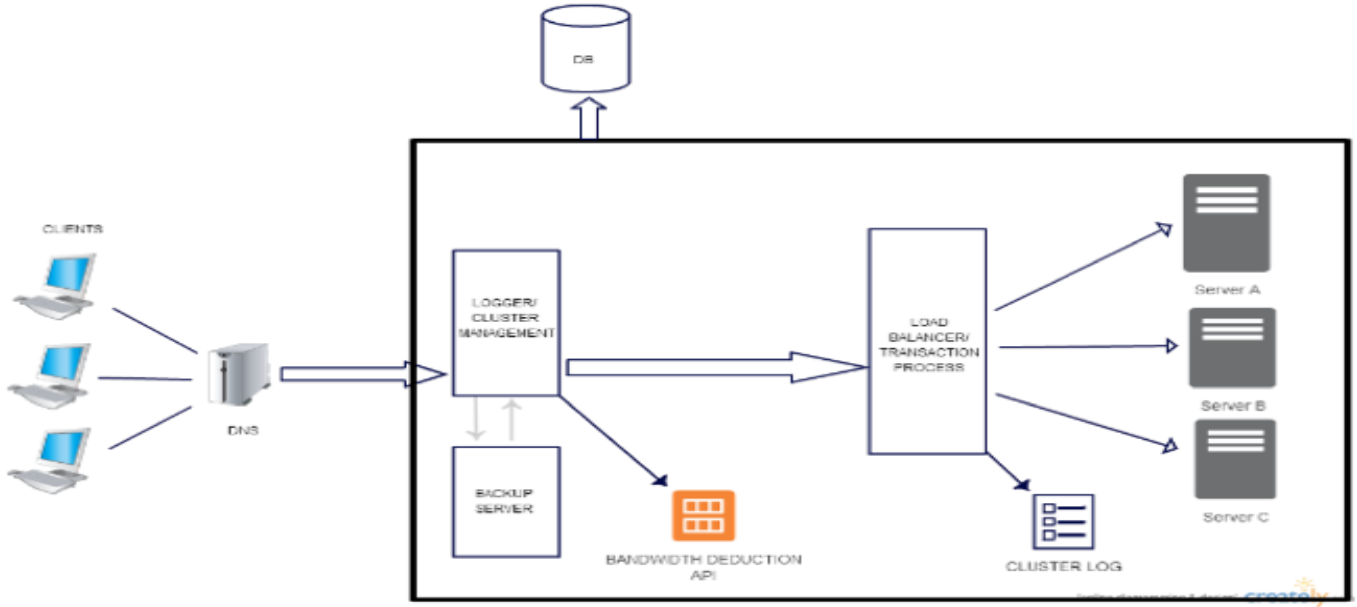

Fig 1: Architecture Diagram

2. RELATED WORKS

2.1 Load Balancing Performed On Optimal Sharing Of Network Bandwidth

Sending data from source to receiver take high transmission time. Only on batteries the sensor notes in networks. More batter power is used by time delay. Thus this reduces the life time of sensor node. In existing systems, in order to identify the receiver, sends messages to all nodes. Huge battery gets consumed due to high energy usage. This the peers participating in DHT networks are not able to maintain load balance when comes to large scale and dynamic nature. Therefore, load balancing overcomes the issue of single point failure as well as performance.

P2P environment is designed consist of system comparing cluster of peer and technology for load balancing in inter as well as intra cluster. In this light peers answers for the queries of virtual servers whereas heavy peers do registration of load value to virtual servers. This system is dynamic and automation self-evolving cluster of peer. Load balancing provides availability and scalability to server farms which appears as single servers to client. Client send the request to server, based on parameters such as availability or current server load, the load balancer responds to traffic. Best load balancing algorithm is identified through availability and scalability. Availability - time between failures. Scalability – capacity to serve simultaneously many clients.

2.2 Load Balancing Techniques On Cloud Computing

Presently enormous growth of internet and running over it has been highly experienced by humans. In order to increase computing utility and server as pay- as-you-go model, cloud computing became popular.

Cloud computing is internal technology with high features. Cloud computing is easy and flexible way to store huge volume of data without worrying about hardware requirement. Recently prominent research topics in cloud computing concentrates on load balancing. In other to have maximum throughputs and minimum time among nodes when given maximum workload through load balancing approaches. Load balancing comes in picture in cloud computing as when the number of user increases in the cloud with the decrease in resources existing which results in time delays between user and recourses provided by cloud. Situation of imbalance between nodes occur with some being overloaded as well as under loaded. Traffic needs to be balanced.

Factors affecting load balancing in cloud computing:

- Dynamic vs static behavior of algorithm.
- Geographical distribution of nodes.
- Algorithm complexity.
- Traffic analysis over different geographical location.

2.3 Clustering Techniques And Current Trends

Clustering is a technique that physically stores the logically related information together. Clustering enables to organize and access huge volume of data. In general, clustering can be characterized as the task to decompose the larger data set into smaller data subset such that the similar data are placed within one cluster whereas different clusters are segregated. Continuous Trend based clustering: In order to predict the future, it is important to identify trend. Thus it is an important problem to analyze trend of time series. In past trend analysis was done in static and streaming time series. In recent, methods have been proposed based on trend characteristics to continuously cluster a no. of streaming time series. Streaming time series are represented as vectors by PLA (Piecewise Linear Approximation) technique. Split and merge has been taken as criteria to continuously update information of clustering.

3. OUR APPROACH

Resource Management is the integral part of load balancing of various type of Network based model. Here, in this approach before the actual load balancing of the servers, it is assured that user should spend the minimum time for the selection of services from the various services present on the server. The

system predicts the application on according to user behavior which is time based and then provide shortcut selection of application. We are taking bandwidth as one of the parameter for the load balancing which is a less time taking process. Whatever time it takes we compensate that time by predicting shortcut selection of application. If bandwidth is known, then the load over the server can be maximized without any server issues as the misleading data of user and bandwidth is minimum in this case.

We are using clustering of different user on the basis of their bandwidth. It is known from the previous load balancing strategies that clustering technology is been one of the best approaches for load balancing. Here a modified single pass method is used for the clustering of users.

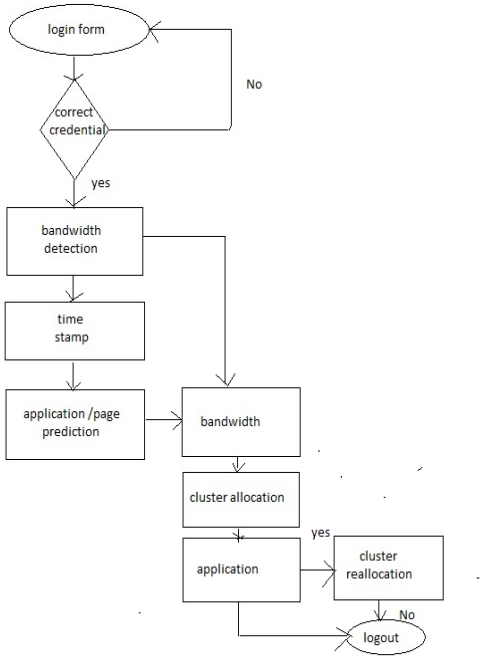

Fig.2 Data-Flow Diagram

At the login page the user is detected after that bandwidth detection is done. For bandwidth detection any bandwidth API can be used whichever is fast and accurate. After that an algorithm predicts the application that are being done for that particular time slot of the user.

3.1 Algo Login

1. Get the service request.
2. Open login page.
3. Start bandwidth detection (using API).
4. Get service (by service selection
5. Get the bandwidth.
6. Allocate cluster (cluster allocation)

Here server receive the service request and login of the user is done and procedure of bandwidth detection starts. Meanwhile the service prediction is done for selection by user.

3.2 Algo Service Selection

If (USER. View = 0)
{
Open Homepage USER. View++}
Else for (i=0;i<x;i++)
if (USER. View[i]>n)
{create a tab
}
If user.selectsapp[i]

{user.view[i]++}

Here if the user is there for the first time he/she is taken to the home page and whichever application they select. It increases the view count of that particular application in that time slot and also the overall view count. If the user already is in user DB that a standard procedure for selection of application for that particular time slot is done and whichever application is selected, it's view count is increased. The system always learns from the user behavior and process the request according to that.

3.3 Similarity Measure For Cluster

The similarity measure that is used for the clustering of user bandwidth and we use the dice coefficient for the user clustering. Dice Coefficient,

$$S_{Di,Dj} = \frac{2 \sum_{k=1}^{L} (bandwidth_{ik}\, bandwidth_{jk})}{\sum_{k-1}^{L} bandwidth_{ik}^{2} + \sum_{k-1}^{l} bandwidth_{jk}^{2}}$$

3.4 Algorithm For Cluster Creation

Single pass [3] constraint based method:
1.Assign the first user U_1 for the representative for C_1.
2.For U_i Calculate the similarity S (bandwidth) with the representative for each existing cluster.
3.If S_{max} is greater than constraint value S_C, add user to that cluster and recalculate the cluster representative to initiate a new cluster.
4.For user U_i which remains to be clustered, return to step 2.
5.For each cluster S assign a maximum capacity S_{max} so that cluster size not exceeds.

This algorithm is used for the creation of the cluster; it requires the user set be process only one. In this algorithm first user becomes the seed and its bandwidth is the similarity measure based on that user are categories and allocated a new cluster. After any new bandwidth size user is sited a new cluster is created. The size of each cluster is fixed so that not a very big cluster is created which can cause redundancy of user and misleading of server's load.

3.5 Algorithm For Cluster Allocation

1.At first the user is allocated a cluster with same bandwidth and less number of user.

2.Adaptation of user DB is done.

3.Cluster log is update so that at reallocation time no mishap occurs.

3.6 CLUSTER REALLOCATION

1.At first the cluster capacity is checked in the log.

2.If number of user in the cluster is more than the threshold/constraint that change the next cluster which has the less number of user than its capacity. When a cluster which is

having less load is found then reallocate all the extra users in the cluster log with more load to the cluster having less load.

3.Now check for the user session in user DB. If user is active for more than some specified time it is reallocated to a higher degree cluster which has less bandwidth allocation and if the user is in active his/her is auto logged out after some specified time.

1. Check cluster log for cluster capcity/load –

If ClusterLoad [x]>threshold[x]

Loop

{check for

ClusterLoad [x+1]<threshold[x+1]

If cluster[i] found

End loop}

Uthreshold[x]+Uthreshold[x+1]+ . . . . . +

Uthreshold[x+n] !cluster[i]

2. Check for user session –

If Ui session>/n min in cluster[x]

Allocate Ui to lower degree cluster[y]

3. Check for data usage -

If Ui data is 0 kb in last n min

Auto logout

Terminate

3.7 CLUSTER MANAGEMENT

- Allocates cluster based on the type of user and their usage.
- Stores log of Cluster
- Administers clusters' branching

This is the module where the actual load balancing of server with reference to cluster is done here. Here according to the usage time and data usage, users are reallocated to other lower level cluster and in some cases the connections are terminated

in this module. Transection processor/load balancer checks the cluster log for the load of servers and clusters and act accordingly.

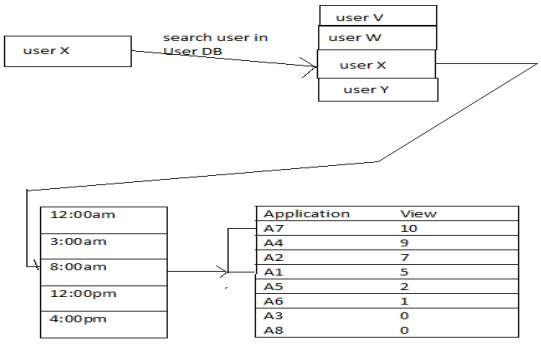

Fig.3 Work-Flow Diagram for client management

4. SUMMARY

The load balancing architecture described in this paper consists of TCP based user session identification and implementing clusters dynamically. Also, an Artificial Intelligence Algorithm has been proposed to bypass the homepage or landing page and directly send the user to the pages analyzed and ranked by the Engine based on the user's browsing behaviors. While the other clustering based scenarios included *Aprori* threshold to defined independently, theoretically our approach doesn't require any such threshold definition which can be disastrous in current scenario where users can't be guessed easily and for condition like this it is needed for a load-balancing algorithm to be more dynamic and can change based on the user behavior.

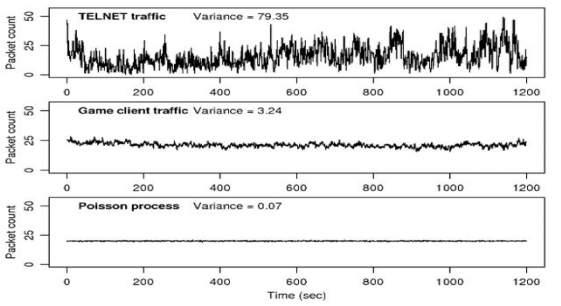

Fig.4 Traffic Comparison

As depicted and theoretically established, a login server contains a user database which is formally used to store user logon credentials especially the basic account details of the user. Moreover, this section also contains an Artificial Intelligence algorithm which can be used for user behavior analysis on various databases or servers also, subject to the user behavior analysis by the algorithm, we can easily predict the efficiency of the system or infrastructure that is being provided for the same and in future course it can be increased or decreased subject to *scalability* issues. A special provision is made to create and allocate clusters to different users based on their bandwidth, so that resources can be made available to more number of users. For maintaining clusters, a cluster management tool will be working to facilitate dynamic cluster creation and amalgamation of different clusters.

a) *Client-side information* for the purpose of bandwidth detection a client-side tool may be used on the sole discretion of the user which will be helpful in predicting bandwidth precisely which in turns results in to smoother flow of data minimizing data-loss. A cache may be maintained for verifying transactions and user – sessions from the clients' web-browser.

b) *Tier I Algorithms* for this tier, specifically numerous task has to be done, which roughly includes Authentication and revoking user access permissions, determining the bandwidth of the user and predict any user page requirements based on the previously faced data. Usually one Algorithm will be authenticating the user credentials, once the credentials are verified the user reaches to the next algorithm which based on previous logs suggest user to choose from frequently visited pages with a timestamp or give them an option to go to pages of their choice manually. While these process are being run another algorithm starts sending hops to the users' browser and calculates the bandwidth based on the interval between hops being sent and the same being received.

c) *Tier II* in this particular tier user is provided with a

smooth functioning cluster which enables it to send and receive data with negligible latency after the user gets verification clearance from the *Tier I* architectural algorithms. Once the verification is done for a particular user they are then then sent to different clusters. Attention is paid to the cluster creation algorithm which is solely responsible for cluster definition initially. Once the cluster is created it has to be allocated based on some pre-determined parameters which presumably is the bandwidth in our case. Based on the users' bandwidth a client is allocated servers or say clusters in decreasing order of their bandwidth i.e. more the bandwidth faster is the server allotted and vice versa. After the users are put into their requisite servers it is necessary to maintain those clusters in order to provide maximum throughput [4] and also to maximize the resource utilization. An algorithm is defined to keep analyzing those clusters based on the user requirement i.e. if the users are increasing the cluster manager may tell cluster creation algorithm to initiate new cluster creation, however if the user requirements are decreasing and the number of users are also decreasing gradually, then surely a probable wastage of resource may take place. To get rid or avoid such situation the cluster management algorithm amalgamates different clusters into one single cluster and hence that bigger cluster might be divided again to facilitate more users to utilize greater amount of resources.

A sample graph for the bandwidth throughput can be drawn, considering bandwidth to be the root parameter. Without considering any external hindrances, the output can be measured categorically and more precisely. Also the bandwidth curve strongly explains the detected fluctuations in the in recorded times intervals. It may be noted the bandwidth increases and touches its peak during 9-10 pm of any particular month commonly. The interval considered in all months are between 5pm to 10 pm and has been subdivided on hourly basis to precisely establish the readings. The read bandwidth could be superseded by bandwidth throughput higher than normal.

$$\left|\sum_{v\in V_i^{\mathcal{A}}} L_v - \frac{\sum_{v\in V} L_v}{\sum_{k\in N} C_k^{max}} \times C_i^{max}\right| = \left|C_i^{max}\left(\frac{\sum_{v\in V_i^{\mathcal{A}}} L_v}{C_i^{max}} - \mu\right)\right| = |C_i^{max}(LBF_i - \mu)|.$$

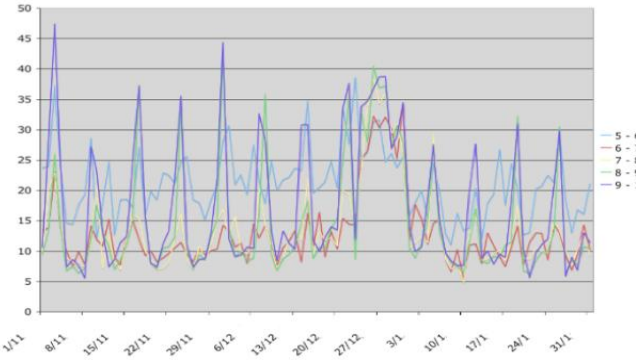

Fig.5 Average Bandwidth (For a given Period)

Once the graph is plotted it is considered to be defining the bandwidth throughput and hence peak bandwidth period [5] can be defined easily. With analysis of recorded data user can manage their timeframe to make use of available resources based on analysis.

5. CONCLUSION

This paper presents a cluster based dynamic [6] load balancing server cluster architecture to solve the problems that traditional dedicated load balancer faced. Our Architecture consist of a cluster allocation algorithm which delivers creation, allocation and cluster-management in a dynamic strategy leading to efficient load balancing. Also, the architecture contains an Artificial Intelligence Engine which can help predict user-behavior to help with directing the user to their intended page, saving time and space. Future papers will deal with better handling in real time scenarios and a smart Artificial Algorithm. So, that we can achieve more efficient load balancing.

6. REFERENCES

[1] Design and implementation of Server-Cluster dynamic load balancing based on Open-Flow. Zhihao Shang, Wenbo Chen, Qiang Ma, Bin WU Lanzhou University Communication Network Center Lanzhou, China

[2] Clustering data streams: Theory and practice. Guha, S.; Dept. of Comput. Sci., Pennsylvania Univ., Philadelphia, PA, USA; Meyerson, A.; Mishra, N.; Motwani, R.

[3] An improved Single-Pass clustering algorithm internet-oriented network topic detection. Yi Xiaolin; Coll. of Comput. Sci., Beijing Univ. of Technol., Beijing, China; Zhao Xiao; Ke Nan; Zhao Fengchao

[4] Continuous One-Way Detection of Available Bandwidth Changes for Video Streaming Over Best- Effort Networks. Javadtalab, A.; Distrib. & Collaborative Virtual Environ. Res. Lab., Univ. of Ottawa, Ottawa, ON, Canada; Semsarzadeh, M.; Khanchi, A.; Shirmohammadi, S.

[5] Determining the time window threshold to identify user sessions of stakeholders of a commercial bank portal. Jozef Kapusta1, Michal Munk1, Peter Svec1* and Anna Pilkova21Constantine the Philosopher University in Nitra, Nitra, Slovakia 2Commenius University in Bratislava, Bratislava, Slovakia.

[6] Dynamic Clustering in Object-Oriented Databases: An Advocacy for Simplicity J. Darmont1, C. Fromantin2, S. Régnier2+3, L. Gruenwald3, M. Schneider2.