Adaptable Load Balancing

Sung Kim, Youngsu Son, Gaeyoung Lee
Home Solution Group
Samsung Electronics

ABSTRACT
The proposed load balancing system includes multiple counts of servers for processing network traffics and a client for transmitting connection request signals to those network traffic processing servers.
First, the client sends request signals to all the servers. After receiving those request signals, based on the server resource availability, they calculate the wait time before it sends a response signal back to the client. The client makes the connection to the first server that transmits the response signal and ignores all servers response. Delay time is calculated from system resource availability and predefined maximum tolerance response time for the service. This system combines the use of both local load balancing and global load balancing. By controlling the delay time, it decides which one to put more focus than the other one.
By using this system, the client request can be efficiently distributed. This system is cost efficient because it does not need load balancer and it has flexible architecture.
.

Categories and Subject Descriptors
D.3.3 [**Programming Languages**]: Language Contructs and Features – *patterns*.

D.2.11 [**Software Engineering**]: Software Architectures – *patterns*.

General Terms
Algorithms, Design

Keywords
Load Balancing, RTT Control, Delay time, Maximum response time, Flexible architecture

1. INTRODUCTION
As requests from the client increases, the servers have to process more transactions.
As the transaction increases, the number of servers in the system has to increase in order to handle those transactions. To efficiently control those increased servers, load balancing method is used. Normally, Load Balancer is needed in order to perform load balancing. However, in this method, it does not require one.

Because Load Balancer is the center gateway of all the transaction, as the transaction count increases, it gets overloaded. Complex load balancing algorithm also adds additional stress to the Load Balancer. In here, for some reason, if Load Balancer breaks down, it will cause the entire service to stop. However, our proposed system does not get this issue since it does not require a load balancer [1].

In the proposed system, the combination of the local load balancing and the global load balancing system is used to adjust the system dynamically depending on the situation/environment.

2. BACKGROUND
Local Load Balancing - Local load Balancing System distributes the clients request to multiple server. With distribution of the load, availability of systems is enhanced. This cause the smart use of servers resource, thus it produce efficient system. Load balancing servers are placed in same location and set up. When the client makes requests to the servers, the client will connect to the server with the lowest load [2].

Global Load Balancing – Global load Balancing System are used when the servers are located in different networks. It is used to located the server with fastest network response. So this considers efficiency of network. The global load balancing system uses redirection method [3]. Load balancing servers are deployed to other place on network. When client request to server, client will connect server that is fastest response.

3. EXAMPLE
For example, let's assume that we are developing an office automation system for buildings located closely together in a downtown. There are various types of devices in the system and they are connected to a wired or wireless network. In addition, it is required to keep software in each device up-to-date. The server will provide the latest software via client-server model. In Polling method, each client requests data from the server without considering any other clients. So, it would cause server overload. On the other hand, in Push method, a client that is turned off or malfunctions at the time of upgrade wouldn't be updated.

Let's take a specific example with the figure below. The server has to update various types(green, yellow, red) of devices that are placed in different location. Some office would have all types of devices but some would not. In this situation, it is possible for the server to manage devices in a way that groups them by device type or location.

4. CONTEXT
Environments like websites which services various type of contents, such as video feeds, html/image rendering, puts stress to both network and server resources. These require 2 types of load balancing. The global load balancing focuses on network related

issues. The local load balancing focuses on server resource related issues [4].

In Local Load Balancing, a load balancer is located at the server side and distributes clients' requests only based on the resource availability of the servers. It does not take into consideration of the network status or service type (whether it is video streaming service or html data transmission service)

Global load balancing balances the load by taking into consideration of requests Round Trip Time(RTT).

The local load balancing is able to balance the server overload, but it is difficult to consider external conditions, e.g. network bandwidth, systematic vaccine update, and so.

The global load balancing is apt for providing clients with fast response.

5. PROBLEM

Due to the workload of server and the status of network changes constantly, these changes should be considered and reflected in real time. So to balance the load, we should consider multiple factors, e.g. Round Trip, Load of Sever, Weight of Load, load item count, maximum response time, Delay time, RTT

6. FORECES

The following items should be regarded as forces:

- Providing load balancing without the load balancer. Thus prevents the service stop when the load balancer breaks down. Being able to adjust which one to focus more, between local load balancing and global load balancing, depending on the situation.
- Ignoring any server that does not respond, from the load balancing list.

7. SOLUTION

7.1 Structure

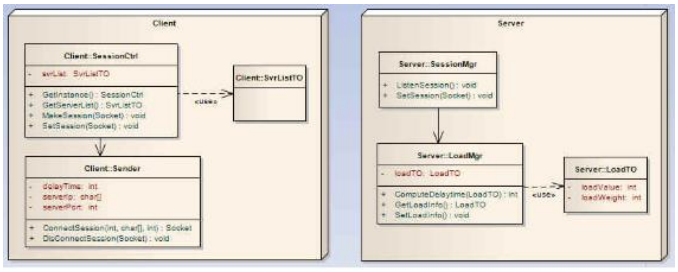

Figure 1. Half-Push/Half-Polling Structure

multiple count of servers for processing network traffics and a client for transmitting connection request signals to those network traffic processing servers. The proposed method includes multiple counts of servers for processing network traffics and a client for transmitting connection request signals to those network traffic processing servers. The proposed load balancing method structure is shown in Figure 1.

SessionCtrl Component controls the session between server and client. Sender Component performs connection and disconnection.

- GetInstance : Start load balancing.
- GetServerList : Get loading balancing target server list..
- MakeSession : Make session to server.
- SetSession : Set session that connects to a server transmitting the first received response signal.
- ConnectSession : Sends request signal to server.
- DisconnectSession : Disconnects all others except to the first session.

SessionMgr Component transmits connection request signals and registers client session. LoadMgr Component computes delay time according to the algorithm.

- ListenSession : Receives client request signal and responses to the client after delay time
- Set(Register)Session : Registers a client session.
- ComputeDelayTime : Compute the delay time from the server load.
- GetLoadInfo : Get values of server load items.
- SeLoadInfo : Set weight of load items.

7.2 Algorithm

An algorithm exists that controls the server response time. Clients transmit connection request signals to the servers. The servers receive clients connection request signals and compute the delay time (DT) from the server load. During the computed Delay Time (DT), the servers will delay to send response signal to the client. In result, if the server load is big, the response time will be delayed that much.

Variables for load balancing algorithm are shown in table 1.

Variable	Value
RTT	Round Trip Time
LT	Load Type- - CPU, Memory … of server.
LW	Weight of load
LC	Count of load item
MT	Maximum response time
DT	Delay time
DRTT	RTT considered load.

Table 1. Variables for load balancing algorithm

RTT is the required time for network communication to travel from the client to the server and back. LT is the server load type – CPU usage, memory usage, etc. LW is the priorities amongst the load types. LC is count of LT. MT is the maximum response time that takes client to get the response back from the server after sending request signal.Load balancing algorithm is expressed in following equations.

DT (Delay Time)

$$DT = \frac{\sum_{i=1}^{LC} LTi \times LWi}{LC} \times MT$$
$$(i = 1,2,3..., LC)$$

DRTT (RTT considered Server Load)

$$DRTT = RTT + DT$$

(Round Trip Time(RTT) gets added to the delay as well)

The type of data being transmitted/processed affects the load of server resource, Load Type(LT).
Depending on the data being video steaming or html data, the server resources would have different load, one resource (such as CPU) getting priority from other resources. An algorithm can be used to apply different weight to these resources.

Also server load count (LC) can be increased or decreased depending on the type of data.. For example, Load Types are memory usage, CPU usage, disk usage, etc. If server is influenced by CPU usage the most, The CPU usage will have the biggest weight. As result, the increase of CPU usage will influence DT, in turn, DT will be increased. In other words, DT is sensitive to the CPU usage change.

MT decides how big the delay time can get. If MT is increases, DRTT will be influenced that much If MT decreases, DRTT will be influenced that much less. The other hand, if MT decreases, the influence of RTT will be bigger than DT because the weight of RTT is more than weight of server load. In result, the balance of focus between the local load balancing and the global load balancing is controlled by MT.

7.3 Dynamics

Sequence of load balancing method is shown in Figure 2.

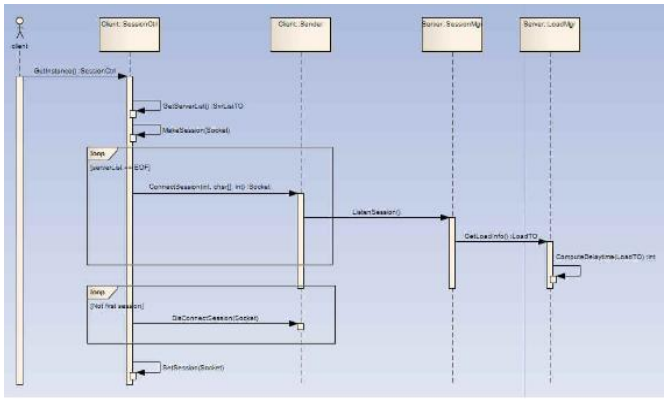

Figure 2. Sequence of load balancing method

Client has server list. It also sends the request signal to server. First, the client sends request signals to all the servers. After receiving those request signals, based on the server resource availability, they calculate the wait time before it sends a response signal back to the client. The client makes the connection to the first server that transmits the response signal and ignores all servers response.

When client sends request signal and receives response signal from server, DT is influenced by RTT automatically. Thus, RTT is important. Therefore client is connected to server that has the best one when considering the low load or the fast RTT.

If a server breaks down, the server will not be able to respond. Those broken-down servers will be ignored from load balancing.

8. Experiments

In our experiment environment, the system was composed of two servers and one client. Network is LAN environment. The experiment compared the round robin method with the proposed method. Different request types affect server resource usage differently as shown in below table.

Request Type	CPU Usage	Memory Usage
HTTP Transaction	1%	1%
DB Transaction (A Type)	3%	1%
DB Transaction (B Type)	1%	3%

Table 2. Assumption values of load

As shown in the table 2, A Type of DB Transaction influence more on CPU usage than memory usage. B Type of DB transaction influences more on memory usage than CPU usage.

#Experiment 1 – Were tested with Server1 and Sever2 having same Value of variables (CPU, Memory, RTT).

Name of variable	Value of variable
RTT	Server1 : 300ms, Server 2 : 300ms
LW	CPU : 50, Memory : 50
LC	2 (CPU, Memory)
MT	3000ms

Table 3. Value of variable for LW experiment

Result of the experiment is shown in Figure 3.

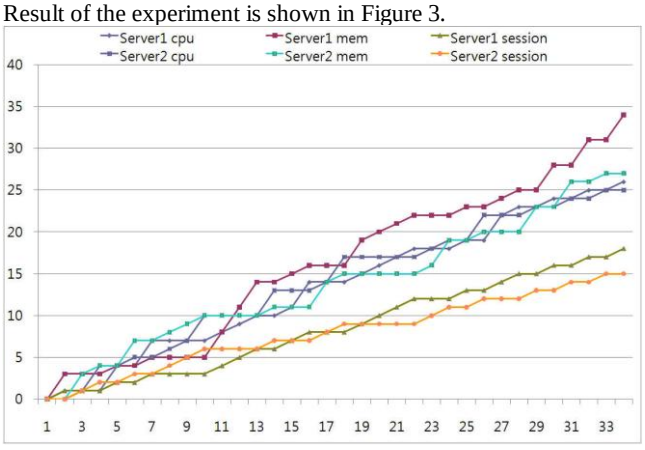

Figure 3. result of LW Experiment

In Result, the number of sessions in two servers came out to be different. However resource usage between server 1 and server2 is similar. As Figure3 indicates, the experiment #1, caused the

server's load balancing to act like the local load balancing.

Experiment 2 – Servers were setup with different network environments.

Name of variable	Value of variable
RTT	Server1 : 100ms, Server 2 : 300ms
LW	CPU : 50, Memory : 50
LC	2 (CPU, Memory)
MT	3000ms

Table 4. Value of variable for RTT experiment

Result of the experiment is the same as Figure 4.

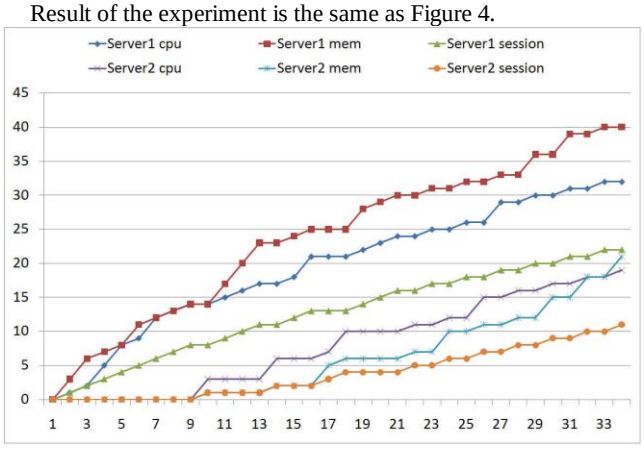

Figure 4. Result of RTT experiment

As Equation #1 suggest, the experiment #2 with low MT value caused DT to be small. Since DRTT is the sum of DT and RTT, low DT insinuates DT having less influence to the DRTT than the RTT value influencing DRTT. As Figure 4 indicates, the experiment #2, caused the server experiment #2, cause to act like the global load balancing.

Experiment 3 – Added more value on CPU than memory.

Name of variable	Value of variable
LW	CPU : 70, Memory : 30
LC	2 (CPU, Memory)
MT	3000ms

Table 5. Value of variable for LW experiment

Result of the experiment is shown in Figure 5.

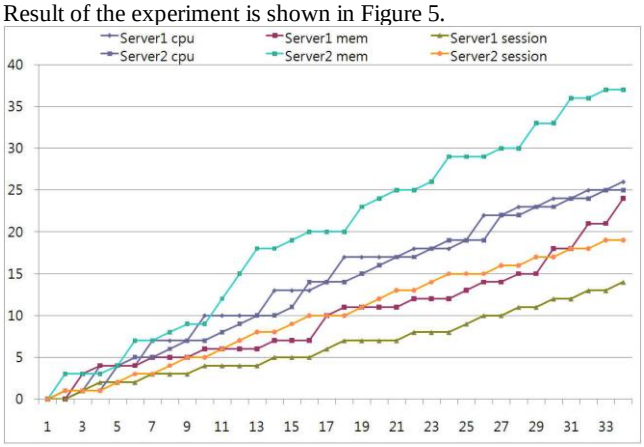

Figure 5. Result of LW experiment

As seen in Figure #5, the experiment result shows the servers having different session counts and memory usage from each other. However, CPU usage seems to be similar on those servers.

#Experiment 4 – Increased the values of MT.

Name of variable	Value of variable
RTT	Server1 : 100ms, Server 2 : 300ms
LW	CPU : 50, Memory : 50
LC	2 (CPU, Memory)
MT	10000ms

Table 7. Value of variable for RTT and MT experiment

Result of the experiment is shown in Figure 7.

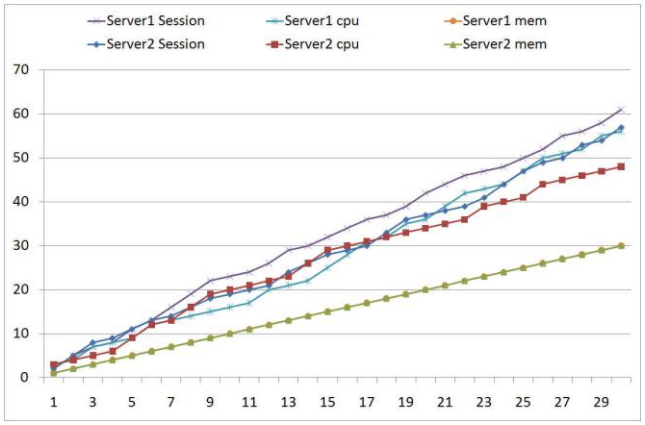

Figure 6. Result of RTT and MT experiment

As Equation #1 suggest, the experiment #4 with high MT value caused DT to be high. Since DRTT is the sum of DT and RTT, high DT value insinuates DT having bigger influence to the DRTT than the RTT value influencing DRTT.
As Figure 6 indicates, the experiment #4, caused the server's load balancing to act like the local load balancing due to having less influencing RTT value to DRTT.

9. Side Effects

As the number of the servers increases, the difference in the time that each servers receive the request signal sent by the client increase. This time difference affects the load balancing.

10. RESULTING CONTEXT

The proposed method can perform load balancing without the load balancer. Thus, it automatically ignores any server that does not respond.
The Value of variable gets set depending on the importance of each resource in server.
The load balancing is performed by considering these resources with high values. From observing the resource usage and RTT, the proposed method uses the load balancing according to the environment.

ACKNOWLEDGMENTS

We thank our shepherd Norihiro Yosida for his valuable comments that contributed to improve this paper. The EVA (Pattern Evangelist Group), James Chang provided useful improvements and corrections.

REFERENCES

[1] Valeria Cardellini, Michele Colajanni, Philip S. YU, "Dynamic Load Balancing On Web-Server System", IEEE Internet Computing, June 1999.

[2] Network Load Balancing Technical Overview White Paper

[3] Valeria Cardellini and Michele Colajanni and Philip S. Yu, "Redirection Algorithms for Load Sharing in Distributed Web-server Systems", IEEE 1999.

[4] CDN Conference, NetTrend 2001, 2001.

[5] Msurice Castro, Michael Dwyer and Michael Rumsewicz, "Load balancing and control for distributed World Wide Web servers", IEEE, August 1999

[6] Rajkumar Byyya, "high performance cluster computing", Prentice Hall PTR, 1999.

[7] R.j. Schemers, "lbmnamed: A Load Balancing Name Server in perl", Proc, 9th Systems Administration Conf, Uscnix Assoc, Berkely, Calif, Sept 1995.

[8] A. Bestavros et al, "Distributed Packet Rewriting and its Application to Scalable Web Server Architectures", IEEE Computer Soc. Press, Los Alamitors, Calif., 1998

[9] Carla Sadtler, John Chambers, Ariane Schuldhaus, "Load Balancing for eNetwork Communications Server" International Technical Support Organization