International Journal of Grid Distribution Computing
Vol. 8, No.4, (2015), pp. 83-90
http://dx.doi.org/10.14257/ijgdc.2015.8.4.08

Optimization of Edge Server Selection Technique using Local Server and System Manager in content delivery network

Debabrata Sarddar[1] and Enakshmi Nandi[2]

[1]*Assistant Professor, Department of Computer Science & Engineering, University of Kalyani, Kalyani, India*
[2]*Ph.D. Student, Department of Computer Science & Engineering, University of Kalyani, Kalyani, India*
[1]*dsarddar1@gmail.com*, [2]*pamelaroychowdhurikalyani@gmail.com*

Abstract

Cloud Computing is a distributing computing technology which is used as pay per use basis. Now a day's Cloud Computing is the most reputed topic due to its ability to offer guaranteed quality of service atmosphere, dynamic IT infrastructures, and conFigureurable software services. But many users could not satisfy on cloud services completely due to their uncovering security purpose for handling large numbers of data. Even the network becomes uncontrollable, when large numbers of user's request to the server create network congestion and data losses vigorously. Content Delivery Network OR CDN is an eminent solution of this problem. Our objective is to create local and global server and connect the global server to system manager, which is worked over Content Delivery Network to deliver and direct the user request to the nearest global edge server except local server and establish the connection between them and transfer the respective content. For optimization of edge selection process and reduce the load over content delivery network we approach local server concept in this paper.

***Keywords*:** CDN, Euclidean Distance, Dijsktra Algorithm

1. Introduction

A content distribution network is a large, geographically distributed network of specialized servers that deliver the web content and web pages to the huge amount of users in different geographic locations. CDN provides fast delivery of content of websites with high traffic. If the CDN server is closer to users then the content deliver to them with earlier. CDN not only provides high availability and good performance but also provides high security from DOS attacks through their large distributed server infrastructure to absorb the attack traffic. Here the replica of end user's content are distributed among different edge servers using content replica placement algorithm. In this paper we want to optimize the searching technique of edge server using local and global server and system manager concept over content delivery network, so that the searching process of edge servers of content delivery network will be very fast.

1.1. Related Works

S. Saroiu, R. J. Dunn, H. M. Levy and so on are the important name in the field of CDN. They examined on the basis of four content delivery system such as web traffic, Content Delivery Network of the Akamai and Kazza, Gnutella peer-to-peer file sharing traffic [1].Recently Rajkumar Buyya, Al-Mukaddim works on detail taxonomy of Content Delivery Network with respect to four impact factors such as request-routing procedure, content replication mechanisms, cache management ,load balancing methodology [2].

ISSN: 2005-4262 IJGDC

Phooi Yee Lau,Sungkwon Park research on the basis of relationship between the placement strategy and throughput among different edge servers [3].

2. Proposed Work

In this paper our aim is to create local server and host server and connected host server to system manager which is connect to different edge servers via content delivery network. We want to optimize the edge server selection process and reduce the load over content delivery network. So we approach local server concept at first of request response process. At first that the earth is divided into six regions: North and South America , Europe ,Asia, Africa, Australia as shown in the Figureure below(R0,R1,R2,R3,R4,R5,R6) . At first users send request for a definite content to their local server, if they do not find the proper server then local server terminate the request to host server. Such as an Australian's users send the request to local server if they can't find the required content then local server terminate the request to host server. Host server connected with system manager, where an indexed table found. The replica of all contents lies in local server and this indexed table. The replica of contents lies in this indexed table according to shortest distance and high throughput basis. This table holds the relevant information of the connected edge servers and the associated databases of the web-sites for which the request is to be handled. To optimize the load over CDN, local server concept has approached so that the request are generated to local server, if the content does not match with the local edge server's content then automatically request will be transfer to CDN through system manager. In system manager contains the indexed table, where contents contain in the edge server according to shortest distance edge server basis. This shortest distance edge server works on Dijsktra algorithm basis.

2.1. Find the Euclidean Distance for Finding the Distance of Different Edge Server from Client

The Euclidean distance between p and q is the length of the line segment connecting then (pq) is

$$d(pq) = \sqrt{\sum_{i=0}^{n}(p_i - q_i)^2}$$

2.2. After Finding the Distance from Client to Different Edge Server We Use Dijsktra Algorithm for Finding Shortest Distance Edge Server Is As Follows

Step1. Create a set *sptSet* (shortest path tree set) that keeps track of edge servers included in shortest path edge server, i.e., whose minimum distance from user is calculated and finalized. Initially, this set is empty.

Step2. Assign a distance value to all in the input edge server. Initialize all distance values as INFINITE. Assign distance value as 0 for the user so that it is picked first.

Step3. While *sptSet* doesn't include all edge server.

….a) Pick a edge server u which is not there in *sptSet* and has minimum distance value.

....b) Include u to *sptSet*.

…c) Update distances value of all adjacent vertices of u. To update the distance values, iterate through all adjacent edge server. For every adjacent edge server v, if sum of distance value of u (from user) and weight of edge u-v, is less than the distance value of v, then update the distance value of v[4].

2.3. Most Frequently Used Algorithm

Edge server also follows the table and update this table accordingly, that is most recently used edge server table. MFU or Most Frequently Used algorithm based on the argument that the page with smallest count was probably just brought in and has yet to be

used [5]. This table also maintains another criterion that is, most frequently used edge servers have arranged with their scheduling values accordingly. The respective algorithm for most recently used is as follows

Step1. Consider the edge servers string value and their respective buffer size.

Step2. For MFUk =0 To Val(MFUbn) - 1 ‘match cIf MFUpage(MFUi) = MFUb(MFUk) Then MFUr(MFUk) = Val(MFUr(MFUk)) + 1;

Step3. MFUflag = 1
Exit For ‘exit loop MFUK
End

Step4. If Next MFUk; If MFUflag = 0
Then If MFUbn < MFUbf Then'MsgBox "flag0"

Step5. MFUb(MFUbn) = MFUpage(MFUi);
MFUr(MFUbn) = Val(MFUr(MFUbn)) + 1;
Step6. MFUbn = Val(MFUbn) + 1;'MsgBox "bn: - " & MFUbn
Else 'MsgBox "2"
Step7. MFUl = 0'MsgBox MFUnFor MFUk = 0 To Val(MFUbf) -1
Step9. If MFUr(MFUk) > MFUl
Then MFUl = MFUr(MFUk) ;
MFUlc = MFUk ;
Step10. End
If Next MFUk
MFUb(MFUlc) = MFUpage(MFUi)
Step11. MFUc = 0
For MFUk = 0 To MFUi
If MFUpage(MFUi) = MFUpage(MFUk)
Then MFUc = Val(MFUc) + 1MFUr(MFUlc) = MFUc ;

Step11. End
If Next MFUk
End If
End If

2.4. System Manager

System manager maintains the following track record

i. System manager connected to all edge servers.

ii. System manager maintains an indexed table based on shortest distance edge server from user which contains replica of contents of web service.

iii. This indexed table maintains all records of parameters and arranged according to shortest distance edge server and high throughput basis.

iv. System manager maintains another table based on most frequently used edge server which contains information about edge servers’ parameter like workload, network throughput, average waiting time, storage capacity, channel capacity and so on. According to these features all the most frequently used edge servers in this second table are arranged and updated periodically after performing the request response operation of delivery content from respective edge server.

v. System manager provides secured file sharing services between edge server and system manager.

vi. When the request come from host server to system manager then it checks to its indexed list and most frequently used edge server list that which server has that content that match the request content. If match found then that respective IP address of that edge server send to user and the corresponding edge server then directly connected to user for delivering the content. If match not found then it checks the next server from table. If in case it is found that the requested content will contain in two nearest edge servers in

system manger then system manager select the edge server which gives better performance according to checking the scheduling criteria from the table.

2.5. Calculating Throughput from Sender to Receiver in the Network

Throughput measures the number of jobs completed per second i.e. one measure of work is the number of processes that are completed per time unit is called throughput. For long process throughput measures the rate of one process per hour and for short process this will be process per second [6].

Now we consider that sender's node and destination node, k+1 node all together with k-1intermediary nodes. There are k links between each pair of edge server nodes. For transmitting one bit it takes τ seconds and the bandwidth of each link is W bits/sec. Before a single message can be sent the sender sends a connection request message, the receiver replies with an accept/deny message to the sender, and then the real message transfer begins. Now the sender's message a connection disconnect message is sent. These 3 messages are H bits long and occur so that the destination must receive the entire message before the next one is sent out. For 1 message that is *H* bits long the time taken to transmit the entire message should be *H* bits * τ seconds. So, to transmit 3 messages across k nodes I have a total time of $3k(H*\tau)$seconds needed.

- In steady state after handshaking, a message of maximum segment size (*MSS*) can be sent to the destination by time
- Therefore, Propagation delay + Transmission delay

$= k\tau + k\frac{MSS}{W}$

The return acknowledgement has a delay of

- Therefore, Propagation delay + Transmission delay

$= k\tau + k\frac{H}{W}$

So, the sum of these two gives the whole round trip time. And the actual message length sent without header should be *MSS−H*, which gives throughput

- Throughput = $\frac{MSS-H}{2K\tau+K\frac{MSS+H}{w}}$

2.6. Algorithm for Selecting Edge Server for a Particular CDN

1. Let U= {u_1,u_2, u_3,u_4…….u_n} be the set of users from six geographic locations.
2. And E= {e1, e2, e3…en} be the set of edge servers and System Manager situated centrally and connected to both host server and CDN edge servers.
3. When a client from set U sends a request to a specific region to local server, Then local server check the requested content to its contain servers, if match found then stop process. Otherwise local server terminates the request to host server.
4. When a request arrives at the original web server, it first identifies which type of content is to be served; it sends the request to the system manager system of the registered CDN.
5. Now system manager first identifies the client's location i.e. longitude, latitude and IP address and then it looks up indexed table to find that which nearest edge sever have the replica of requested content and then check the next most recently used edge server table to get the proper edge server with its scheduling criteria accordingly.
    a. If it finds the IP address and associated geo-coordinates with edge parametric in these tables then it immediately directed the associated edge server with the corresponding delivery content to the respective client.

b. Else go to step 6;

6. Now the request comes from a new client –
   a. Calculate the shortest distance between the user and edge servers of the CDN using Dijsktra algorithm method, and store the shortest distance edge server with its IP address to indexed table accordingly.
7. Now choose first shortest edge server and check it to most frequently edge server list for its respective parametric values.
   a. If match found then For each temporary edge server ,i.e. temp_server

i. If(getNetworkthroughput(edge_server)<getNetworkThroughput(temp_serve)) then selected_edge_server=edge_server ;

ii. Else selected_edge_server=temp_server;

End for

8. After the selection of the edge server, the parameters of the selected_edge_server and the client is written to the most frequently used list.
9. Now the original request of the client is forwarded to the selected_edge_server and then selected_edge_server transmit the definite contents immediately.
10. Now the connection between the selected_edge_server and the client is established.
11. If match not found go to first indexed table and search for requested content to all edge server.
12. If user again sends a request for either some dynamic content or for static content then
    a. If request is for dynamic content then:
       i. selected_edge_server acts as the proxy and fetch the dynamic content from the host web server and sends it to the client.
    b. If request is for content then :
       i. If content is available then transmit the static content immediately.
       ii. Else if it is not available in selected_edge_server then it will search for content by sending a search query to its siblings and if found redirects it to the client.
       iii. Else if it is not available within the CDN, then a brought back request is sent to the host server and the static content is fetched, simultaneously content is distributed over the CDN with the help of RMS.
13. For each request repeat step 5 to step 11
14. End

For adding and searching content in the most frequently used list we can use HASHING technique also.

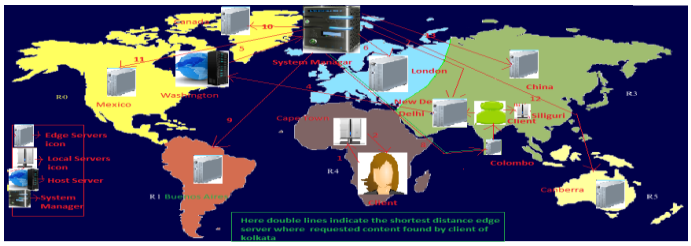

Figure 1. An Example of CDN in Six Different Regions of World Map with Host Server, System Manager, Edge Server and Users with Their Request for Content

3. Practical Example with Result Analysis

- As we see in the above Figure the respective client stay in R5 region, i.e. Africa. She sends request to her local server in Africa for respective content. She found the respective content to her local edge server. She found that content then no need for CDN.
- The client from R4 region i.e. Asia (Kolkata) ,send request for mail service for sending 1400 bytes data using the mail server to his local server but no one match with requested content ,then local server terminate the request to host server at Washington. Host server sends it to system manager [7].
- System manager then find shortest distance edge server from the indexed table by using Dijsktra algorithm and finding the definite mail service from two respective tables with highly throughput value and minimal response time from corresponding list and directed the particular edge server in Content Delivery Network with the Kolkata's client.
- CDN provider stores the details information of all edge servers through System Manager. As shown in the table1. System Manager calculates the mean waiting time using multi-level queuing formula and Throughput of every edge servers from previous formula.

Table 1. Consider the Name of Places of Edge Server of Particular CDN

Name of the region	Name of the place of edge server
1. Asia	Colombo
	New Delhi
2. Europe	London
3. North America	Mexico
	Washington (Host Server)
4. South America	Buenos Aires
5. Africa	Cape Town
6. Australia	Canberra

Table 2. Mean Waiting Time of Each Edge Servers Using Multilevel Queue

Host Server	**Arrival rate**	**Service rate**	**Mean Waiting Time using Multilevel Queue**
Washington	19	30	0.000058
Edge Server	**Arrival rate**	**Service rate**	**Mean Waiting Time using Multilevel Queue**
London	18	30	0.000046
New Delhi	115	118	0.000022
Colombo	112	116	0.000035

Table 3. Shortest Edge Server Distance Using Dijsktra Algorithm and Bandwidth of Host Server and Each Edge Server

Host Server	Shortest Distance edge server using Dijsktra algorithm	Bandwidth
Washington	13840 (Distance from host server)	9 Gbps
Edge Server	**Shortest Distance edge server using Dijsktra algorithm**	**Bandwidth**
New Delhi	1231	1Mbps
Colombo	1851	450Mbps
London	6832	200Mbps

Table 4. Throughput of Host Server and each Edge Server using throughput formula in multilevel queuing level

Operating Host Server	Throughput (ms)
Washington	55.321
Edge Server	**Throughput (ms)**
New Delhi	12.5542
Colombo	16.5747
London	6.4385

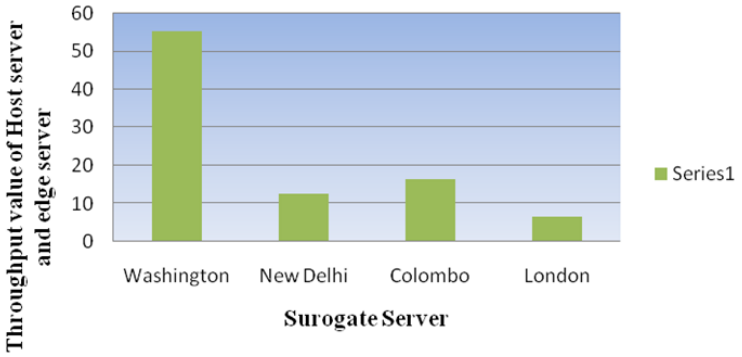

Figure 1. Through Put Values Of Host And Deferent Edge Servers

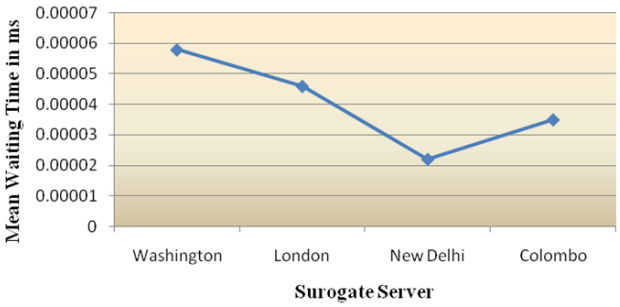

Figure 2. Mean Waiting Values of Host and Different Edge Servers

3.1. Result Analysis

By analysing above result, we see that the client of Asia (Kolkata) send request to his local server at Siliguri, but he can't find the respective content what he wants. So the request automatically terminates from Siliguri to Washington, where host server presents. Host Server transfers the request to System manager in CDN.CDN check the required content to its table. System manage first calculate the shortest distance edge server from Kolkata using Dijskatra algorithm where the required content founds from its contained indexed and most frequently used edge server tables. These tables track all records about all edge server as we see these tables calculates that Colombo is that shortest distance place where that respective content found and throughput value is high mean waiting time is less comparative to next shortest distance edge server of client from Kolkata. If the required contents found in Siliguri then there is no need for CDN in that case load for edge servers in CDN will be reduce. Here throughput calculation has done by multilevel queuing algorithm and other parametric values of each edge server through this method.

4. Conclusion and Future scope

In this paper we want to optimize the edge server selection process using local server and system manager concept over CDN. We have done that work with effectively. Our simulation analysis shows that result that CDN becomes an efficient way of delivering both static and dynamic contents faster to client with the help of local server and system manager. Local server and system manager reduce the workload over CDN and help to improve the network performance with efficiently. But if we use pipelining concept for handling more request in parallel over CDN then the performance will be very effective in future.

Acknowledgements

We would like to thank our HOD Dr. kalyani Mali in department of Computer Science and Engineering in University of Kalyani for purpose of laboratory and university infrastructure

References

[1] S. Saroiu, *et al.*, "An analysis of internet content delivery systems", ACM SIGOPS Operating Systems Review 36, SI **(2002)**, pp. 315-32.
[2] A. –M. K. Pathan and R. Buyya, "A taxonomy and survey of content delivery networks", Grid Computing and Distributed Systems Laboratory, University of Melbourne, Technical Report **(2007)**.
[3] K. H. Wong, "Using surrogate servers for content delivery network infrastructure with guaranteed QoS. Diss. UTAR, 201.
[4] file:///C:/Users/User/Desktop/ph.d%20documents/Dijsktra%27s%20algorithm.htm
[5] Silberschatz, Galvin and Gagne, "Books of Operating System Concepts, International Student Version Paperback", (2009) April 20.
[6] https://www.excentis.com/blog/measuring-throughput-effect-used-tcp-settings
[7] Sarddar, Debabrata, S. Roy and R. Bose, "Queueing Based Edge Server Selection in Content Delivery Network Using Haversine Distance".

Authors

Debabrata Sarddar, is an Assistant Professor at the Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India. He completed his PhD from Jadavpur University. He did his M. Tech in Computer Science & Engineering from DAVV, Indore in 2006, and his B.E in Computer Science & Engineering from NIT, Durgapur in 2001. He has published more than 75 research papers in different journals and conferences. His research interests include Wireless and Mobile Systems and WSN, and Cloud computing.

Enakshmi Nandi, received her M.Tech in VLSI and Microelectronics from Techno India, Salt Lake, West Bengal and B.Tech in Electronics and Communication Engineering from JIS College Of Engineering, West Bengal under West Bengal University of Technology, West Bengal, India. At present, she is Research scholar in Computer Science and Engineering from University of Kalyani. Her research interests include Cloud Computing, Mobile Communication system, Device and Nanotechnology

