An Efficient Edge Servers Selection in Content Delivery Network Using Voronoi Diagram

Debabrata Sarddar[1]
Department of Computer Science & Engineering
University of Kalyani
Kalyani, India
dsarddar1@gmail.com

Sandip Roy[2]
Department of Information Technology
Brainware Group of Institutions
Kolkata, India
sandiproy86@gmail.com

Rajesh Bose[3]
Senior Project Engineer
Simplex Infrastructures Ltd.
Kolkata, India
bose.raj00028@gmail.com

***Abstract*—**Handle on demand network popularity of the Content Delivery Network and solve the flash crowd problem, caching web content at the internet's edge server has been emerged. To provide faster service of the web users, content come from nearby edge servers. Therefore nearest edge server finding of a particular web user is an open research problem and it ensures a faster response time and download time of the requested content due to reduced latency. Our simulation study and application to a real set of geographical coordinates of the edge servers which are geographically dispersed and a well-known geometric device, Voronoi diagram is used for decomposing the earth surface around each location of a particular edge server and closest edge servers of the requested web user is searched by the nearest neighbor queries using Delaunay triangulation property over the aforesaid decomposed earth surface.

Keywords*-Cloud computing; Content Delivery Network; Delaunay triangulation; Nearest neighbor queries; Voronoi diagram*

*****

I. INTRODUCTION

A Content Delivery Network (CDN) is a large distributed system of edge servers that deliver web content to the requested users based on their geographic locations [1], [2]. But the average response time of the original host server of popular website requested by copious users becomes higher. Solution to this problem is caching the web content to the edge servers which are geographically scattered. When a user requests for web content the CDN redirects the user's request from the originating host server to an edge server that is closest to the requested user. This technique helps to reduce the average response time and improves the average packet loss of the CDN [3].

In this paper we have introduced a decomposing method using Voronoi diagram and finding nearest neighbor edge servers using Delaunay triangulation. Using "ping" command the accurate network latency time of nearest neighbor edge servers is captured. The connection is established between minimum network latency edge server and requested user.

II. BACKGROUND STUDY

A. Content Delivery Network (CDN)

A Content Delivery Network (CDN) is a distributed network of geographically scattered edge servers which caches the static and some dynamic content of a website like the images, Cascading Style Sheet (CSS) and Jave Script (JS) or other users' related files. When users from different geographical locations request a website then the nearest edge server to the user will deliver that static and dynamic content with the help of CDN and reduce the network latency time which is depicted in Figure 1 [3].

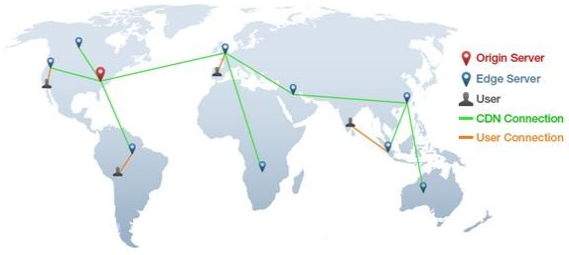

Figure 1. An example of Content Delivery Network

B. Voronoi diagram

A Voronoi diagram is a way of decomposing earth surface into a number of Voronoi cells as shown in Figure 2. Let us consider a finite set of geographical points $P = \{P_1, P_2 \ldots P_N\}$ over the earth surface [11]. Each point P_k has its corresponding Voronoi region R_k consisting of every point whose distance to P_k is less than or equal to its distance to any other coordinate of other Voronoi region.

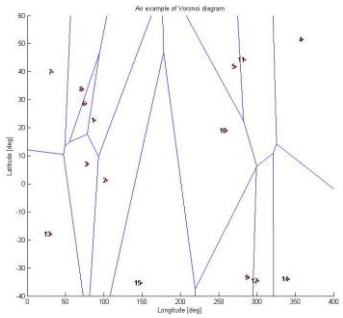

Figure 2. An example of Voronoi diagram

Shamos and Hoey explained Voronoi diagram using a divide- and-conquer algorithm [17]. Afterwards many efficient data structures and algorithms were designed for executing Voronoi diagram [4], [5], [6], [10], [18] and it founds their application in many different fields in Computer Science [5], [6], [8], [18].

C. *Delaunay triangulation*

A Delaunay triangulation for above set of geographical points P is a triangulation DT(P) such that no point of P lies inside the circumcircle of any triangle in DT(P) [7], [9].

Connecting the centers of the circumcircles of Delaunay triangulation in the Figure 3 produces the Voronoi diagram as like Figure 2.

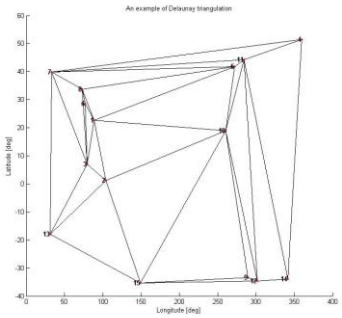

Figure 3. An example of Delaunay triangulation

III. PROPOSED ALGORITHM

In our proposed algorithm we have formulated an efficient technique for CDN provider which is monitoring a particular CDN to select appropriate edge server to reduce the average response time and to improve the average packet loss [3]. Here the set of edge servers are considered as the set of points P scattered over geographical region. This set of edge servers P is decomposed using Voronoi diagram in our context [11]. Then using Delaunay triangulation we can find out the nearest neighbor edge servers of requested user. Executing "ping" command over the nearest neighbor server's IP address generates the accurate network latency time, which is used to select the minimum average latency time edge server for delivering web content of a particular host server [15], [16]. The average packet loss is minimized as the connection is established with one of the closest edge server [3].

Our proposed method is implemented using Matlab R2012b [12], [13], [14]. Here we have considered the following enlisted edge servers as Table I. The geographical location of enlisted edge servers is portrayed in Figure 4.

TABLE I. LATITUDE AND LONGITUDE OF THE DIFFERENT EDGE SERVERS

	Location of edge server	Latitude	Longitude
1	Kolkata	22.5667° N	88.3667° E
2	Singapore	1.3000° N	103.8000° E
3	Colombo	6.9344° N	79.8428° E
4	London	51.5072° N	0.1275° W
5	Chicago	41.8819° N	87.6278° W
6	New Delhi	28.6139° N	77.2089° E
7	Ankara	39.9300° N	32.8600° E
8	Islamabad	33.7167° N	73.0667° E
9	Santiago	33.4500° S	70.6667° W
10	Mexico	19.000° N	99.1333° W
11	Kingston	44.2333° N	75.6919° W
12	Buenos Aires	34.6033° S	58.3817° W
13	Harare	17.8639° S	31.0297° E
14	Cape Town	33.9253° S	18.4239° E
15	Canberra	35.3075° S	149.1244° E

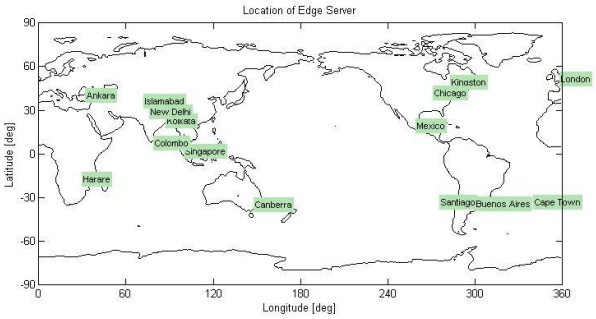

Figure 4. Location of Edge Servers over the earth surface

Algorithm for selecting edge servers for a particular CDN

1. A set lat = {lat_1, lat_2, lat_3, …, lat_M} of latitudes are assigned in [1 × M] array
2. A set lon = {lon_1, lon_2, lon_3, …, lon_M} of longitudes are assigned in [1 × M] array
3. A set latv = {lat_1, lat_2, lat_3, …, lat_M} of latitudes are assigned in [M × 1] array
4. A set lonv = {lon_1, lon_2, lon_3, …, lon_M} of longitudes are assigned in [M × 1] array
5. A set ipaddr = {$ipaddr_1$, $ipaddr_2$, $ipaddr_3$, …, $ipaddr_M$} of IP addresses in [1 × M] string array
6. For i ← 1 to size (lon)
   - 6.1 If lon (i) ≤ 0
     - 6.1.1 lon (i) ← lon (i) + 360
   - 6.2 End
7. voronoi(lon, lat) // Voronoi diagram
8. dt ← DelaunayTri (lonv, latv) // Delaunay triangulation
9. mylat ← 22.5697200 // Latitude of user's current location (e.g. Kolkata)
10. mylon ← 88.3697200 // Longitude of user's current location (e.g. Kolkata)
11. qrypts ← [mylat mylon] is [1 × 2] array
12. [pid, D] ← nearestNeighbor(dt, qrypts) // returns the index of nearest point in dt and returns in addition of the corresponding Euclidean distances D between the query points and their nearest neighbors
13. For i ← 1 to 3
    - 13.1 neighlat (i, 1) ← lat (dt (pid, i))
    - 13.2 neighlat (i, 2) ← lon (dt (pid, i))
    - 13.3 [status, result] ← dos(['ping -n 3 ' ipaddr (dt (pid, i), :)]) // Assign status and result value using ping command
    - 13.4 If (status == 0)
      - 13.4.1 If (strfind (result, 'Average = ')) // Matching 'Average = ' from result
        - 13.4.1.1 ans (i, 1) = result (length (result) – 6 : length (result) – 3)
        - 13.4.1.2 data.avgrestime (i, 1) = str2double (ans (i, 1)) // Average response time
        - 13.4.1.3 data.sltneighip (i, 1) = ipaddr (dt (pid, i), 1) // IP address of neighbors
        - 13.4.1.4 data.ipindex (i) = dt (pid, i)
      - 13.4.2 End
    - 13.5 else
      - 13.5.1 ans (i, 1) = '4000' // Default Time out time
      - 13.5.2 data.avgrestime (i, 1) = str2double (ans(i, 1))

13.5.3 data.sltneighip (i, 1) = '127.0.0.1' // IP Address of local host
13.5.4 data.ipindex (i) = 0
13.6 End
14. End
// Calculate the minimum average time from data.avgrestime
15. minavg = data.avgrestime (1);
16. ip = data.sltneighip (1, 1)
17. If (minavg > data.avgrestime (2))
17.1 minavg = data.avgrestime (2)
17.2 ip = data.sltneighip (2, 1)
18. End
19. If (minavg > data.avgrestime (3))
19.1 minavg = data.avgrestime(3)
19.2 ip = data.sltneighip (3, 1)
20. End
21. dos (['nslookup' ip]) // Finds name server information
22. End

IV. SIMULATION ANALYSIS

Step 1. Assigning latitude and longitude value in lat and lon array variables, which are depicted in third and fourth column of Table II respectively. The transformed negative longitude values by adding 360° are enlisted in Table II.

Step 2. Then IP addresses of different locations of edge servers are assigned in ipaddr array variable. These edge servers along with their IP addresses and domain names are enlisted in Table III.

Step 3. The set of edge servers are decomposed using Voronoi diagram as shown in Figure 5.

Step 4. Figure 6 shows Delaunay triangulation object from set of latitude and longitude values of edge servers over the earth surface.

TABLE II. LATITUDE AND LONGITUDE OF THE DIFFERENT EDGE SERVERS OVER THE EARTH SURFACE

	Location of Edge Servers	Latitude	Longitude	Modified Longitude
1	Kolkata	22.5667	88.3667	88.3667
2	Singapore	1.3000	103.8000	103.8000
3	Colombo	6.9344	79.8428	79.8428
4	London	51.5072	-0.1275	359.8725
5	Chicago	41.8819	-87.6278	272.3722
6	New Delhi	28.6139	77.2089	77.2089
7	Ankara	39.9300	32.8600	32.8600
8	Islamabad	33.7167	73.0667	73.0667
9	Santiago	-33.4500	-70.6667	289.3333
10	Mexico	19.0000	-99.1333	260.8667
11	Kingston	44.2333	-75.6919	284.3081
12	Buenos Aires	-34.6033	-58.3817	301.6183
13	Harare	-17.8639	31.0297	31.0297
14	Cape Town	-33.9253	-18.4239	341.5761
15	Canberra	-35.3075	149.1244	149.1244

TABLE III. IP ADDRESS AND DOMAIN NAME OF DIFFERENT LOCATION OF EDGE SERVERS

	Location of Edge Servers	IP Address	Domain Name
1	Kolkata	203.197.118.81	www.jaduniv.edu.in
2	Singapore	137.132.21.27	www.nus.edu.sg
3	Colombo	192.248.17.88	www.cmb.ac.lk
4	London	212.113.11.22	www.lse.ac.uk
5	Chicago	198.101.129.15	www.uchicago.edu
6	New Delhi	103.27.9.20	www.du.ac.in
7	Ankara	80.251.40.153	www.ankara.edu.tr
8	Islamabad	61.5.158.124	www.islamabadairport.com.pk
9	Santiago	158.170.64.116	www.udesantiago.cl
10	Mexico	128.123.3.2	www.nmsu.edu
11	Kingston	130.15.126.136	www.queensu.ca
12	Buenos Aires	190.224.163.234	www.buenosairesherald.com
13	Harare	196.201.17.237	www.caaz.co.zw
14	Cape Town	41.72.141.237	www.capetown.travel
15	Canberra	137.92.97.88	www.canberra.edu.au

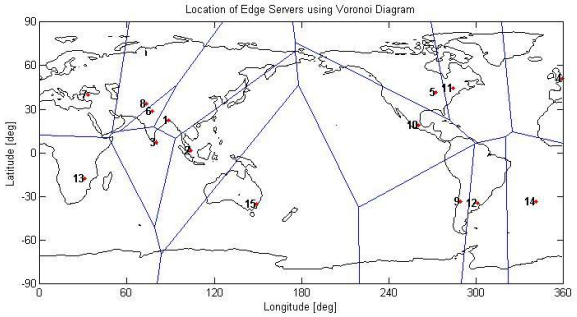

Figure 5. Decomposing Edge Servers using Voronoi diagram based upon geographical coordinates

Step 5. The nearest neighbor points are generated from the current latitude and longitude values of requested user and dt 2D array variable of Delaunay triangulation as shown in Table IV. Let us consider Kolkata as the current location of requested user. As a consequence pid variable is returned with value 7 and edge servers placed at Kolkata, New Delhi and Colombo are selected as nearest neighbor edge servers of requested user for delivering web content of underlying CDN.

Step 6. Executing "ping" command over the nearest neighbor edge server's IP address, i.e. the IP addresses of edge servers placed at Kolkata, New Delhi and Colombo. CDN provider dynamically captures actual network latency time of aforesaid edge servers.

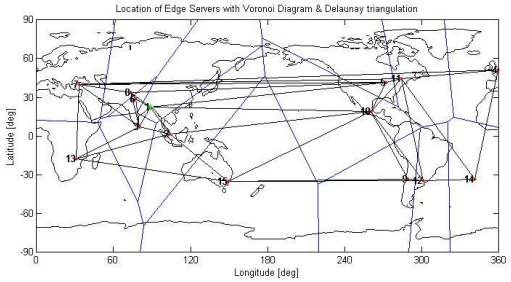

Figure 6. Simulation result of Delaunay triangulation & Voronoi diagram over the earth surface

TABLE IV. DT VALUE OF DELAUNAY TRIANGULATION

	Neighbor I	Neighbor II	Neighbor III
1	8	7	3
2	3	13	2
3	13	3	7
4	1	3	2
5	1	8	6
6	1	5	8
7	**1**	**6**	**3**
8	8	3	6
9	10	1	2
10	10	5	1
11	13	5	2
12	5	7	8
13	11	5	10
14	11	7	5
15	9	12	10
16	4	7	11
17	15	9	10
18	2	15	10
19	11	10	12
20	11	12	14
21	9	15	12
22	11	14	4

TABLE V. NETWORK LATENCY TIME OF NEAREST NEIGHBOR EDGE SERVERS FROM THE CDN PROVIDER OF PARTICULAR CDN USING WIRELESS 2G BANDWIDTH DATA CONNECTION

	Nearest neighbor	IP Address	Time I (ms)	Time II (ms)	Time III (ms)	Average time (ms)
1	Kolkata	203.197.118.81	1990	541	1400	1310
6	New Delhi	103.27.9.20	2677	840	638	1385
3	Colombo	192.248.17.88	2620	1040	2816	2158

TABLE VI. NETWORK LATENCY TIME OF NEAREST NEIGHBOR EDGE SERVERS FROM THE CDN PROVIDER OF PARTICULAR CDN USING WIRED 720 Kbps BANDWIDTH DATA CONNECTION

	Nearest neighbor	IP Address	Time I (ms)	Time II (ms)	Time III (ms)	Average time (ms)
1	Kolkata	203.197.118.81	539	562	567	556
6	New Delhi	103.27.9.20	334	357	380	357
3	Colombo	192.248.17.88	662	571	594	609

Step 7. CDN provider calculates minimum average response time among the nearest neighbor edge servers.

Step 8. Connection is established with edge server corresponding to the minimum average response time.

Edge server located at Kolkata is selected using wireless 2G bandwidth connection and edge server located at New Delhi is selected using bandwidth of 720 Kbps for considerate scenario.

V. CONCLUSION

Our proposed algorithm and simulation results articulate the fact of attaining minimum network latency time by selecting proper edge servers dispersed over the earth surface. These results also assert about attaining minimum packet loss during transmission. These results are produced with the use of Voronoi diagram and Delaunay triangulation for decomposing earth surface and nearest neighbor edge servers' selection respectively. Our proposed methodology can be used for mobile and standalone system in wireless and wired network and Content Delivery Network is one of the best applications where web content is delivered from closet edge server.

REFERENCES

[1] E. Nygren, R. K. Sitaraman and J. Sun, "The Akamai Network: A Platform for High-Performance Internet Applications," ACM SIGOPS OSR, vol. 44, no. 3, pp. 2–19, 2010.
[2] T. Repantis, J. Cohen, S. Smith and J. Wein, "Scaling a Monitoring Infrastructure for the Akamai Network," ACM SIGOPS Operating Systems Review, vol. 44, no. 3, July 2010.
[3] J. Parikh, H. Prokop, R. Sitaraman, J. Dilley, B. Maggs and B. Weihl, "Globally Distributed Content Delivery," IEEE INTERNET COMPUTING, pp. 50-58, September-October 2002
[4] F. Aurenhammer, "Voronoi Diagrams—A Survey of a Fundamental Geometric Data Structure," ACM Computing Surveys, vol. 23, no. 3, pp. 345–405, September 1991.
[5] S.W. Bae and K.-Y. Chwa, "Voronoi Diagrams with a Transportation Network on the Euclidean Plane," In: Algorithms and computation: 15th international symposium, ISAAC, Hong Kong, pp. 101–124, 2004.
[6] O. Beaumont, A.-M. Kermarrec, L. Marchal and E. Rivière, "VoroNet: A scalable object network based on Voronoi tessellations," Laboratoire de l'Informatique du Parallélisme, Lyon, France, RR n° 5833, February 2006.
[7] S.J. Fortune, "Voronoi diagrams and Delaunay triangulations, "In Computing and Euclidean Geometry, by D.-Z. Du, F.K. Hwang, World Scientific, Singapore, pp. 193–233, 1992.
[8] A. Landström, A. Simonsson & & H. Jonsson, "Voronoi-based ISD and site density characteristics for mobile networks," IEEE 76th Vehicular Technology Conference: VTC2012-Fall: Towards Sustainable Mobility; Proceedings. Piscataway, NJ: IEEE, 1-4673, 5 p. (Proceedings of the IEEE VTS Vehicular Technology Conference), 2012.
[9] C. Shouraboura and P. Bleher, "Placement of applications in computing clouds using Voronoi diagrams," Springer Journal of Internet Services and Applications, Special Issue: Cloud Computing DOI 10.1007/s13174-011-0037-8, September 2011.
[10] B. Kao, S. D. Lee, F. K.F. Lee, D. W.-l. Cheung and W.-S. Ho, "Clustering Uncertain Data Using Voronoi Diagrams and R-Tree Index," IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, vol. 22, no. 9, pp. 1219-1233, September 2010.
[11] H.-S. Na, C.-N. Lee, O. Cheong, "Voronoi diagrams on the sphere," In: Computational Geometry: Theory and Applications, pp 183–194, September 2002.
[12] http://www.mathworks.in/help/matlab/ref/voronoi.html
[13] http://www.mathworks.in/help/matlab/delaunay-triangulation.html
[14] http://www.mathworks.in/help/matlab/ref/delaunaytriangulation.nearestneighbor.html
[15] https://www.microsoft.com/resources/documentation/windows/xp/all/proddocs/en-us/ping.mspx?mfr=true
[16] http://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus5000/sw/command/reference/fund/n5k-fund-cr/n5k-fund_cmds_p.pdf
[17] M. Shamos and D. Hoey, "Closest-Point Problems," 16th annual IEEE symposium on Foundations of Computer Science, pp. 151–162, 1975.
[18] A. Okabe, B. Boots and K. Sugihara, "Spatial Tessellations: Concepts and Applications of Voronoi Diagrams," Wiley, Chichester, 1992.

AUTHORS PROFILE

Debabrata Sarddar[1], Assistant Professor in the Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India. He has done PhD from Jadavpur University. He completed 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 interest includes Mobile System, Wireless Sensor Network and Cloud Computing.

Sandip Roy[2], Assistant Professor in the Department of Information Technology, Brainware Group of Institutions, Kolkata, West Bengal, India. He has completed M.Tech in Computer Science & Engineering from HIT under WBUT in 2011. He has also done his B.Tech in Information Technology from WBUT in 2008. His main areas of research interest are Cloud Computing, Data Structure and Algorithm.

Rajesh Bose[3], Senior Project Engineer in the Simplex Infrastructures Limited, located in Kolkata. He received his M.Tech in Mobile Communication and Networking from WBUT in 2007. He had also received his B.E. degree in Computer Science and Engineering from BPUT in 2004. His research interest includes Wireless Communication, Networking and Cloud Computing.