International Journal of Grid and Distributed Computing
Vol. 10, No. 3 (2017), pp.41-52
http://dx.doi.org/10.14257/ijgdc.2017.10.3.04

Edge Multilevel Edge Server Co-operation in Content Delivery Network using Hierarchical Classification

Debabrata Sarddar, Subhankar Roy and Priyajit Sen

Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India
roysubhankarcs@gmail.com

Abstract

A Content Delivery Network (CDN) is a system of distributed servers that deliver web pages and other web contents to a user from a suitable Edge server. The purpose of a CDN is to provide web contents faster and with a higher availability. This service is effective in speeding up the delivery of content of websites with high traffic and websites that have global reach. The edge servers are cache servers that are deployed over the geography. These servers are used to store same contents from main server i.e. the content replicas are cached in these servers. User's requests for a web content may be served directly from any of these edge servers instead of the main server or origin server. This approach overcomes the limitations of network infrastructure in the internet. In this paper we propose a way to select appropriate edge servers when there is cache miss in the primary Edge Server up to the Origin Server. We have developed a technique for CDN provider to route user's request from the requesting machine to edge servers or to the origin in a Content Delivery Network (CDN) by applying hierarchical classification of edge servers to cluster them based on their response delay i.e. effectively based on the distance among the edge server.

Keywords*: Content Delivery Network, hierarchical or agglomerative approach, clustering, edge server, dendogram tree.*

1. Introduction

The internet has changed our lifestyle, business, entertainment, social activity at a huge that we cannot even take a single footstep without it. The revolution in the networking technology has given us the pace of life now we have. Yet internet was not engineered to have such elevation of reliability, robustness, scalability and performance. Modern human life consumes more data that few years ago we cannot even imagine. However internet service has emerged since then becoming an essential part of our modern world. The internet is a network of networks administered by different autonomous bodies. Internet Service Providers (ISP) are the one who manages this network, links, routers and infrastructure to provide internet service to the end users.

As web sites grow to be popular, they're often become exposed to bottleneck of server. Excessive requests contaminate the infrastructure of internet, such as the frontend Web server, bandwidth, or the back-end transaction-processing infrastructure. The request burden can crash a site or cause unusual delay in response times resulting in a negative impact on customers toward a product or brand. To help overcoming this problem the concept of edge server is introduced. As the demand for data grow very high, replicating popular contents and storing them in geographically nearer to the user has given us one solution and popular as Content Delivery Network (CDN).

Akamai Technologies, Inc. an American content delivery network and cloud services provider has first introduced the concept of CDN and now provides Web and IP application acceleration, including High Definition live media, on demand media streaming. Now, there are many organizations and companies providing CDN

ISSN: 2005-4262 IJGDC

infrastructure like Amazon Cloudfront, a global CDN for serving static and streaming content. It works seamlessly with any origin server and other Amazon Web Services (S3, EC2), supports secure web pages. Azion is a modern CDN with real-time analytics, comprehensive API and fast purging. BelugaCDN is a pay-as-you-go, low-cost CDN provider with POPs on 5 continents. They provide features include: full-featured IPv4+IPv6 network, instant purging, log streaming and HTTPS at no extra charge. ChinaCache is well-known as the CDN market leader in China, which has to do with 127 POP locations and 11000 servers in mainland China. The CDN features list is impressive, including hotlink protection, custom CNAME for SSL and Purge All. Verizon Digital Media Services is a top player in the CDN market with 47 'SuperPOPs' in North-America, EU, Asia, Australia, South-America and the Middle-East. VDMS offers useful features like a HTTP Rules Engine, multiple origins and CNAMEs and Token Authentication.

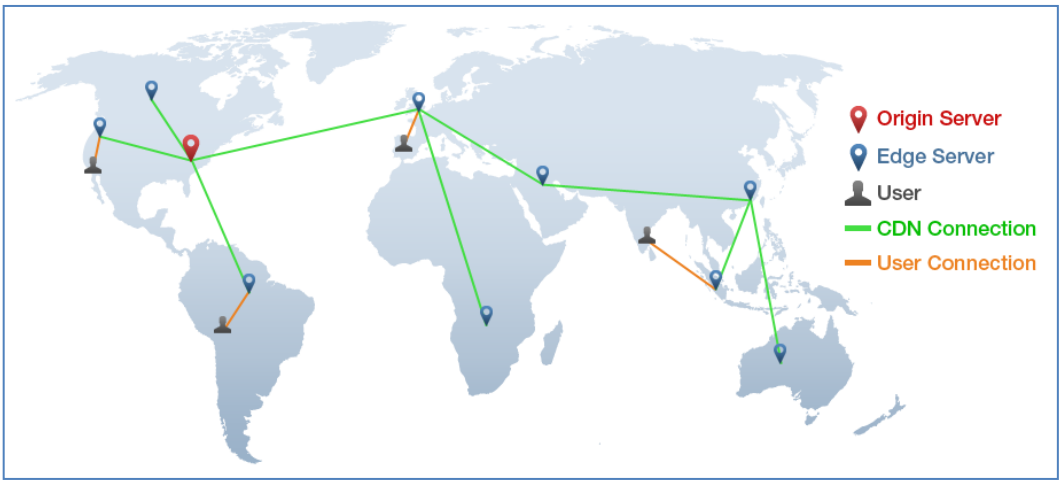

Figure 1. Network Distribution in the Geographical Region for CDN

Request Re-direction

When a user requests for a domain the request is re-routed via DNS to the Edge Servers instead of the main server. When the browser looks up this hostname, the user's DNS server will contact DNS servers. DNS servers are typically deployed in layers: the first layer is named a Top Level Name Server (TLNS)[1], which talk to the lower layers which are composed of Low Level Name Servers (LLNS)[1]. The user's DNS server will locate an edge server deployment near the user. The user's DNS server is used as a proxy for the actual location of the user. Once a server has been chosen the TLNS will provide a pointer to the LLNS within the deployment .The user's DNS server will contact the LLNS server in order to learn the IP address of the edge server in that deployment which can serve the customer's requested content.

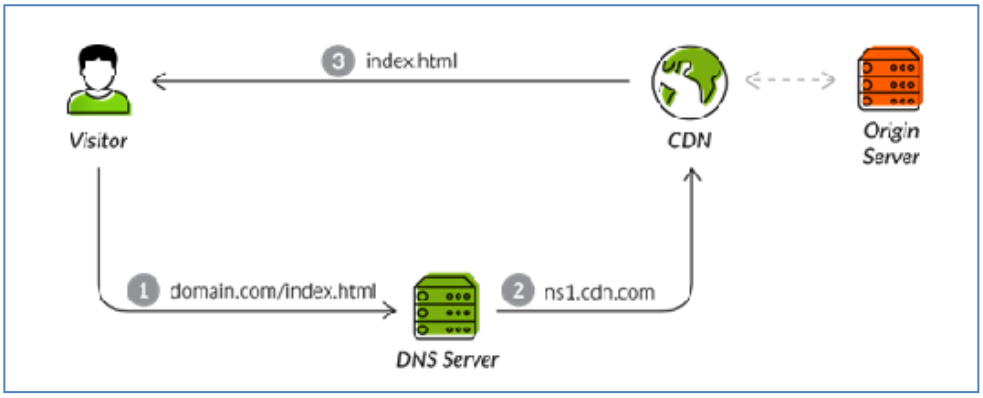

Figure 2. Request Redirection in CDN

The edge server will check its local cache for the requested content. If the requested content is available and the contents are not stale, then the contents are returned to the user. But when the requested content is not available in the current edge server or the contents are stalled in that particular edge server's cache, the request is forwarded to the origin server and the content is copied to the correspond edge server. For caching purpose suitable algorithms are used.

CDN is necessarily a complex system with many components. Two of the major concerns addressed by CDNs are ensuring efficient content distribution and freshness of content given to the customer. An enterprise of ISPs would never accept misses on certain objects. Retrieving large objects across the Internet during bandwidth-constrained hours could result in unacceptable latencies. Pushing time sensitive data (e.g. news, share prices, entertainment, live sporting events) to servers to ensure fresh content is a challenge, which CDN solution providers face. Most important of them is efficient distribution of all content to the respective servers.

2. Issues of CDN

The CDN requires establishing replica or surrogate servers and some supporting service infrastructure at the edge of the networks. To ensure the success of CDN a huge no. of edge server are required to deployed over the geography and there are several issues involved in CDN since there are several decisions to be made like edge server placement, contents updating mechanism, suitable edge server selection, re-routing algorithm. The optimal performance and reliability depend on these mechanisms.

Edge server Placement Mechanism

Edge server placement in appropriate location is critical to the performance of CDN. CDN topology is formulated such that the performance obtained to the client is maximized and cost to the provider is minimized. Therefore efficient surrogate server placement will reduce the no. of servers established and size of the cache such that client's requests are served with high availability and reduced cost. The parameters for this placement algorithms are mainly classified in two categories: (1) Cost related parameters, (2) Network related parameters.

Cost Related Parameters:

1. Resource Deployment cost: The cost of installing edge servers or leasing computing, storing, bandwidth resources. This is calculated as the unit resource cost with respect to the size of replica at the server. 2. Content Delivery cost: the cost of network involved in delivering content from the edge servers to the end users. It is calculated in terms of unit

bandwidth cost and user demand. 3. Content Replacement cost: the network cost involved in updating content from origin server to edge sever or a group of servers.

Network Related Parameters:

Network Performance: These are the parameters that suggest the performance and quality of service that the network topology provides between end users and edge servers. The parameters involved in this regard are latency, bandwidth, hop-count and link quality in replica placement algorithms in CDN.

Latency in the edge server placement is the time required to deliver content from edge server to the end-users. Most algorithms minimize this latency as objective function or uses latency as constraint to find effective distribution of servers. Bandwidth available is also considered in some algorithms proposed to place servers where available bandwidth is used as a QoS parameter. Hop-count is used as input in some of the algorithms to find location of edge server. Hop-count is the no. of hops required in the shortest path between edge server and the end users to route a request. Hop-count is used as objective function and the algorithm tries to minimize it keeping in mind that it will eventually reduce the latency.

Contents Updating Mechanism:

Updating contents time by time in the edge servers is important in CDN, if not client may get invalid response from the edge server creating inconsistency. Whenever contents are updated or changed in the hosting site or contents become stale, all the copies residing in surrogate servers should be replaced with most updated one. Mechanisms proposed in the literature are mainly as follows periodic update, expiration based update and cache miss based update.

Periodic update: Various dynamic algorithms in periodic updating mechanism are proposed for tree topologies. The origin server periodically servers contents to the edge server to replace old copy with new one. In[13] each node in the tree topology is an edge server which serves the client holds contents and manages the frequency of request for a copy indicating the popularity of the content in that node. In the update method of [12], a update distribution tree is formed to multicast copies periodically from origin to edge servers. The root of the multicast tree is the origin server that connects all edge servers through shortest path link. Each child node gets updates from its parent node up to bottom along the tree.

Expiration based update: Here TTL (Time to Live) based consistency is taken into account for each edge server. The edge servers make request for update when the TTL value is expired until a valid copy is reached. In [14] expiration based update mechanism is considered and replicas are placed in a network. A dynamic algorithm is proposed to place replica and update cost, delivery cost are minimized.

Cache based update: Two types of content outsourcing for cache based update are there: Co-operative caching and uncooperative caching scheme.

In co-operative caching scheme the suitable edge servers co-operates each other to serve client requests and update contents for stale content in order to reduce cost. In co-operative push based scheme the origin server pushes the contents from origin server to the surrogate servers and at first the contents are perfected in the surrogate servers. Client's request for a content is directed to the closest edge server and the CDN keeps the mapping between contents and corresponding edge servers. If not found then other edge servers cooperate to serve the request or the request is forwarded to the origin server. In co-operative pull based scheme when a miss occurs then the other edge servers cooperate each other to serve the request. In [16] several replication strategies for push-based co-operative caching have been discussed. It is observed that greedy heuristic algorithms are better for replication.

In uncooperative pull based caching if the requested content is not found in the edge server or the content is stale, the request is directed to the origin server or to a peering edge server in the CDN i.e. the edge servers pull contents from the origin server in case of a miss. The performance of uncooperative caching based update mechanisms has been discussed in [15].

Suitable Edge Server Selection

Choosing the appropriate edge server for delivery of a content is critical to the performance of a CDN. In general the edge server closest to the client is selected for the delivery of content. But there exists several matrices which play major role in the selection algorithms proposed in the literature like network proximity, response delay, geographical distance, load of the edge server. Considering this factors algorithms have been proposed to redirect client request to the most suitable close edge server. This request routing algorithms are largely divided into two types namely non-adaptive adaptive and routing algorithms.

Non-adaptive algorithms do not consider the current situation or load status of the edge server and an edge server is selected based on some heuristics. Non-adaptive algorithms are simple and easy to implement. The most common non-adaptive routing algorithm is round robin algorithm which tries to distribute load among all the edge servers [17]. The performance of round robin algorithm is limited by the distance of the edge servers from a client. It is assumed that all the edge servers are placed closely and they have same computing capability but it does not work well for widely distributed edge server network.

Some algorithms predict the load on the edge servers and rank them as per load to distribute client request. Here load assumptions are made based on the no. of requests served previously and also distance is considered a factor to select a proper edge server. Several other algorithms are implemented Cisco DistributedDirector [18]. One of the algorithms considers the no. of request received as a factor for choosing the edge server. Here it is assumed that the edge server with more no. of request is more powerful to serve the content and client's requests are forwarded to this powerful edge server. Karger et al. [19] proposed a non-adaptive algorithm where the selection of edge server is done as a result of a hash function based on the URL of the content.

On the other hand adaptive algorithms consider the current status of the edge servers and re-direct the request as per. These algorithms are complex in nature and matrices involved in measuring the status of edge servers are load of the server, congestion in the network, reliability of edge server, bandwidth available to edge server.

Cisco DistributedDirector [18] has implemented an adaptive server selection algorithm deployed in this network considers a weighted combination of three metrics, namely – inter-AS distance, intra-AS distance, and end-to-end latency.

Globule [20] uses an adaptive algorithm that chooses edge server closest to the clients in terms of network proximity. The path length considered as a metric in the algorithm is updated at regular intervals and it is done passively without introducing any supplementary traffic to the network.

Akamai [1] introduced a complex adaptive server selection algorithm which takes into account matrices like server load, reliability of the server, bandwidth available to the edge server to redirect request to the most suitable edger server.

Request Redirection

There are several mechanisms to redirect client's request to an appropriate edge server. Redirection mechanisms tell how one request to a server is transferred to another server. There are DNS based rerouting, HTTP redirections, URL rewriting, anycast.

DNS based redirection is simple and popular as it uses naming resolution so it can be used in any Web applications. In this approach a domain name is associated with a number of IP addresses of edge servers. The DNS server resolves this mapping of edge servers to its IP addresses. When a request arrives any of the associated IP address is returned and the corresponding edge server serves the client. In a single reply approach the DNS server returns only one IP address which best serves the request and in multiple replies approach a pool of edge server is returned and generally selected in round robin fashion. However the main drawback of DNS based routing is that it is associated with domain name whereas the system should serve requests at object granularity. The performance of DNS redirection algorithms are discussed in [21][22]. Another issue with DNS redirection is that it do not take into account the client IP i.e. is the proximity of the user is not consider and may redirect to servers that may degrade the overall performance of the CDN.

In HTTP redirection mechanism the server responds the client with a message to redirect the request to another surrogate server. This approach is simple, flexible and object granularity is maintained for individual web page. But it requires extra messages to circulate between the client and server is an overhead and redirection also lacks transparency.

In URL rewriting the server redirect the client request to an edge server by dynamically created URL links. This mechanism is mainly used where some embedded objects are returned to the client as a response. It uses some special scripts to rewrite the embedded URLs and redirects clients to fetch objects from other surrogate servers. There are two types of this approach namely proactive and reactive rewriting. In proactive approach the URL rewriting of embedded objects is done before the contents are hosted in the origin server. In reactive approach rewriting of embedded object URLs are done after a request from client is made at the origin server.

Anycasting redirection scheme uses IP level redirection where set of servers have the same IP address. The contents are replicated on those servers and any of those equivalent servers may be accessed to serve a request. The user generates request for a content with anycast name as destination address and the anycast service redirect this request to any of the servers having same anycast address. Servers are chosen based on characteristics that best serve the client. Anycasting can work with servers that are heterogeneous and each router holds path to the closest server thus different router has different paths to different server with same IP address.

In our paper we propose that the edge servers will co-operate to each other to serve a content based on our algorithm. It will reduce the delay to fetch data from the main server as it is provided by a nearer edge server. A content table will be maintained at each Edge server to reduce searching time and will be updated as per data swapping and caching policy. We used agglomerative clustering to form a tree structure of the edge server. We consider that a user will request its nearest Edge server for a request or some other characteristics like processing power; availability etc may also be considered.

Agglomerative Clustering

The agglomerative clustering method formulates a hierarchical tree structure of provided data points in the geography. The procedure to cluster starts with n clusters for n data points, that is, each single data point is a cluster of its own. Using latency time as a count of distance two nearest clusters are merged at each steps of the algorithm, therefore the number of clusters is lessened and a successively larger cluster is formed. It is a bottom-up approach. When all data points are in a same cluster, then the process is stopped. Here each Edge server along with the Origin server is taken as data points in the algorithm.

Steps involved in agglomerative clustering are as follows:

1. Assign each Edge Server and also the Origin Server to separate clusters of its own.
2. Formulate a distance matrix by determining the latencies between all pair of clusters. Sort these latencies in ascending order.
3. Look for the two clusters that have the smallest distance.
4. Combine the pair with smallest latency into one cluster.
5. Stop until there is only a single cluster remaining.
6. Re-evaluate all distances with the new cluster and revise the distance matrix after the recombination and go to step 3.

3. Related Work

The CDN edge servers are distributed all over the earth surface to increase the global availability of web contents and to reduce the bandwidth cost. CDN Users may request from different location for particular web content and a relevant edge server is selected based on the characteristics of delay, availability, bandwidth etc [2]. In paper [3], authors have presented a detail description about CDNs with respect to four different factors as request-routing procedure, content replication mechanisms, load balancing methodology and cache management. Co-operative caching improves the response time by reducing VM synthesis time by caching the previous state [4].

In their paper, P. Sen et al. have suggested that, in cloud computing, instead of co-operative caching one of the devices can be chosen as a central device called Central Controller [5] that keeps track of the status of all the other devices connected with it using a star topology network. During this time the Central Controller scan passively all the devices dynamically to get the current status about storage whereas a device need to store data will perform an active scan over the Central Controller to know the status of all other devices. As soon as, it gets the information it will proceed directly onto that particular device to store the data. Thus Response time and efficiency can both be effective and valuable for future course. In their paper Centralized scheduler for content delivery network, Jun Li et al. have proposed the Centralized Scheduler [6] Concept for the content delivery network that keeps track of all the request and service provided in the network.

The caching system largely is classified into two branches namely CDN, transparent caching. In CDN content provider is in businesses relationship with the content publisher and in transparent caching the service provider performs all the caching without awareness of caching. For cache sharing, protocols like Summery Cache [7], Cache digests [8], Cache array routing protocol (CARP)[9], Internet Cache Protocol (ICP)[10], Hypertext Caching Protocol (HTCP)[11] etc have been proposed.

4. Proposed Work

In our algorithm we proposed which servers are to be requested and what will be the order for forwarding the request whenever there is a miss or a stale content .Here we form clusters of edge servers using the hierarchical clustering. Each edge server is taken as a point in the domain and the delay of response as the distance between the respective servers. By the execution of 'ping' command over all the edge server‘s IP address, we can simply fetch the exact network latency time. Here we are not concern about the actual geographical location of any server .Time delays between each individual edge servers are calculated and also the time delay from the origin server is calculated. The time delay is taken as distance between edge servers in Agglomerative Hierarchical Method to cluster the servers. Hierarchical clustering produces a set of nested clusters organized as a hierarchical tree. If the content is not found in the primary edge server then request is forwarded to next the edge servers, belonging to the same cluster as per the dendrogram tree hierarchy. In our algorithm we proposed the order in which servers are to be

requested for forwarding the request whenever there is a miss or a stale content. Here we form clusters of edge servers using the hierarchical clustering. Each edge server is taken as a point in the space and the delay of response as the distance between the respective servers. By the execution of 'ping' command over all the edge server‘s IP address, we can simply fetch the exact network latency time. Here we are not concern about the actual geographical location of any server .Time delays between each individual edge server and origin servers are calculated. This response delay is taken as distance between servers in Agglomerative Hierarchical Method to cluster the servers. Hierarchical clustering produces a set of nested clusters organized as a hierarchical tree. Here we propose that every edge server maintains a content table showing which contents are available in that edge server. The content table shows the corresponding memory location of the contents, making the whole searching process within the server faster. After searching the members of the primary cluster the request will be forwarded to the next Edge Server up in the nested tree. If the requested object is still not found at this level of Edge Servers, the request will be forwarded to the next Edge Server as per the proposed tree. As the Origin Server is a part of the clustering hierarchy, the request will eventually find the Origin Server at some level. The tree structure suggests the path through which a request should be routed in the geography. In every Edge Server a content table will be maintained every time some content is written or replaced in the cache. The content table indicates the memory location of a content and shows whether the content is actually available or not in the corresponding edge server. These will minimize the time for searching content in any server. As the requests are not directly forwarded to the Origin server after the very first cache miss in most cases, it reduces the load in the Origin server and also distributes object requests over the whole Edge Server network making it more responsive and robust to sever failures. It reduces the packet loss and response delay to serve any content to the end user. The Origin server is not get chocked by the requests after cache misses over the whole network, as other Edge Servers cooperate to find the contents in their Edge Servers.

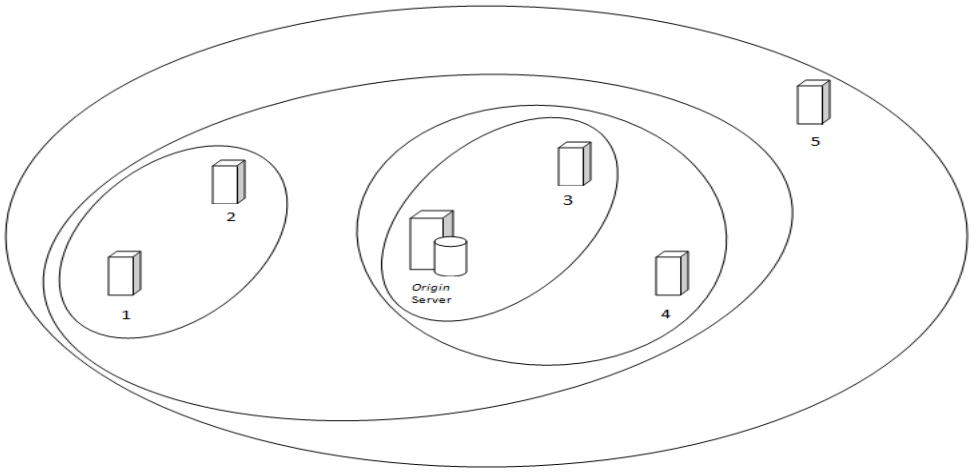

Figure 1. Origin Server and Edge Servers Clustered Over the Region

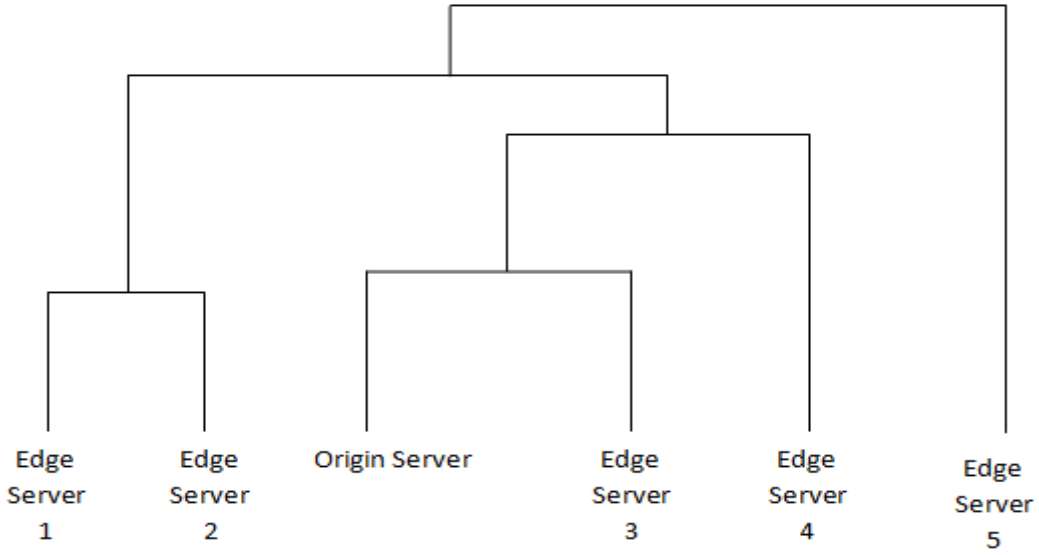

Figure 2. Dendrogram Tree Records the Sequence of Edge Server and the Origin Server

The requests are routed from one edge server to another based on the dendrogram sequence tree and ultimately it reaches to the main origin server. For example if a request is made at Server 4 and the content is not found then the request is forwarded to the Edge Server up in the tree. In this case the request is routed through the path like Server 4 -> Server 3 -> Origin Server. This is shown in figure 3. In the same way a request in Server 4 will have a route like Server 4 -> Server 3 – Server ->2 -> Origin Server. Whenever the contents are already found on the Edge Servers before reaching to the Origin Server the contents are sent to the requester. So the Edge Servers closer to the client are approached with a probability in the whole process and the contents are provided faster.

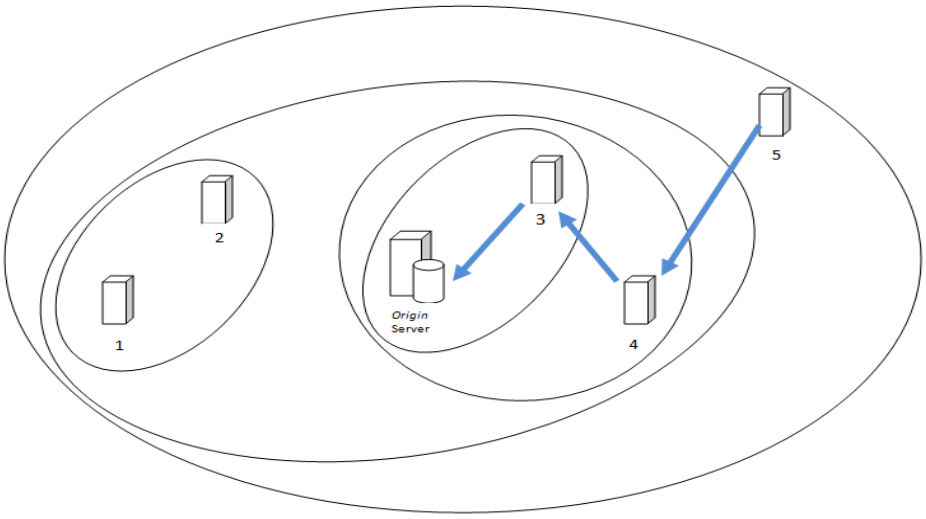

Figure 3. Request Forwarding

5. Conclusion

As the request for data is increasing and people are accessing huge data contents using their devices through internet it has become mandatory about the contents to be fast, available and robust to bottleneck. Our approach tries to satisfy the requirements mentioned by lowering the no. of requests made to the main Server or the Origin Server. It tries to serve contents from other Edge Servers as near as possible to the end user and reduce the delay. Here the origin server does not get chocked due to huge requests every time a miss occurs within an edge server as other edge servers try to satisfy the requests by themselves. It states the routing path for any request to the main server.

However, there are some concerns like if the requests of the users are very deviant or irregular it may increase the cache miss in successive edge servers and eventually ending up with a response from the Origin Server. This may have an adverse affect on the network.

References

[1] Akamai Technologies, Inc., www.akamai.com, 2016.
[2] Sougata Chakraborty and Debabrata Sarddar, “An Efficient Edge Server Selection in Content Delivery Network using Dijkstra’s Shortest Path Routing Algorithm with Euclidean Distance”, International Journal of Computer Applications (0975 – 8887) Volume 117 – No. 4, May 2015 24.
[3] 2Al-M. Khan Pathan and R. Buyya, “A Taxonomy and Survey of Content Delivery Networks,”www.cloudbus.org/reports/CDN-taxonomy.pdf”.
[4] Preetha Theresa Joy and K.Poulose Jacob : Cooperative Caching Framework for Mobile Cloud Computing, Global Journals Inc.(USA), Volume 13, Issue 8 Version 1.0, Online ISSN: 0975-4172 & Print ISSN: 0975-4350,2013.
[5] Debabrata Sarddar, Priyajit Sen and Manas Kumar Sanyal, “ Central Controller Framework for Mobile Cloud Computing”, International Journal of Grid and Distributed Computing Vol. 9, No. 4 (2016), pp.233-240 http://dx.doi.org/10.14257/ijgdc.2016.9.4.21.
[6] Jun Li, SnigdhaVerma and Kumar Ramaswamy. ”Centralized scheduler for content delivery network”, Link: http://www.google.co.in/patents/WO2007111588A1?cl=en Publication Number: WO2007111588 A1.
[7] L. Fan, P. Cao, J. Almeida and A.Z. Broder, “Summery Cache:m a scalable wide-area web cache sharing protocol”, ACM SIGCOMM Computer Communication Review, vol 23,(1993), pp.239-248
[8] Rousskov and D. Wesssels, “Cache Digests”, Computer Network and ISDN Systems, vol. 30, (1998) pp. 22-23.
[9] V. Vallopillil and K. W. Ross, “Cache Array Routing Protocol v1.0”, IETF Internet Draft, (1998), February, http://tools.ietf.org/id/draft-vinod-carp-v1-03.txt.
[10] D. Wessels and K. Claffy, “Internet Cache Protocol(ICP) version 2”, IETF RFC 2186, (1997) September, http://www.ietf.org/rfe/rfc/rfc2186.txt.
[11] P. B. Danzing, C. Neerdaels, M .F. Schwartz and K. J. Worrell, “A Hierarchical Internet Object Cache”, Proceedings of the Annual Technical Conference on USENIX, (1996),pp.13.
[12] S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt and L. Zhang, "On the placement of Internet instrumentation," in INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE, vol.1, no., pp.2.
[13] M. Shorfuzzaman, P. Graham and R. Eskicioglu, "QoS-Aware Distributed Replica Placement in Hierarchical Data Grids," in Advanced Information Networking and Applications (AINA), 2011 IEEE International Conference on, pp.291-299, 22-25 March 2011.
[14] X. Tang and S. Chanson, "Analysis of Replica Placement under Expiration-Based Consistency Management," IEEE Transactions on Parallel and Distributed Systems, vol. 17, no. 11, pp. 1253-1263, 2006.
[15] Johnson, K.L. et al. The measured performance of content distribution networks. Computer Communications24,2(Feb.2001),202–206.
[16] Kangasharju, J. et al. Object replication strategies in content distribution networks. Computer Communications25,4(Mar.2002),367–383.
[17] M. Szymaniak, G. Pierre, and M. van Steen, “Netairt: A DNS-based Redirection System for Apache,” In Proceedings of International Conference WWW/Internet, Algrave, Portugal, 2003.
[18] K. Delgadillo, “Cisco DistributedDirector,” Cisco White Paper, Cisco Systems, Inc., June 1997.
[19] D. Karger, A. Sherman, A. Berkheimer, B. Bogstad, R. Dhanidina, K. Iwamoto, B. Kim, L. Matkins, and Y. Yerushalmi, “Web Caching with Consistent Hashing,” Computer Networks, Vol. 31, No. 11-16, pp.1203-1213,1999.

[20] G. Pierre, and M. van Steen, "Globule: A Collaborative Content Delivery Network," IEEE Communications, Vol44, No.8, August2006.
[21] A. Shaikh, R. Tewari, and M. Agrawal, "On the Effectiveness of DNS-Based Server Selection," In Proceedings of IEEE INFOCOM, Anchorage, AK, USA, pp. 1801-1810, April 2001.
[22] Z. M. Mao, C. D. Cranor, F. Boughs, M. Rabinovich, O. Spatscheck, and J. Wang, "A Precise and Efficient Evaluation of the Proximity between Web Clients and their Local DNS Servers," In Proceedings of the USENIX 2002 Annual Technical Conference, Monterey, CA, USA, pp. 229-242, June 2002.

Authors

Debabrata Sarddar, Assistant Professor in the Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, INDIA, completed Ph.D at Jadavpur University. He completed M. Tech in Computer Science & Engineering from DAVV, Indore in 2006, and B.E in Computer Science & Engineering from NIT, Durgapur in 2001. He published more than 150 research papers in different journals and conferences. His research interest includes wireless and mobile system and Cloud computing. Email: dsarddar@klyuniv.ac.in.

Subhankar Roy is presently pursuing M.Tech in Computer Science and Engineering in the Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India. He completed his B.Tech from Govt. College of Engg & Textile Technology, Berhampore, Murshidabad, West Bengal, India in 2015. His research interest includes Mobile Computing, Wireless Sensor Network and Cloud Computing. Email: roysubhankarcs@gmail.com

Priyajit Sen is presently pursuing M.Tech in Computer Science and Engineering in the Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India. He completed his MCA from Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India in 2015. His research interest includes Mobile Computing, Wireless Sensor Network and Cloud Computing. Email: priyajit91@gmail.com

