A Study of Request-Routing in Content Delivery Networks

Shreeshrita Patnaik
University of Wisconsin-Madison
spatnaik2@wisc.edu

Arjun Kashyap
University of Wisconsin-Madison
akashyap3@wisc.edu

ABSTRACT

Content Delivery Networks (CDNs) have seen significant growth and evolution, due to the huge amounts of online content, specifically media content being generated continuously. As the demand for dynamic and streaming web content increases, it is important to inspect the several modules that make up a CDN. This paper focuses on looking into request-routing algorithms and mechanisms, and the role they play in Content Delivery, quantified by the use of relevant metrics.

1 OVERVIEW

1.1 Motivation

CDNs, today comprise of several heterogeneous components and mechanisms, each of which plays a role in delivering requested content to end-user in a faster, reliable and more secure way. Of all the blocks that make a CDN, we believe that request routing plays a significant role in ensuring CDNs are able to deliver content as guaranteed. Request-Routing is the process by which a client's request for content, gets routed to the specific surrogate server hosting the content and with minimum cost, which is defined by the use of several metrics. The ability to route a client request for content to an edge server that is chosen, so as to minimise the delay between request and response, would be an important factor in terms of both commercial benefit to CDNs and the delivered user experience.

A Request Routing comprises of two parts: the algorithm which chooses the specific server that should serve the request, and the mechanism which specifies how the request would be routed to the server that the algorithm picks. There are several different algorithms and mechanisms that exist today, in use by commercial CDN providers and also small- scale web content providers. However, we do not see a comprehensive comparison of these mechanisms and algorithms, under varied network and congestion conditions. We believe that an analysis of the widely used algorithms and mechanisms, their implementations and their behaviour under network loads or congested routes, would pave the way for a better understanding and selection of these mechanisms, according to the dynamic network conditions.

1.2 Research Problem

To evaluate the efficiency and accuracy of request-routing algorithms, subject to varied network conditions and discover whether these algorithms truly choose the optimal edge server to serve a client request.

2 RELATED WORK

2.1 A Taxonomy and survey of content delivery networks[1]

This paper serves as an entry point to the dynamic world of CDNs. The paper provides a comprehensive survey of all the terminologies and processes that make up a CDN. It provides a high-level overview of the workings of the components of a CDN and the interaction among them. It talks about: the architecture of a CDN, the various types of servers, the protocols used for communication by caches and network elements, about the concepts of replica placement, request routing and also gives a brief overview of the industry trends and popular Content Delivery Network providers at the time, the paper was released.

2.2 A Walk through Content Delivery Networks [2]

This paper also provides a high-level overview of the various components of a CDN. It delves into more details about request-routing algorithms. It talks about the implementation of a few request routing mechanisms, and gives a process walkthrough. The most significant takeaway of this paper were the sections on measurement and metrics used for analysing a Request- Routing Algorithm. It talks about the different types of measurement that we can use during our study. It also lists various metrics we can use, like latency, packet loss and network proximity. This paper, however does not consist of any experimental methodology to compare or analyse a request- routing mechanism.

2.3 Request-routing Trends and Techniques in Content Distribution Networks [3]

This paper provides an in-detail description of various Request-Routing solutions, like DNS-based, Transport-layer based, Application-layer based and content-layer based Routing mechanisms , used today is CDNs. It provides a more detailed outlook of the various components in each mechanism. It, however, does not provide any insight into the performance measure of any one .

2.4 Unreeling netflix: Understanding and improving multi-CDN movie delivery [4]

This paper was valuable in understanding how to set up measurement for analysing CDNs in use by a content provider like Netflix, how to discover servers, how to manually override the server selection and details about CDNs used by content providers like Netflix.

2.5 CDN DNS - An Efficient DNS Request Routing Technique in Content Delivery Network [5]

This paper was useful in explaining some details about the experimental methodology that can be set up to obtain an analysis of certain request-routing mechanisms like DNS-redirection. It delves into the selected mechanism, by both simulating an appropriate environment and also analysing datasets from popular CDNs. It also provides an insight into the effect of AS and network clustering on latency.

2.6 Drafting Behind Akamai: Inferring Network Conditions Based on CDN Redirections [6]

This paper introduces us to the measurement methodology of Akamai redirection mechanisms but does not speak about analyzing the request routing algorithms. It highlights the idea of how request routing mechanisms and algorithms depend on network conditions and server loads.

3 PROPOSED APPROACH AND CHALLENGES

3.1 Choosing a content provider that uses one or more CDNs

In order to start our study of comparing request routing algorithms being deployed today, and their performance measure, we need to choose a content provider which uses one or mode CDNs to host their content. For instance Netflix uses at least 3 CDNs: Akamai, LimeLight and Level-3[5]

3.2 Discovering Edge Servers

The next logical step would be to discover the edge server, which the request routing algorithm shall choose, where the server is placed (in terms of network proximity and AS no.) and the other servers present, that the request routing algorithm may bypass. There are a few tools to accomplish this, the simplest of which would be a nslookup, that can give us the identities of a couple of servers.

Challenge 1: A challenge in our proposed step, would be to find a way via which we can discover all the other edge servers that the content provider uses to host their content and how we can identify them. If we want to figure out the servers distributed across the world, one possible step as cited in [5]. would be to host VMs in the cloud-providers across geographic locations and send a GET request from these servers. We hope to find a simpler way to complete this step.

3.3 Finding which request routing algorithm a CDN uses

To map a CDN's performance to a request routing algorithm, we need to figure out which request routing algorithm is being used by a CDN. There may be several approaches to it.

3.3.1 Literature Review and white papers: We intend to look at publications and white papers that address or refer to the request routing algorithm in use by a CDN

3.3.2 Analyse headers: The best way to validate a request-routing mechanism would be deciphering the packet format sent back by the chosen server. We need to figure out if we can look at the packets and map it to a certain request-routing algorithm. This is also **Challenge 2**.

3.4 Starting data transfer and Observation

In this step we would need to start the data transfer and observe the server that gets chosen at different points in time during our project, measuring performance in terms of delay or latency between request and response and the throughput of the transfer.

Challenge 3: A challenge in this step would be to validate if the proposed metrics are a good measure of the performance of a request routing algorithm.

Challenge 4: Another challenge in this step would be breaking down the response time into factors accounting for: (1) the time taken for the redirection mechanism to choose a server based on its algorithm and the mechanism to deliver the server address and (2): the actual round trip time that is indicative of the server load and path congestion or throttled bandwidth.

Perhaps, a comparative study of the behaviour may help isolate the components. We can vary or manipulate the conditions by: (1) throttling bandwidth manually or by even (2) overriding the edge server selection.

3.5 Overriding Edge Server Selection

We would like to, in the course of our project, be able to manually override the edge server selection to choose a different edge server and verify if indeed the decision reached by the request routing algorithm, was indeed justified.

Challenge 5: To figure out how to manually override the edge server selection.

[5] talks of a way to trick the servers by modifying the HTTP host header.

3.6 Analyse Gathered Data

The tentative last step in our project would be to analyse all the data we've gathered and understand the performance of the selected request routing algorithms and the variance in behaviour under different network conditions.

4 METHODOLOGY

In order to understand and evaluate the request routing algorithms implemented by leading content providers, we needed to gather data that would be pertinent in decoding the efficacy and utilization of the algorithms. From the data collected we could observe the trends and analyze them. In order to do so, we followed a series of steps, ranging from experimental setup, to data collection and analysis.

4.1 Choosing a content provider

In order to begin data collection, we needed to zero in on one content provider, which in turn relied heavily on CDNs. Several content providers leverage multiple content delivery networks, both for redundancy and cost-effectiveness [2]. After careful consideration of a few choices of content providers including Netflix, Hulu, and

Adobe, we found analyzing Hulu to be the best choice. This choice was based on factors including a moderate number of servers from multiple different CDNs, impact and direction provided by related work we sampled and lastly, the monetary cost of membership of the content provider.

According to *comscore*, a media company, Hulu occupies the third place in leading video subscription services in the United States, by number of subscribers. Hulu leverages 3 content delivery networks: Akamai, Level 3 and Edgecast. Akamai primarily hosts the static content, specifically js(javascript) objects that are most frequently used, while Level3 and Edgecast are used to host the media content in mp3 and mp4 formats.

4.2 Choosing Edge Servers

Discovering edge servers was the next step in our proposed approach, and determining the edge server that the request routing algorithm chooses at the moment and the other proxy servers that could be potential alternate choices. We had here posed the first challenge that we expected to face in terms of carrying out our requests in a distributed fashion from multiple locations so as to determine the maximum subset of edge servers. We had envisioned that the process would deal with deploying our script containing the http requests on VMs at multiple geographic locations. To achieve this, we considered several options. The two most viable options were using PlanetLab [7] Nodes and using the WebPageTest [8] tool.

We chose WebPageTest as the primary instrument for data collection for the following reasons:

(1) Ease of use: The process of using WebPageTest appeared to be simpler, in the sense that there was no extra processes required to add locations or set up any other infrastructure.
(2) APIs: Since we were planning to undertake repeated tests (to avoid skewness of results), from multiple locations, with different objects at different points of time, the APIs exposed were helpful in automating the tasks.
(3) Variety of metrics: The most important reason behind our choice of WebPageTest was its ability to return data in terms of a variety of metrics including Time-to-First-Byte (TTFB), DNS lookup time, server round-trip times, etc.

We then use several of our python scripts to leverage WebPageTest API's for data collection. WebPageTest was originally developed by AOL for internal use. It is open sourced under a BSD license. The platform consists of a web UI using PHP and an agent that runs tests on browsers at different locations across the world. The main agent is named wptdriver and supports multiple browsers. The main components of the file are:

(1) The wptdriver : This is the main executable that gets launched on receiving a request. It a) polls the server for work, b)launches the browse, c) injects the scripts or issues the URL request, d) reports the metrics, and e) updates as required
(2) Wpthook: This is the main code that gets injected into the browser, mimics the actions required to request a web page or an object. It runs on port 8888.
(3) Wptwatchdog, wptglobal and wptupdate are the other components

4.3 Finding which request routing algorithm a CDN uses

Request routing algorithms are proprietary and hence cannot be determined by just measurement.

4.4 Starting data transfer and Observation

- After determining WebPageTest as our instrument of data collection, we wrote several python scripts that took as input a) the URL of the web object that we wanted to fetch from the servers, b) the location from which we wanted to fetch the object and c) the metrics we wanted to extract like TTFB and DNS lookup time.
- One of the significant challenges we faced, during the process of data collection was dealing with unauthorized access. When we deploy our scripts from browsers at different locations of the world, we needed a way to also pass our authentication parameters, so that we could avoid receiving HTTP error code 403 (forbidden). Hence, to also pass on our credentials, we used two approaches.
  (1) Appending session token to the URL: We used this approach for collecting audio data, since the range of bytes of audio are hosted on servers whose only means of authentication is comparing the authentication tokens received from the customer and the content provider. However, we observed that the session keys expired after a certain time period (in hours).
  (2) Injecting Java Script with login and password details.
- We collected the IP addresses of the edge servers by requesting either audio or static content along with TTFB times.
- Once we had the list of servers, we needed to determine if the request routing algorithm was efficiently choosing the correct edge server, based on the minimum TTFB time which represented network proximity, network conditions and server load. To do so, we needed to first determine if the returned edge server's TTFB was indeed optimal.
- The next step in the order of things, was to force our request to fetch data from all servers, instead of just one and note their respective times. To do so, we injected another Java Script, that would override the Domain Name System mapping for the domain name to the IP address specified by us. We repeated the process for all the servers in each of our lists.
- We then obtained the optimal server for each location based on minimum TTFB times. The final step was compare the difference between the server chosen by the request routing algorithm and the optimal server observed, and to analyze the trends observed in the data.

5 DATA COLLECTED

We had automated the task of fetching data from the following locations in Europe and locations in North and South America over several time periods in a day:

- *Europe*: Amsterdam, Berlin, Falkenstein, gce-europe, Italy, London, Maidenhead, Poland, Prague, Spain, Stockholm and Strasburg.

- *America*: Argentina, Chicago, Colorado, Dulles, ec2-sa-east-1, ec2-us-east1, ec2-us-west1, Florida, gce-us-west1-linux, MinasGerais, Nebraska, New Jersey and Texas.

After several runs of our scripts, we were able to collate or data into server lists, one each for Akamai and Level-3, for static and audio content respectively:

- Akamai, America
- Akamai, Europe
- Level 3, America
- Level3, Europe

We discovered *11* Level 3 and *11* Akamai edge servers in Europe. And *15* Level 3 and *19* Akamai edge servers in North and South America.

6 RESULTS

Analysis of the collected data was based on the metric : Time to First Byte (TTFB). TTFB is the time spent waiting for the initial response. This time captures the latency of a round trip to the server in addition to the time spent waiting for the server to deliver the response [4]. We think that the TTFB is a good metric to use, given that it depends on the round trip time indicating network conditions and server response which indicates the server load. Both these factors are important to the request routing algorithm in choosing an appropriate edge server.

Another important note is that we were unable to trace ping times, since our Internet Control Message Protocol (ICMP) packets were dropped indicating some sort of access-control by the content delivery networks.

After running our script several times, we observed that the below set of server's in Figure 1 and Figure 4 remained mostly constant, indicating the set represented a good estimate of Level 3's servers being used by Hulu. We then sketched a bar plot (Figure 2) and a line plot (Figure 3) for Level 3 CDN which hosted audio objects in Europe.

Location	RR IP address	RR TTFB (ms)	Opt IP Address	OPT TTFB (ms)
Amsterdam	8.253.157.244	142	8.250.87.248	135
Berlin	8.250.103.245	161	67.24.143.244, 8.253.140.106, 8.250.103.245, 8.250.89.243	160
Falkenstein	67.24.139.24	131	67.24.139.244	130
gce-europe	8.250.87.248	148	8.253.157.244	143
Italy	67.24.143.244	156	67.24.139.244	151
London2	8.252.29.244	141	8.252.8.244	135
Maidenhead	8.252.29.24	145	8.250.87.248	138
Poland	8.250.89.243	175	67.24.139.244	167
Prague	8.253.140.106	165	67.24.143.244, 8.253.156.110	154
Spain	8.253.156.110	178	8.253.140.106, 8.250.103.245, 67.24.139.244	165
Stockholm2	8.252.8.244	174	67.24.143.244	166
Strasburg_FR	8.253.140.10	156	8.252.8.244	152

Figure 1: Table containing locations in Europe, the edge server IP address chosen by the RR algorithm, its TTFB, the optimal edge server observed, and its corresponding TTFB for the audio object

We also sketched a bar plot (Figure 5) and line plot (Figure 6) for Level 3 CDN which hosted audio objects in America.

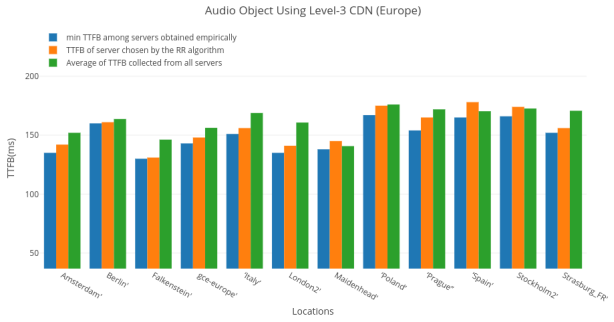

Figure 2: Bar Plot for an audio object using a Level 3 CDN for locations across Europe. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

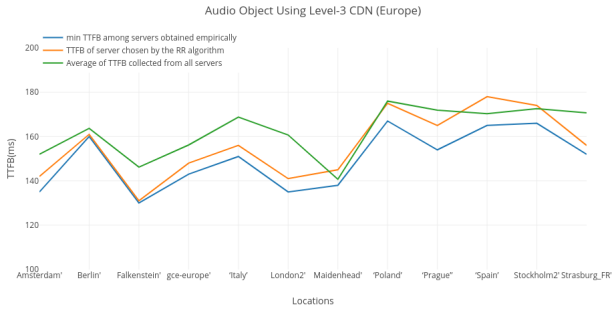

Figure 3: Line Plot for an audio object using a Level-3 CDN for locations across Europe. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

Similar data was collected for a javascript file (static content) which is present in the homepage of Hulu. We repeated our experiments with this web object and obtained a list of servers and sketched bar and line plots for both Europe and America. Figure 7 and Figure 10 represents the IP addresses of the edge servers in Europe and America respectively. Figures 8, 9, 11, 12 represent the various bar and line plots for static objects across Europe and America.

Following is the analysis for Level 3 CDN request routing:

- Level 3 CDN's request routing algorithm does not have a good match with the optimal server. For Europe, the match rate is 0, while in America the average match rate is <50%.

Location	RR IP address	RR TTFB (ms)	Opt IP Address	OPT TTFB (ms)
Argentina	8.252.88.116	183	8.252.16.244	181
Argentina2	8.252.16.244	191	8.253.165.238	186
Chicago Chrome	8.252.10.116	180	8.252.10.116	179
Colorado	8.253.148.236	59	8.253.148.236	57
Dulles	67.26.241.244	179	67.26.241.244	163
ec2-sa-east-1	8.253.165.238	176	8.252.85.116	174
ec2-us-east1	8.249.225.243	90	8.249.225.243	71
ec2-us-west1	8.253.129.177	79	8.253.129.177	82
Florida	8.27.231.244	71	8.27.231.244	67
gce-us-west1-linux	8.253.133.238	75	8.253.133.238	74
MinasGerais_BR	8.252.85.116	196	8.252.16.244	194
Nebraska	8.253.197.45	117	67.24.197.248	97
New Jersey	8.250.99.243	71	8.250.99.243	71
San Francisco	8.27.231.244	144	8.253.129.177	82
Texas	67.24.197.248	71	67.24.197.248	70
Toronto	8.252.103.116	194	8.252.10.116	179

Figure 4: Table containing locations in America, the edge server IP address chosen by the RR algorithm, its TTFB, the optimal edge server observed, and its corresponding TTFB for the audio object

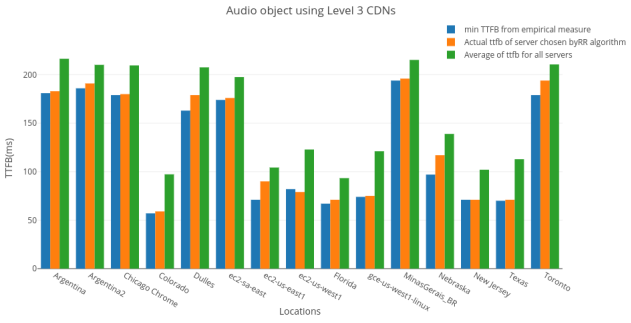

Figure 5: Bar Plot for an audio object using a Level 3 CDN for locations across America. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

The redirection, we guess maybe based on network proximity, rather than a combination of network conditions, proximity and server load.

- There is a high variation in the TTFB times among locations in America, and less variation in TTFB times among locations in Europe.
- Although the chosen server may not be optimum, it has better TTFB than the average over all other locations.

Following is the analysis for Akamai CDN request routing:

- Akamai's request routing algorithm had a much better match rate as compared to Level 3. The mismatch rate was always < 30%, ranging around 25%, with the mismatch seen over fixed locations only.
- Similar to Level 3 CDN if there was a mismatch, the TTFB time was still much better than the average rate.
- As we ran the experiment several times, we found that unlike Level 3, Akamai frequently updated its choice of edge server.

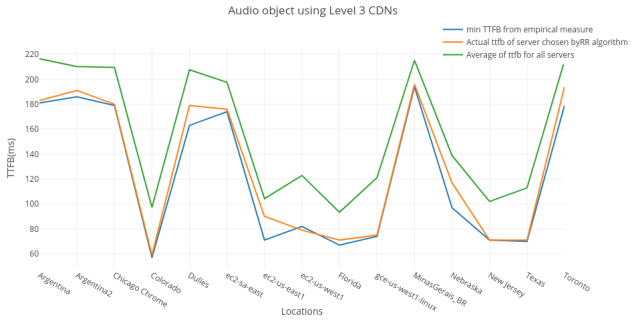

Figure 6: Line Plot for an audio object using a Level 3 CDN for locations across America. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

Location	RR IP address	RR TTFB (ms)	Opt IP Address	OPT TTFB (ms)
Amsterdam	69.192.70.38	37	69.192.70.38	35
Berlin	104.86.40.25	35	104.86.40.25	37
Falkenstein	104.111.244.87	38	104.111.244.87	38
gce-europe-west1-linux	23.216.206.221	44	69.192.70.38	43
Italy	104.83.156.69	35	104.83.156.69	36
London_EC2	23.214.189.19	36	23.214.189.19	38
Maidenhead	23.214.189.19	37	23.214.189.19	39
Poland	104.81.97.152	35	104.81.97.152	35
Prague	104.103.109.11	36	104.103.109.11	37
Spain	104.126.95.88	42	104.126.95.88	42
Stockholm2	2.23.131.158	35	2.23.131.158	35

Figure 7: Table containing locations in Europe, the edge server IP address chosen by the request routing algorithm, its TTFB, the optimal edge server observed, and its corresponding TTFB for the Static Object.

This leads us to believe that the change may have been due to changing network conditions which is again correctly reflected in the TTFB times. We also observed a variation in TTFB times with respect to different times of the day. Analyzing this trend and verifying optimal request routing utilization at different points of a day could indicate user behavior and a CDN's efficiency. This could studied more in the future.

7 FUTURE WORK

- Although we believe that the TTFB was a significant metric in estimation of an optimal server, we believe that using additional metrics (like download complete time, which may also be indicative of a server's capabilities) and modeling the effect of these metrics, would help us choose an uncontested optimal server.

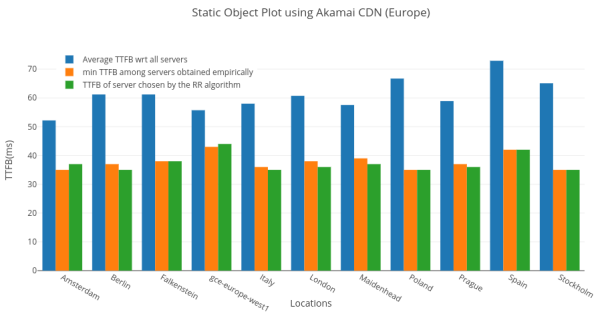

Figure 8: Bar Plot for a static object using a Akamai CDN for locations across Europe. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

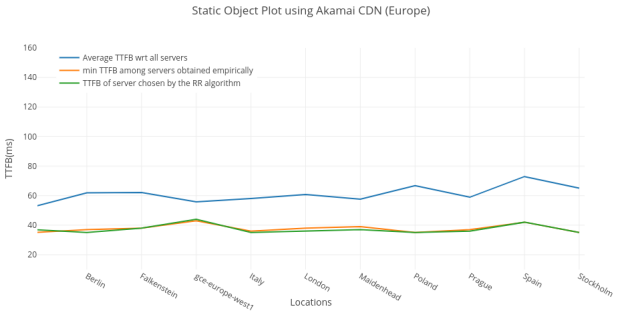

Figure 9: Line Plot for a static object using a Akamai CDN for locations across Europe. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

- We would also like to have analyzed the trends of user behaviour and its effects on optimal CDN utilization.
- Lastly, we were extremely fascinated with the key exchange mechanism between the three parties involved: the content provider, the content delivery network and the user, and how these affect Digital Rights Management in an increasing video-on-demand world.

8 SUMMARY

The implications of the above analysis to the question posed by us at the start of the paper are as follows:

- Given that the number of internet users who watch video is growing at a rapid rate, and the quality of experience of an user is affected by slower page loads, delay in video starts and interruptions, it becomes imperative for content providers to provide a seamless experience to the users and content delivery networks play a pivotal role in this [4]. We have analyzed a couple of content providers and the trend is to use multiple content delivery networks for data hosting and distribution.
- From our analysis of two content delivery networks we observed that Akamai's request routing algorithm performed significantly better than Level 3's in relation to Hulu's content.
  - Akamai's low mismatch rate indicates to us that Akamai must be using an adaptive algorithm that takes as parameters, network proximity (path length), network conditions and server load (client-server latency). Akamai also uses DNS-redirection based request routing, but since the DNS time for redirect would be similar for a particular VM, we can say that our data is free from DNS time bias.

Location	RR IP address	RR TTFB (ms)	Opt IP Address	OPT TTFB (ms)
Argentina	72.246.215.93	87	104.104.166.91	60
Argentina2	23.197.232.59	36	23.197.232.59	36
Chicago Chrome	23.63.203.42	67	23.63.203.42	68
Colorado	184.29.152.237	52	184.29.152.237	53
Dulles	104.119.136.151	67	104.119.136.151	67
ec2-sa-east-1	104.104.166.91	36	104.104.166.91	36
ec2-us-east1	104.117.22.193	35	104.117.22.193	36
ec2-us-west1	172.231.28.66	36	172.231.28.66	36
Florida	23.62.169.196	41	23.62.169.196	42
gce-us-west1-linux	23.195.239.200	49	23.195.239.200	42
MinasGerais_BR	173.222.84.211	54	173.222.84.211	56
Nebraska	96.7.86.249	74	96.7.86.249	72
New Jersey	104.106.243.165	37	104.106.243.165	43
Texas	172.230.208.91	38	96.7.86.249	35

Figure 10: Table containing locations in America, the edge server IP address chosen by the request routing algorithm, its TTFB, the optimal edge server observed, and its corresponding TTFB for the static object.

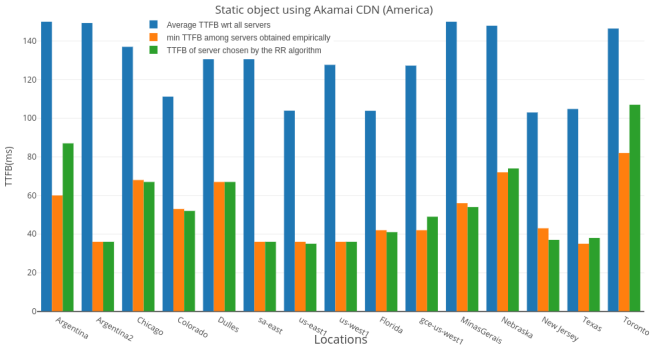

Figure 11: Bar Plot for an static object using a Akamai CDN for locations across America. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

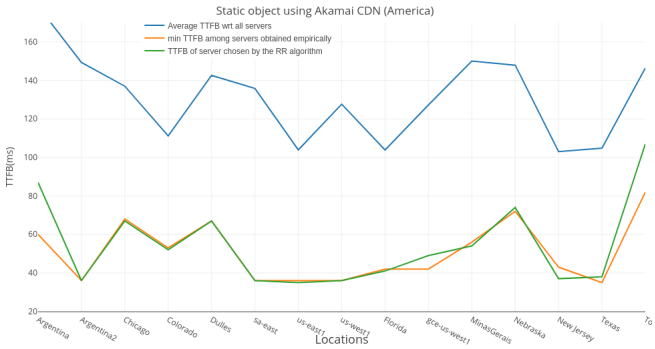

Figure 12: Line Plot for a static object using a Akamai CDN for locations across America. The blue trace represents overriding the DNS and finding the optimal server with minimum TTFB. The orange trace represents the actual server TTFB (that is the server chosen by the request routing algorithm). The green trace is the average of all the TTFBs obtained by DNS override.

  - Level 3 does not effectively track changes in network conditions. In fact, as per our observation, for each video we played, for the entire duration of the period, the choice of the edge server did not change once it had been initialized. This tells us that Level 3 must be using either a non-adaptive request routing algorithm (based on a simple round-robin mechanism) or an adaptive algorithm that simply considers path length as the only metric.
- Skewness of data: In Spite of our attempt to avoid any sort of skewness of the data, we have to note the fact that for some locations, there were test queues, in which there may have been certain amount of delay between the time at which the edge server was chosen by the request routing algorithms and the time at which our request was sent to the other servers in the list. However, even for the busiest places on our list, the window was less than 20 minutes.

REFERENCES

[1] Al-Mukaddim Khan Pathan and Raj Kumar Buyya , “A Taxonomy and survey of content delivery networks”, in Content Delivery Networks, Chapter 2, Springer Press, ISBN 978-3-540- 77886-8.

[2] Bartolini N., Casalicchio E., Tucci S. (2004) A Walk through Content Delivery Networks. In: Calzarossa M.C., Gelenbe E. (eds) Performance Tools and Applications to Networked Systems. MASCOTS 2003. Lecture Notes in Computer Science, vol 2965. Springer, Berlin, Heidelberg

[3] Md. Humayun Kabir, E. G. Manning, G. C. Shoja, Request-routing Trends and Techniques in Content Distribution Networks, International Conference on Computer and Information Technology (ICCIT), , 315-320, 2002.

[4] V. K. Adhikari et al., "Unreeling netflix: Understanding and improving multi-CDN movie delivery," 2012 Proceedings IEEE INFOCOM, Orlando, FL, 2012, pp. 1620-1628. doi: 10.1109/INFCOM.2012.6195531

[5] Sandeep Kath, Manoj Kumar and Ajay Sharma, “CDN DNS - An Efficient DNS Request Routing Technique in Content Delivery Networks” International Journal in Advances in Computational Sciences and Technology , ISSN 0973-6107 Vol 3 Number 2 (2010) pp. 147-154.

[6] A.-J. Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante,“Drafting behind Akamai: Inferring network conditions based on CDN redirections,” IEEE/ACM Trans. Netw., vol. 17, no. 6, pp. 1752–1765, Dec. 2009.

[7] *PlanetLab*, An open platform for developing, deploying and accessing planetary scale services.

[8] *WebPageTest*, Website Performance and Optimization Test.