Resource Allocation Techniques for CDN Latency Reduction.

Christina Pavlopoulou
cpavl001@ucr.edu

Niloufar Hosseini Pour
nhoss003@ucr.edu

Andres Calderon
acald013@ucr.edu

June 9, 2016

Abstract

CDN goal is to serve content to end-users with high availability and high performance. In order to do that, locations for new CDN server deployment should be carefully analyzed. In this project, we try to apply well-known resource allocation techniques to find the optimal locations to deploy new CDN servers. We collect RTT data between different cities in North America and hypothetical current CDN servers. We find the areas with the lowest time response and propose them as potential new locations for CDN server deployment.

1 Introduction

Internet had experienced huge changes in the last decades. Content and user have increased exponentially each year. The quality of service and the global access to information lead to content providers to search for mechanisms in order to improve the experience of their current and potential users.

Content delivery networks aim to reduce the time access of specific contents by bringing them closer to the user (figure 1[1]). Different providers around the world offer both commercial (Akamai and Limelight) and open source (CoDeeN) solutions [1].

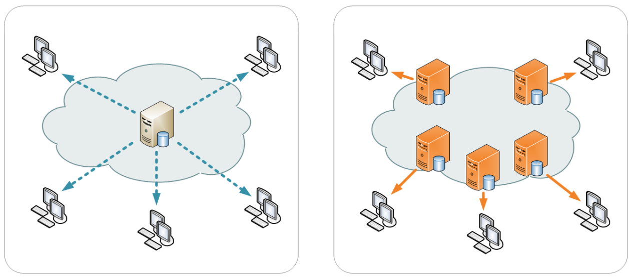
Figure 1: CDN example.

With the constant increase of contents, one of the most important concerns of CDN providers is where to deploy a new server. In this project, we implemented a resource allocation technique to minimize the time response of candidates and proposed an ordered list of possible locations.

[1] https://commons.wikimedia.org/wiki/File%3ANCDN_-_CDN.png

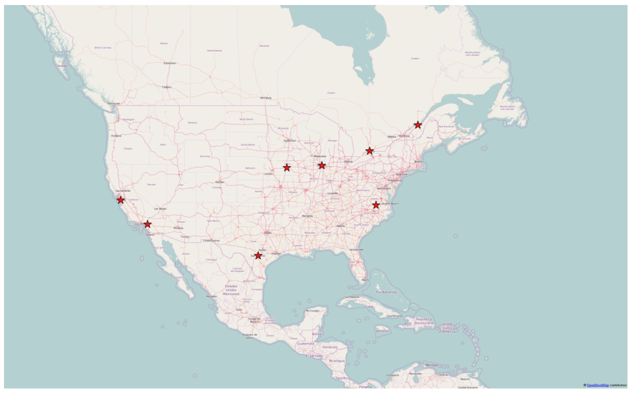
Figure 2: Current CDN servers.

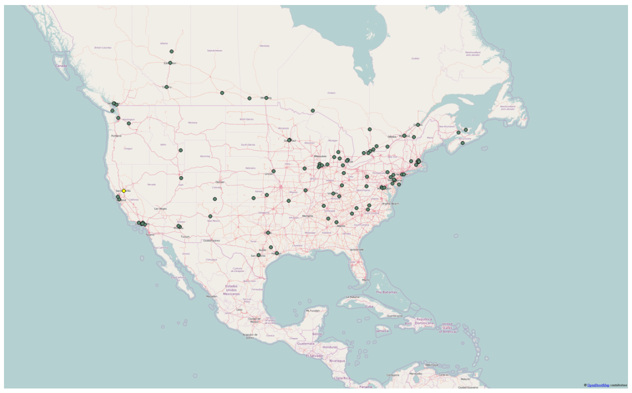
Figure 3: Candidates location.

2 Methodology

Our methodology consists of five steps. First, we had to simulate a number of current CDN server and then collect a sample of possible candidates for future deployments. Next, we had to measure the RTT response between our current servers and the candidates. Finally, we used the collected data to built a graph and analyzed the performance of the candidates. The following sections will provide more details for each of the steps.

2.1 CDNs Collection

We used Microsoft Azure: Cloud Computing Platform & Services to collect current CDN servers. We focused on USA and Canada and allocated servers on the following regions: East USA, Central USA, South Central USA, West USA, North Central USA, East Canada, Central Canada and Riverside. Figure 2 shows the aforementioned locations.

2.2 Candidates Collection

For the candidates we used the Universities Worldwide webpage [2] to collect a sample of university urls in the area of study. We used python code to parse the web page and extract the url addresses and also for geocoding the real location of the url. We started with 200 urls but after processing them we concluded with 97 valid urls. The cleaning process consisted of removing duplicates, incorrect formats and those that did not response to a ping command. Figure 3 presents the locations of the candidates.

2.3 Measuring RTT Metric

We developed a set of python scripts to run ping commands between the current servers deployed on the Azure network and the set of candidates located in different cities. We transmitted 5 packages and computed the average ping time. We collected a file from each current CDN server and then we merged them in order to build a graph.

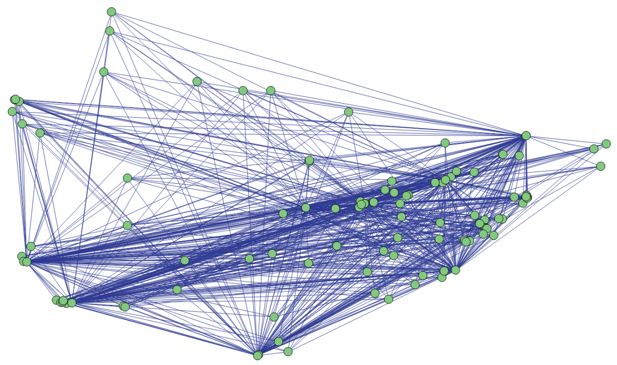

Figure 4: Graph representation.

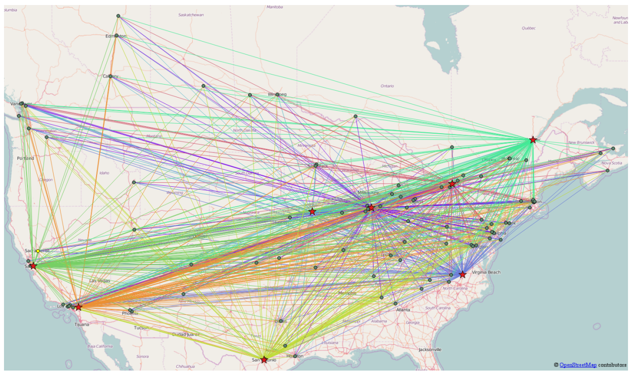

Figure 5: Graph representation.

2.4 Building the Graph

To implement our graph, we used as nodes the current and candidates CDNs and as edges the ping time between them. We used the NetworkX python package to build the graph. The final graph has 105 nodes and 776 edges. Figure 4 shows a representation of this graph.

2.5 Visualization

We exported the graph using a `shapefile` format. It is a well-known format used for many GIS applications to visualize geographic data. We used QGIS [3] to illustrate the location of current and candidates CDN in the study area. Figure 5 shows this visualization.

City	State	Avg Ping Time
Clarksville	TN	71.19775
Bloomsburg	PA	69.86825
Athabasca	AB	57.35125
Philadelphia	PA	56.912875
Charlottetown	PE	56.19075
Chicago	IL	54.007875
Smithfield	RI	52.443
Langley	BC	52.4085
Thousand Oaks	CA	50.99025
Boston	MA	50.548125

Table 1: Top 10 best locations.

3 Results

To identify the candidate with the higher time response, we iterated at each candidate in the graph and computed its average ping time with each of the current CDNs. Finally we ordered the list of candidates ascendantly according to the average ping time. Figure 6 shows the location of the two highest ping time response and probable future locations for a new CDN deployment. The two cities that we propose are Clarksville (Tennessee) and Bloomsburg (Pennsylvania). We propose these two locations given that their average ping time was very similar in comparison with the rest of possible candidates

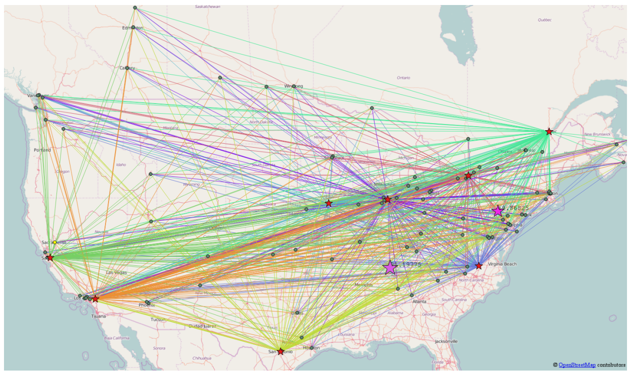

Figure 6: Output of test 2.

(see table 1).

4 Conclusions

In this project we present a methodology to identify CDN optimal locations using resource allocation techniques. We end up with proposing two possible locations due to their high latency. We observed that the actual distance between the current and candidates CDNs does not influence the results since cities located far away from current CDN servers still present better ping time. Another metric that could further improve our results is the standard deviation since first packages that we transmit have always higher ping time. Thus, the average ping time that we get may not be a good representative of the actual relation between the nodes.

References

[1] Cheng Huang, Angela Wang, Jin Li, and Keith W. Ross. Measuring and evaluating large-scale cdns paper withdrawn at mirosoft's request. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, pages 15–29, New York, NY, USA, 2008. ACM.

[2] Universities worldwide. `http://univ.cc/`.

[3] Qgis: A free and open source geographic information system. `http://www.qgis.org/`.