Gerhard van Andel [*]

[*]Colorado University, CSCI/ECEN5273

ABSTRACT

The thirty year history of web crawling began in the days when a single book contained all the websites of the world and has advanced rapidly to a multi-billion dollar industry crawling millions of pages per day. [2] In the past few years, the growth of mobile internet usage has reshaped the way companies have designed and formulated their page-rank algorithms. For example in 2016, Google started working on mobile-first indexing, the crawling of the web using a smartphone Googlebot, forcing companies who want to stay competitive in the web market to adapt to these new habits of the 21st century internet traveler. The rules are not so much enforced but created superficially through Google and other companies' page ranking algorithm, the great and powerful placing of what links will be first in the modern age. Being at the top of a search can be the difference in getting business or being cast out to the depths of the second page. This project studies the web and load times across regions to try and simulate how user experience of a website can change depending on where you are.

General Terms

Search Engines, Information Retrieval, PageRank, User Experience

1. INTRODUCTION

Search engines typically look at the desktop version of a page's content to evaluate its relevance to the user, but in the last four years the growth of mobile devices has increased dramatically to around 50% of web traffic using mobile devices. So why does any of this matter? Sites must continue to grow their online presence while balancing content and load times across the globe. The last thing businesses want is for customers to complain on social media that the website is slow or unresponsive. Customer experience metrics are important to alert staff of production issues across websites just like any other service level indicator (SLI). An SLI is a carefully defined quantitative measure of some aspect of the level of service that provides an idea on the health of the service. An SLI may be as simple as a web page load time. If a website does not load within three seconds, the chance that the user will give up and go to a different page increases by 32%. [1] Additionally a slow response time can push a company's web site off the front page. Therefore, monitoring traffic and response time from servers is important data for web experiences across all industry sectors.

2. RELATED WORK

To be listed on the front page of a search, companies need to understand search engine optimization (SEO). SEO is often about making small modifications to parts of the website. When viewed individually, these changes could have a noticeable impact on a site's user experience and performance in the world of organic search results.

Websites are added to search engines by crawling pages which are triggered by many different methods. New web pages can be discovered when the web crawler follows a link from a known page to a new page. This new website can then lead to the reading of a sitemap normally linked from a robots.txt file that lives at the root of the site. Once a page is discovered, the cream of the search engine industry is figuring out what a page is about. Each search engine uses a different method to classify and index each web page on the internet. This can be improved by using page headers and content from the text paragraphs to the images and videos to convey content. Once the pages have been crawled and indexed into the search engine then a user is able to search for something. [3]

An example is a user searching for a place to buy cat food, which requires the user to provide the location of the search to improve results as a user who lives in Boulder would not need results of cat food stores in Iceland. This is just the start to find the user's furry friend some food. Using response times the search engine can filter the fastest loading web pages to the top of the results providing a good user experience through the search engine. When the user clicks on the web page the search engine has provided, it will not fail to load or worse load very slowly.

3. PROBLEM CHARACTERIZATION

Speed is a crucial aspect in providing a pleasant user experience to visitors of any site. It ensures that a visitor's limited attention span and time are spent on actual content and not wasted waiting for images and scripts to load. Studies have shown that users will not tolerate more than a three second load time. [1] If a site fails to offer a quick response, users will leave, resulting in a loss of revenue. gMonitor is a simple way to monitor and crawl web pages to track the user experience.

4. METHODOLOGY

gMonitor takes a request that contains a url to search. The design of gMonitor is based on a micro-service architecture with message queues between workers. This allows each system to scale independently. The service then creates a task for the url and uses a producer consumer model so the rest interface places a message on the spider queue. Spider workers pull the website using the Python Requests library for this url and store it into Google Cloud Storage, creating a task for the scanner. The scanner picks up this message to load the web page from Google Cloud Storage and parses using BeautifulSoup. The scanner filters out included links that the web page contains creating additional tasks for a spider to work. Once this is complete, the scanner adds a message on the last queue for the cleaner. The cleaner worker helps reduce the footprint of the cache and pushes items that have been scanned and updated into Google Cloud Storage. If a user requests the data or the results of the scan, the controller can pull this data from storage with an increase of time to the call. This helps keep unused data off the cache and allows data that is being accessed more frequently to be on the cache. This design is visually represented in figure 1.

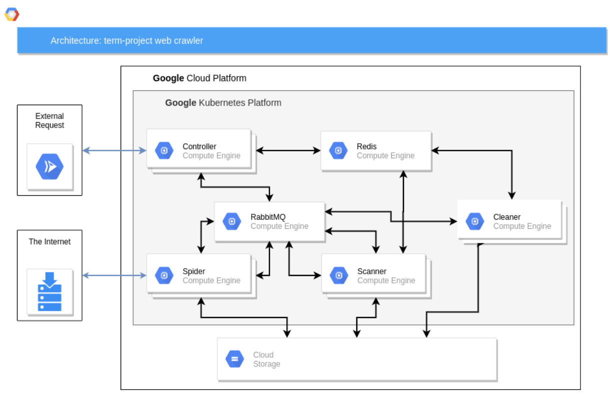

Figure 1: Architectural Diagram

Google Kubernetes Engine (GKE) provides a managed environment for deploying, managing, and scaling the containerized applications along with providing a simple interface to Google infrastructure like cloud storage. The same template is used in three regions, United-States central, Europe central, and East Asia. The requests were kept similar in these three regions, however, the nature of the distributed system did not allow for control of the queues and what jobs were being worked.

5. MILESTONES

5.1 Checkpoint 1

The first step of gMonitor was to make a working prototype pulling http/https response times based metrics. Python Requests library was used to pull data from a website, then BeautifulSoup library was used to parse the website, to pull all the links from the site and follow the links to determine site speed on leaf sites. The worker used Redis as a distributed lock manager, locking the domain so only a single worker could request a web page from that domain at a time. The worker then stored this web page data into Google Cloud Storage. To test this ecosystem docker-compose was used which allowed faster changes locally. This increased productivity as changes no longer had to be pushed to the docker registry.

The Redis database worked well but can be filled very fast as the crawler grows exponentially. To reduce the number of tasks in the database by pushing completed jobs back into cloud storage, and on request, load it out of cloud storage. Link information took up more space than expected, as gMonitor crawled 70 web pages, linking to more than 1200 other pages, resulting in a need to figure out where to store information in a scalable manner.

5.2 Checkpoint 2

While working through the small problems for database locks, Redis documentation revealed set operations have the option of an expiry. A domain based entry will remove the key value pair in cache after the expiration. This reduced the domain database from growing very large. Domains are generally grouped as links from one page link to pages within the same domain. After the domain would expire from the database, an updated version of the robots.txt is obtained with a new request to the domain. A deploy in the middle of a search could result in a deadlock on a message, so the expiry of the domain data ensured the release of the domain lock. The domain locks and RabbitMQ created a separate issue such that when a message is rejected from a worker it should then put the message at the back of the queue. Instead, if a domain lock was being used, the message would be rejected and then placed in the front of the queue, only to get picked back up right away.

6. INSIGHTS GLEANED

The first test was run in the US central region. The popular website news.ycombinator was used for the data set. This website links to many different types of websites and uses a voting system to bring favorite / popular content to the top of the front page. This served as a baseline as to what to expect from the other regions as shown in figure 2.

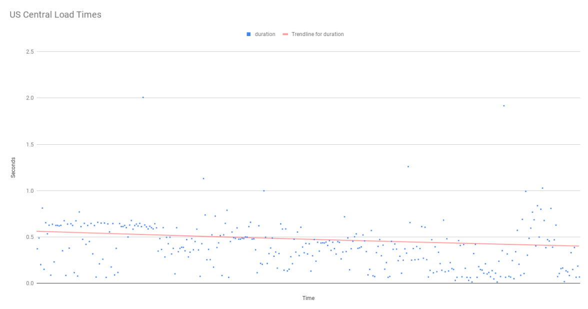

Figure 2: Central United States Load Times Graph

Surprisingly, the load times in the Europe central region were not so different from the us central region. Assuming that news.ycombinator uses some sort of content delivery network (CDN), the same data can be easily provided in the European region. Another assumption is that the website probably has more active users as English is a common language in both the US and Europe. Europe response time is shown in figure 3.

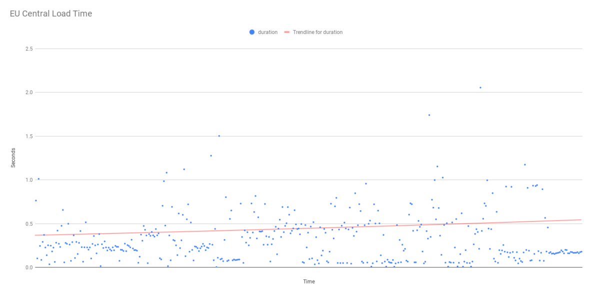

Figure 3: Central Europe Load Times Graph

The response times for the East Asia region were slower than in the previous two regions, though not so different as to cause alarm. The same assumptions made in Europe are made here. With this the request duration's were above average compared to the United-States and Europe as shown in figure 4.

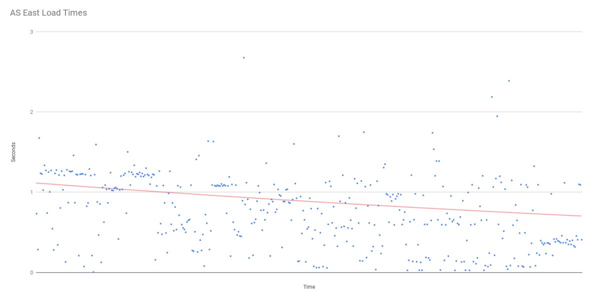

Figure 4: East Asia Load Times Graph

7. FUTURE WORK

To solve the message loop problem in future iterations, a priority queue could be used as it allows the priority of a message to be lowered and tracks the number of times the system sees a message. Offloading completed requests from the Redis job database into cloud storage also reduced the memory footprint of the database as a small amount of jobs are pulled after they complete. Google's big-table service was considered in place of Redis, but found that redesigning the platform for the use of big-table required more work than time provided. Additionally, the scanner could be improved when loading content from a website to include the loading of other content such as style sheets, images, and javascript.

With more time, future iterations of gMonitor would have implemented the current and future industry standards in monitoring speed for web pages, such as PageSpeed from Google and YSlow from Yahoo. When a web page is profiled with PageSpeed, it evaluates the page's conformance to a number of different rules. These rules are general front-end best practices that can apply at any stage of web development. However there are differences in how YSlow and PageSpeed perform the calculations. Each algorithm analyzes a page using a set of rules that they believe are most relevant to page speed and performance. Most of the rules overlap or are very similar to each other, but in general scores should be comparable. gMonitor would also include some of the new advancements from increased usage of HTTP/2.

gMonitor could start collecting the location of the remote server along with using troubleshooting tools such as traceroute to better explain the result obtained.

8. CONCLUSION

The response times varied across regions, but the range was not as different as expected. With future iterations gMonitor could expand to allow for a better explanation of the routes and locations of the servers that responded to the request. All of this data can be used to improve the web browsing experience for both companies having to maintain the systems and users just search for some cat food for their furry friend.

9. REFERENCES

[1] D. An. Think with google, February 2018.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. 1998.

[3] B. Jones, B. Rogan, C. Stahl, D. Creager, H. V. Madhyastha, I. Grigorik, J. E. Tuttle, L. Chen, M. Efimov, P. Papageorge, and S. Burnett. Network error logging: Client-side measurement of end-to-end web service reliability. In *17th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2020*, 2020.