Characterizing Roles of Front-end Servers in End-to-End Performance of Dynamic Content Distribution

Yingying Chen, Sourabh Jain, Vijay Kumar Adhikari, Zhi-Li Zhang
Department of Computer Science & Engineering
University of Minnesota - Twin Cities
Minneapolis, MN 55414, USA
{yingying,sourj,viadhi,zhzhang}@cs.umn.edu

ABSTRACT

This paper investigates the roles of front-end (proxy) servers in improving user-perceived performance of *dynamic* content distribution. Using Bing and Google search services as two case studies, we perform extensive network measurement and analysis to understand several key factors that affect the overall user-perceived performance. In particular, we develop a simple model-based inference framework to indirectly measure and quantify the (directly unobservable) "frontend-to-backend fetching time" comprised of the query processing time at back-end data centers and the delivery time between the back-end data centers and front-end servers. We show that this fetching time plays a critical role in the end-to-end performance of dynamic content delivery.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: Performance attributes

General Terms

Measurement, Performance

Keywords

Dynamic content distribution, Search service, TCP-splitting

1. INTRODUCTION

More and more content on the Internet is now stored at powerful, large-scale data centers in the cloud. A significant portion of this content is *dynamic* in that in response to a user's request for content, the content returned to her is generated dynamically and sometimes personalized. Web search is one common example of such dynamic content generation. With the emergence of cloud computing and cloud-based services, we expect that more data will be stored in the cloud, and more dynamic content will be generated on the fly in response to user requests. Because of the sheer scale and cost of building and operating large-scale powerful data centers, their number is few and far between. Hence they are generally far away from a large majority of users.

One way to mitigate this effect and improve the user-perceived performance (e.g., the overall response time) is to deploy "proxy" servers – hereafter we refer to them as *front-end* (FE) servers – closer to users. The usefulness of such an approach for *static* content distribution (e.g., video) is obvious because of *content caching*. FE servers can also be exploited to improve the *user-perceived* performance of *dynamic* content distribution due to the following two key aspects [9,11]: i) a portion of the (dynamic) content may be static; thus can be cached and delivered immediately from the FE servers; and more importantly, ii) via *split* TCP connections, a FE server can establish a *persistent* TCP connection with the data center which not only eliminates the effect of TCP slow-start congestion window ramp-up on the throughput of the TCP connection between the FE server and the back-end (BE) data center, but also mitigates such effect on the throughput of the TCP connection between the user and the FE server (due to the reduced RTT). Nonetheless, the *overall* user-perceived *end-to-end* performance likely depends on a confluence of various factors such as RTT, loss rate and throughput of the connections between users and FE servers and between FE servers and BE data centers, the load on FE servers, the processing time at BE data centers to generate user-requested dynamic content, and so forth.

To investigate the roles of FE servers in improving user-perceived end-to-end performance of *dynamic* content distribution, we conduct an active measurement-based *comparative* study of Google and Microsoft Bing web search services. Both services utilize a number of FE servers that are placed closer to users to assist *dynamic* content (i.e., search results) distribution: Google deploys a set of its own FE servers, whereas Bing relies on Akamai's content distribution network (CDN). Using the PlanetLab nodes, we perform extensive measurements of Google and Bing search services by emulating and generating a variety of keyword search queries of varying popularity, granularity and complexity, and collect a large amount of dynamically generated content and application-layer measurement data. Through *content analysis* and *temporal clustering* of packet-level events, we confirm that both Bing and Google search results contain a *static* portion, such as the HTTP header, HTML header, etc., which is cached and delivered immediately by the FE servers upon receiving a user request. As the effect of the aforementioned second key aspect cannot be *directly* measured, we develop a novel *model-based* inference framework: we classify and separate the content (i.e., search results) into

IMC'11, November 2–4, 2011, Berlin, Germany.

two parts – the *static* portion that is cached and directly delivered by FE servers, and the *dynamic* portion that is generated by the BE data centers and then passed onto the FE servers for delivery. We define several *directly measurable* parameters to characterize and *predict* the delivery performances of static and dynamic portions. These predictions are indeed borne out by our measurement data, and thus enable us to deduce that despite that one utilizes a third-party CDN (as FE servers) and the other does not, both Bing and Google employ FE servers in a similar fashion.

Furthermore, our inference framework allows us to *bound* the *FE-BE fetch time* (T_{fetch}) of dynamic content – namely, the overall time it takes for a FE server to forward user query to a BE data center, for the data center to dynamically generate the user-requested content and deliver it to the FE server – we note that *this time cannot be directly observed and measured at the end systems*. Comparing Bing and Google search services, we find that the fetch time between Google FE servers and BE data centers tends to be smaller and more stable for Google; in contrast, the fetch time between the Akamai FE servers and Bing data centers tends to be larger and shows higher variability. Hence, despite Akamai FE servers are generally placed closer to users (and their number is larger than that of Google FE servers), user-perceived performance of the Bing search service tends to vary significantly from queries to queries. While it is known that placing FE servers closer to users can generally improve the user-perceived performance (e.g. [9]), our study demonstrates a critical trade-off between placement of FE servers and the FE-BE fetch time which limits this improvement: there is a *distance threshold* within which placing FE servers further closer to users is no longer helpful; instead, the end-to-end performance is now determined solely by the FE-BE fetch time. Thus, to improve the end-to-end performance, it is also crucial to optimize the FE-BE fetch time. Lastly, we develop heuristics to factor the FE-BE fetch time so as to estimate the back-end processing time (T_{proc}) and the BE-FE round-trip delivery time (RTT_{be}) separately.

Related Work. Most prior works in this area focus on understanding the distribution of static content. For instance, in [5], authors study the assignment of clients to the CDN edge servers, in order to maximize the performance for each user. Similarly, several other studies such as [4, 6, 7] develop techniques to use a peer-to-peer based model to distribute the content, which is assumed to be static. In [10] authors study the various caching mechanisms used by CDN networks. Besides, a recent OSN study [12] show that placing more proxy servers can enhance the content distribution for Facebook users sharing similar interests. However, the focus of the study was to exploit the redundancy in the data accessed by users in a given geography, and reduced the user-perceived delay by caching the content at nearby proxy servers. A study [9] that is more closely related to our work compares the performance of cloud service with and without tcp-splitting, and therefore dealt with only the indispensability of TCP-splitting. On the other hand, in this paper, by reverse engineering the strategies used by Google and Bing to distribute the dynamic content, we shed light on the trade-offs among different underlying factors in designing TCP-splitting for dynamic content distribution.

2. PROBLEM SETTING & A SIMPLE MODEL

In this section, we describe the basic infrastructure with FE servers for dynamic content distribution, and present a simple abstract model to capture the interactions between users and FE servers and between FE servers and BE data centers. This model will guide us in the measurement and analysis of dynamically generated search results from Bing and Google.

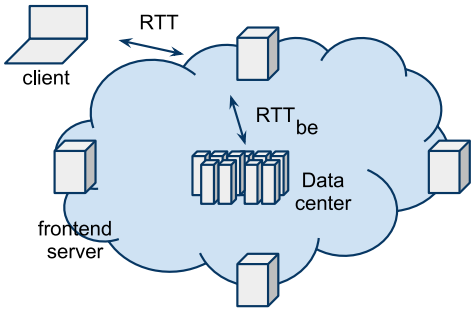

Figure 1: Content distribution infrastructure.

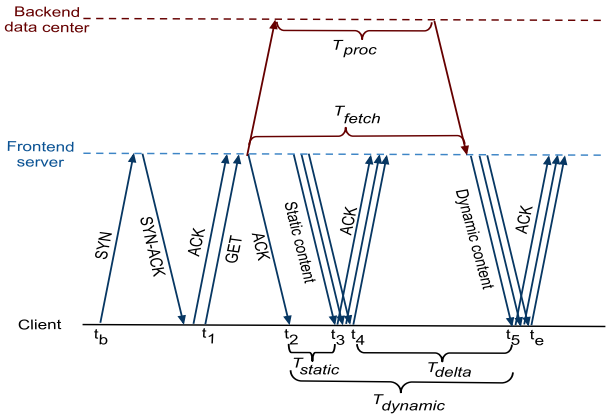

Figure 2: Modeling search query timeline.

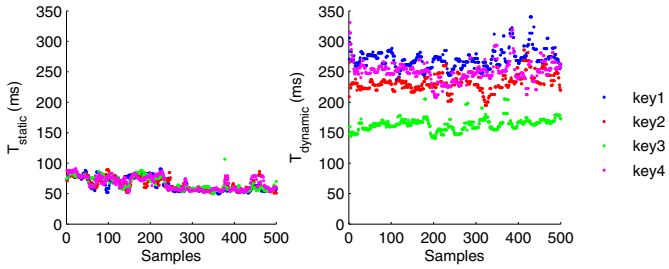

Figure 3: T_{static} and $T_{dynamic}$ for different keywords.

Figure 1 depicts a typical infrastructure set-up for dynamic content delivery consisting of FE servers that are deployed at the "edge of the cloud" (thus relatively closer to users) and BE data centers "deep in the cloud". When serving static content, FE servers often function as caches. When serving dynamic content, FE servers can play two key roles: i) they can cache certain portion of static content that is common to all dynamically generated content, and deliver it immediately upon receiving a user's request; ii) by splitting the end-to-end TCP connection, FE servers can establish persistent TCP connections with BE data center to

speed up the delivery of the dynamically generated content between them.

Unlike the static content distribution, there are several key factors affecting the user-perceived performance of dynamic content distribution: the latency or round-trip time (RTT), available bandwidth and loss rate between a user and a FE server, the load on a FE server, the latency or round-trip time, available bandwidth and loss rate between a FE server and the BE data center, the processing time at the data center to generate dynamic content in response to a user's request, the load on servers at the data centers, and so forth. Unfortunately most of these latter factors cannot be *directly observed and measured at the end hosts*. To address this, we develop a novel inference framework which allows us to indirectly measure and quantify the overall (search query) processing and delivery time between the data center and a FE server.

As shown in Figure 2, we model the packet-level generation and reception process and define several (*measurable*) parameters to capture the events of the *static* and *dynamic* portions of the content distribution. The process starts at t_b with a TCP three-way handshake[1]. At time t_1, a user (client) sends an HTTP GET request, and receives the ACK packet from the FE server after one RTT at t_2. At t_3 the client receives the first packet containing the *static* portion of the content, and at t_4 receives the last packet containing the static content. At time t_5, the first packet containing dynamic content is received, and at time t_e, the final packet of the entire content is received. The correctness of the model is validated in later sections, and is also quite consistent with the descriptions given in [8, 9, 11].

In our model, the time when the first and last packets of the *static* content portion are received should, *by definition*, hinge on the factors involving only the FE server and client, namely, they are independent of the BE data center. We define $T_{static} := t_4 - t_2 (= t_4 - t_1 - RTT)$ which bounds the processing and delivery of the static content portion *at the FE server side* (assuming a constant RTT). We define $T_{dynamic} := t_5 - t_2$ and $T_{delta} := t_4 - t_2$. Then $T_{dynamic}$ upper-bounds the overall fetch time T_{fetch}, while $T_{delta} := t_5 - t_4$ serves as a (potentially loose) lower-bound on this overall time. Meanwhile, T_{fetch} is mainly consisted of the time it takes for the FE server to send the user request to a BE data center, for the data center to process and dynamically generate a response (the *dynamic content* portion) to the user request, and deliver it to the FE server. Namely,

$$T_{delta} \leq T_{fetch} \leq T_{dynamic} \quad (1)$$

$$T_{fetch} = T_{proc} + C * RTT_{be} \quad (2)$$

where C is constant, which depends on the TCP window size on the BE data center. Moreover, fixing a FE server, T_{fetch} should be a constant, assuming the variability introduced by FE server and data center loads, the available bandwidth between them, etc., is small and negligible relative to the RTT between the FE server and a client. In other words, the time it takes for the FE server to receive the delivery of the *dynamically generated* portion of the content from the BE server is (roughly) a constant. On the other hand, the delivery time ($t_4 - t_3$) for the static content is a function of RTT. Hence our model predicts that as RTT increases, T_{delta} decreases. With sufficiently large RTT, $T_{delta} = 0$; thus the last packet of the static content portion and the first packet of the dynamic content portion will be delivered back-to-back or even coalesce as a single packet.

[1]DNS resolution time is not included, as it is negligible as compared to the overall user-perceived response time.

3. ACTIVE MEASUREMENT & CONTENT ANALYSIS

For our study, we develop an in-house user search query emulator, which performs exactly the same functionality as the web-based search box. We deploy the emulator on globally distributed[2] PlanetLab nodes as well as on our lab and home machines. The number of Planetlab nodes participating in each of our experiments ranges from 200 to 250, depending on their reachability at the time being. We conduct extensive measurements by submitting the same search queries to both Bing and Google search engines, and collect detailed TCPdump with full application-layer payloads. We perform two sets of experiments: 1) In the first set, search queries are launched from all measurement nodes to their *default*[3] FE servers every 10 seconds. 2) In the second set, we fix one FE server (of Bing or Google respectively) at a time, and launch queries from all measurement nodes to this server. We repeat these two sets of experiments using different sets of keywords and over different times. We refer to the data collected in the first set/type of experiments as *Datasets A*, and the second as *Datasets B*. Due to space limitation, we omit the details of the active measurement platform and experiment design and execution.

Parsing Application-Layer Packet Traces and Identifying the Static Content Part. Using the packet traces collected via TCPdump, we perform detailed application layer content analysis as well as transport layer temporal classification of packet generation and reception events. We find that in the search results returned by both Bing and Google, there is a portion of the content that is *static*, namely, independent of the search keywords submitted. This *static* content portion includes the HTTP header, HTML header, CSS style files, and the static menu bar, e.g. "Videos," "News," "Shopping," etc. that are placed on top of each search result page. The remaining *dynamic* portion includes the *keyword-dependent* dynamic menu bar, search results and ads. The temporal analysis of the packet-level events confirms that this static content portion is most likely cached at FE servers and delivered immediately, as its delivery time is largely a function of RTT, and does not vary significantly with, say, the types and complexity of search queries as does the dynamic content portion (see below and Section 4).

Choice and Effect of Search Queries. Since the dynamically generated content portion is search query dependent, we use different sets of search keywords with varying *popularity, granularity, and complexity*. For instance, the Bing main page provides a list of most popular keywords at the current time. In terms of granularity, we generate search queries with concatenated keywords which gives us increasingly refined search results (e.g., "Computer Science Department" and "Computer Science Department at University of Minnesota"). In terms of complexity, we use long and com-

[2]Although users are distributed globally, the size of the returned search results are quite similar.

[3]The default server is whatever server IP address the DNS resolution returns to the client.

plex search queries and mixtures of keywords that are not highly correlated (e.g., "computer *and* potato").

As an example, Fig. 3 illustrates the effect of 4 search keywords of different types on the Bing search performance: the left and right panels plot T_{static} and $T_{dynamic}$, respectively, for the 500 sample queries made in chronological order. As the performance is susceptible to short-term fluctuations, we plot the moving median with the sample window size being 10. (The results using Google have similar distributions.) We observe that $T_{dynamic}$ varies significantly with the types of search keywords used, whereas T_{static} is mostly insensitive to the search keywords used.

Do FE Servers Cache Search Results? To answer this question and mitigate the effect of this type of caching on our measurement analysis, we conduct a series of experiments. First, we collect a list of commonly searched keywords (e.g., "mobile cloud computing") as listed in the drop-down "search suggestion box" used by both Bing and Google. We also generate a list of search words not listed by the suggestion bar. A total of 40,000 keywords are used in the experiments. We perform two sets of experiments. In the first set, all measurement nodes submit the same search query sequentially to a fixed FE server. In the second set, each node submits a different search query to a fixed FE server. We repeat the experiments with different search queries and vary the FE server used. Analysis and comparison of the measurement results (in particular, the characteristics and distributions of $T_{dynamic}$) suggest that FE servers do *not* appear to cache any (dynamically generated) search result. This may not be too surprising, as most search engines attempt to *personalize* search results for individual users.

4. DISSECTING END-USER PERFORMANCE

In this section, we present the analysis of search query traces collected by us using Bing and Google. In particular, we use the model described in Sec. 2 to extract the T_{static} and $T_{dynamic}$ from the traces. We present our methodology to understand how the round trip delay between user and FE web servers affects the distribution of T_{static} and $T_{dynamic}$.

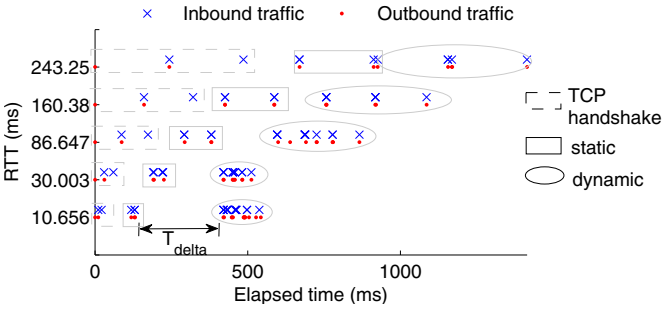

Figure 4: Inbound and outbound traffic events triggered by a single search query.

4.1 Extracting & Analyzing T_{static} and $T_{dynamic}$

Guided by the basic abstract model in Sec. 2, we perform temporal analysis of the packet-level events at the user (client) side using the collected packet traces. Figure 4 plots five sample timelines of packet generation and reception events at the client side, where five PlanetLab nodes are used as clients, each submitting the same search query to the Bing FE server. The x-axis represents the elapsed time since the start of the session, when the first TCP SYN packet is sent to the FE web server. The y-axis represents the round trip time (RTT) between the client and the FE server. Each horizontal array of dots/markers represents the timeline of packet-level events, where each blue cross/red dot marker indicates the sending/receiving time of a TCP packet. When RTT values are small, the temporal clusters of packet events are clearly visible: correlating with the application-layer packet payloads, we find that the first cluster represents the three-way TCP handshake between the client and the FE server; the second and third cluster represent the delivery of static and dynamic contents, respectively, from the server to the client. As the RTT increases, the gap between the end of the second and the beginning of the third clusters decreases, and eventually the two are lumped together, as predicted exactly by our model.

Using the *datasets B* collected via the second type of experiments (see Sec. 3) conducted for both Bing and Google, we extract and analyze the parameters T_{static}, $T_{dynamic}$ and T_{delta}. Figure 5 shows the distribution of these parameters using a sample of measurement data for one Bing FE server (IP address 198.189.255.208) and one Google FE server (IP address 74.125.224.18), where 720 repeated experiments using the same search query are performed over time at each PlanetLab node. In these plots, x-axis represents the RTT between a PlanetLab node and the FE server, while y-axis represents the *median* value of T_{static}, $T_{dynamic}$ and T_{delta} observed at each of the PlanetLab nodes. With a few outliers, the leftmost plot shows that for both Bing and Google, T_{static} is relatively stable and is largely independent of the PlanetLab node that had sent the query. Similarly, $T_{dynamic}$ is roughly a constant when RTT is small. However, when RTT is large, $T_{dynamic}$ increases linearly with RTT. In the case of T_{delta}, when RTT is small, it decreases linearly with RTT; and it becomes zero when RTT is beyond a certain threshold (for Google, this threshold is around 50ms to 100ms, for Bing, around 100ms to 200ms).

All these observations can be explained using our simple abstract model. First note that in the definition of T_{static}, we have subtracted the (initial) effect of RTT. Hence T_{static} depends mostly on the time to generate and deliver the (same) static content portion at the FE server (assuming the available bandwidth and the server load seen by all the PlanetLab node is roughly the same). When RTT is small, the delivery of the static content portion will be finished before the FE receives the dynamically generated search result from the BE data center (the time at which this is received at the FE server is independent of where the client is). Hence when RTT is small, $T_{dynamic}$ is roughly a constant while T_{delta} decreases as a function of RTT. When RTT increases beyond a certain threshold, the dynamic content portion will be received by the FE server before the static content portion is entirely delivered to the client. Hence $T_{dynamic}$ increases as a function of RTT (due to the TCP window mechanism), while T_{delta} becomes zero. The observations therefore match the prediction by our simple abstract model. Our analysis also suggests that below a certain threshold, reducing the RTT further will not drastically improve the overall user perceived performance.

4.2 Comparing Bing & Google Performances

We now compare the performances of Bing and Google by

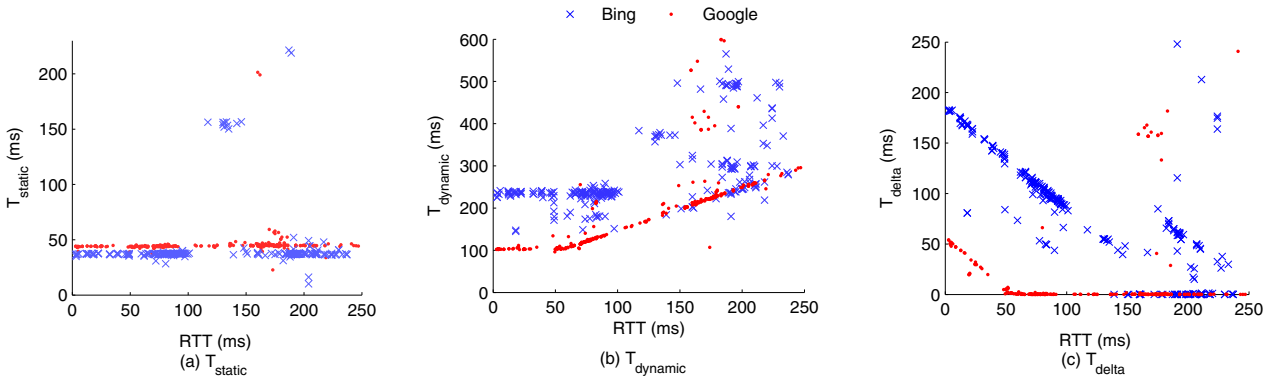

Figure 5: Distribution of T_{static}, $T_{dynamic}$, and T_{delta}.

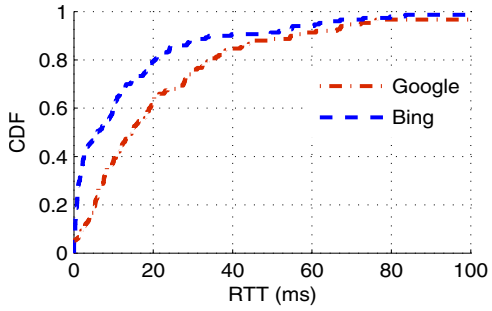

Figure 6: RTT distribution.

examining various factors and time components in affecting the overall user-perceived response times using the *datasets A* collected via the first type of experiments (see Sec. 3).

Comparing RTT Distributions. In Figure 6, we compare the RTTs between the PlanetLab nodes and their *default* FE servers (as determined by the DNS resolution). As seen in this plot, in general, Bing has FE servers (Akamai CDN servers) which are closer to PlanetLab nodes than for Google. In particular, more than 80% of PlanetLab nodes observe an RTT of less than 20ms for reaching the Bing FE servers. On the other hand, only 60% of PlanetLab nodes observe this latency for Google.

Comparing T_{static} and $T_{dynamic}$ Distributions.

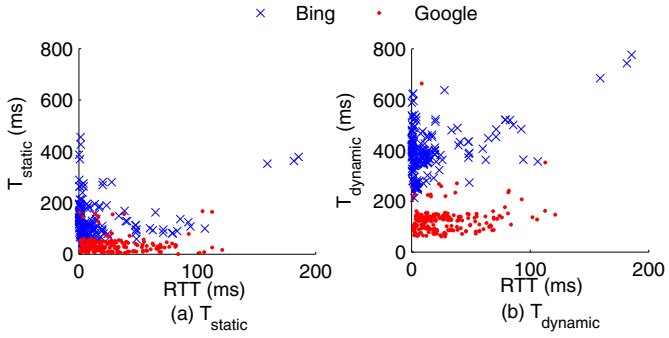

Figure 7: T_{static} and $T_{dynamic}$ for Planetlab nodes using default frontend servers.

Next, we extract and analyze T_{static} and $T_{dynamic}$ seen by each client when the default FE server is used for search queries. Figure 7 shows the distribution of T_{static} and $T_{dynamic}$ for both Google and Bing. As seen in this figure, although the Bing FE servers are generally closer to the clients (PlanetLab nodes), it has significantly higher value of T_{static} and $T_{dynamic}$ than Google. In contrast, Google has slightly farther FE servers from the clients, but has significantly lower T_{static} and $T_{dynamic}$. In addition, Bing exhibits more *variable* performance (i.e., higher variances) in the measured values of T_{static} and $T_{dynamic}$ than Google. These results illustrate that placing FE servers closer to clients does not necessarily reduce T_{static} and $T_{dynamic}$. We speculate that a plausible reason that Bing has higher and more variable T_{static} values may be due to the higher and more variable loads at the Akamai FE servers, as they are shared with a number of other services; while that Google FE servers have smaller and more stable T_{static} values may be attributed to the fact that these servers are likely dedicated to distribution of search results. Similarly, the $T_{dynamic}$ values for Bing FE servers are larger, and have more variability. The contributing factors may involve the processing capability and load fluctuations on the BE data centers, the search algorithm being used, the quality of the connection between FE and BE servers, e.g. loss rate, jitter, throughput, etc. A dedicated connection between FE and BE servers via "internal" network usually provides better connection than that built on the general Internet connections.

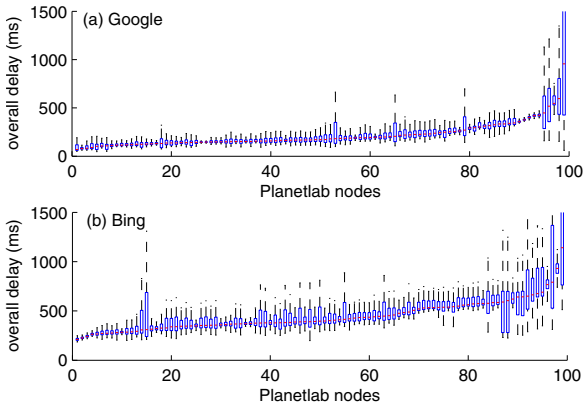

Figure 8: Overall delay performances.

Comparing Overall User Search Experiences. Finally, we compare the overall responsive time for individual search queries performed on both Bing and Google, as shown in Fig-

ure 8. The x-axis represents the PlanetLab nodes, and the y-axis represents the box-plot for the distribution for different samples. The results show that comparing Google, users using the Bing search service tend to experience slightly longer and more variable overall response times. In conclusion, our study shows that simply placing FE servers closer to users may not be entirely effective in improving the overall user-perceived performance in dynamic content distribution. Other key factors such as the processing time, server loads at both FE servers and BE data centers as well as the (physical and TCP) connections between them also play a critical role. Improving and optimizing these factors are therefore important in improving the overall user-perceived performance in dynamic content distribution such as dynamic generation of search results in response to user queries.

5. FACTORING FE-BE FETCH TIME

As discussed in Sec. 2, T_{fetch} consists of two key components, namely, T_{proc} and RTT_{be}. They represent the search query processing time at the BE data center and the delivery time of search results from the BE server to the FE server respectively. As part of our ongoing work, we are exploring various mechanisms to separate these two components.

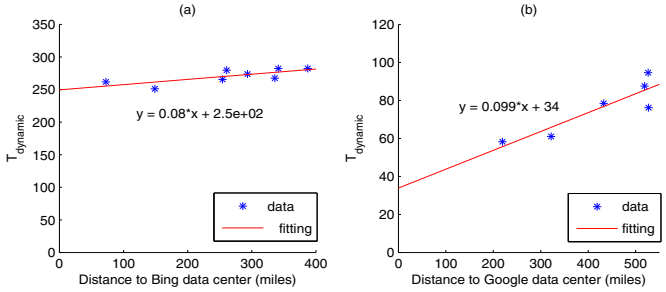

Figure 9: Correlating $T_{dynamic}$ and the distance to BE data center.

To get a better understanding of these components we conducted the following analysis. We first get a list of possible locations for Bing and Google data centers from [1, 2]. Next for each of these data centers we consider the geographically closest FE servers, and plot the distribution of $T_{dynamic}$ with respect to geographical distane between FE and BE. As explained in Sec 4 for smaller values of RTT, $T_{dynamic}$ can be considered as an approximation for the T_{fetch}. Figure 9 shows the distribution of T_{fetch} time for Bing and Google with respect to the geographical distance between the FE and BE locations. For this plot, we consider the Bing data center located in Virginia (US), while we pick the Lenoir, North Carolina data center for Google. As seen in these figures, the T_{fetch} time increases linearly as the distance between BE and FE increases. We perform a linear regression to fit a straight line for these data points, which is shown using the red continuous lines in the figure. As seen in this figure the Y-intercept for the regression is 260ms for Bing, while it is only 34ms for Google data center. This intercept actually represents the computation time for a given search query for Bing and Google data centers. Similarly, the slope of the line represents the contribution of network delay in T_{fetch}, which are similar for both Bing and Google. For the different keywords used in our search queries, we get very similar slope, but pretty ydifferent intercept values. Though our initial results show interesting characteristics of factors affecting the FE-BE fetch time, we are currently conducting extensive experiments and analysis to gain a better understanding behind the factors affecting the FE-BE fetch time, and thus will potentially guide us in designing better content placement and delivery strategies for dynamic content distribution.

6. DISCUSSIONS

In this paper we have focused on the roles of frontend servers on the end-to-end performance of search queries using the standard search fucntions of search engines. More recently, some search engines such as Google has introduced more advanced search features such as the *interactive* "search as you type" feature. Our preliminary investigation of this new feature shows that our basic model and key observations still hold. We find that using the interactive search feature, after each letter a user has typed, a separate query (using a *new* TCP connection) is sent to the FE server. The delivery of each query hence still fits our basic model; although we believe it is likely that the search query processing times at the BE data centers are generally reduced because the subsequent queries are highly correlated with previous queries. We are in the process of conducting more thorough measurements and analysis on this and other search features.

As most Planetlab nodes are located within or close to the University campus networks (and it is known that some Akamai frontend servers are placed closer to University campus networks), we realize that using the PlanetLab as the testbed may introduce some biases. For instance, the RTT between PlanetLab and Akamai FE servers may not be of all users. In addition, in our measurements we do not see any significant packet losses. In an environment where the loss rates are high (e.g., in a wireless network), placing FEs closer to users in fact may significantly improve the user-perceived end-to-end performance by reducing the total time needed to deliver the query result (that has been delivered to the FE server from a BE data center) to a user. As part of ongoing work, in addition to the PlanetLab, we are utilizing other testbeds (e.g., Seattle testbed [3]). We are also investigating the trade-offs between RTTs and loss rates (e.g., in a WiFi environment) in the placement of FE servers.

7. CONCLUSIONS

In this paper we investigated the roles of FE servers in improving the user-perceived performance of dynamic content distribution. Using Bing and Google search services as case studies, we conducted extensive application-layer active measurement and data analysis. Our results demonstrate that there is a critical trade-off between the placement of FE servers and the FE-BE fetch time. While placing FE servers closer to users can help reduce latency, other key factors such as processing times and loads at both FE servers and BE data centers as well as the quality of (physical and TCP) connections between them also play a critical role in determining the overall user-perceived performance.

Acknowledgments

This work is supported in part by the NSF grants CNS-0905037, CNS-1017092 and CNS-1017647.

8. REFERENCES

[1] Google data centers. http://www.google.com/corporate/datacenter/locations.html.

[2] Microsoft online services data center locations. http://www.championcloudservices.com/data-center-locations/.

[3] Seattle. https://seattle.cs.washington.edu/html/.

[4] S. Androutsellis-Theotokis and D. Spinellis. A survey of peer-to-peer content distribution technologies. *ACM Computing Surveys (CSUR)*, 36(4):335–371, 2004.

[5] M. Bateni and M. Hajiaghayi. Assignment problem in content distribution networks: unsplittable hard-capacitated facility location. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 805–814. Society for Industrial and Applied Mathematics, 2009.

[6] M. El Dick, E. Pacitti, R. Akbarinia, and B. Kemme. Building a peer-to-peer content distribution network with high performance, scalability and robustness. *Information Systems*, 2010.

[7] H. Jiang, Z. Wang, A. Wong, J. Li, and Z. Li. A replica placement algorithm for hybrid CDN-P2P architecture. In *2009 15th International Conference on Parallel and Distributed Systems*, pages 758–763. IEEE, 2009.

[8] D. Lewin, A. Davis, S. Gendler, M. Kagan, J. Parikh, and W. Weihl. Dynamic content assembly on edge-of-network servers in a content delivery network, July 6 2010. US Patent 7,752,258.

[9] A. Pathak, Y. Wang, C. Huang, A. Greenberg, Y. Hu, R. Kern, J. Li, and K. Ross. Measuring and evaluating TCP splitting for cloud services. In *Passive and Active Measurement*, pages 41–50. Springer, 2010.

[10] J. Ravi, Z. Yu, and W. Shi. A survey on dynamic Web content generation and delivery techniques. *Journal of Network and Computer Applications*, 32(5):943–960, 2009.

[11] M. Tariq, A. Zeitoun, V. Valancius, N. Feamster, and M. Ammar. Answering what-if deployment and configuration questions with wise. In *Proceedings of the ACM SIGCOMM 2008 conference on Data communication*, SIGCOMM '08, pages 99–110, New York, NY, USA, 2008. ACM.

[12] M. Wittie, V. Pejovic, L. Deek, K. Almeroth, and B. Zhao. Exploiting locality of interest in online social networks. In *CoNEXT*, page 25. ACM, 2010.

Summary Review Documentation for

"Characterizing Roles of Front-end Servers in End-to-End Performance of Dynamic Content Distribution"

Authors: Y. Chen, S. Jain, V. Adhikari, Z. Zhang

Reviewer #1

Strengths: Simple and clear model that is then confirmed by measurements. Comparison of 2 different search engines, instead of just applying the model to one.

Weaknesses: It is surprising to see such differences in the slope of the lines in figure 8, you would expect them to be roughly the same. Model doesn't take into account new search engine webpages where dynamic content is continuously updated as people type and try different searches.

Comments to Authors: Short and clear paper.

1. Introduction: you should mention that the load on the client side might also impact the overall performance
2. As mentioned in your weaknesses, your model assumes a single search when search engines have now dynamic webpages where the static content is initially downloaded and the dynamic content is continuously updated as the user type and try different searches.
3. Do you have any references about the front end architecture for Bing and Google?
4. In section 2, I assume that you define Tdelta as t5-t4 and not t4-t2
5. Did you also observe protocol manipulations when you compared the 2 search engines? Did you see for instance differences in initial congestion windows?
6. I like your section 5 but I'm puzzled by it and by figure 8. I was expecting indeed a different Y-intercept for both search engines. However I don't understand how we can observe a difference of a factor of 3 between the slopes of the lines in the 2 figures. You would expect these numbers to be pretty similar.

Reviewer #2

Strengths: The problem of inference the frontend-to-backend delay is an important one. The paper can be a good reference. Solid measurement methodology and analysis of measurements. Large scale experiments. I like the trade-off between placing caches very close to the user vs. put caches close enough to the datacenters to reduce fetching time. The paper is well-written.

Weaknesses: In section 3 it is not clear how you generate the queries. Queries and answers may very per region, this issue was not enough discussed in the paper. The authors used only PlanetLab nodes; there are other available testbeds that were not utilized.

Comments to Authors: (1) In section 3 you have to elaborate more on how you generate the queries. I guess that the fetching time depends on the complexity of the query. You should expect low fetching time for very popular queries and a higher fetching time for complex queries. It would be nice to present results on this. You may want to evaluate if there is a correlation between the fetching time and the number of words used in the query.

(2) You have to provide evidence that your queries are quite representative for users located in different regions. It is known that queries and answers in both Google and Bing highly depend on the user region, you have to address this issue.

(3) It is also known that both Google and Bing also optimize based on user profile. In your study this can not be simulated. Do you think that this may increase the fetching time you observe?

(4) It is not clear what is the popularity of your queries. Do you expect that the answers are pre-cached or you always receive a fresh answer for dynamic content?

(5) Despite the fact that this is a short paper, you have to provide information for the active measurement platform.

(6) Did you observe any fetching activity if the autocompletion feature is activated?

(7) You may also want to compare the fetching time with the DNS resolution time.

(8) You may want to repeat your experiments using the Seattle testbed (https://seattle.cs.washington.edu/html). The bandwidth limitations should not be a problem for your measurements.

Reviewer #3

Strengths: Very timely and relevant problem. Nice methodology in terms of measurement, model building and validation. Useful reported results. Very good presentation.

Weaknesses: Some missing related work.

Comments to Authors: I really enjoyed reading this paper, my compliments to the authors.

1. Please take a look at: Exploiting Locality of Interest in Online Social Networks, Mike P. Wittie et al., ACM CoNEXT'10. From the title it does not seem that relevant but it is about OSNs FE used to split the TCP connection to the OSN BE and as well cache static parts ... Please discuss how your work compares to this paper.
2. A suggestion: Why don't you use a virtual coordinates system to estimate the RTT between FE and BE servers and then take

this and Tstatic+RTT out from Tdynamic in order to say something about Tproc at the datacenter?

3. Additionally, I think that you can improve (or maybe report if you have already done) the study around the effect of particular queries to the observed results. For example you should explain more clearly at least the two extremes, simple queries for which the answer will come very quick (might be cached at the BE) and more complicated ones.

4. Another thing, in your model I cannot see how the the interactive "search as you type" offered by google the last 1-2 years would fit.

5. Also I think you need to improve the positioning of your work and contributions. Cite more papers on tiered internet services, optimization of intra datacenter communications between FE and BE, etc. There is work here that you don't mention.

Reviewer #4

Strengths: This is a very timely topic as dynamic service delivery is emerging as the next big business for CDNs, after video delivery. the paper´s model and evaluation concurre and validate each other. the results on the differences from Bing using Akamai and Google using their own CDN are interesting too.

Weaknesses: The model does a good job at capturing delays between edge and back servers, processing times, and delivery times. however, it lacks a more fundamental study of the impact that losses can have in TCP splicing and thus the position of the servers. you assume that FE server position is only worth it to have enough pipelining such as to ensure that no idle time exists, and in the case of high processing time, placing servers closer to the user may not have any effect. this is true, as long as there are no losses since splitting TCP connections under losses can have a secondary effect, i.e. that of increasing the throughput of the connection. for instance look at this paper: "TPOT: Translucent Proxying of TCP". P. Rodriguez, S.Sibal, O.Spatscheck, WCW'00, International Caching Workshop.

Comments to Authors: None.

Reviewer #5

Strengths: Nice observations about content delivery performance.

Weaknesses: Study from planetlab only, so RTT is probably not representative but the message should still be valid when measured from other end-hosts.

Comments to Authors: A latency of 20ms even to Akamai is really low. DSL end-hosts would have higher latency, even to Akamai. Maier et al. Characteristics of residential traffic. IMC 2009, showed that often 30 ms is added just by the DSL interleaving, so the latencies you found are certainly not realistic, not even mentioning mobile devices. Your observations are due to Akamai having caches in academic networks and large ISPs. This is not always the case and greatly varies across networks, both academic and commercial. Despite the methodological limitations, this paper is of great interest to Akamai, send it to them.

Response from the Authors

We thank the anonymous reviewers for providing valuable comments and feedback on our paper. We have addressed a majority of them in this final version of the paper, and few others are listed as future work. Rest of the comments contain interesting suggestions, but unfortunately are out of the scope of this paper.

We add one separate discussion section to discuss the major concerns raised by the reviewers. One such concern is that we only dealt with the traditional search engine, in which advanced features such as the interactive search, autocomplete function are not considered. To address this problem, we elaborate on how these features work, and how they still fit our basic model.

Some reviewers suggest that the use of Planetlab may not be representative of the typical RTT between the users and the FE servers. Also, the loss rate might be another factor that should be taken into consideration in placing the FE servers. In particular, with the increasing popularity of wireless users, the loss rate at the last hop can be a major issue. We discuss these issues in the discussion section, and also list them as our ongoing work.

Some reviewers asked for clarification about the results presented in the last section, i.e. factoring FE-BE fetch time. The slope values shown in the last figure was quite different. To validate the correctness of our results, we repeat the same measurements described in that section for different keywords, and only consider front-end servers close enough to the BE servers, and new results are reported in the final version of the paper.

To address concerns that the impact of the DNS resolution time, and the personalization of the search results generated for users in different regions were not explicitly mentioned, we put several footnotes in the final version of the paper to better clarify them. We also added some of the citations suggested by the reviewers, and fixed several typos.