Characterizing and Mitigating Web Performance Bottlenecks in Broadband Access Networks

Srikanth Sundaresan, Nazanin Magharei, Nick Feamster, Renata Teixeira
{srikanth.sundaresan,nazanin,feamster}@cc.gatech.edu, renata.teixeira@lip6.fr

ABSTRACT

We present the first large-scale analysis of Web performance bottlenecks as measured from broadband access networks, using data collected from two extensive home router deployments. We design and implement tools and methods to identify the contribution of critical factors such as DNS lookups and TCP connection establishment to Web page load times and characterize how they contribute to page load times in broadband networks. We find that, as the connection speeds of broadband networks continue to increase, other factors such as TCP connection setup time, server response time, and network latency are often dominant performance bottlenecks. Thus, realizing a "faster Web" requires not only higher download throughput, but also optimizations to reduce both client and server-side latency. We deploy three common caching optimizations inside home networks to reduce latency—content caching, TCP connection caching, and DNS caching—and evaluate their effects on the factors that contribute to page load times in broadband networks.

1. INTRODUCTION

Broadband Internet access at home is getting faster: the OECD reports that broadband speeds are increasing by about 15–20% every year; average advertised broadband speeds are now about 16 Mbits/s in the U.S. and 37.5 Mbits/s across OECD areas [43]. As downstream throughput continues to increase, one might expect the Web to get faster at home, as well; yet, Internet service providers and application providers are increasingly finding that *latency* is becoming a critical performance bottleneck [29, 40]. The Bing search engine experiences reduced revenue of 1.2% with just a 500-millisecond delay [51], and a 400-millisecond delay resulted in a 0.74% decrease in searches on the Google search engine [14]. Forrester research found that most users expected online shopping sites to load in two seconds or fewer [40].

This paper presents a large-scale, longitudinal study of the effects of broadband access network performance on Web page load times. Our study finds that latency contributes significantly to slow page load times, particularly for access links with higher downstream throughput. Because previous studies have observed that characteristics of the access network can introduce significant latency [53], we were motivated to develop and evaluate techniques for mitigating latency bottlenecks in the last mile. Although many Web browsers already perform caching and prefetching optimizations, the growing number of embedded devices in the home and the potential for caching and prefetching across devices makes the home router an interesting place to evaluate these techniques. Our extensive discussions with content providers highlight a perennial quest to mitigate any factor that can slow down Web performance by even tens of milliseconds.

Although there is a vast body of work on both modeling Web performance and making it faster, our work is the first large-scale study of Web performance bottlenecks from broadband access networks. To our knowledge, it is also the first to directly compare the benefits of three different caching optimizations (*i.e.*, caching and prefetching of content, DNS records, and TCP connections), which we have deployed and evaluated in real broadband access networks. Identifying the extent to which performance characteristics of the underlying network introduces bottlenecks in a Web download—and the extent to which various optimizations can mitigate these bottlenecks—requires developing a careful measurement method, which we outline in detail in Section 3. This paper makes contributions in three areas:

First, **we characterize Web performance from more than 5,000 broadband access networks to nine popular Web sites and develop a tool for identifying factors that contribute to Web page load time (Section 4)**. We use data from the FCC/SamKnows study in the United States from September 2012, as well as data collected from the BISmark deployment in North America, Europe, and Asia during May–June 2012. We find that throughput is the dominant bottleneck for access links with downstream throughput rates of less than 8 Mbits/s. In the case of the increasing number of access links with higher downstream throughput, latency has become the bottleneck for Web performance. Latency affects many aspects of page load time, from DNS lookups to the time to complete a three-way TCP handshake; it also plays a significant role in the speed of small-object transfers. Our results suggest that an increase in last-mile latency of just 10 milliseconds can sometimes induce delays of hundreds of milliseconds for page load times of popular sites.

Second, **we develop and evaluate the effects of different optimizations on Web performance in broadband access networks (Section 5)**. To our knowledge, this paper presents the first study to compare the relative benefits of content caching, DNS caching, and TCP connection caching from within home networks. We instrument the home router to evaluate the benefits of performing these three common Web performance optimizations. We find that, as expected,

content caching offers the most significant reductions in page load time and can reduce page load times by more than 50% in some cases. Even simply caching commonly used TCP connections at the home router can reduce expected page load times by more than 20% in many cases. Performing all three optimizations from the home router can reduce page load times by 25–60% for many popular Web pages when cache hits occur for DNS records, TCP connections, and content. Our experiments show that caching content in edge ISPs may not fully realize the potential improvements in page load times, since last-mile latency is often significant.

Third, **we design, implement, and deploy mechanisms to improve cache hit rates in home networks to allow users to realize faster page load times in practice (Section 6)**. We develop an OpenWrt module that runs on a home router and tracks the Web sites that users in that home commonly visit. Based on its assessment of popular sites in the home, the module proactively prefetches and caches DNS records and TCP connections for these popular sites. This approach, which we call *popularity-based prefetching*, allows users to realize lower latencies and faster page load times in practice. Using traffic traces capturing user behavior from twelve homes, we find that proactively performing DNS prefetching and TCP connection caching for the twenty most popular popular sites inside the home significantly increases DNS and connection cache hit rates.

We have published both the active measurement data from the BISmark experiments [13] and the OpenWrt module for performing popularity-based prefetching [47] at private URLs that we will make public upon publication of the paper. The FCC/SamKnows Web performance measurements will soon be available on the FCC Web site [29].

2. BACKGROUND

We describe the factors that affect Web page load times and previous optimizations to make page load times faster.

2.1 Factors That Affect Web Page Load Times

Web downloads begin with a DNS lookup for the Web page. The Web browser retrieves the requested object by requesting a uniform resource locator (URL) over the hypertext transport protocol (HTTP); the object may itself contain references to other objects that the browser must subsequently retrieve. Each of these "embedded" objects is typically referenced with another URL, which the browser retrieves with additional HTTP requests.

Because the time for the browser to render a page depends on the user's choice of browser and machine configuration, we instead study *page load time*, which is the time from the initial request to the time when all objects for the page have been retrieved. Many factors contribute to Web page load time. The retrieval time for each object has the following components: (1) the *DNS lookup time* for the domain referenced in the object URL; (2) the *TCP connection time*, which is the time to complete the TCP three-way handshake to the server; (3) the *server response time*, which is the time it takes for the server to start sending the object once it has been requested; and (4) the *object fetch time*, which is the time to download the object itself over the TCP connection. Some of these factors, such as the DNS lookup and TCP connection times, are bound by latency; others, such as the object fetch time, are bound by both latency and throughput.

2.2 Web Performance Optimizations

We now discuss both server-side and client-side optimizations to improve page load time.

Server-side optimizations Server-side optimizations include HTTP replacements [52] and TCP modifications [11, 12, 19, 20, 26]. Recent proposals suggest using a larger initial congestion window sizes on servers for TCP connection, so that small objects can be transferred with significantly fewer round trips [26]. Al-Fares *et al.* studied the effects of server-side optimizations, such as increasing TCP's initial congestion window (ICW) and enabling HTTP pipelining on Web page load times [7]. They found that increasing the ICW can reduce page load times by several hundred milliseconds in many case. Although these server-side optimizations can improve page load times, they do not reduce certain components that contribute to page load time, including DNS lookup and the TCP connection setup time. Zhou *et al.* propose a new protocol that minimizes connection time by having DNS resolvers set up TCP connection on the client's behalf [56].

Client-side optimizations Many client-side optimizations from the browser have also been developed. Nielsen *et al.* introduced pipelining and persistent HTTP connections and showed the superior performance of HTTP/1.1 with pipelining and over HTTP/1.0 [42]. HTTP/1.1 allows *persistent connections*, so that a client can retrieve multiple objects over the same TCP connection (thereby amortizing the three-way handshake and TCP congestion window ramp up cost over multiple objects). HTTP/1.1 also allows *pipelining*, whereby the client initiates a request for the next object as soon as it sees a reference to that object in another object (rather than waiting for the object download to complete). Most browsers do not enable pipelining by default, and some servers do not enable persistent connections.

Content caching is also a common optimization. Ihm *et al.* characterized five years of Web traffic traces from a globally distributed Web proxy service; they observe that Web caches typically have a 15–25% hit rate, and these rates could almost double if caches operated on 128-byte blocks [32]. Previous studies have reported object cache hit rates in the range of 35–50%, although these cache hit rates have continued to drop over time [2, 16, 31, 41, 55].

To improve cache hit ratios, Web browsers prefetch DNS records before the client requests an object for that domain; the browser parses certain downloaded pages (*e.g.*, a search result page) for domain names and resolves them before the user clicks on them [24]. Some browsers also support content

Target	Objects			Lookups			Connections			Size (KB)		
	Sam-	BISmark		Sam-	BISmark		Sam-	BISmark		Sam-	BISmark	
	Knows	US	non-US	Knows	US	non-US	Knows	US	non-US	Knows	US	non-US
edition.cnn.com	26	25	26	4	4	4	12	12	12	1199	1022	1023
www.amazon.com	24	31	32	4	4	4	21	24	23	589	840	851
www.ebay.com	29	33	32	12	14	14	16	17	19	595	613	615
www.facebook.com	8	8	7	2	2	2	7	8	7	437	389	289
www.google.com/mobile	32	20	20	1	1	1	8	8	8	1398	289	291
www.msn.com	24	24	54	8	8	8	14	14	16	377	348	641
www.wikipedia.org	16	15	16	1	1	1	16	15	15	56	56	56
www.yahoo.com	74	69	66	7	7	8	32	32	29	927	887	818
www.youtube.com	8	7	8	2	2	2	9	8	8	488	423	414

Table 1: *Properties of the Web sites in our data set. The values represent the average of the parameters over all transactions.* Objects *denotes the number of objects that must be fetched;* lookups *are the number of DNS lookups required; and* connections *are the number of unique TCP connections the client set up to fetch all objects in the page. The number of connections depends on whether the server supports persistent connections, whether the objects are located on one or more domains, and the order in which objects are retrieved.* Size *denotes the number of bytes for all of the objects for a particular page.*

Location	Number of homes	Avg. last-mile latency (ms)	Avg. downstream tput (Mbits/s)
United States	43	12	24.2
Europe	3	24	8.3
N. Amer. (non-U.S.)	3	15	5.5
E. Asia/Australia	4	3	46.5
Southeast Asia	8	12	5.7

Table 2: *Nodes in the BISmark deployment.*

Metric	Type	Description
Page fetch time	Total	The time to set up TCP connections and retrieve all objects.
Page load time	Total	Page fetch time, plus the DNS lookup time
DNS lookup time	Per Domain	The DNS lookup time for the main domain of the site.
Time to first byte	Per Object	The time from the initiation of the TCP connection to the arrival of the first byte of the requested object (including server processing time).
Object fetch time	Per Object	The time to download an object, excluding the DNS lookup time and time to first byte.

Table 3: *Performance metrics. For each per-object metric, the test measures the maximum, minimum, and average times for each object in the transaction.*

prefetching [39]; Padmanabhan *et al.* proposed predictive content prefetching, using server hints [45]. To reduce the time associated with DNS lookups, browsers and intermediate DNS servers employ caching and prefetching [21, 22]. Jung *et al.* studied DNS performance and the effectiveness of DNS caching [35], and saw that DNS cache hit rates can reach as high as 80%, even with only a few clients [34]. We study the performance of DNS caching in the home and see that it can reduce the maximum DNS lookup time for loading a page's objects by about 15–50 ms. Feldmann *et al.* observed in Web traces from AT&T home users that 47% of objects retrieved incur more than half of the total download time from TCP connection setup [30]. Based on this observation, the study proposes a *connection cache* in the ISP network to reduce connection setup time, which reduces download times by upto 40%.

3. MEASUREMENTS AND METHOD

Although a single number that represents the Web page load time that a user experiences is appealing, devising such a measure is unfortunately not possible, for many reasons. Notably, every browser downloads pages using a different method (and different optimizations), and the pages themselves may be optimized for different browsers and devices.

Regardless of the optimizations that clients and servers may perform to improve a user's actual experience, the effects of network parameters such as the access network throughput and latency on various components of Web page load time are worth understanding. Understanding how network effects contribute to various factors that affect Web page load times can better inform designers of both Web sites and browsers about bottlenecks that contribute to page load times. Our goal in this work is not to estimate the page load time that a real user would experience, but rather to establish methods for understanding baseline measurements against which we can benchmark both the contributions of various factors to page load time, the network bottlenecks that affect page load time, and the benefits of different network optimizations towards reducing page load time.

3.1 Measurements

We collect three different types of measurements in our study: Web performance measurements from the home router, Web performance measurements from the browser, and passive measurements of user browsing activity collected from the home router. Ideally, we would measure Web performance from each access link using a variety of browsers, since each Web browser implements a different set of optimizations; unfortunately, deploying many Web browsers in thousands of access networks is not feasible, since it is difficult to instrument browser-based experiments in a repeat-

able, controlled environment. To deal with this tradeoff, we perform active performance measurements from routers in a large number of homes, and more detailed browser-based measurements in a controlled setting.

Active measurements from the home router We measure Web performance by periodically requesting the home page of the nine Web sites in Table 1. We use two home-router deployments:

- We run measurements from the BISmark deployment [48], an open platform for home network research. Table 2 characterizes the homes in the BISmark deployment by region. *BISmark-US* collects measurements from 44 routers across the US, and *BISmark-nonUS* collects measurements from 18 routers in other parts of the world, including Europe, North America (excluding the US), and Asia. The BISmark-US dataset is from May 17–June 7, 2012, and the BISmark-nonUS is from May 24–June 7, 2012. The URLs are the same for both datasets, and we rely on DNS to locate the local version of a site, if one exists.
- We use measurements from 5,667 participants in the SamKnows/Federal Communications Commission study across 11 ISPs in the US from September 1–31, 2012. We include only users who have reported more than 100 measurements during the duration of the study from ISPs with more than 100 users. A previous study of broadband Internet performance describes the deployment in more detail [53]. We verified that the results that we present in this paper are consistent with similar sets of measurements from October 2011 and April 2012.

Active measurements from the Web browser We also perform controlled experiments to measure page load times from a Web browser. We run controlled experiments from a browser in a laptop running Ubuntu 10.04 LTS that is connected to the access point directly on a wired gigabit Ethernet link. We use the Selenium library to automate fetching of Web sites using Firefox 3.6. We shape the uplink to 10 Mbits/s with last-mile latencies of 10 and 40 ms to emulate home access links. To eliminate any optimizations that the browser itself might perform, we use a fresh browser instance with a new user profile for each Web page load. This test measures whether the optimizations that we implement result in perceptible performance differences from a browser.

Passive measurements of browsing activity We also capture Web browsing activity of users from routers deployed in twelve homes for the period October 15–31, 2012 to evaluate the potential improvements from proactive DNS and connection caching in home networks. We record the timestamp, hashed domain, and TTL of all resolved DNS responses. Each HTTP flow event has a timestamp of all connections, hashes of their corresponding IP addresses, flow duration, and TCP connection time.

3.2 Method

We now briefly describe the methods that we use to characterize Web page load times from broadband access networks. In the first part of our study, we characterize page load times using a tool that offers a breakdown of how different components contribute to the retrieval of each Web object, as well as the total time to retrieve all of the objects from a Web page. We then develop a controlled approach to investigate how three different caching optimizations—DNS caching, TCP connection caching, and content caching—contribute to reducing page load time. Finally, we characterize the benefits of these optimizations in actual home networks, if they were deployed on a home router. To do so, we analyze browsing patterns in our passive data and evaluate performance benefits in a real network.

3.2.1 Measuring Components of Page Load Time

We call the process of downloading all objects for a Web site a *transaction*. We measure the time for each component of a Web transaction, as shown in Table 3. We periodically request the home page of a Web site, determine the objects that must be downloaded to render the page, perform all the DNS lookups, and then download all the objects. This allows us to separate the page load time into the time to perform DNS lookups and the page fetch time. We can also measure individual components of Web page load time, including the DNS lookup time and the time to first byte, for each object. The tool uses persistent TCP connections if the server supports them and up to eight concurrent TCP connections to download objects in parallel. The same tool is deployed in both the SamKnows and the BISmark deployments and is publicly available [50].

Although we acknowledge that the measurements that we collect are not representative of the page load times that a real Web browser might experience, they do reflect the times associated with individual object fetches. Any optimizations that a Web browser might perform (*e.g.*, fetching objects in parallel, prefetching objects or DNS records) would still have to confront the network metrics that our study characterizes.

3.2.2 Measuring Performance Benefits of Caching

We use the BISmark deployment described in Section 3.1 to evaluate how caching in the home network can improve Web page load times. The Web tool running on the BISmark router resolves DNS names for Web sites by sending queries to `dnsmasq`, a lightweight caching DNS resolver [25] that runs on the router. The resolver caches up to 150 domains and honors the TTL values of the lookups. To evaluate TCP connection and content caching, we use `polipo`, a caching HTTP proxy [46]; `polipo` splits the TCP connection by opening a connection to the requested domain on behalf of the client and communicates with the client over a separate connection and reuses TCP connections where possible. We run `polipo` with a 4 MByte cache in RAM.

Measurement	Proxy Location	DNS	Conn.	Content
Baseline Measurements				
No Proxy, ISP DNS	—			
Cold Home Proxy	—	•		
ISP Proxy	Network	•	•	•
Optimizations				
Home DNS	Home	•		
Home Conn. Caching	Home	•	•	
Home Proxy	Home	•	•	•

Table 4: *The measurements we perform to evaluate the benefits of DNS, connection, and content caching in the home.*

In general, page fetch times for users outside the United States are significantly higher for many sites; these higher times result mostly because of higher latencies to these sites.	4.1
Page fetch times tend to stop improving beyond a downstream throughput of about 8 Mbps. At this point, latency becomes the main performance bottleneck.	4.2
The time to first byte can exceed the object fetch time in the cases of many small objects and for some pages can be as much as 21% of the page fetch time. The last-mile latency can be up to 23% of the time to first byte.	4.3

Table 5: *Highlights of Section 4 results.*

Measuring Baseline Performance Table 4 illustrates how we measure baseline performance and compare it with the performance of each optimization. To measure the three *baseline* performance measurements, the client first performs the following three sets of requests:

1. **ISP DNS Cache ("No Proxy, ISP DNS").** The client clears the local DNS cache and fetches the page directly from the server. This measures the baseline performance of fetching the Web page. The DNS lookups required for this measurement may reflect caching in the ISP's DNS resolver.
2. **Empty Caching Proxy in the Home ("Cold Home Proxy").** The client fetches the same page directing its HTTP request through a fresh `polipo` instance running in the router. Because `polipo`'s cache is empty at this point, this measurement reflects the overhead of using a "cold" proxy. This measurement does take advantage of any DNS caching that `dnsmasq` performs in the previous step.
3. **Shared ISP Caching Proxy ("ISP Proxy").** To measure the benefits of performing DNS, connection, and content caching at a shared proxy (the most common setup for content caches), the client first fetches the page through a `polipo` proxy that is running on a server in our university to "warm" the cache. It then immediately fetches the same page again through the same remote `polipo` proxy. Because we cannot deploy (or control) a Web cache in a real ISP's network, we approximate the behavior of an ISP cache by deploying a caching proxy in our university and perform Web page load measurements from eleven BISmark routers that are less than 35 ms away from the proxy.

Quantifying Caching Benefits in Access Networks After collecting the baseline measurements, the client then performs three additional sets of requests, in the following order, to measure the relative benefit of performing different performance optimizations in the home.

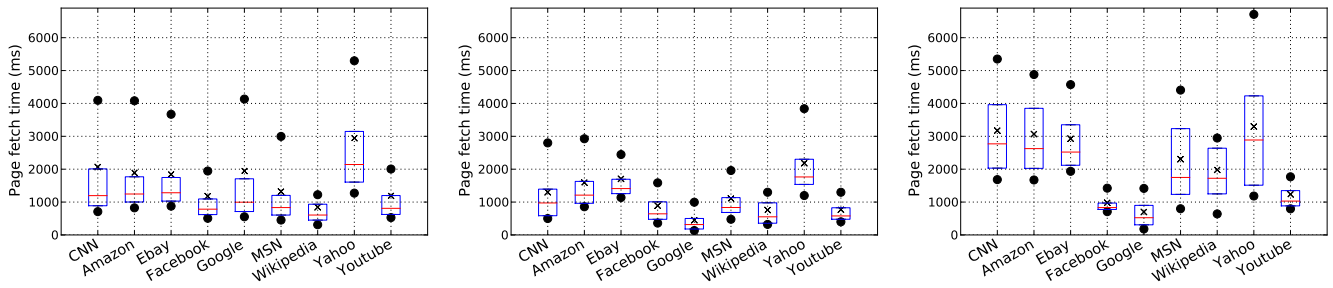
4. **Home DNS caching only ("Home DNS").** The client fetches the same page directly from the servers. This measures the benefits of DNS caching in the home (since `dnsmasq` caches the DNS responses from earlier measurements).
5. **Home proxy with DNS caching, persistent connections, and content caching ("Home Proxy")** The client fetches the page by directing its HTTP requests through `polipo` again; this measurement takes advantage of DNS, content and connection caching, since the proxy would have cached any cacheable objects and reused TCP connections where possible from the requests in the "Cold Home Proxy" experiment.
6. **Home proxy with DNS caching and persistent connections only ("Home Connection Caching")** The client clears the `polipo` cache on the home router and fetches the page by directing its HTTP requests through the home proxy again. During this step, all content must be retrieved again from the origin service. But because we run this experiment immediately after several other HTTP requests to the same set of sites the experiment can still reuse TCP connections from the previous experiment.

These experiments allow us to isolate the effects of (1) placing a cache inside the home versus elsewhere (*e.g.*, in the ISP); and (2) the relative benefits of performing DNS, connection, and content caching inside the home network.

3.2.3 Measuring Cache Hit Rates in Home Networks

The performance gains associated with caching DNS records, TCP connections, or the objects themselves can only be realized if users inside a home network experience cache "hits" at the home router. To characterize the likelihood of these cache hits in practice, we analyzed cache hit ratios in twelve homes by performing a trace-driven simulation using the passive traces described in Section 3.1. These traces contain anonymized DNS lookups and HTTP connection requests across all devices in a household. To preserve user privacy, we do not capture packet payloads, so we do not evaluate cache hit ratios for content.

4. CHARACTERIZING BOTTLENECKS



(a) *SamKnows* **(b)** *BISmark-US* **(c)** *BISmark-nonUS*

Figure 1: *Page fetch times for popular sites. The lower edge of each box indicates the 25th percentile of the distribution of page fetch times for each site, the upper edge is the 75th percentile, the middle line is the median, the cross represents the average, and the dots the 10th and 90th page fetch times.*

We characterize page fetch times from clients in the BISmark and SamKnows deployments and evaluate the effects of downstream throughput and latency on page fetch times. Table 5 summarizes the results from this section. We first analyze the page fetch times of popular Web sites and then characterize how downstream throughput and latency affect fetch times. A major takeaway is that *latency is increasingly becoming a performance bottleneck for Web performance in home networks*, particularly as users upgrade to faster service plans. Latency also remains a significant performance bottleneck for users outside the United States. Last-mile latency is a significant overall contributor to both DNS lookup times and the time to first byte. Therefore, *even when Web caches are close to users, implementing optimizations in the home network to reduce the effects of last-mile latency can offer significant performance improvements in page load time*.

4.1 Page Fetch Times of Popular Web Sites

We study the page fetch times to nine popular Web sites from clients around the world. Figure 1 shows the fetch time for each site for the SamKnows and BISmark deployments. Figures 1a and 1b present fetch times for homes in the United States; Figure 1c shows the fetch times for homes outside of the United States. As expected, the fetch time varies both by site and the location of the access network. Some variability results from differences in page size and design (see Table 1); the largest four sites (CNN, Yahoo, Amazon, and Ebay) also have the largest fetch times (*e.g.*, the median for CNN in the United States is more than one second).

Figure 1c shows that homes outside of US typically have higher page fetch times for a given site than homes in the United States. The median and variance is higher for all sites we measure from outside the United States, as well. A few sites have different sizes across locations (see Table 1), but most performance differences are not due to size effects. One explanation for the higher median outside of the United States is that content is closer to United States homes.

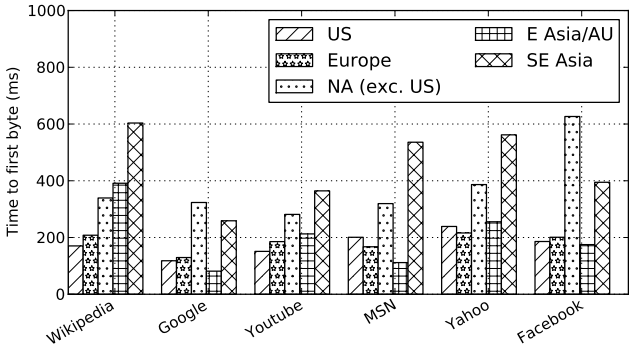


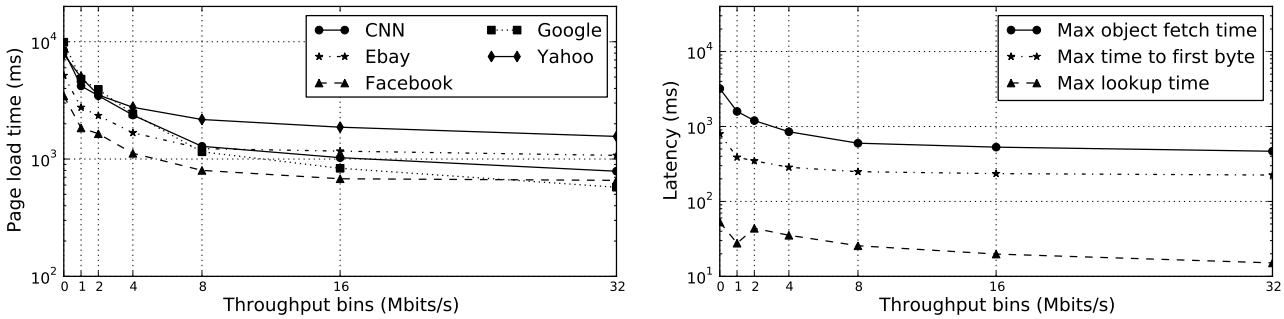
Figure 2: *Average time to first byte to six representative sites from BISmark nodes broken down by location.*

Figure 2 shows that the average time to first byte is in general higher in most regions outside the US. Site operators sometimes deploy content delivery networks to improve performance, though these results indicate that their prevalence is variable across the world. Sites with extensive content delivery networks (*e.g.*, Google, YouTube) have low median and maximum page fetch times, whereas other sites have more variable performance, both in the US and abroad.

4.2 Downstream Throughput

We study how page fetch time and its components vary with downstream throughput, using active measurements from the SamKnows testbed. We use the 95th percentile of the distribution of downstream throughput over the duration of the measurements for a given user to capture the capacity of each access link. We group access links according to downstream throughput into seven bins that reflect common ranges of Internet access plans in the dataset: 0–1 Mbits/s, 1–2 Mbits/s, 2–4 Mbits/s, 4–8 Mbits/s, 8–16 Mbits/s, 16–32 Mbits/s, and 32–64 Mbits/s. Figure 3a shows the median page fetch time for each category for five representative sites.

Median page fetch time decreases as downstream throughput increases, up to 8–16 Mbits/s. As downstream through-



(a) *Page fetch times* **(b)** *Components of page load time*

Figure 3: *Page fetch times decrease with downstream throughput, but only up to 8–16 Mbits/s. X-axis labels denote the start of each throughput bin (*e.g.*, "0" is the set of users with downstream throughput up to 1 Mbits/s.) (SamKnows)*

put increases further, page fetch times decrease only modestly. For example, the median time for CNN is 8.4 seconds for links with throughput 0–1 Mbits/s and 1.3 seconds when throughput is 8–16 Mbits/s. Yet, when downstream throughput exceeds 32 Mbits/s, the page fetch time is 790 ms, only slightly better than for links with 8–16 Mbits/s.

The fetch times in east Asia are similar to those in Europe, despite the disparity in downstream throughput for the nodes in these respective regions. For example, the average page fetch time for Facebook is 1.7 seconds in southeast Asia, 990 ms in east Asia and Australia, and 924 ms in Europe. Table 2 shows that east Asia has much higher average throughput than southeast Asia and Europe, but do not necessarily see a corresponding improvement in page load times because latency becomes a bottleneck as throughput increases.

Figure 3b shows how the maximum object fetch time, maximum time to first byte, and DNS lookup time decrease as throughput increases. For each group with a particular downstream throughput range, we plot the median of each of these values. As downstream throughput increases to 32–64 Mbits/sec, the object fetch time decreases from 3.2 seconds to 530 ms; in contrast, the time to first byte decreases from 800 ms to 230 ms and DNS lookup time decreases from about 50 ms to about 15 ms. Thus, as downstream throughput increases beyond 8–16 Mbits/sec, time to first byte and DNS lookup times become a larger component of page fetch time. When downstream throughput is less than 1 Mbit/s, last-mile buffering also increases latency.

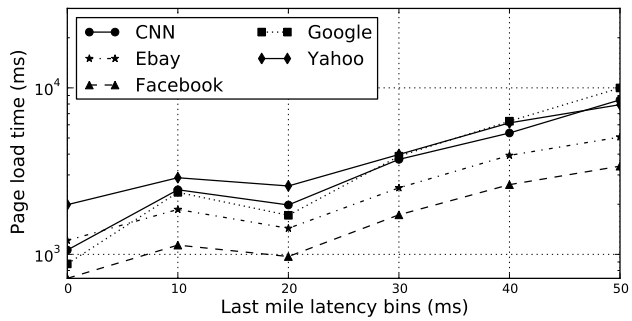
Improving Object Fetch Time Because most pages have many objects, the time to download all of the objects for a page can dominate other factors when downstream throughput is small. Hence, links with less than 8 Mbits/s will benefit the most from caching content inside the home. For links with higher downstream throughput, improving object fetch time is less important than reducing latency. We discuss the effects of latency in the next section.

4.3 Last-Mile Latency

A previous study observed that last-mile latency is a significant contributor to end-to-end latencies from homes in the United States [53]. To study the effect of last-mile latency on page fetch times, we group the SamKnows links into 10 ms bins according to the 10th percentile last-mile latency of their access link. Figure 4a shows the median page fetch time for links in each group; we show results from the same five representative sites. Page fetch times of all Web sites increase with last-mile latency, but the increase is not monotonic because other factors such as downstream throughput also affect page fetch time and some groups have more links than others: 75% of links have less than 10 ms last-mile latency. Increasing downstream throughput beyond 8 Mbits/s yields only marginal improvements in fetch time, but decreasing last-mile latency can consistently reduce fetch time.

Figure 4b shows the effect of last-mile latency on the minimum time to first byte and object fetch time for each transaction, as well as on DNS lookup time. The figure plots the median of each of these values for each group of links. Comparing the minimum object fetch time for a transaction to the minimum time to first byte shows that it can often take longer to establish the TCP connection to the Web server than to actually transfer the object, especially for smaller objects. In fact, we found that the average time to first byte ranges from 6.1% (for Yahoo) to 23% (for Wikipedia) of the total page fetch time. Figure 2 shows that the time to first byte vary from 250 ms in the United States to 800 ms in other parts of the world. We observed that last-mile latency can be up to 23% of the time to first byte.

Improving DNS Lookup Time Figure 4b shows that minimum lookup time increases as last-mile latency increases. Minimum lookup times are about 15 ms; lookup time increases as the last-mile latency increases. The only approach to eliminate the last-mile latency is to cache DNS records inside the home itself. When users configure their end-systems to use a DNS resolver elsewhere, they may be hurting their performance. Even when the resolver is well-placed in the access ISP network, DNS lookup latencies are bound by last-



(a) *Page fetch times*

(b) *Components of page load time*

Figure 4: *Page fetch times increase with last-mile latency. X-axis labels denote the start of each latency bin. (SamKnows)*

Placing a content proxy inside the home versus in a nearby network improve median page fetch times between 150 and 600 milliseconds, depending on the site.	5.1
Performing DNS caching inside the home can improve maximum lookup times by about 15–50 milliseconds, compared to simply relying on the ISP's DNS resolver.	5.2
Connection caching reduces median page fetch time by 100–600 milliseconds depending on the Web site.	5.2
Applying all three optimizations together can reduce Web page fetch times in a browser by upto 43%.	6.2

Table 6: *Highlights of Section 5 results.*

mile latency (previous work confirms this observation [1]).

Improving Time to First Byte Optimizations in the home cannot improve server processing time, but they can improve TCP connection setup time. The connection setup time depends on the round-trip latency to the server. Web service providers use content distribution networks to place servers as close to users as possible. Figure 2 shows that servers are in general closer to homes in the United States, but even users in the United States can experience slowdowns in TCP connection setup time due to the last-mile latency. Client-side optimizations such as connection caching [30] can reduce this latency by maintaining TCP connections to popular sites, thereby eliminating the setup time for new connections.

5. MITIGATING BOTTLENECKS

In this section, we analyze the effect of DNS caching, TCP connection caching, and content caching in a home network Web page performance using the method described in Section 3.2.2. The optimizations we evaluate are not new, but these experiments are a first attempt to quantify the benefits of deploying them in a home network. Table 6 outlines the main results and where we discuss them in more detail.

5.1 Mitigating Throughput Bottlenecks

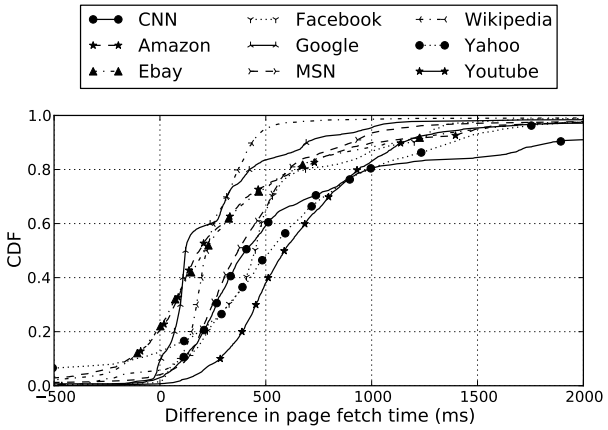


Figure 5: *Running a proxy in the home improves median page fetch times by 150–600 ms versus running a proxy in the ISP. (*Home Proxy *vs.* ISP Proxy *Measurement)*

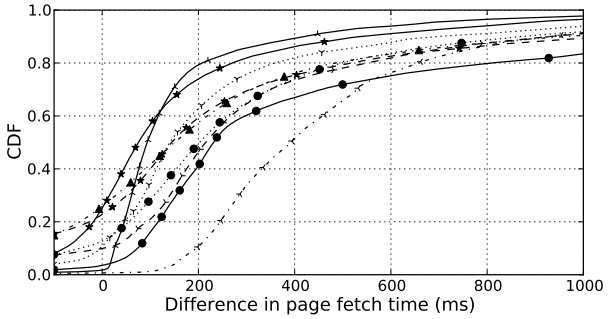


Figure 6: *Content caching reduces the median page fetch time by 75–400 ms over connection caching alone. For sites with more cacheable content, the benefit is greater (*Home Proxy *vs.* Home Connection Caching *Measurements)*

Benefits of content caching in the home vs. in the ISP We compare the Web page fetch time when using a remote HTTP proxy (the *ISP Proxy* measurement from Table 4) ver-

sus using a local HTTP proxy running on the router (the *Home Proxy* measurement). Figure 5 shows that a proxy in the home can offer a median improvement in page fetch time of 150–600 ms, depending on the site. Yahoo and CNN experience the most benefits, likely because these pages are larger and have many objects (Table 1). Caching objects in the home when possible mitigates the access link throughput bottleneck and reduces the latency to the Web content. A cache in the upstream ISP is still constrained by the access link's throughput and last-mile latency. For some sites, the remote proxy performs better than the home proxy about 20% of the time, perhaps because of varying access link characteristics across tests (due to cross traffic) or because the proxy is in a university network that potentially has better connectivity to these sites.

Benefits of content caching vs. connection caching Figure 6 shows the improvement in page fetch time due to content caching. We compute the improvement by subtracting the page fetch time for the *Home Proxy* experiment from that for the *Home Connection Caching* experiment. Caching content inside the home can decrease median page fetch times in the US by 75–400 ms, depending on the site. Sites with more cacheable content will benefit more. Our analysis shows that this benefit is even more significant for clients outside the US; at least 20% of clients experienced an improvement of 500 ms or more for all sites.

5.2 Mitigating Latency Bottlenecks

Benefits of DNS caching in the home vs. in the ISP To quantify the benefits of DNS caching in the home, we compare the maximum lookup time for a page load for the *Home DNS Cache* and *No Proxy, ISP DNS* cases. Figure 7 shows the CDF of the improvement in the maximum lookup time in a transaction. In the median case, placing a DNS cache in the home reduces the maximum lookup time by 15–50 ms, depending on the site. Clients outside the US can reduce their lookup time by several hundred milliseconds for certain sites (*e.g.*, Ebay, CNN) by caching lookups in the home.

Benefits of TCP connection caching in the home Figure 8 shows the additional improvement in page fetch time due to connection caching by measuring the difference between the fetch times for the *Home Connection Caching* and the *Home DNS Cache* measurements. The median improvement varies from 100–750 ms depending on the site. Ebay and Yahoo enjoy the most improvement in fetch times; because both sites require many objects from many domains to render, connection caching can significantly reduce the time to establish TCP connections.

6. REALIZING FASTER PAGE LOADS

In the previous section we saw how caching techniques can improve Web performance. In this section, we explore how to achieve these improvements in practice. We design

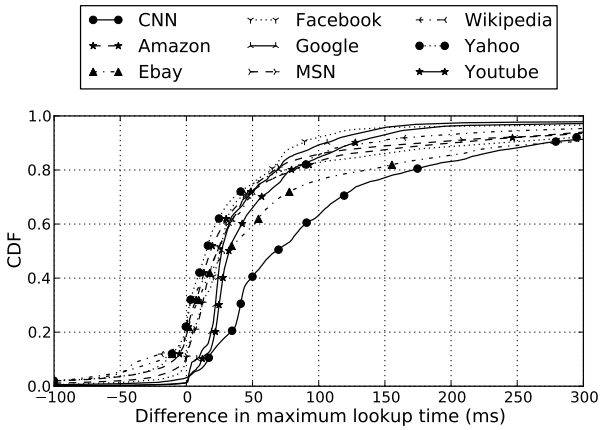


Figure 7: *Caching DNS in the home can reduce the maximum DNS lookup time by 15–50 ms. (*Home DNS *Measurement vs.* No Proxy, ISP DNS *Measurement)*

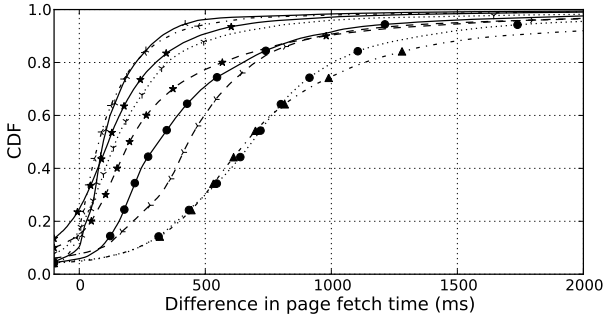
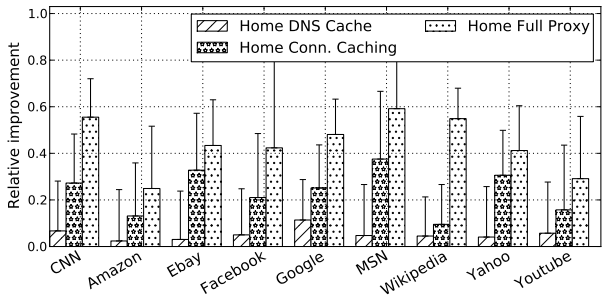


Figure 8: *Connection caching in the home can reduce median page fetch times by 100–750 ms.(*Home Connection Proxy *vs.* Home DNS *Measurements)*

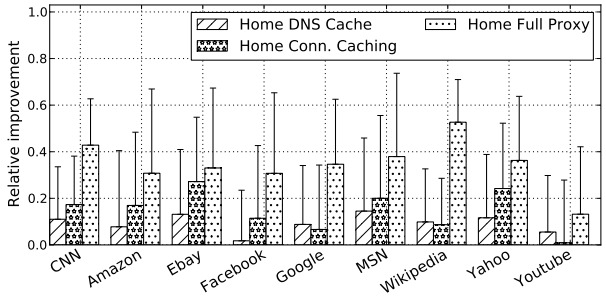
and implement a popularity-based system that prefetches and caches DNS records and maintains active TCP connections to popular domains. We evaluate the best case performance with both from the router, and also a real browser. We then analyze the effect of the system in real home settings.

6.1 Popularity-based Caching & Prefetching

We design, implement, and evaluate a *popularity-based caching and prefetching* system that prefetches DNS records and keeps TCP connections to Web servers active based on the sites that users in the household visit most frequently. We develop a proof-of-concept OpenWrt module, which is publicly available [47]. The system consists of a caching DNS resolver (`dnsmasq`) and an HTTP proxy (`polipo`), instrumented to track popular DNS lookups and HTTP domains respectively. By using a simple caching mechanism it tracks popularity and refreshes DNS lookups and maintains an active TCP connection to popular domains. The system aims to maximize the hit rate of the DNS and TCP connection caches. The two parameters that affect the hit rate are a) *the number of domains to be tracked*: the system actively



(a) *BISmark-US*



(b) *BISmark-nonUS*

Figure 9: *Average relative improvement in page fetch times for various optimizations, as observed from the* **router**. *Error bars denote standard deviation.*

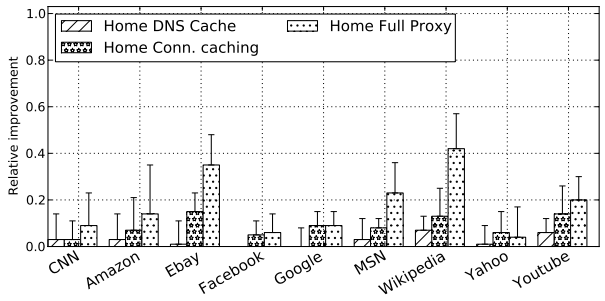


Figure 10: *Average relative improvement in page fetch times for various optimizations, as observed from the* **browser**. *Error bars denote standard deviation.*

prefetches DNS records and maintains active connections to these domains; the system maintains the two lists separately; and b) *timeout thresholds*: the system tracks the time since a lookup or a TCP connection was requested to a domain and removes the domain from the popular list if this time exceeds a threshold. The system maintains only one active connection per domain. It also does not prefetch content but exploits any content caching that `polipo` performs by default.

6.2 Benefit of Caching

We obtain the benefit of caching DNS and TCP connections by analyzing the improvement in page load times in the best-case scenario both at the router and from a browser.

Improvements at the Router. Since the Web tool from our earlier experiments yields page fetch times, we approximate the page load time as the sum of the fetch time and the maximum DNS lookup time over all of the domains in a transaction. We use the *No Proxy, ISP DNS* measurement as the baseline. We compute the *relative improvement* as $(b-v)/b$, where b is the load time without any optimizations, and v is the load time with the specified optimization. Figure 9a shows the relative improvement of each optimization relative to the baseline of performing no optimizations in BISmark-US. Applying all three optimizations improves load times by as much as 60%. Even without content caching, connection caching can yield up to a 35% improvement in load time, and DNS caching alone can improve load time by as much as 10%. Figure 9b shows relative improvement for clients outside the US The improvement is slightly less than for the users within the US simply because the absolute load times for users outside the US is already substantially higher (Figure 1). The actual improvements are correspondingly higher. This is the best-case scenario because the quick succession of tests will always induce a cache hit at the router.

Improvements in the Browser. We now evaluate whether caching improves page load times in real browsers, under the best-case scenario of a warm cache. We measure improvements in an end-host as described in Section 3.1), with the router immediately upstream of the host performing caching. Due to the variance in actual page load times that even a single browser experiences, we measure fetch times, but the improvements for the load time are similar (with higher variance). Figure 10 shows the relative improvement in the page fetch times due to the optimizations, as observed from the browser. DNS caching improves page fetch times by as much as 7%; connection caching and DNS caching improve fetch times by about 16%; all three optimizations together reduces fetch times by as much as 43%. The benefits as measured from the browser are somewhat different from the improvements we observed at the router and likely results from how the browser fetches objects; many browsers perform DNS prefetching and may also perform other optimizations.

6.3 Benefit of Prefetching

We now analyze whether popularity-based prefetching can improve cache hit rates using traces collected from twelve homes. To analyze DNS prefetching, we run a trace-driven simulation that maintains the DNS cache using the timestamp, hashed domain, and TTL of resolved DNS responses from the trace. When prefetching a domain, the simulation delays each incoming DNS response by a random interval between 50 and 200 ms; the TTL value is set according to the value in the trace. Requests for domains waiting for a DNS response are considered cache misses.

To characterize the benefit of TCP connection caching, the simulation maintains the TCP connection cache based on the

timestamp, five-tuple flow identifier, and duration of both the entire TCP connection and its connection establishment of all port 80 requests from the passive traces. For each new request, the simulation compares it with the entries in connection cache. In the event of a cache miss, or when a connection to an IP address is in use by another request, we establish a new connection with the flow and TCP connection establishment durations that correspond to the TCP connection in the trace. Upon a cache hit, the simulation denotes the cache entry to be in use for the duration of TCP connection minus the connection establishment time to that IP.

We find that baseline DNS cache hit rates (*i.e.*, without popularity-based prefetching) vary from 11–50%, depending on the home; the baseline cache hit ratio for TCP connection ranges from 1–8% across homes. By setting a cache size of 20 and a timeout threshold of 120 seconds for both DNS and connection prefetching, DNS hit ratios improve to 19–93%, representing at least a 40% improvement over the baseline in every home, Connection cache hit ratios increase to 6–21% across homes. These results indicate that there is scope for optimizing fetch times above the optimizations already existing in current browsers.

7. RELATED WORK

The vast body of work on Web performance ranges from modeling protocol behavior to empirical analysis. Past work has neither performed longitudinal studies of Web performance bottlenecks from broadband networks nor examined how various optimizations might mitigate these bottlenecks.

Modeling Web performance. Previous work has developed models for TCP and HTTP performance based on various network properties such as latency and packet loss [8, 9, 17, 18, 44]. Krishnamurthy and Wills analyzed the impact of variants of TCP (*i.e.*, Tahoe, Reno) and HTTP (*i.e.*, persistent, parallel, or pipelined) on Web download performance; they also studied the effects of network latency and server load on the observed performance [36]. Cohen *et al.* [23] studied how to manage the overhead of persistent connections on the server. WebProphet [38] analyzes dependencies between Web-page objects to predict browser-level response time. WISE uses passive traffic traces of Web downloads to help operators determine the effects of a configuration or infrastructure change on Web page load times [54].

Measuring Web performance. Barford and Crovella analyzed how server, client and network delays each contribute to HTTP 1.0 transaction times [10]; the authors placed virtual clients on several campus networks while requesting files from a virtual Web server. They found that for small file sizes, the server load is the dominant factor for performance, while for large file sizes, the network properties dominate. More recent work has studied the performance of CDNs [33] and caching [27]. Akella *et al*. [3–6] studied the effects of the performance of 68 CDN sites across 17 cities, focusing on how server multihoming can improve CDN performance for clients. "WhyHigh?" identifies and explains cases where certain set of clients experience higher Web page load times [37]. Inside the home, Erman *et al.* find that 13% of all HTTP traffic is from "non traditional" sources, with about 5% originating from consumer electronics devices [28]. Butkiewicz *et al.* recently studied how the complexity of modern Web sites may contribute to slower page load times and found that more than 60% of the Web sites they profiled retrieve content from 5 non-origin sources that contribute to more than 35% of the bytes downloaded [15].

8. CONCLUSION

This paper presents the first large-scale, longitudinal study of Web performance bottlenecks in broadband access networks. We characterize performance from more than 5,000 broadband access networks to nine popular Web sites and identify factors that create Web performance bottlenecks. Our results show that as broadband access speeds continue to increase, other network characteristics, such as latency, become performance bottlenecks. Page load times stop improving as throughput rates increase beyond 8–16 Mbits/s. Last-mile latency contributes to the time required for performing DNS lookups, completing TCP connection setup, and downloading Web objects and can be up to 23% of the time to first byte. With an eye towards reducing this latency, we evaluate how three different caching optimization techniques (DNS caching, TCP connection caching, and content caching) improve performance. Caching content inside the home versus in a nearby network improves median page fetch times by up to 600 ms, while connection caching improves fetch times by 100–600 ms. Performing all three optimizations together can reduce page load times by 30–60%.

We believe that the home router may ultimately be a reasonable location for deploying the caching and prefetching optimizations that we have implemented, in addition to the similar optimizations that browsers already implement. Any device that connects to the home network will directly benefit, even when applications are not running on top of a browser. Indeed, as Singhal and Paoli state, "apps—not just browsers—should get faster too" [49]. Our publicly available OpenWrt module for popularity-based prefetching may serve as yet another important component for reducing page load times in home networks.

REFERENCES

[1] B. Ager, W. Mühlbauer, G. Smaragdakis, and S. Uhlig. Comparing dns resolvers in the wild. In *Proceedings of the 10th annual conference on Internet measurement*, IMC '10, pages 15–21, New York, NY, USA, 2010. ACM.

[2] B. Ager, F. Schneider, J. Kim, and A. Feldmann. Revisiting cacheability in times of user generated content. In *INFOCOM IEEE Conference on Computer Communications Workshops, 2010*, pages 1–6. IEEE, 2010.

[3] A. Akella, B. Maggs, S. Seshan, and A. Shaikh. On the performance benefits of multihoming route control. *IEEE/ACM Transactions on Networking*, 16(1), Feb. 2008.

[4] A. Akella, B. Maggs, S. Seshan, A. Shaikh, and R. Sitaraman. A measurement-based analysis of multihoming. In *Proc. ACM SIGCOMM*, Karlsruhe, Germany, Aug. 2003.

[5] A. Akella, J. Pang, B. Maggs, S. Seshan, and A. Shaikh. A comparison of overlay routing and multihoming route control. In *Proc. ACM SIGCOMM*, Portland, OR, Aug. 2004.

[6] A. Akella, S. Seshan, and A. Shaikh. Multihoming performance benefits: An experimental evaluation of practical enterprise strategies. In *Proc. USENIX Annual Technical Conference*, Boston, MA, June 2004.

[7] M. Al-Fares, K. Elmeleegy, B. Reed, and I. Gashinsky. Overclocking the Yahoo! CDN for faster Web page loads. In *Proceedings of Internet Measurement Conference*, 2011.

[8] E. Altman, K. Avrachenkov, and C. Barakat. A stochastic model of tcp/ip with stationary random losses. In *ACM SIGCOMM*, 2000.

[9] M. Arlitt, B. Krishnamurthy, and J. Mogul. Predicting Short-transfer Latency from TCP arcana: a Trace-based Validation. In *Proc. ACM SIGCOMM Internet Measurement Conference*, New Orleans, LA, Oct. 2005.

[10] P. Barford and M. Crovella. Critical path analysis of tcp transactions. In *IEEE/ACM Transactions on Networking*, 2000.

[11] M. Belshe. A Client-Side Argument for Changing TCP Slow Start. `http://goo.gl/UDKXz`.

[12] M. Belshe. More Bandwidth Doesn't Matter (much). `http://goo.gl/OIv47`.

[13] BISmark Web Performance data. `http://dl.dropbox.com/u/13045329/bismark_webperf_data.tar.gz`.

[14] J. Brutlag. Speed matters for Google Web search. `http://services.google.com/fh/files/blogs/google_delayexp.pdf`, June 2009.

[15] M. Butkiewicz, H. Madhyastha, and V. Sekar. Understanding website complexity: Measurements, metrics, and implications. In *Proc. Internet Measurement Conference*, Berlin, Germany, Nov. 2010.

[16] R. Caceres, F. Douglis, A. Feldmann, G. Glass, and M. Rabinovich. Web proxy caching: The devil is in the details. June 1998.

[17] J. Cao, W. S. Cleveland, Y. Gao, K. Jeffay, F. D. Smith, and M. Weigle. Stochastic models for generating synthetic http source traffic. In *IN PROCEEDINGS OF IEEE INFOCOM*, 2004.

[18] N. Cardwell, S. Savage, and T. Anderson. Modeling tcp latency. In *Proc. IEEE INFOCOM*, Tel-Aviv, Israel, Mar. 2000.

[19] Y. Cheng and Others. *TCP Fast Open*. IETF, Sept. 2011. `http://www.ietf.org/id/draft-cheng-tcpm-fastopen-00.txt`.

[20] J. Chu and Others. *Increasing TCP's Initial Window*. IETF, Oct. 2011. `http://tools.ietf.org/html/draft-ietf-tcpm-initcwnd-01`.

[21] E. Cohen and H. Kaplan. Prefetching the means for document transfer: A new approach for reducing Web latency. In *Proc. IEEE INFOCOM*, volume 2, pages 854–863, Tel-Aviv, Israel, Mar. 2000.

[22] E. Cohen and H. Kaplan. Proactive caching of DNS records: Addressing a performance bottleneck. In *Symposium on Applications and the Internet (SAINT)*, pages 85–94, 2001.

[23] E. Cohen, H. Kaplan, and J. Oldham. Managing tcp connection under persistent http. *Comput. Netw.*, 31:1709–1723, May 1999.

[24] DNS Prefetching (or Pre-Resolving). `http://blog.chromium.org/2008/09/dns-prefetching-or-pre-resolving.html`.

[25] Dnsmasq. `http://thekelleys.org.uk/dnsmasq/doc.html`.

[26] N. Dukkipati, T. Refice, Y. Cheng, J. Chu, T. Herbert, A. Agarwal, A. Jain, and N. Sutin. An argument for increasing tcp's initial congestion window. *SIGCOMM Comput. Commun. Rev.*, 40:26–33, June 2010.

[27] J. Erman, A. Gerber, M. Hajiaghayi, D. Pei, and O. Spatscheck. Network-aware forward caching. In *Proceedings of the 18th international conference on World wide web*, 2009.

[28] J. Erman, A. Gerber, and S. Sen. Http in the home: It is not just about pcs. In *HomeNets: ACM SIGCOMM Workshop on Home Networks*, Aug. 2011.

[29] FCC Measuring Broadband America Report. `http://www.fcc.gov/measuring-broadband-america/2012/july`, July 2012.

[30] A. Feldmann, R. Caceres, F. Douglis, G. Glass, and M. Rabinovich. Performance of web proxy caching in heterogeneous bandwidth environments. In *Proc. IEEE INFOCOM*, New York, NY, Mar. 1999.

[31] S. Gribble and E. Brewer. System Design Issues for Internet Middleware Services: Deductions from a Large Client Trace. In *Proc. 1st USENIX Symposium on Internet Technologies and Systems (USITS)*, Monterey, CA, Dec. 1997.

[32] S. Ihm and V. Pai. Towards understanding modern web traffic. In *Proc. Internet Measurement Conference*, Berlin, Germany, Nov. 2010.

[33] A. jan Su, D. R. Choffnes, A. Kuzmanovic, and F. E. Bustamante. Drafting behind akamai (travelocity-based detouring. In *Proc. ACM SIGCOMM*, Pisa, Italy, Aug. 2006.

[34] J. Jung, A. W. Berger, and H. Balakrishnan. Modeling TTL-based Internet Caches. In *IEEE Infocom 2003*, San Francisco, CA, April 2003.

[35] J. Jung, E. Sit, H. Balakrishnan, and R. Morris. DNS Performance and the Effectiveness of Caching. In *Proc. ACM SIGCOMM Internet Measurement Workshop*, San Fransisco, CA, Nov. 2001.

[36] B. Krishnamurthy and C. Wills. Analyzing factors that influence end-to-end Web performance. In *Proc. Twelfth International World Wide Web Conference*, Amsterdam, The Netherlands, May 2000.

[37] R. Krishnan, H. V. Madhyastha, S. Jain, S. Srinivasan, A. Krishnamurthy, T. Anderson, and J. Gao. Moving beyond end-to-end path information to optimize CDN performance. In *Proc. Internet Measurement Conference*, 2009.

[38] Z. Li, M. Zhang, Z. Zhu, Y. Chen, A. Greenberg, and Y.-M. Wang. Webprophet: Automating performance prediction for web services. In *Proc. 7th USENIX NSDI*, San Jose, CA, Apr. 2010.

[39] Link Prefetching FAQ. `https://developer.mozilla.org/En/Link_prefetching_FAQ`.

[40] S. Lohr. For Impatient Web Users, an Eye Blink Is Just Too Long to Wait. `http://www.nytimes.com/2012/03/01/technology/impatient-web-users-flee-slow-loading-sites.html`, Mar. 2012.

[41] J. C. Mogul, Y. M. Chan, and T. Kelly. Design, implementation, and evaluation of duplicate transfer detection in HTTP. In *Proc. First Symposium on Networked Systems Design and Implementation (NSDI)*, San Francisco, CA, Mar. 2004.

[42] H. Nielsen, J. Gettys, A. Baird-Smith, E. Prud'hommeaux, H. W. Lie, and C. Lilley. Network performance effects of http/1.1, css1, and png. In *Proc. ACM SIGCOMM*, Cannes, France, Sept. 1997.

[43] OECD. *OECD Communications Outlook*. OECD Publishing, July 2011.

[44] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP Throughput: A Simple Model and its Empirical Validation. In *Proc. ACM SIGCOMM*, pages 303–323, Vancouver, British Columbia, Canada, Sept. 1998.

[45] V. Padmanabhan and J. Mogul. Using predictive prefetching to improve world wide web latency. *ACM SIGCOMM Computer Communication Review*, 26(3):22–36, 1996.

[46] Polipo. `http://www.pps.jussieu.fr/~jch/software/polipo/`.

[47] OpenWRT Module for Popularity-based Prefetching. `http://dl.dropbox.com/u/13045329/popularity_prefetch.tgz`.

[48] Project bismark. `http://projectbismark.net`.

[49] S. Singhal and J. Paoli. Speed and Mobility: An Approach for HTTP 2.0 to Make Mobile Apps and the Web Faster, Mar. 2012. `http://goo.gl/1uWCl`.

[50] SamKnows Webget Tool. `https://files.samknows.com/~gpl/`.

[51] S. Souders. Velocity and the bottom line. `http://radar.oreilly.com/2009/07/velocity-making-your-site-fast.html`, July 2009.

[52] SPDY: An experimental protocol for a faster web. `http://www.chromium.org/spdy/spdy-whitepaper`.

[53] S. Sundaresan, W. de Donato, N. Feamster, R. Teixeira, S. Crawford, and A. Pescapè. Broadband internet performance: A view from the gateway. In *Proc. ACM SIGCOMM*, Toronto, Ontario, Aug. 2011.

[54] M. B. Tariq, A. Zeitoun, V. Valancius, N. Feamster, and M. Ammar. Answering "What-if" Deployment and Configuration Questions with

WISE. In *Proc. ACM SIGCOMM*, Seattle, WA, Aug. 2008.

[55] A. Wolman, G. M. Voelker, N. Sharma, N. Cardwell, A. Karlin, and H. M. Levy. On the scale and performance of cooperative web proxy caching. In *Proc. 17th ACM Symposium on Operating Systems Principles (SOSP)*, Kiawah Island, SC, Dec. 1999.

[56] W. Zhou, Q. Li, M. Caesar, and P. Godfrey. Asap: A low-latency transport layer. In *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*, page 20. ACM, 2011.