Modeling Mobile Web Characteristics for Energy Optimized Delivery

Troy Johnson

A thesis submitted in partial fulfillment of
the requirements for the degree of
Master of Science

Department of Computer Science

Central Michigan University
Mount Pleasant, Michigan
September 2014

ACKNOWLEDGMENTS

ABSTRACT

Modeling Mobile Web Characteristics for Energy Optimized Delivery

by Troy Johnson

As mobile traffic and data consumption continue to rise, there is a growing need to investigate increased energy efficiency and optimizations to reduce the bandwidth when browsing the mobile web. To determine the reduction in energy consumption of mobile devices, there is also a need for a way to measure the energy consumption of mobile devices. By investigating the composition and characteristics of mobile web pages, statistical models can be derived for describing the characteristics of a typical mobile web page, such as the individual response sizes and expiration ages of responses that mobile browsers request for web pages. HTTP Archive will be a great source of data that may be utilized to derive models for describing the mobile web. Additionally, this pool of data is updated on a bimonthly basis, providing a constantly updated pool of data to update the developed models with and validate them. These models can then in turn be used to provide more accurate results when estimating the possible energy and bandwidth savings of optimizations by using these models for generating artificial web pages that will contain characteristics that closely resemble those characteristics often found on the actual mobile web. Investigating the models and data further, they can be extended to create prediction models that will describe the growing mobile web for future years. With these models in place, they can be applied to projects for optimizing energy and bandwidth consumption on mobile devices, such as the possible energy and bandwidth savings that can result from cache forwarding between desktop computers and mobile devices. To measure the possible energy consumption savings from these projects, a low cost test bed for measuring the power consumption of mobile devices can be employed as a baseline. The measurement data ,experimentally determined, can then be used for follow-up performance analyses.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER I

INTRODUCTION

The increased ubiquity of mobile devices has accelerated the growth of mobile data usage tremendously. Cisco, Inc. predicts that the total monthly global mobile data traffic will exceed 15 exabytes by 2018 [1]. With this increase in traffic also comes the increased need for optimizations on how content is delivered to mobile devices to reduce the load being put on mobile networks. To determine the potential achievable savings of these optimizations, simulations can be performed which simulate a typical user session when browsing the mobile over an extended period of time. These simulations can use data from an actual pool of mobile web pages, or statistical models that be developed that describe web page. HTTP Archive [30] is one such tool that provides pools of data about the characteristics of the most popular desktop and mobile web pages. Statistical models have the advantage of being able to be extended to show what we expect web pages to look like in the future and don't require the overhead required for having a pool of data containing all the characteristics of the most popular web pages.

HTTP Archive

The data contained in HTTP Archive is mined and utilized for several of the projects in the following chapters. HTTP Archive is an online repository of web performance information containing information on both desktop and mobile versions of websites. Information gathered includes all the details about the responses web pages make such as the size and expiration age. An exhaustive list of the characteristics gathered by HTTP Archive can be found on their website (http://httparchive.org). The information gathered only pertains to the characteristics of the landing page of the web site. HTTP Archive gathers their data using a private instance of WebPageTest (http://www.webpagetest.org). Ac-

cording to HTTP Archive, the determination of which web pages are to be archived is based on set

derived from the union of the following sets of sources: Alexa 500, Alexa US 500, Alexa 10,000,

Fortune 500, Global 500, and Quantcast10K.

CHAPTER II

An Inexpensive Testbed for Mobile Device Power Measurement

Introduction

Over the last few decades, there has been an enormous increase in the ubiquity of mobile devices. With this increase, has also come the increase in demand for data-driven services and Cisco, Inc. predicts this demand to continue [1] into the foreseeable future. The battery consumption of mobile devices represents a limiting bottleneck and thus power optimizations suggestions have been suggested [2] to reduce energy consumption on mobile devices. Software-based energy profilers do exist [3], however they are not always feasible for implementing in a straightforward manner or desirable due to rapid development cycles. To overcome these barriers, a real world test bed can be implemented to perform measurements for the power consumptions on mobile devices to obtain a real life gauge for power estimations of optimizations. In this chapter, I will discuss the development and set up of a testbed for this purpose.

Hardware Configuration

The main component in this testbed is the mobile device. This can be realized by using a smartphone and replacing the battery with connectors to a power supply; alternatively one of the common development board packages, such as Pandaboard (see www.pandaboard.org) or Wandboard (see www.wandboard.org) packages. Development boards and smartphones can be utilized together with the Android operating system, which provides log output via USB to the measurement control device, which can be a regular PC or another development board with Android debugging support. The mobile device is then networked with a wireless access point, which allows for wired and wireless evaluations. The switchable power supply has an external serial or USB port to communicate the

current and power in small time intervals to the control device. A BK Precision 1696 switchable power supply is utilized, as it offers fine granularity in power, current, and time intervals. While other equipment, such as an Arduino with custom circuits, were used in other measurement approaches, these power supplies are common lab equipment and offer overall robust features.

Software Configuration

The software components are comprised of several Python scripts that execute the Android Debugging Bridge (ADB) and capture the output either to a local file or allow sending of the output to a remote receiver, as illustrated in Figure 1. These scripts allow for easy customization on the locally connected control device or at a remote location, e.g., filtering by specific events in the log. Similarly, a locally executing script captures the output from the power supply and is enabled to forward the data to a remote location as well.

Demonstration Description

To demonstrate the usefulness of this testbed, two different aspects of the measurement setup were utilized using an example Android application that performs web requests (wired and wirelessly). This application makes requests to a local proxy server, either serially or in parallel, and the proxy server goes out and fetches what the phone requested. The workflow for the application can be seen in figure 2. Both, wired and wireless access scenarios were exhibited for accessing a remote web service and retrieving results in order to demonstrate the functionality of the testbed. With these demonstrations, real time visualizations of the data about power consumption were created and stored on log files on a remote computer where they can be readily parsed for automatic evaluation of application power consumption on the mobile device.
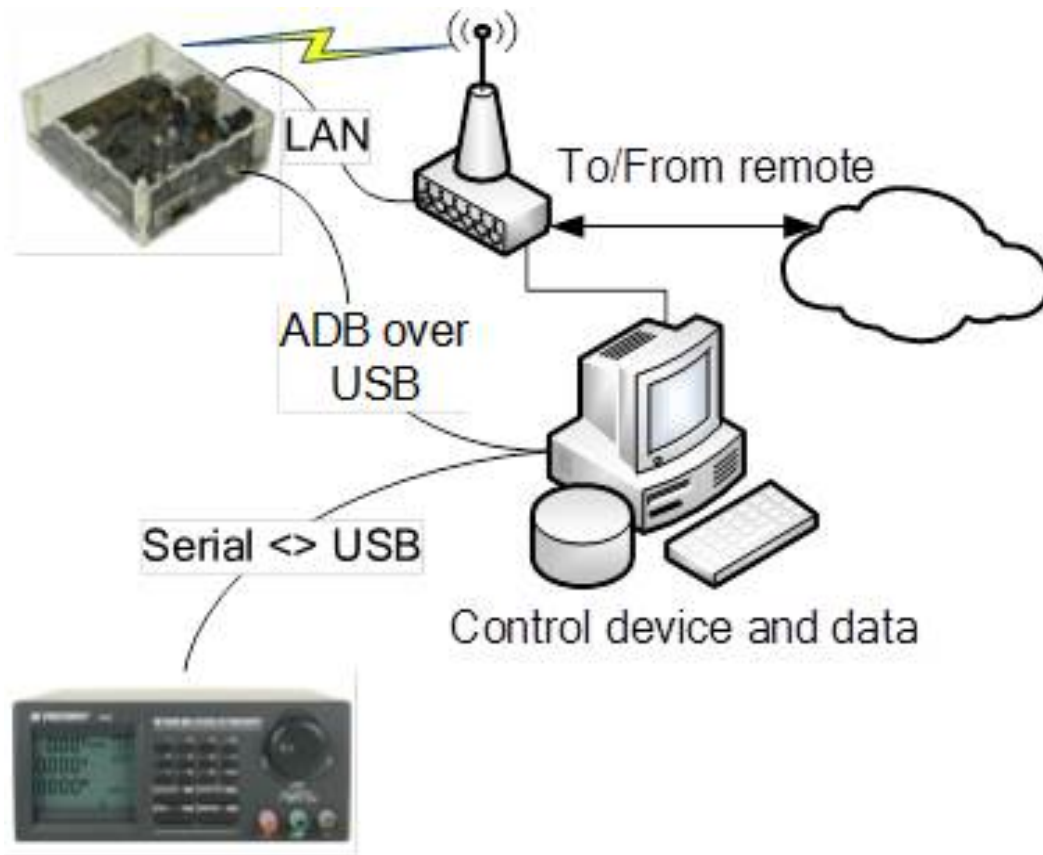
Figure 1. Illustration of measurement testbed.
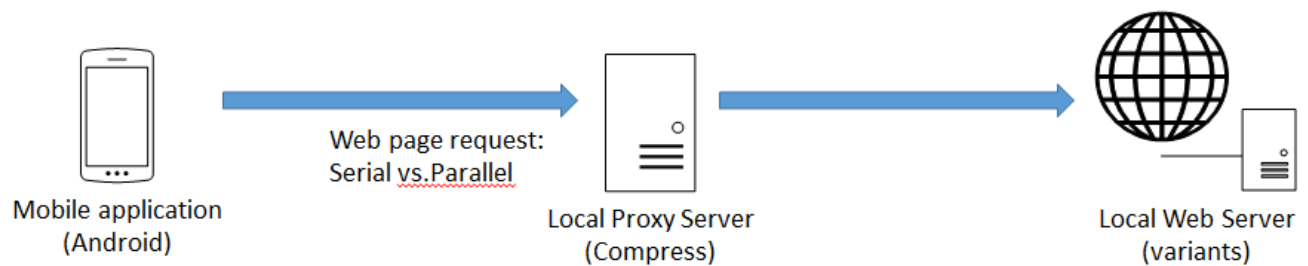
Figure 2. Workflow of mobile application tested on testbed.

CHAPTER III

Power Consumption Overhead for Proxy Services on Mobile Device Platforms

Current predictions by Cisco, Inc. show that the amount of data that users consume has increased significantly and will continue to increase into the foreseeable future [1]. Previous studies have shown that the network interfaces of mobile devices consume much of the limited battery life [4]. Thus, heavy research efforts have been poured into studying the possibilities of energy efficient mobile data delivery. Some research avenues have middleware that acts as an on-device proxy service to realize benefits or enable new interaction paradigms, such as display networks [5] or mobile content sharing [6],[7]. To determine whether or not local proxy servers result in a large overhead in terms of power consumption and time delays, the measurement framework testbed described in Chapter II can be implemented to determine what kind of overheads can be expected from utilizing local proxy servers.

Methodology and Metrics

Measurement Setup

To determine energy consumption, a set up similar to the one descibed in Chapter II is utlized. At the core of the setup, a Pandaboard ES mobile software development platform is utilized, which features a Texas Instruments OMAP 4460 dual core ARM Cortex-A9 processor with 1 GB of DDR2 RAM, SMSC 10/100 Mbps Ethernet port, and LS Research WLAN/Bluetooth wireless module, next to other components. (Please refer to http://www.pandaboard.org for more details. The open-source Android distribution (version 4.1.2, âĂŸJelly BeanâĂŹ) is used as the operating system software for the mobile device. An illustration of the overall measurement setup can be seen in Figure 3. The Pandaboard is powered by a BK Precision 1696 switchable power supply, which features serial port

6

access to read out voltage and ampere values over time. The power supply is connected to a Linux desktop computer serial port which timestamps the values obtained over time to measure the power consumption incurred by the Pandaboard. The Pandaboard is connected through a 1 Gbps maximum speed Ethernet campus network, which eliminates potential bottlenecks. For wireless measurements, an externally connected WLAN antenna is utilized to connect to the campus network through a dedicated WLAN access point, again eliminating potential bottlenecks for the amounts of data considered throughout. It's also important to note that a combination of input devices and an external monitor were connected as well. On the server side, a locally hosted virtual machine next to Internet-routed web requests is utilized. The local server employs the Debian Linux distribution as its' operating system with the Apache2 HTTP server and the popular Video Lan Client (VLC) as the media streaming application. A pre-encoded video sequence of the popular open-source movie Tears of Steel (see http://www.tearsofsteel.org for more information) is streamed utilizing HTML5 video streaming. The video-only sequence was transcoded offline into a resolution of 864 ÃŮ 480 at 24 frames per second in the Theora video codec and encapsulated using the OGG container format, both commonly utilized for HTTP video streaming âĂIJon the webâĂİ and suitable for mobile playout. The resulting video bit stream has a duration of 12 minutes, 14 seconds and an average bit rate of 1.42 Mbps. The bit rate in turn falls well within range of the network bandwidth capacity.

Mobile SOCKSv5 Proxy Server

Several implementations of the SOCKSv5 standard exist to date [8], which allow utilization of a remotely hosted standard-conforming SOCKSv5 proxy server (typically from a desktop computer through an organizational server). Mobile implementations, however, are less frequent. One example of an implementation for the Android operating system is the anonymity generating Orbot application

(see http://www. torproject.us for more details), which routes traffic into the TOR network and contains âĂIJproxificationâĂİ methods for applications as well (i.e., transparently forcing the usage of the proxy through, e.g., modifications of the iptables firewall). A basic Android service application was generated that is based on the jSOCKS proxy server implementation [9], which is open source (entirely written in JAVA) and does not require any privileges, such as root level system access. As the service is executed within the Dalvik VM utilized on Android devices, it incurs a minimal computational overhead when compared to native applications. This approach, however, is commonplace to allow broadest application compatibility and encouraged for developers of the platform.

Performance Metrics

In the following, we briefly outline the metrics used to evaluate the performance of either scheme. Initially, we capture the reported voltage level *v(tl)* [V] and the current *i(tl)* [A] as reported by the power supply and timestamped at time *tl* on the connected desktop computer. We similarly calculate the instant power consumption as *p(tl) = v(tl) * i(tl)* [W]. As the reported values are instantaneous snapshots in time from *l = 0* at *t(l) = 0* (denoting the first measurement) to *l = L*, which happened at *t(l) = T* (whereby *T* denotes the last measurement), we calculate the time passed between consecutive measurement instances as $t(l) = t(l) - t_{l-1}$. To determine the energy that was used in the *l* - th measurement period, we calculate $e(l) = t(l) * w(t_l)$ [J]. We denote the energy that was used in a measurement period up to *t(l)* as

$$o = \frac{\overline{E}_{Proxy} - \overline{E}_{Direct}}{\overline{E}_{Direct}}, \tag{III.1}$$

8
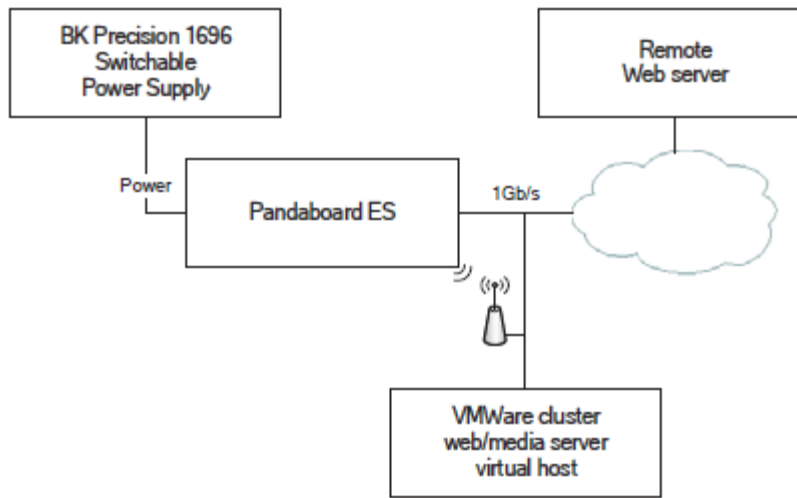
Figure 3. Overview of the measurement setup

Performance Evaluation For Web Request

To perform a representative evaluation of frequent HTTP web requests, web requests for Google's home page were utilized. The goal of this particular measurement scenario is to evaluate the performance impact of frequent requests through the local proxy server, which has to perform the additional connection tasks each time a request is made. A direct measurement application for Android was developed, which will request http://www.google.com without further resolving any HTTP objects within. Individual requests are followed by a sleep period of 2 seconds for both, direct requests and requests through the mobile SOCKSv5 proxy service. As requests are made without utilizing a browser, no caching is involved client-side.

Fixed Network

In the fixed network scenario, the Pandaboard is connected through wired Ethernet to the campus network while performing the web requests. The requests typically coincide with high power consumption levels, as illustrated in Figure 4 for an exemplary 100 web requests with both configurations. We observe that both approaches exhibit an initial "spike" behavior and an otherwise

low level (with some general noise due to overall device activities). There is no immediately visible trend for the momentary power consumptions, as in both approaches, there are several bursty periods of slightly elevated consumptions on top of the actual web requests. Next, we evaluate the total (compounded) energy consumption that is observed when performing these requests for a certain period of time. We illustrate 100 subsequent requests directly and through the mobile proxy server in Figure 5. Initially, it is observed that despite the short-term fluctuations, there is a linear increase in the energy consumed while using either approach. More significantly, there is no immediately noticeable significant difference between the approaches, which is indicated by the almost indistinguishable values in the plot. Lastly, comparing the overhead between the approaches numerically in Table III.1, where the 300 highest levels of energy consumption measured for periods of placing 300 requests were analyzed. It's also notable that the direct approach results in a higher average level of energy consumption (albeit with a larger variability), whereas the proxy-based approach yields a lower average and variability of energy usage values determined. Overall, this results in an overhead of o = âĹŠ0.0563, which presents an initially counter-intuitive result. (Differently worded, by utilizing the mobile proxy server consuming additional CPU cycles, potential energy savings of 5.6% could be realized without requiring any additional modifications.) Taking into account that partially significant variability exists due to some outliers in the total duration (as some measurement points can exhibit significant delays or coincide with other unrelated system activities), both approaches are very similar.

Interface	Approach	Average[J]	Standard Deviation [J]	Confidence Interval (99 %)
LAN	Direct	0.0912	0.0139	0.0021
LAN	Proxy	0.0860	0.0062	0.0009
WLAN	Direct	0.0900	0.0125	0.0019
WLAN	Proxy	0.0902	0.0089	0.0013

Table III.1. Summary values for 300 direct and proxy-routed web requests over traditional ethernet and wireless LAN networks.
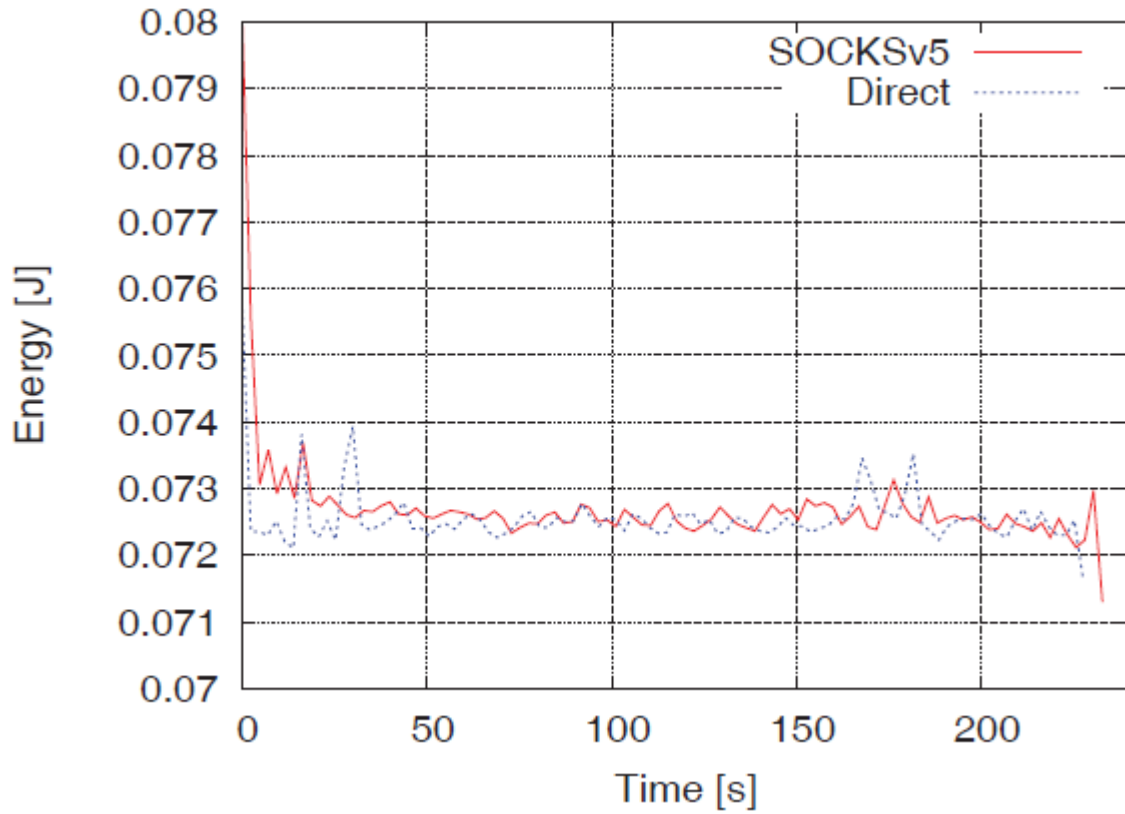
Figure 4. Energy consumption while performing 100 web requests directly or through a mobile SOCKSv5 service, smoothed over time.
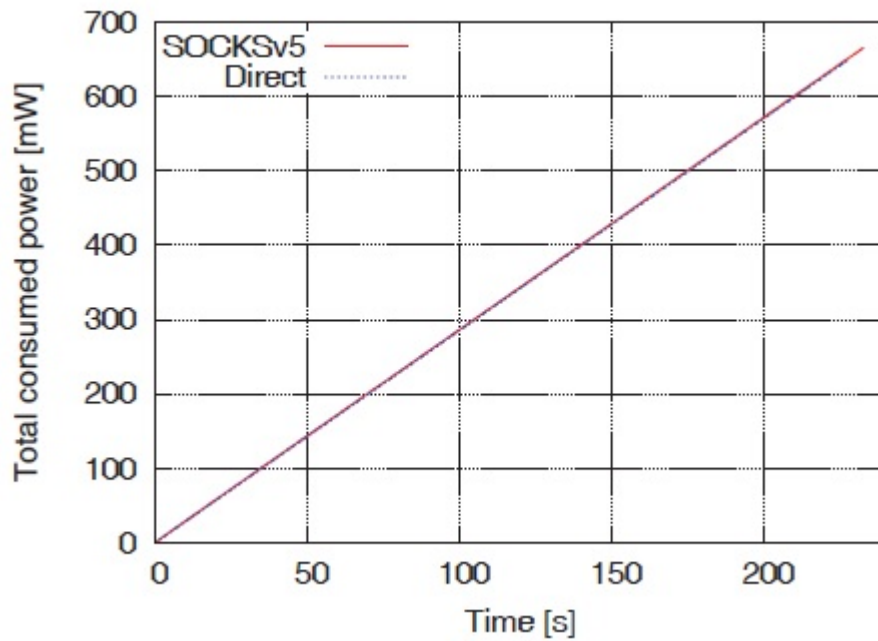
Figure 5. Compounded energy consumption for performing 100 web requests directly or through a mobile SOCKSv5 service.

11

Wireless Network

Shifting the evaluation to HTTP requests made over the wireless network interface, we present

our results in Table III.1. (We note that a graphical evaluation would yield results similar to those

presented in Figures 4 and 5.) We initially note that both approaches are very close with respect

to their measured average energy consumption, resulting standard deviations, and narrow confidence

intervals.

Comparing these results with those obtained for the Ethernet scenario, which outlines the base case

without active wireless communications overheads, we do not note a significant difference in the

average energy usage for the individual web requests.

Impact of Web Request Variations

Motivated by the closeness of requests, the next step is to more closely evaluate the impact of

the web request size over a wireless LAN on the overall power consumption. To limit the impact that

external networks can have (such as different delays), the direct performance comparison is performed

within the on-campus VM environment illustrated in Figure 3. A dedicated virtual machine uses the

Apache2 web server and hosts a Python script that generates a requested number of bytes, additionally

eliminating potential caching impacts. 100 repeated measurements are performed for each different

web request size and significant outliers are deleted.

Power Consumption

The average energy used per different request size is illustrated in in Figure 6. It's observed

that the mobile SOCKSv5 proxy approach always incurs a penalty over the direct connection, which

is readily explained by the additional local processing overhead on the device. Furthermore, it can be
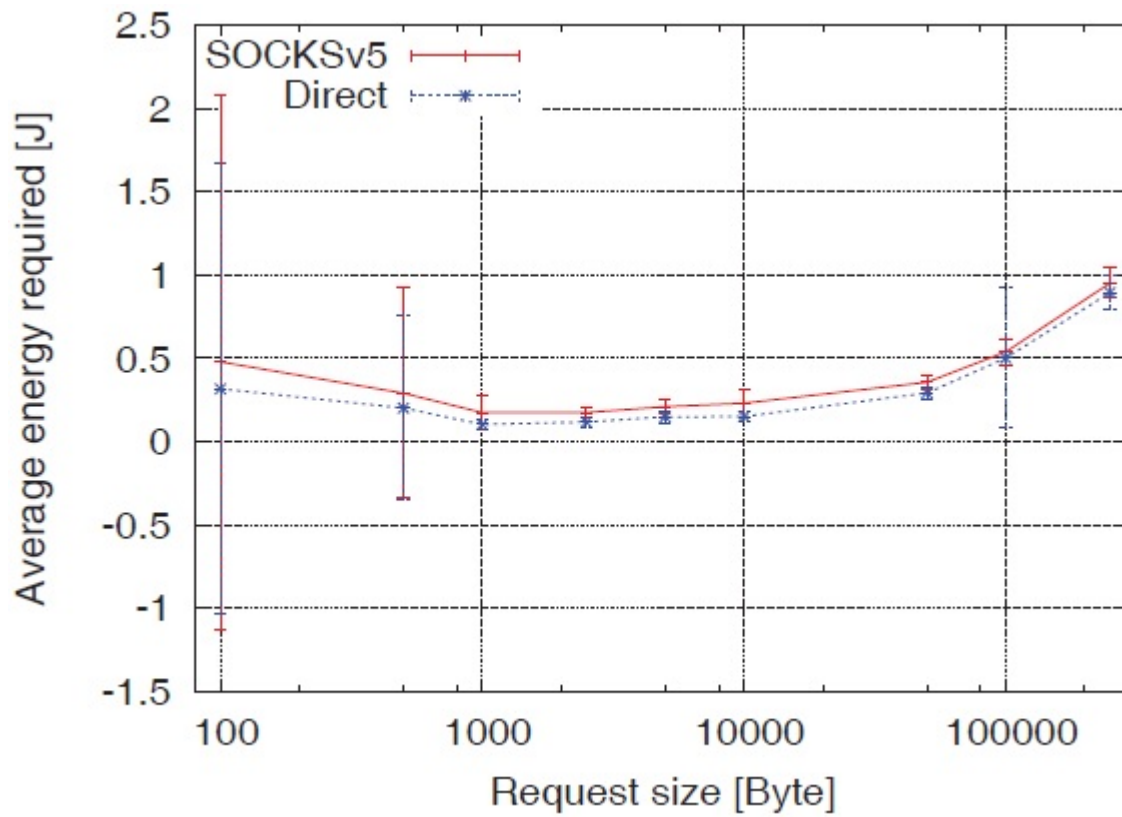
Figure 6. Average energy consumption and standard deviation for requesting different amounts of data from a local server using a direct or mobile SOCKSv5 proxy server.
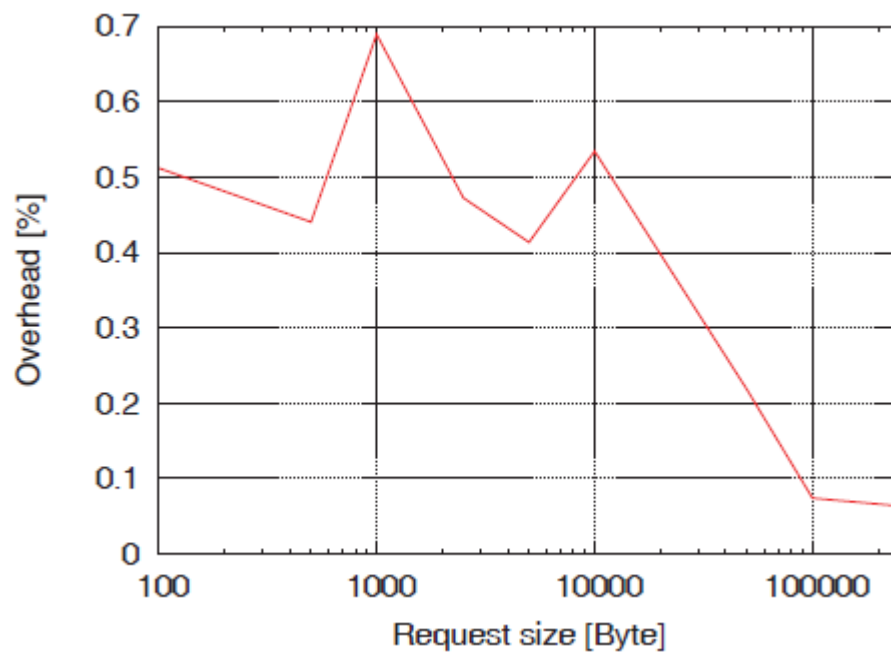
Figure 7. Average energy consumption overhead for requesting different amounts of data from a local server

13

noted that both depicted connection methods follow a âĂIJslumpâĂİ-like behavior. Taking the overall variability of measurements into account, until the amount of requested data approaches the single packet payload region, the lowest energy usage is recorded. This initial behavior can be explained by the overhead to establish the initial server connection, which is dominant for the small request size regions. With a further increase of the payload to very large sizes, the number of required transmissions drastically increases, which now accounts for the majority of the energy consumption (rather than the initial connection setup). This increase in the number of packets is now causing longer periods of energy expensive network activities, causing the rise in energy consumption illustrated. The overall absolute difference between the employment of a mobile proxy and direct connection, however, remains fairly constant, which indicates that the processing overhead for local packet processing prior delivery to an application is rather negligible. Next, the overhead incurred by the mobile proxy server (as in Equation 1) in Figure 7.Initially, an overall âĂIJhumpâĂİ-like behavior is observable which decreases as the request sizes increase. Part of this behavior is explained by the rather large variability (which isn't illustrated here for clarity) in the regions of smaller request sizes. The overall diminishing overhead is explainable by the increase in the web request sizes with the actual overhead in the processing and local loop-back data connections for the proxy service. As the proxy service requires an initial connection setup before performing the pass-through connection from the remote server, additional time and processing resources are required before the connection can be fully passed through the proxy. Both resources incur processing and delay penalties which in turn have negative power consumption impacts. Larger request sizes result in increased numbers of packets to transmit, which âĂIJsmoothesâĂİ the fixed setup overhead over more packets, which results in the decreasing overhead. At 250 kB of requested data, an observable reduction occurs to just below six percent. Overall, that based on these measurements, it's observed that the implementation of a mobile proxy server

directly on a mobile device does not result in considerable additional power demands for larger-sized web requests, which are common today.

Delay Difference

With most interactive settings, the overhead in terms of additional delays can have a negative impact on user utility or perceived quality of service. In networked mobile settings, increased delays additionally have a negative battery impact through the direct correlation between the two. The average delay differences between the proxy and the direct connections for different request sizes is illustrated in Figure 8. Initially, there is a fairly steady level of added average delays to the mobile proxy service in the region of 25 ms with higher levels of standard deviation occurring where the larger amounts of data requested; an overall maximum of about 37 ms can be noted for requesting 250 kB of data. The variability for the delays is a combination of the script producing the larger chunks of data with now slightly increased variability, as well as the increased number of packets sent for each approach incurring an additional network delay variation. Overall, it can be concluded that the observed low level of additional delay represents the fairly constant overhead of setting up the proxy service on the mobile device and processing the initial connection requests.

Media Streaming

The next step is to evaluate the impact of a long duration connection through the mobile SOCKSv5 server, in contrast to the previous evaluation of frequent connection requests. Here, a continuous data stream needs to be forwarded to a local application, with the initial setup overhead becoming negligible. The playback of the web video on the Pandaboard is performed using the Firefox for Android web browser. For measurements of the incurred proxy overhead, the browser is reconfig-
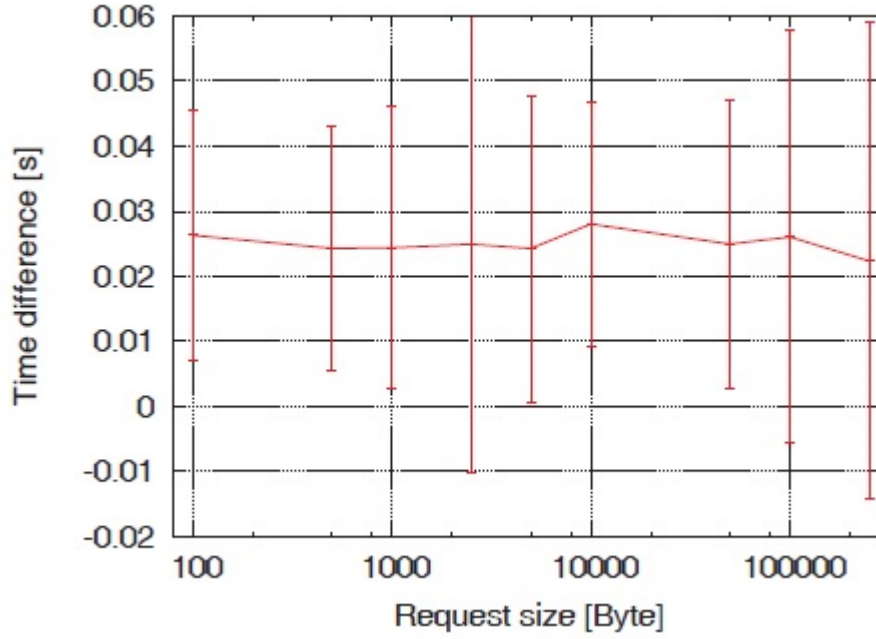
Figure 8. Average time difference and standard deviation between the direct and the proxy approach for requesting different amounts of data from a local server.

ured to utilize the local SOCKSv5 proxy service, similar to the previous web request scenarios. The

HTML5 video is in turn displayed on the browser for the entire duration.

Fixed Network

The sampled power consumption for video streaming over a wired Ethernet network is illustrated in Figure 9. Observing the graph, we can see that most of the measured power consumption values fall into a range between 3.25 W and 3.75 W for both approaches. Both approaches additionally exhibit overall periods of heightened power consumption, e.g., from about 250s to about 290s. This time frame coincides with a fast camera movement following an actor (very high level of background content change, motion) and represents a more challenging video decoding task, in turn leading to higher power consumption. Comparatively, however, it's also observable that no significant differences exist between the two approaches. The aggregated energy consumption resulting from media streaming results in a linear behavior, which additionally is almost indistinguishable between the two

16
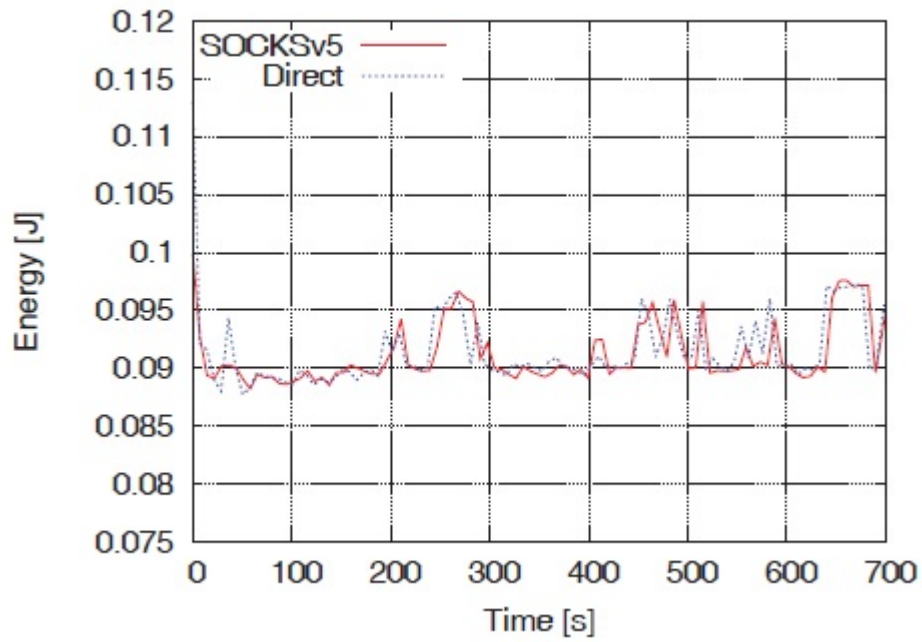
Figure 9. Energy consumption while performing the video streaming of the Tears of Steel open-source video sequence directly and through a mobile SOCKSv5 proxy server, smoothed over time.

approaches as observed for the prior web requests scenario. As indicated by the comparable energy consumption over time in Figure 9, the additional video decoding power requirements let the overall power consumption differences remain minor. We in turn omit a visual representation that closely resembles Figure 5 due to space constraints. The aggregated performance values for this scenario in Table 10. NEED TO CHANGE THIS TO TABLE FORMAT IN LATEX INSTEAD OF PICTURE) Looking at the table, I note that the average energy used is very similar for either approach, with slightly elevated values for the direct approach. Taking the standard deviation and the very narrow 99% confidence intervals into account, we note that the proxy-based streaming incurs no significant penalties. Comparing this streaming example to the individual web requests, I note that overall, the proxy-based approach now is at the same overall level that I observed for the direct case.

17

Interface	Approach	Average [J]	Standard Deviation [J]	Confidence Interval (99%)
LAN	Direct	0.0917	0.0051	$7.81 \cdot 10^{-5}$
		0.0913	0.0050	$7.77 \cdot 10^{-5}$
	Proxy	0.0912	0.0052	$7.99 \cdot 10^{-5}$
		0.0912	0.0052	$7.95 \cdot 10^{-5}$
WLAN	Direct	0.0844	0.0047	$7.87 \cdot 10^{-5}$
		0.0847	0.0049	$8.26 \cdot 10^{-5}$
		0.0844	0.0050	$8.35 \cdot 10^{-5}$
		0.0846	0.0056	$9.37 \cdot 10^{-5}$
	Proxy	0.0849	0.0052	$8.65 \cdot 10^{-5}$
		0.0854	0.0051	$8.53 \cdot 10^{-5}$
		0.0849	0.0047	$7.83 \cdot 10^{-5}$
		0.0855	0.0051	$8.50 \cdot 10^{-5}$

Figure 10. Summary values for HTML5 video streaming of the open-source Tears of Steel movie (video-only) over an ethernet and a wireless network.

Wireless Network

As in the previous web request scenario, I additionally investigate the alternative approach whereby the video streaming is now utilizing a local wireless network. The measurement results in Table 10 for equal numbers of measurement points. (Also note that for means of better comparison, the number of measurement points are limited to be equal.) I note that the overall averages are fairly close and furthermore characterized by relatively narrow confidence intervals. The resulting overhead, when comparing the averages of the direct and proxy-based video streaming is 0.0081, or less than one percent. Comparing the overall WLAN levels to the wired measurements, I note a decrease in the overall power consumption. This behavior is explained with the full availability of all network interfaces, whereby the employed operating system seems unable to properly put all different interfaces into a nopower state. Overall, I conclude that the introduction of the mobile proxy server has only limited negative impacts on the power consumption of about one percent when streaming multimedia due to the significantly higher impact of the decoding processes on the energy consumption and the negligible packet processing overheads.

Conclusions and Future Work

I approximated the energy consumption overhead of mobile optimization frameworks through a mobile SOCKSv5 proxy server as a low-end baseline. The selected implementation is based on JAVA and performs all network traffic forwarding on the application layer within the Dalvik VM for Android devices. I found that the overall usage of a proxy service on the mobile device in a web request scenario does not incur any significant power consumption penalties. For HTML5 video streaming (continuous packet processing), I note an overhead of about one percent. I determined that any existing overhead is relatively constant and explained through the initial communication setup overheads, rather than ondevice processing overheads, which were found to be approximately one percent. A basic optimization framework would in turn only need to overcome this very low overhead to enable energy savings potentials.In future research avenues, I plan to investigate further energy savings potentials based on mobile device proxying, in combination with caching and application layer content awareness in tandem with a remote server.

CHAPTER IV

Web Cache Object Forwarding From Desktop to Mobile for Energy Consumption Optimizations

Introduction

With the beginning of the 21st century, networking support for wirelessly connected mobile user devices has fueled a continuous increase in the demand for mobile data. Web requests now originate in a majority from wirelessly connected user devices – a trend that Cisco, Inc. predicts to continue in the foreseeable future [1]. Simultaneously, the overall user behavior and demand for more rich media inclusion into web pages has increased the overall amount of data that is required to be transmitted per page, see, e.g., [10, 11]. Caching on the client side has been effectively used in the past and was, together with increased numbers of parallel object downloads, able to decrease wait times for desktop clients as reported in [10]. A first view of mobile web page characteristics (which were found to exhibit lower complexity than regular desktop browser versions) and non-landing pages (which were found to be less complex than landing pages) was evaluated by the authors of [11]. Moreover, a significant body of research has emerged that focuses on content delivery optimizations to mobile devices.

Typically, these optimization approaches are targeting on-device optimizations or off–device cloud–based improvements. For mobile applications, for example, significant energy savings were found to be attainable when grouping application requests so as to avoid prolonged cellular network interface activity, see, e.g., [12, 13]. Other approaches optimize the delivery to mobile devices through proxies and cloud-based data anticipation and traffic shaping, see, e.g., [14].

For web data, typically caching is used to limit the amount of data that has to be transfered to requesting clients. In prior works focusing on mobile web page delivery optimizations, such as [15],

Figure 11. Screenshot of rendered web page from WebPageSpeedTest.org. As illustrated, the main textual and pictorial content items are flanked by background and interactive advertisements.

energy optimization has been a key element, due to the limiting restrictions for mobile clients. One particular area for optimization is the pre–fetching of web page objects combined with caching (which allows to download data before usage), such as presented in [16] with about 10 % energy savings. More recently, these approaches were combined with user connectivity predictions, see, e.g., [17].

We propose to utilize a basic cache forwarding method that can be utilized by users to synchronize from, e.g., a desktop computer, to their mobile device, e.g., a smartphone. We illustrate this approach in Figure 11, which contains the desktop and a mobile device. As most devices are charged over night, or are at least stationary within local area network ranges, we propose to utilize this general idle period to allow direct forwarding of cached objects. We presume that browser clients will have a shared information source in the cloud that allows the identification of visited web pages, an approach most browser clients take to date. In turn, clients are enabled to identify web page objects that are identical between the different device display modalities, i.e., identical web objects delivered

when requesting a desktop/mobile versions of a web page. The thus transmitted objects, in turn, reside locally within the mobile device cache and do not require an energy–expensive mobile download through cellular interfaces. We note that the reverse of this approach is possible as well. To support our approach, we gather data through the publicly accessible WebpageSpeedTest.org website as well as the httparchive.org archive of a large dataset of performance evaluations for popular web pages; we refer the interested reader to [18] for a more detailed discussion.

Individual Example

In this section, we outline an individual example for the German news web page "Der Spiegel," accessible at http://www.spiegel.de. This web page serves as an example for a rich media and advertising material containing web presence that is frequently updated. On February 12, 2014, we performed a speed test online using (i) Internet Explorer 9 as the browser instance for a desktop client, (ii) Chrome as the mobile browser on a Google Nexus 5 instance, and (iii) Safari for an iPhone 4 instance provided by WebpageSpeedTest.org (both mobile clients were traffic shaped to 3G connection emulations). An example screenshot of the web page as rendered is provided in Figure 12.

As illustrated, the web page exhibits a significant amount of objects that are required for advertisements, visual items (images, layout), and scripting.

We evaluate the reported data as follows. For each display modality and browser client, we gather the individual web objects requested and the response sizes and headers for caching information (for HTTP response codes 200 only, as we are not interested in redirects). We denote the number of all objects returned for a request modality (i.e., Internet Explorer for the desktop client example and iPhone or Nexus 5 for mobile counterparts) as N, their average size as $\overline{X}$, the standard deviation of their sizes as σ_X, and the Coefficient of Variation (CoV) of the returned object sizes as CoV_X. As in
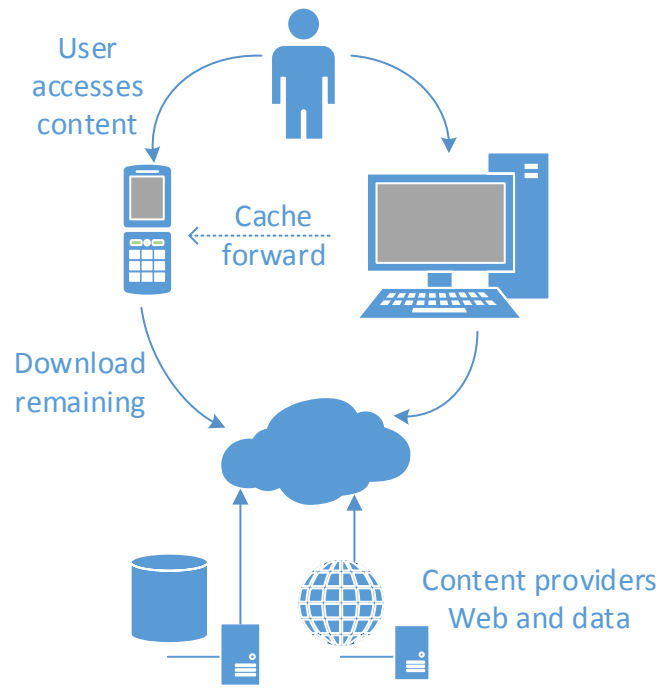
22

Figure 12. Screenshot of rendered web page from WebPageSpeedTest.org. As illustrated, the main textual and pictorial content items are flanked by background and interactive advertisements.

the HTTP specification, the *max–age* directive overrides others, so we initially consider that directive

for the cache longevity of the individual objects. If no explicit information is found, we consider the

response header's *expires* information, which provides a secondary cache lifetime. If both are not

found, or if the *expires* date is in the past or right at the request time, we set the cache lifetime to zero.

Similar to the notation for the object sizes, we denote the expiration time characteristics as $\overline{T}$, σ_T, and

CoV_T, respectively.

Table IV.1. High-level overview of different client web page statistics for http://www.spiegel.de

Statistic	IE Deskt.	iPhone	Nexus 5
N	154	132	145
$\overline{X}$	10402.77	8846.39	10201.63
σ	19103.19	13438.05	25684.58
CoV_X	1.84	1.52	2.52
$\overline{T}$	106285.14	114780.69	114818.28
σ_T	125080.27	125576.65	126090.90
CoV_T	1.18	1.09	1.10

Data Description

We provide an initial high-level overview of the web page characteristics for http://www. spiegel.de as requested by the different clients in Table IV.1.

We initially observe that the desktop version (with Internet Explorer 9 as requesting browser client) exhibits the highest number of objects, followed by the mobile Chrome/Android and Safari mobile/iOS versions. Interestingly, there is only a minor difference in the number of objects between these versions. Next, we observe that the trend for the average number of bytes is aligned with the number of elements. In total, the mobile Chrome version is almost on par with the desktop one (at 92 % of total data); only the Safari mobile version seems optimized (at 72 % of total data). The standard deviation and Coefficient of Variation (CoV) amongst individual element sizes are highest for the mobile Chrome version, indicating more significant size differences than for the safari version (which is one unit lower), while the desktop version falls into the middle.

Next, we evaluate the elements' caching properties on a high level as presented in Table IV.1 as well. Overall, we note that the highest average caching time is observed for the mobile Chrome access with the Nexus 5 device, trailed immediately by the iOS access and (with distance) by the desktop browser. We note that the overall variability in terms of expiration times amongst elements is fairly low and comparable, as indicated by CoV values around 1.1.

24

We now shift the view to the possibility for identical objects to be re–utilized locally to avoid additional download penalties in time and energy consumption. Simultaneous with a web page request, a local broadcast could "ask" for the cached elements for a website to be delivered locally from participating clients. Alternatively, cellular provisioning methods, such as in LTE–A, could initiate the device–to–device local exchange as well. Once the request is received, the local coordination can take place, which inherently results in the forwarding of elements that are the same across browser instances and which have a future cache lifetime expiration.

We filter out elements that are not similar amongst the different requesting devices, which results in 82 objects with the same URL and size combination. In other words, just above half of the objects constituting the web page under consideration are identical between different clients.

While we note an identical cache lifetime for most of these objects, a slight increase is notable from the Desktop over Safari to Chrome clients. Next, we remove items that exhibit a cache lifetime of zero for any of the three requesting clients, resulting in 59 objects. These remaining objects display a reduced average cache lifetime for the mobile browsers, whereby mobile Chrome exhibits the shortest. In turn, we choose the iOS Safari mobile as our exemplary base for calculations. We utilize the approximations for energy consumptions presented in [19] to determine the amount of energy required if local data forwarding can be performed using either Bluetooth or WLAN technologies. We illustrate the energy consumption as function of time in Figure 13.

The complete download energy for exclusive use of cellular is the upper limit on energy spent downloading the data associated with the web page. The Bluetooth and WLAN data exchanges with other local clients both follow the same underlying trend with increasing energy required to transmit
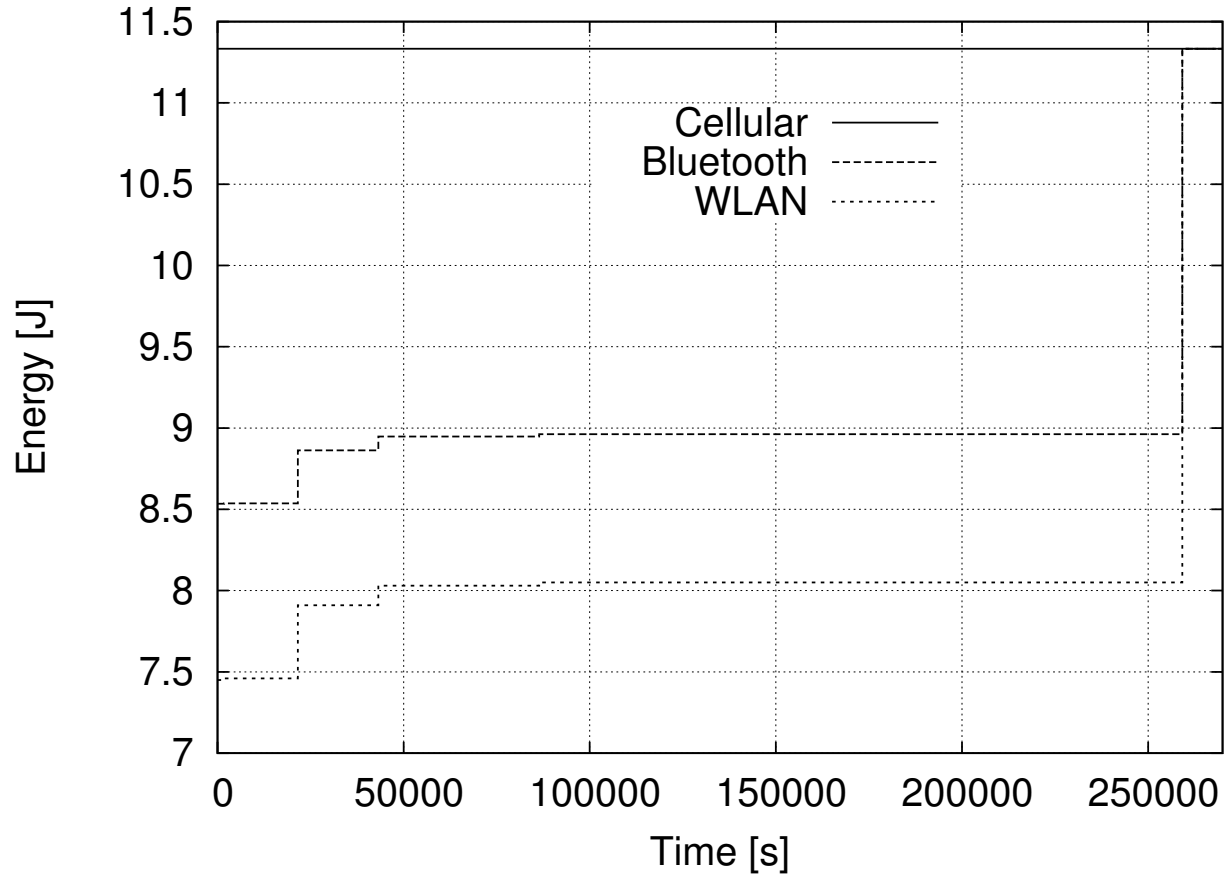
Figure 13. Energy required to download web page data via cellular connection or through combination with partial local exchange with Bluetooth/WLAN.

all data as time progresses from the last access point in time (due to cache expirations).

We illustrate the relative amount of data/items within the cache in addition to the attainable savings in Figure 14 as a function of delayed access time.

We observe that a significant amount of the overall data can be cached (and would in turn be suitable for localized forwarding). We additionally observe that even for longer durations up to multiple days, significant amounts of data remain in the cache. Combining the forwarding from locally cooperating clients, e.g., user-owned desktop computer in the same network through either Bluetooth or WLAN technologies, we utilize the approximations outlined in [20] to derive the relative gains of our approach in terms of energy consumption. Specifically, we compare our approach (combining
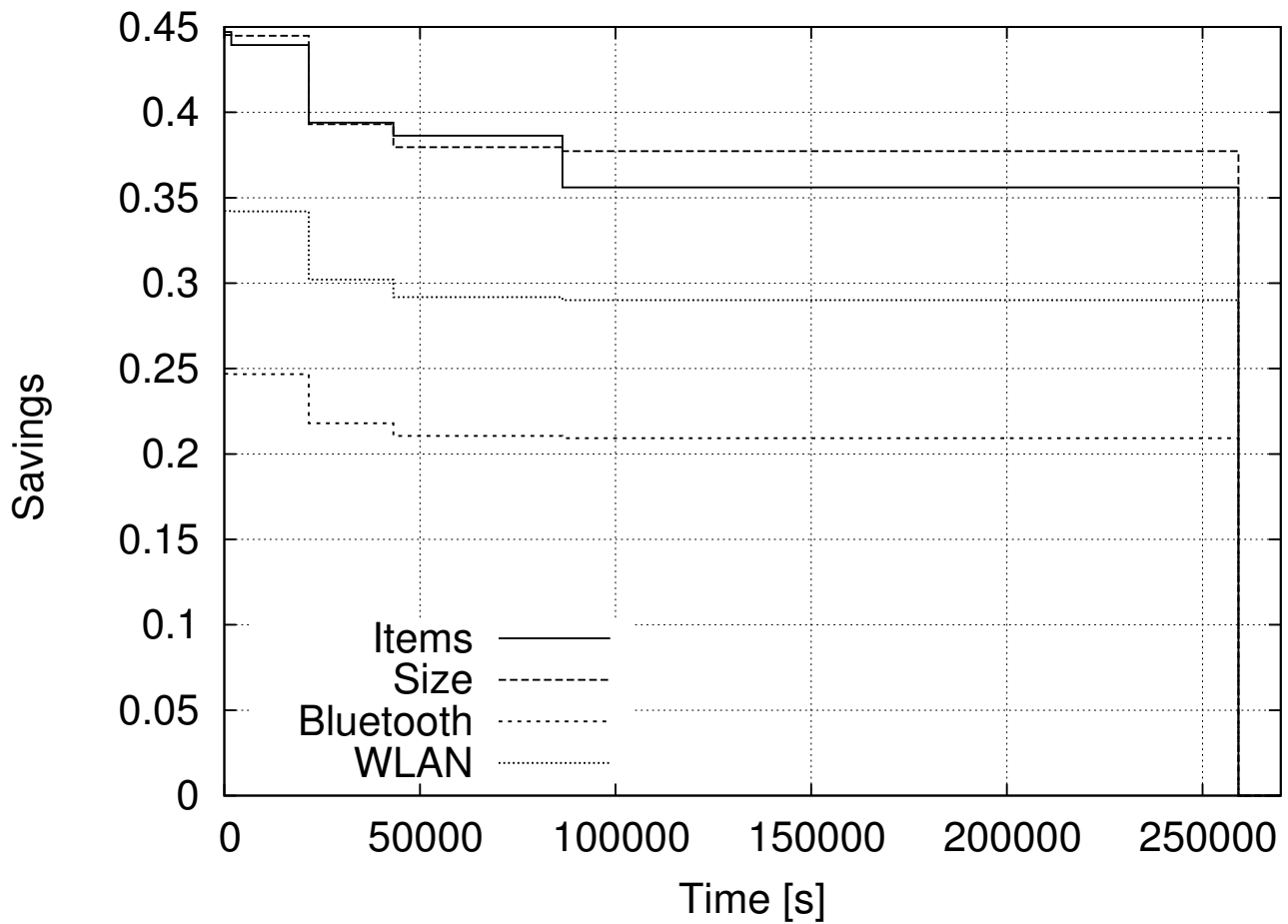
Figure 14. Relative number of items and data amounts in cache and savings resulting from local data exchange with a desktop client.

locally forwarded cached objects and cellular downloads of the remainder) with the regular download of all objects through the cellular network interface. We note that significant savings above 20 % are attainable with our approach, even if the filling of the device's cache is performed wirelessly as well.

Large Dataset Performance Evaluation

We now evaluate our proposed approach through simulations based on a large dataset of landing web pages of popular websites. For this purpose, we retrieved the publicly available web page response details from *httparchive.org* and created a subsequent dataset for October 1st, 2013.

Table IV.2. Overview of the large dataset characteristics for all pages and response objects with a
focus on the cache lifetime.

Category	Desktop (IE)				Mobile (iOS)			
	$\sum X_i$ [MB]	N	$\sigma(X_i)$ [kB]	$\overline{X_i}$ [kB]	$\sum X_i$ [MB]	N	$\sigma(X_i)$ [kB]	$\overline{X_i}$ [kB]
expAge < 30 Seconds	238490.28	15670679	1698.59	15.22	1355.24	122598	582.98	11.0:
30 ≤ expAge < 1 Minute	170.94	29130	27.38	5.87	12.68	898	70.38	14.1:
1 Minute ≤ expAge < 1 Hour	15192.40	797450	270.06	19.05	282.36	15854	226.62	17.8
1 Hour ≤ expAge < 12 hours	29798.68	1186947	751.45	25.11	244.14	17424	209.84	14.0
12 hours ≤ expAge < 1 day	6188.57	358998	227.63	17.24	91.37	5952	91.37	15.3:
1 day ≤ expAge	195784.64	9888147	1337.36	19.80	2122.96	126629	750.51	16.7
Average	80937.59	4655225	718.75	17.39	684.79	48226	321.95	14.2(

Dataset Description

The dataset we utilize for the performance evaluation contains the individual web responses for
the most popular websites of fixed (Internet Explorer for the desktop) and mobile (iOS for iPhone 4)
websites, ranked through the Alexa popularity index. Furthermore, the dataset contains the individual
response details, including cache expiration times and sizes.

Similar to the individual web page example, we provide an overall description of the response
sizes and maximum expiration ages in Table IV.2. We note that the overall average response size as
result of desktop requests (when batched into the outlined time frames) is around 17 kB, with the
largest average sizes occurring between one and twelve hours. Mostly, we note that the majority of
objects exhibit a short expiration time between zero and 30 seconds. For the responses to mobile
devices, we note that the majority of objects are in the same short time span and the long time span of
more than one day. The highest average response sizes, however, are observed between one hour and
half a day, in difference to the desktop counterpart. For more details of the differences between fixed
and mobile web pages, we refer to [21].

We illustrate the cumulative distribution for the desktop and mobile client response expiration
ages in Figure 15.
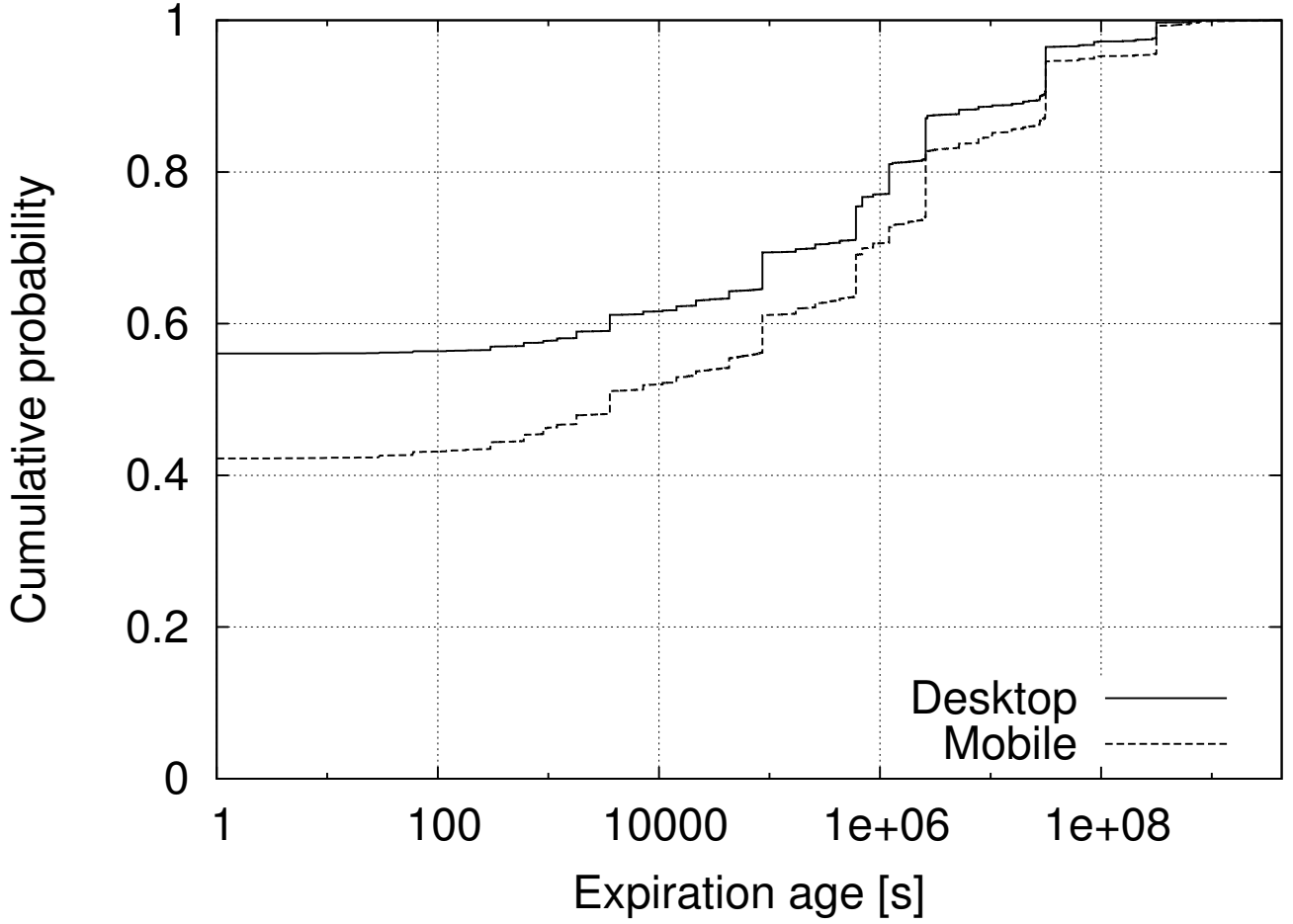
Figure 15. Cumulative probability of expiration times for desktop and mobile responses.

We observe that a significant portion (> 40 %) of responses from web servers exhibits an immediate expiration and would, in turn, require an immediate retrieval by clients. The cumulative expiration probability is higher for responses to desktop clients, which can be attributed to the typically more abundant bandwidth (and less optimization requirements) in contrast to a mobile setting (where web content developers might be more considerate of limitations). In addition, we perform matching of request to mirror the individual example with a direction from desktop to mobile client.

Simulation Model Description

We evaluate the effectiveness of our approach through simulation, which we perform as follows. For the time period of one week, we randomly select a web page from the mobile dataset and

29

separate the data that is required to be downloaded and the data that is still available in cache due to the maximum expiration age of the individual objects. We subsequently wait for a random amount of time before repeating the procedure.

Initially, we sort all landing web pages $l, l = 0, \ldots, L$ from the dataset based on their popularity ranking. To simulate the popularity of web pages, it was found that the Zipf distribution effectively describes the popularity of individual requests. We chose to utilize the non-modified Zipf distribution for our simulation purposes, noting that additional modifications to the distribution exist that can further enhance it, see, e.g., [22]. For the total number of web pages L, we derive the probability of page l being selected in turn as

$$p(l) = \frac{l^{-\alpha}}{\zeta(l)}, \tag{IV.1}$$

where $\zeta(\cdot)$ denotes the Zeta function. We furthermore set $\alpha = 0.85$ as an average choice in common ranges utilized for this parameter.

Once the individual mobile web page l was randomly chosen, we compare it to its desktop counterpart, with $\{d, m\}$ denoting the request modality. As each page l exhibits a time-sensitive number of responses, we denote them as $r^{\{d,m\}}(l, t)$ and each page exhibits $R^{\{d,m\}}(l, t = 0)$ in total. Let $t_c(r^{\{d,m\}}(l))$ denote the expiration or max-age directive received with the response at $r^{\{d,m\}}(l, t = 0)$. The number of objects or responses to be retrieved at time t in turn are given as

$$R^{\{d,m\}}(l, t) = \sum_{r^{\{d,m\}}(l,0)} \left[t_c(r^{\{d,m\}}(l)) \geq t \right] \tag{IV.2}$$

where $[\cdot]$ denotes the Iverson Bracket. In the following, we abbreviate to t_c for readability when in direct context.

We furthermore denote the size of the individual response retrieved at time t as

$$x_r^{\{d,m\}}(l, t) = x_r^{\{d,m\}}(l, 0) \cdot [t_c \geq t]. \tag{IV.3}$$

30

The total size of objects or responses retrieved at simulation time t thus is given as

$$X^{\{d,m\}}(l,t) = \sum_{r^{\{d,m\}}(l,0)} x_r^{\{d,m\}}(l,t). \tag{IV.4}$$

For performance evaluation purposes, we simulate the user behavior similar to the process outlined in [23], whereby we randomly draw the time between requests t_u as Pareto distributed using

$$p(t_u) = \beta k^\beta t_u^{-(\beta+1)}, \tag{IV.5}$$

with $\beta = 1.5$, $k = 30$. We assume that no delays are accrued due to instantaneous cache retrievals or downloads. Using time index i, we thus derive $t_{i+1} = t_i + t_u$. We continue the simulation until $t_{i+1} = t \geq T = 640800$ [s], i.e., one week. To maintain tractability, we bundle the presented results into bins of 3600 seconds, i.e., one hour, through averaging.

Performance Evaluation Results

We present our results for the individual responses or web objects and the total number of bytes. We note that through approximations, such as the ones described in Section IV, an inference of energy consumption would be possible as well. We perform the simulations with an average $\alpha = 0.85$ for the utilized Zipf distribution and repeat each week-long evaluation 2000 times. Initially, we present the total number of responses and bytes that would not require a download (i.e., would reside in the mobile cache) as a function of time in Figure 16.

We note an immediate decrease in the number of requests, which is mirrored by the number of bytes as well (indicating a linear relationship as observed in, e.g., [21]). We furthermore observe that the initial decline levels out at the simulation time of around one day, and remains steady afterwards. This indicates that within the simulation period of one week, initially a large number of items is non–shared between devices or exhibits no significant cache lifetime. In a following group of objects, cache

[Objects]
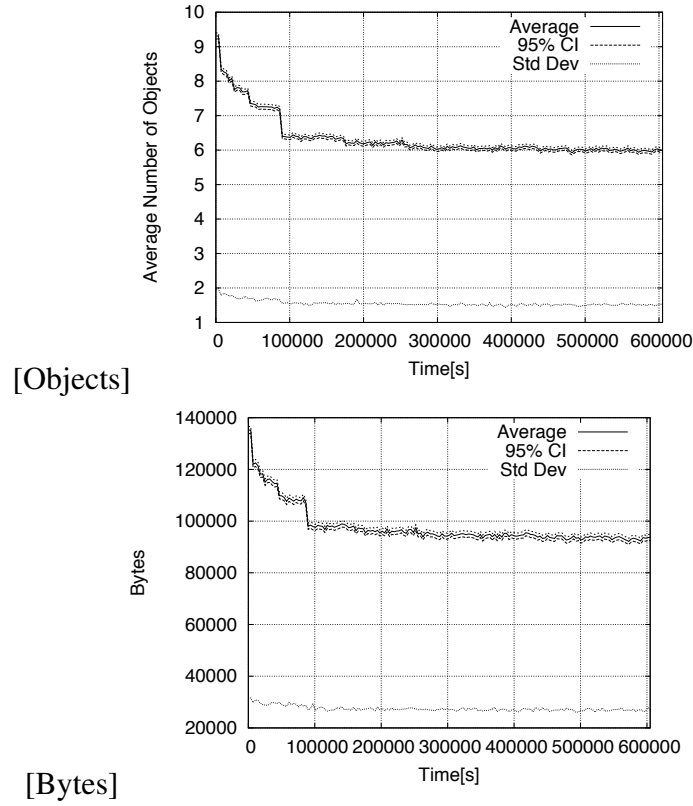
[Bytes]

Figure 16. Simulation results for the number of responses and bytes that are in the transfered mobile device cache as averages in 1-hour bins.

lifetimes increase logarithmically, which leads to the exponential decline we observe here. The narrow confidence intervals and low level of standard deviation indicate that the presented results are stable within simulation confines.

Next, we present how these findings translate into attainable savings with respect to requests for objects and bytes in Figure 17 in relationship to the total web page data without caching.

We initially note a declining trend similar to the one observed for the cached items in Figure 16, but with more distinct "jumps" observable at half-day and day times of the simulation. Overall, we note that almost 15 % savings for objects and bytes decline to around 10 % after the boundaries of a day (whereby objects exhibit lower levels and bytes exhibit higher levels).

To evaluate the impact of different web page popularity distributions, we present an evaluation

32

[Objects]
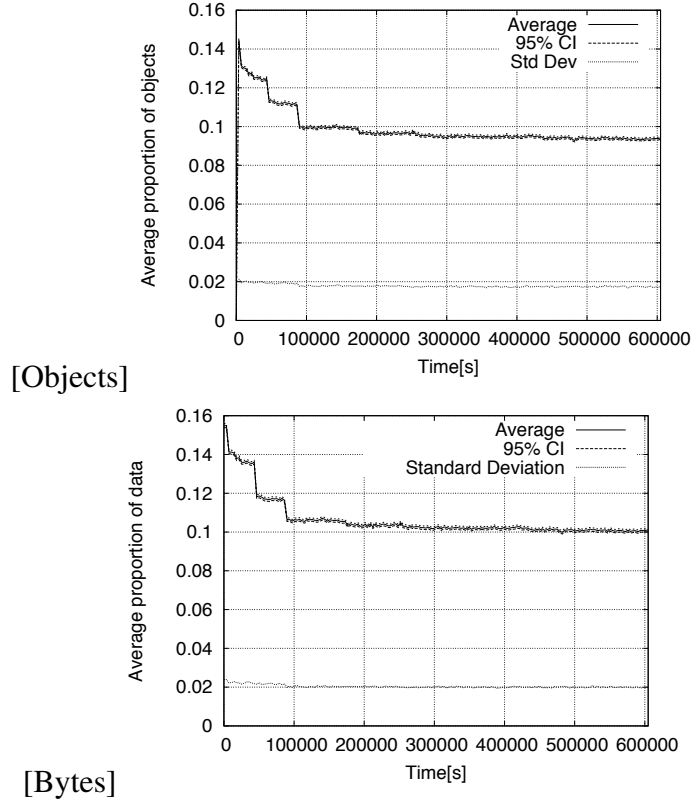
[Bytes]

Figure 17. Simulation results for the attained savings for the number of responses and bytes by transferring to the mobile device cache in 1-hour bins.
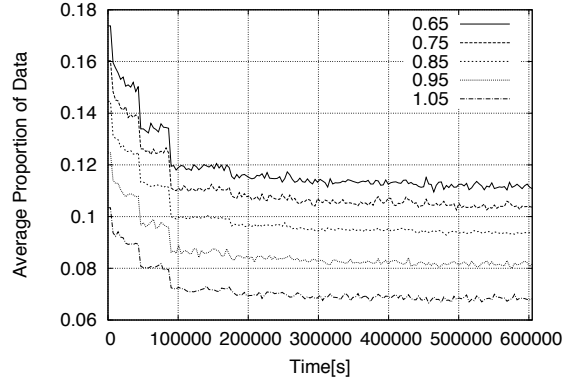
Figure 18. Average proportions of data in mobile cache for different Zipf distribution parameters α.

of different α for the Zipf distribution in Figure 18, each with 500 simulation repetitions for $0.65 \leq \alpha \leq 1.05$.

We limit our comparison to bytes due to space constraints, noting that results obtained for objects are similar. We observe that all levels of α exhibit the same underlying trends, with slight and

non-linear differences between levels. We conclude that the distribution of web page popularity and the composition thereof has an overall impact on the long-term level of savings that can be attained.

Overall, we find a significant reduction in the required downloads for mobile clients that can be directly attributed to lower energy consumption levels and potential download latencies (and even costs for the mobile user through saved data transmissions). Furthermore, this approach does not require sophisticated live evaluations, but only relies on cache forwarding to the mobile device.

Conclusion

Performance Evaluation Results We exemplarily evaluated the content of modern web pages and the identical objects that can be found when requesting the same web page from different devices with different display modalities, such as desktop and smartphone. Assuming mobile users visit the same pages on their mobile device that they visit on their desktop computer as well, we simulated the access of landing web pages and evaluated the potential for cache forwarding. Our basic approach is able to save upward of 7.5 % of mobile request or bytes even for very distant time horizons. In the more typical daily time range, our approach can yield almost linearly decreasing savings stating at about 14 %, without requiring any sophisticated prediction mechanisms, but only a direct cache transfer. We note that similarly, exchanges between mobile and fixed devices belonging to a user are possible, including keeping a cloud-based reference of pages that a user visits (similar to Google Chrome's synchronization features).

CHAPTER V

Desktop and Mobile Web Page Comparison: Characteristics, Trends, and Implications

Introduction

The broad proliferation of web-based services and the trend to outsource formerly server-based and/or locally maintained services into "the cloud" have significantly altered the manner of web page designs and compositions. As the nature that underlies a typical web page changed over time, so have their characteristics and resulting implications for the networks that transport them. In turn, a plethora of popular applications and interactions that are performed by users are enabled by using web-based services and have fueled the growth of Internet traffic.

Popular past web characteristics, model usages, and performance analysis approaches, see, e.g., [24] or, more recently, [25], do not necessarily reflect the current state of the World Wide Web anymore. While Cisco, Inc. predicts that video will account for the largest portion of networked traffic (and its growth) in the near future [26], consumer-based web traffic is forecasted to exponentially increase during the same time frame as well, putting an overall burden on access networks.

Longitudinal studies have recently emerged that target capturing the dynamic behavior of the World Wide Web over time, such as [27]. In [10], the authors investigate five years (2006-2010) of fixed web site traffic captured through a proxy system and found significant impacts as a result of interactions. Furthermore, they note that the overall loading time of web pages has been reduced due to higher levels of caching and an increase in concurrent connections made by desktop browsers. Similarly, the complexity of web sites was evaluated in [28, 11]. The authors found that the number of objects that were loaded (independent from their relative location) has the highest impact on web page load times. Interestingly, the authors additionally found that for their dataset, (i) non-landing pages

tended to be less complex and (ii) mobile web pages tended to be of lesser complexity.

The broad proliferation of mobile devices that can connect to Internet-based services has given rise to considerable amounts of data that are exchanged with web-based services. Several predictions indicate that there will be a continuous increase in the demand for mobile data, see, e.g. [26], and that access to web-based services will soon be mainly performed through mobile devices. While the outlined recent studies and ongoing works mainly investigate the traditional desktop-based web access, the emergence of mobile web access gives rise to a new set of problems that are direct derivatives of the characteristics of mobile devices. In a recent overview of the battery impact that different web page elements have on the power consumption of mobile devices, Java Script and CSS were identified as the main contributors to the power consumption during rendering, see [29]. In addition to the complexity of web pages, their composition can have a direct impact on the web-related performance of mobile devices.

Our contribution fills the currently existing gap of an in-depth evaluation of the trends that emerge for desktop and mobile client versions of web pages, for which we cover the time period from mid-2011 to the end of 2013. We demonstrate the similarities and disparities of the current fixed and mobile web and outline overall notable characteristics and trends that web page developers as well as networking researchers and practitioners should consider in their respective optimization efforts.

The remainder of this paper is organized as follows. In the next section, we describe the underlying dataset from httparchive.org and how we processed it. Subsequently, we compare the characteristics for desktop and mobile clients with respect to web page objects, sizes, and caching in Section V. We discuss the results and implications in Section V before we conclude in Section V.
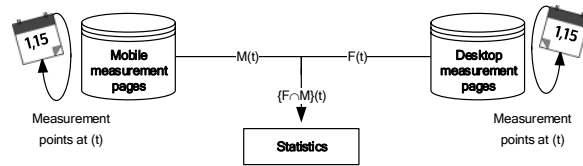
Figure 19. Approach for gathering the general statistics evaluated in this contribution: based on the original archives, we compare the fixed and mobile web pages contained in each bi-monthly measurement and evaluate only those that can be found in desktop and mobile web page requests to allow for a direct comparison.

Data Set and Pre-Processing

In this paper, we utilize the httparchive.org [30] publicly available dataset of captured web performance metrics. As an industry-supported project, its goal is to provide "a permanent repository of web performance information such as size of pages, failed requests, and technologies utilized." The overall starting points are the initial client view statistics, i.e., non-cached web page views, that are gathered by the httparchive.org projects at the beginning and in the middle of each month. We illustrate the overall process in Figure 19. As with any project that has to evolve over time to account for changes in technologies, some underlying measurement setups have to change over time as well. In the remainder of this section, we describe the initial gathering and processing of this dataset in greater detail.

General Notes

As we utilize the measurements over a significant amount of time from June 1, 2011 to December 15, 2013, we initially note that some of the underlying measurement configurations have changed over time, which includes multiple facets, such as Unique Resource Locators (URLs), browser versions, connection speeds, or incorporation of 'lazy loading' of resources. We reason that overall, however, the changes made were reflecting industry trends as well as personal connectivity trends (such as modified access network speeds) and can be seen as representative of the typical connec-

tion scenarios for the World Wide Web. We refer the interested reader to the online documentation of the httparchive.org project for more details pertaining to the measurement setup used and detailed information about changes.

Web Page Selection and Processing

We select the available page statistics for both desktop (fixed) and mobile web pages over the range of more than two years; the available points in time are at the beginning and the middle of each month. This results in a total of 62 datasets each for fixed and mobile web clients, respectively. The evaluation by the httparchive.org project used the Apple iPhone's built-in browser client for mobile requests and the Microsoft Internet Explorer browser client for desktop requests (represented by different user agent strings in the HTTP requests sent). We initially determine all web sites that are common for both archives, i.e., web sites that for each measurement time were evaluated for both, fixed and mobile clients, to allow for a comparative evaluation of their described metrics. This results in a fair comparison at each individual measurement point, but yields an initial reduction of the total number of web pages that we compare. We note, however, that there are more than 900 pages in the initial (smallest) set. For each of the measurement times, we subsequently aggregate the measured web page characteristics over all web sites that were selected in a particular client role – as a result, we derive a representative average snapshot of the characteristics that make up "the web" as accessed using different web clients over time. In addition to the time-varying selection of web pages, we compare the web pages that are found for desktop and mobile clients throughout the longitudinal evaluation time period, which results in a smaller subset of 46 web pages.

The underlying dataset that we utilize has undergone changes over time. Specifically, the base URLs that comprise the original dataset from httparchive.org (i.e., before we match fixed and mobile pages at each measurement point) were changed, namely (i) by switching to the Alexa top 1000000 web sites in November 2011 and (ii) by increasing the number of URLs evaluated in September 2012, see [30]. Other changes that were performed are more network-level oriented and should have a lesser impact on the static results we report here. To evaluate how these changes impact the diversity of values encountered in the dataset over time, we employ the entropy of the Theil population, which is itself based on the Theil index [31].

Specifically, we evaluate the number of web requests and the total number of bytes for each request type over time, as we are more interested in the overall quality of the selected data subset (i.e., statistics for pages that can be found in both datasets). We illustrate the result as function of time in Figure 20, whereby we initially note that higher values represent a higher diversity of the underlying dataset (Theil population). We observe that the entropy curves for each of the selected features (requests, bytes) are fairly close for each of the two modes (fixed, mobile), with significant "jumps," i.e., changes of their overall level, at two distinct measurement points for the larger dataset. These "jumps" occur at times where the underlying number and source set of the httparchive.org base URLs were changed, as outlined above. We note that these changes, however, affect all client modes and type of characteristics evaluated in a similar manner by significantly increasing the diversity of measurement results, which is immediately visible from Figure 20. For the smaller subset, we observe a more steady behavior, with only small deviations and outliers over the entire measurement time period. Overall, we note that the Theil entropy levels for the desktop client values are slightly higher

[Number of requests per page]

[Number of bytes

per page]
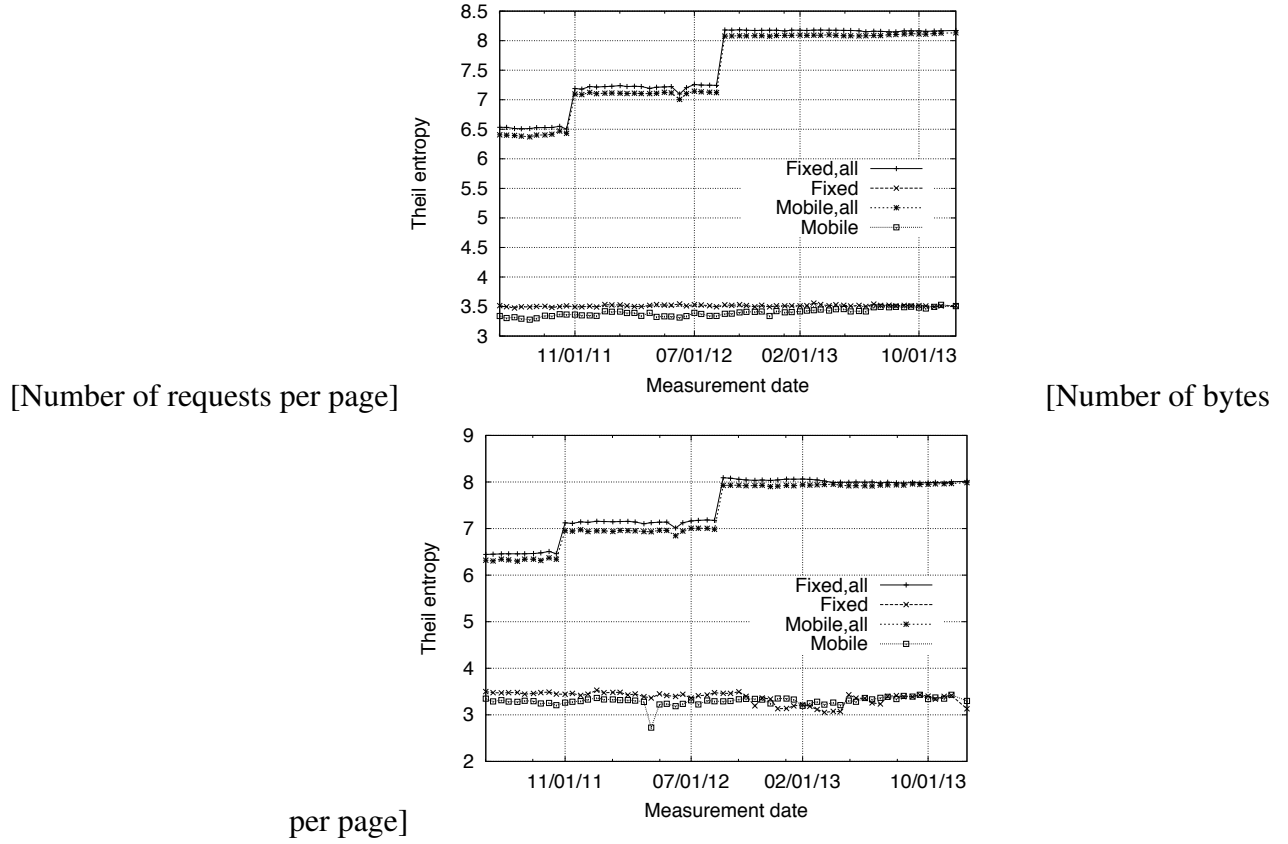
Figure 20. Theil entropy for the number of objects requested and the number of bytes for web pages present in the httparchive.org's desktop and mobile client data files for all pages at each measurement point (all) and the subset of continuously included pages.

than their mobile client counterparts; however, they remain within close range. This closeness in the calculated Theil entropy additionally motivates us to focus on the overall averages of the values we compare in the remainder of this paper.

Comparison of Desktop and Mobile Web Page Characteristics

In this section, we compare the average values we obtained from the joint httparchive.org datasets for the desktop and mobile web client versions of the same set of web pages requested, as given by their URLs.

[Categorized requests for desktop client (all pages).]
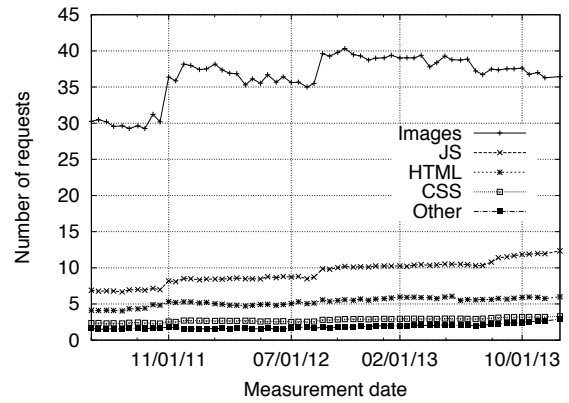
[Categorized requests for mobile client (all pages)]

Figure 21. Average total number of requests for objects constituting a web page in desktop and mobile versions and decomposition into HTML, CSS, Java Script, Images, and Other object categories.

Average Number of Web Site Objects

Initially, we investigate the average number of objects requested when accessing a web page

for the first time and illustrate the results in Figure 21 for both client types and all web pages compared

to the subset present throughout all measurement points. We initially note that the desktop versions

exhibit a significantly larger number of requested objects in comparison to the mobile counterparts. We additionally observe an overall continuous increase, which tends to have slight "jumps" in certain time frames (more pronounced for mobile client requests). These slight increases are in line with the earlier observations concerning changes to the underlying web pages constituting the measurement points in time. While significant variability between individual measurement points can be noted, the overall trend remains steady and slowly growing.

Most notably, we witness the increase to over 100 requests on average per web page requested by a desktop client and its continued growth since the second half of 2012. Within the same time period in 2012, the mobile web request counterpart rose to over 60 requests, with a steadily trend thereafter. We compare these trends for all web pages at each measurement point with the continuously present subset in Figure 21 (a) as well. We note that the web pages of the subset exhibit significantly lower levels of the average number of requests per web page for both versions. When evaluating the overall trends, however, we find that the subsets exhibit similar ones, hence corroborating our earlier observations for all pages.

Investigating the origins of the average number of web object requests in greater detail, we illustrate the composition of the average number of objects requested separated into the categories of HTML, CSS, Java Script, Images, and Other objects over time in Figure 21 (b) for desktop client requests. The Other category contains items such as fonts, Flash elements, as well as any remaining downloaded web page components.

We initially observe a rising trend for all categories, with the image category accounting for the most requests and the remaining "other" category continuing somewhat steady over time. Keeping the overall increasing trend in mind, we note the distribution between the different categories of objects remains rather steady over time (with categories in desktop client requests accounting for approximately

42

the following long-term averages: Images 74%, Java Script 19%, HTML 11%, Others 9% , and CSS 5%). For mobile requests, on the other hand, we observe in Figure 21 (c) a nearly identical order at a lower level (with categories in mobile client requests accounting for approximately the following long-term averages: Images 80%, Java Script 20%, HTML 11%, CSS 6%, and Others 4%). We note that an evaluation of the categorized average number of requests per web page yields similar results for the subset of pages (omitted due to space constraints).

We conclude from these numbers that for both scenarios, the main culprit for web requests is provided by the images contained in web sites, and to a lesser extent the oft-mentioned Java Script, found in modern AJAX and HTML5 based pages in fixed as well as mobile environments.

Average Web Page Sizes

We now shift our view to the average sizes of desktop and mobile client requested web pages over time as well as their most contributing factors. We initially illustrate the total number of bytes that were required on average to download a web page to a fixed and a mobile client in Figure 22.

We observe that the fixed sizes are always larger than their mobile counterparts and that both continuously increase over time. This leads to the average web page size to have increased from 0.71 MB in June 2011 to over 1.5 MB in December 2013 for desktop client and from 0.38 MB to almost 0.9 MB for the mobile counterparts in the same time frame. In greater detail, we note that the average increase in average web page sizes over time for fixed web pages during the evaluated time period is around 3% per month or approximately 41% per year, while the mobile counterparts feature an increase of 3% or 40%, respectively. We also find that the average mobile web page sizes currently trail their desktop counterparts by approximately 1.75 years, which likely will increase due to the difference in average growth rates (1.57% vs. 1.47% on average in every two week time period for
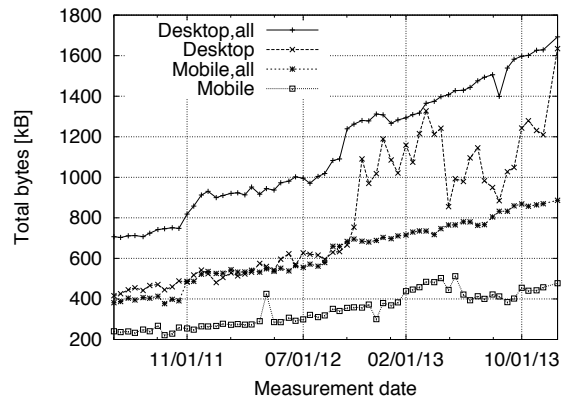
43

[Averages for desktop and mobile client versions.]
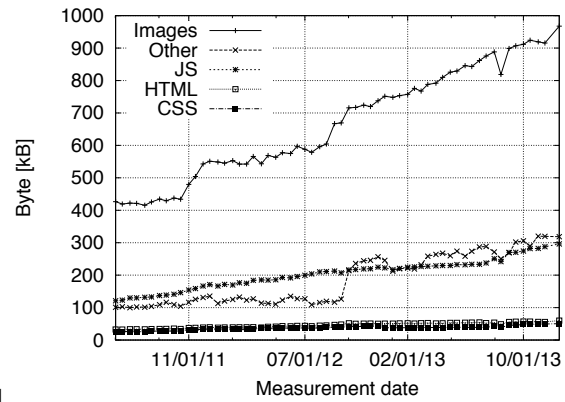
[Categorized for desktop client (all pages).]
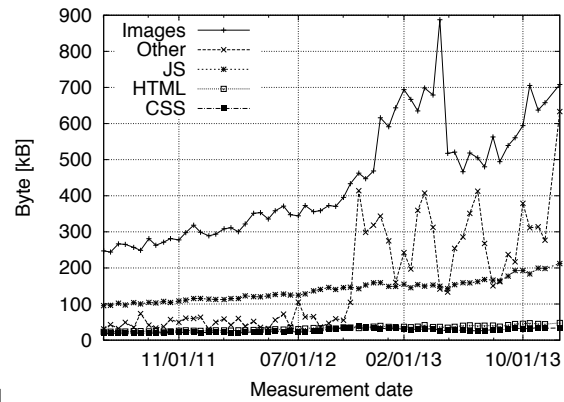
[Categorized for desktop client (subset).]

[Categorized by category for mobile client (all pages).]

Figure 22. Average total number of bytes constituting a web page in desktop and mobile versions and decomposition into HTML, CSS, Java Script, Images, and Other object categories.

44

fixed and mobile web pages, respectively). Similar to the average number of requests, we compare the

subset of 46 web pages to the complete dataset in Figure 22 as well. We observe a similar behavior

here with the subset's reduced average number of bytes for the first half of the observation period. In

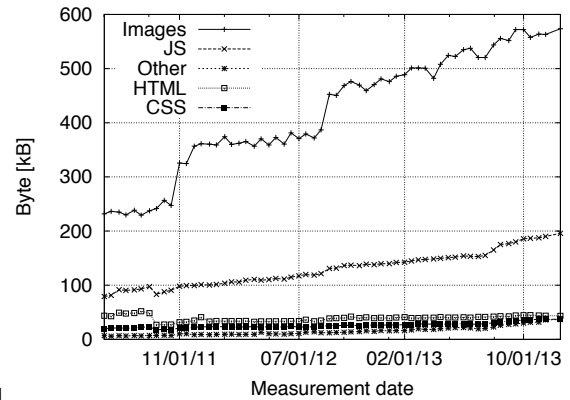the second half, the fixed web pages exhibit a significant increase in the average sizes and variabilities

thereof. However, the overall rising trend is present for both versions and corroborates our overall

rising trend observations.

Investigating the origins of the average number of web page sizes, we illustrate the composition

of the average number of bytes requested separated into HTML, CSS, Java Script, Images, and Other

object categories over time in Figure 22 (b) for desktop client requests. We immediately note that while

all components are increasing in size over time, the main contributors to the growth are the Images and

Other web page object categories. More specifically, we note that the three main contributors to total

web page sizes are images, other components, and Java Script. While the number of bytes contributed

by Java Script has slowed in growth, images (which have almost doubled in the number of bytes) and

other web page items (which more than doubled) continue their growth over time. We can attribute

this behavior to the richer web experience that users demand, with additional interactivity and visually

stimulating appearance. HTML and CSS components of web pages, on the other hand contribute only

a minimum amount of data, with little increase over time. Comparing the detailed categorized values

with the subset of web pages in their desktop versions, we note that the origin of the radical increase in

average sizes can be attributed to the increase of data in the images and other categories. Some of these

individual spikes can be attributed to individual outliers, such as a handful of web pages exhibiting

very large object sizes. On http://www.tumblr.com on April 15, 2013, an approximate 15 MB image

was encountered, which can be seen as anecdotal representation of automated processes (such as the

background image for that particular homepage being picked from user posts) lacking optimization

45

routines (such as downscaling in this case).

Shifting the view to the average sizes of the web page components for mobile clients, we also note an overall growth trend for all categories as illustrated in Figure 22 (d). For mobile clients, however, the overall average web page sizes stem mainly from images and secondly Java Script. The remaining three categories contribute significantly less data and exhibit a slower growth. It is interesting to note from comparison with the desktop counterparts, that the average mobile client image sizes are approximately two thirds of their desktop counterparts. Unlike the average number of requests, however, the average number of bytes clearly identifies Java Script for mobile devices as one of the main causes of increased web page sizes. We exclude the detailed evaluation of the subset's characteristics here, as the results resemble earlier observations and their conclusions.

Data per Web Page Object

Combining the two former evaluations, we now evaluate the average web page object request size. We illustrate the overall and categorized views in Figure 23. For the average number of bytes per web page request, illustrated in Figure 23 (a), we note a steady increase for both desktop and mobile client requests. In greater detail, we note that the overall trend for both request origins is rather linear and close. In turn, we note that both feature an almost identical overall growth rate of about 0.9% every two weeks. Verifying these original findings by comparing them with the smaller size subset of web pages, we note that even these pages exhibit the same underlying trend, albeit in conjunction with increased variability. This is a finding with great long-term implications for simplified approaches to modeling, e.g, in capacity determination scenarios.

Next, we investigate the behavior more closely by evaluating the HTML, CSS, Java Script, Images, and Other categories separately for the desktop and mobile clients in Figures 23 (b) and
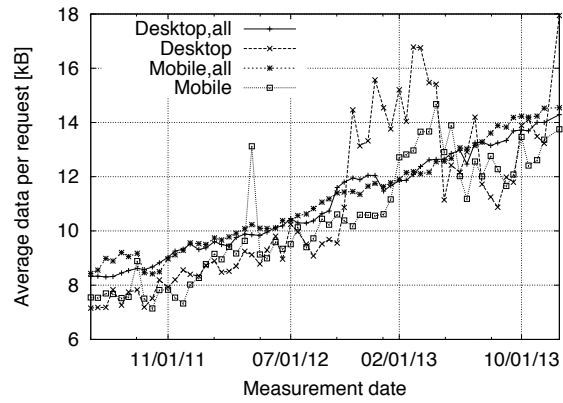
46

[Averages for desktop and mobile client versions.]
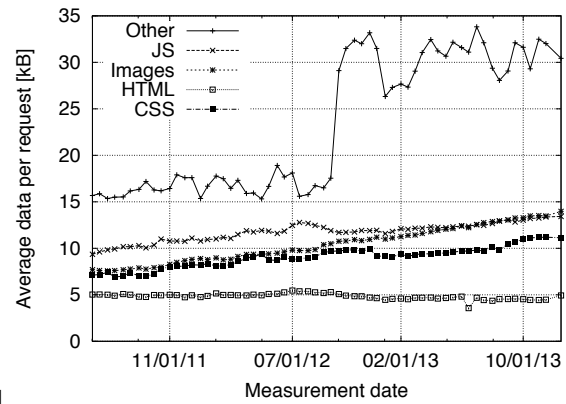
[Categorized for desktop client (all pages).]
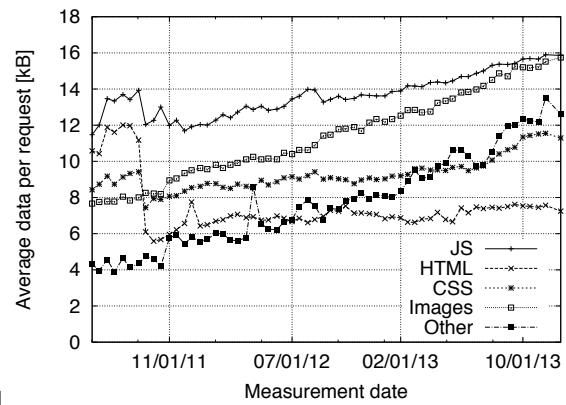
[Categorized for mobile client (all pages).]

Figure 23. Average number of bytes per web page object request for desktop and mobile versions and decomposition into HTML, CSS, Java Script, Images, and Other object categories.

23 (c), respectively. For desktop requests, we note that the highest number of bytes per request can be observed for the Other category, with a pronounced "jump" in the time frame of base URL adjustments in the underlying dataset. In turn, we reason that more components, such as Flash or font data, are included in the larger underlying set of web pages evaluated, which results in the current level of around 33 kB per request in this category. For Java Script and CSS data per request, we observe initially growing trends, which have slowed over time and now are moving beyond 14 and 11 kB, respectively. Images, on the other hand, continue their growing trend and now feature about 14 kB per request. The actual HTML markup request size has remained relatively constant at around 5 kB. For requests from mobile clients, on the other hand, we note that the largest amount per request is observed for Java Script, with currently growing sizes above 15 kB. The Images and Other categories also feature a significant growth for mobile client requests, which have risen above 15 and 12 kB, respectively. The HTML and CSS categories are fairly stable with only minimal growth over time, which is similar to the trends observed for the desktop clients.

In comparison, we note that while both average client request sizes are comparable with respect to their level and growth rates, the actual composition is significantly different. While desktop request sizes are mainly determined by the Other category data, such as Flash or fonts, and increasingly images, mobile requests are mainly characterized by Java Script and Images. The reason behind this behavior could be the increased level of rich media inclusion for desktop clients (e.g., using Flash), whereas automatic adjustments performed (e.g., using Java Script) disable this for mobile clients, where this is not desired. Furthermore, we note that the average amount of bytes per image is fairly close (even a bit higher for mobile), which could be due to increased utilization of content management systems. These commonly utilized frameworks typically adjust web page layouts and theming elements based on the requesting client, e.g., format for mobile display sizes and utilize smaller sized

[Desktop clients]
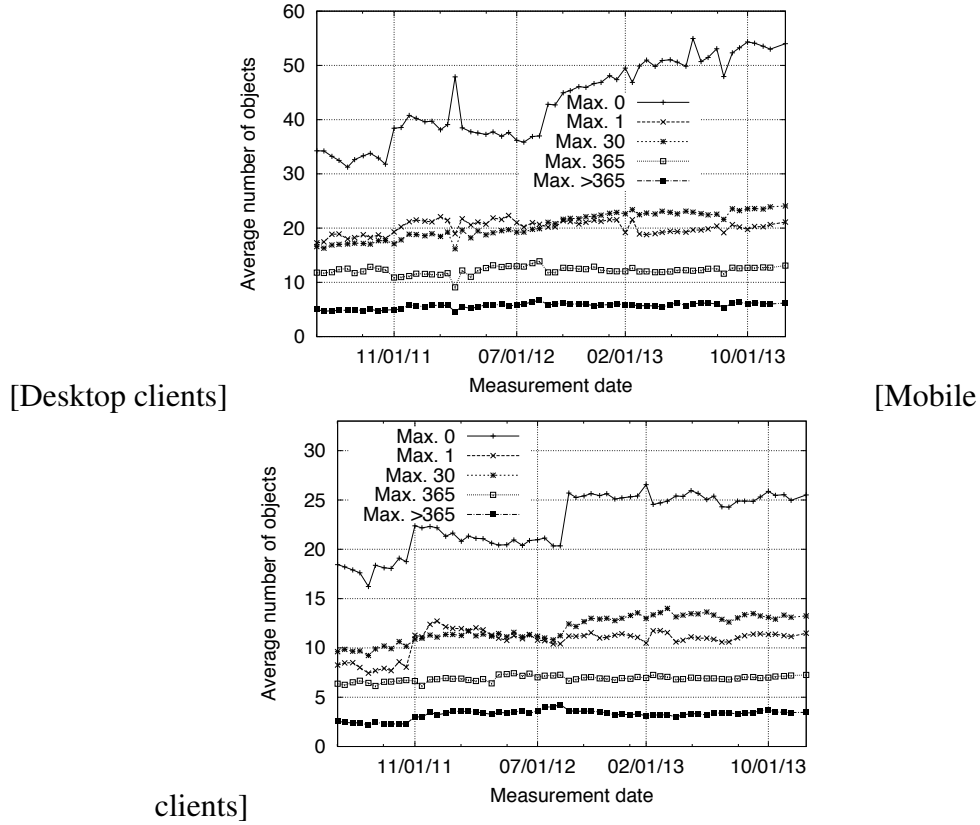
[Mobile clients]

Figure 24. Maximum age of web page objects for desktop and mobile web clients.

theming elements. The actual main content elements, however, typically remain unchanged (which, in

turn, results in identical downloads of the larger sized content images).

Caching of Web Page Objects

One of the main assumptions for the prior evaluation was that the data represents "first views,"

i.e., the dataset's underlying measurements do not consider the caching of web pages after initial visits.

The individual entries in the datasets obtained from httparchive.org, however, additionally contain the

max-age header (amongst other, such as expiration) directives for evaluation of the maximum lifetime

of objects on the requesting device. The locally cached data can have a positive impact on the required

network access, especially for mobile devices. We illustrate the different max-age values obtained

from the data in Figure 24 (a) for desktop client requests. We observe that the largest fraction of

49

items cannot be cached beyond a day, and in turn requires frequent downloads if web pages are visited infrequently. More importantly, this is the only category that exhibits significant continued growth. Over the time horizon we consider here, items that are stored between one and up to 30 days can be considered regularly downloaded at a comparable level, while only a relatively small number of objects can be considered invariant due to larger maximum allowable cache ages. For the mobile client counterparts illustrated in Figure 24 (b), we initially note a similar level and continued growth for the average number of objects that are able to be cached below a day (considering the overall average number of objects requested as discussed in Section V). Objects that require regular, but less frequent downloads as well as more invariant items follow the trends observed for the desktop client. Overall, the implication for mobile clients is that the fraction of requests with a shorter cache lifetime constitutes the majority of requests and in turn will require frequent downloads if web pages are visited infrequently.

Implications

One initial limitation of our dataset is the evaluation of landing pages only, which, however, could be seen as an actual upper limit, according to [11]. Overall, we find significantly growing trends for the average web page number of objects and their sizes. The immediate implications from the growing trend of the number of web request per fixed and mobile website versions and their respective size is that the pressure on access networks capacity growths will continue, see, e.g., [26] for a detailed separation into typical business and consumer categories of accesses Internet data. We find significant similarities between the desktop and the mobile versions of the popular web pages that were part of this evaluation. Mobile counterparts of desktop page versions typically feature less objects and smaller sizes; both characteristics typically also exhibit a significantly slower growth. As briefly outlined by

50

the authors of [11], the reduced number of servers and number of objects involved likely can be seen as an indicator of the correlation between the two different versions of popular web pages.

As previously outlined in [29], Java Script objects in mobile web pages are a main contributor to the rendering power consumption, and we find significant amounts of Java Scripts to be contained in average web pages. More importantly, we conceive that there is a significant amount of data and requests in average mobile pages that is dedicated to image and other objects, which will contribute significantly to the combined transmission and rendering energy costs, as outlined in [29] as well. Optimization efforts that target mobile clients specifically should in turn focus on these categories of data, specifically while taking more advantage of caching possibilities, as the transmission energy contributes a significant amount to the total energy required. Combining the two prior evaluation categories, the average total number of bytes per request for desktop and mobile web pages are very similar and exhibit a linearly growing trend. This finding for the larger dataset is additionally corroborated by evaluation of the continuously present web pages, which exhibit the same trend, albeit at higher levels of variability. While some of the greater details highlight smaller differences by respective type of object requested, the overall direct correlation between the two gives rise to future research questions. Comparing our results with prior longitudinal studies in [27, 10], we note several interesting similarities and other findings, within the constraints given by each of the three studies. First, both prior studies cover date ranges prior to our study's time frame, ending in 2009 and 2010, respectively. They are based on actual user request analyses derived from proxy monitoring, whereas our study specifically utilizes an evaluation of the landing pages of popular web sites, which can be regarded as more static. The HTTP GET transaction size evaluation performed in [27], which is also corroborated by the median page size growth ending around 133 kB reported in [10], provides an almost seamless trend continuation over time (especially when considering the different study parameters, which

include follow-up session browsing).

As indicated by [10] for prior years, our study also finds a significant shift in the bytes per object – with images constituting the majority of traffic in both request scenarios. In the recent years we evaluate, however, we now note a shift to the desktop client's Other category, while mobile clients exhibit a significantly smaller overall size and lower amounts of data in the Other category. Comparing initial caching observations here with the results provided in [27, 10], we note that the overall number of cacheable objects is increasing for the desktop requests, which is in line with the GET request savings described in [27], where about 20-25% were identified. Similarly, [10] reports 27-37.1 % of objects to fall into this category for US-based traffic, resulting in byte savings of 16.8-28.1%. In turn, the cache-based object retrieval can reduce the waiting times to amounts that fall within positive user experience time frames, see, e.g., [32]. In conclusion, our study finds a continuation of prior longitudinal studies when overall trends are regarded, but differences in the lower-level shifts of requests and data by categories and requesting device.

Conclusion

This overview represents the first longitudinal and comparative study of the current state and changes of the World Wide Web, as seen by current fixed and mobile clients when requesting web pages. We utilized the publicly available data from httparchive.org, an industry-supported web page statistics archival project, to outline major trends and resulting implications. For the number of web-page requests, we noted an overall growing trend, with a slightly higher rate for desktop versions of web pages, continuing trends observed in the past. Modern web desktop page versions feature well above 100 object requests on first views, while mobile versions trail above 60. Similarly, we noted a rapidly increasing trend with regard to the average web page sizes, with desktop pages surpassing

1.5 MB in initial view size, while mobile pages trail at more than 50%. Most astonishingly, this study found that the average bytes per request are within close proximity for fixed and mobile clients and exhibit both comparable as well as significant growth rates. When performing decomposition into categories, we found that Image, Other (such as Flash or fonts), and Java Script objects are the main contributors to web page sizes for both clients. Furthermore, the most requests for objects on first view are attained for the Image category of web page objects, again independent of the client type. We also evaluated the potential of caching objects for subsequent views of web pages, whereby we found that most web page objects exhibit very short cache life times of a one day maximum. Our overview thus suggests that there are significant opportunities for future research and optimization efforts for efficient mobile web content delivery. Specifically, our findings indicate the need for more optimization and client considerations when designing modern web pages and frameworks that are utilized for their delivery.

CHAPTER VI

CONCLUSION

Future Work

REFERENCES

[1] Cisco, Inc. Cisco visual networking index: Global mobile data traffic forecast update, 2013–2018. Technical report, February 2014.

[2] Feng Qian, Zhaoguang Wang, Yudong Gao, Junxian Huang, Alexandre Gerber, Zhuoqing Mao, Subhabrata Sen, and Oliver Spatscheck. Periodic transfers in mobile applications: Network-wide origin, impact, and optimization. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 51–60, New York, NY, USA, 2012. ACM.

[3] Andre Luiz Tinassi DAmato, Linnyer Beatrys Ruiz, Anderson Faustino da Silva, and Jose Camargo da Costa. Eprof: An accurate energy consumption estimation tool. In *Proceedings of the 2011 30th International Conference of the Chilean Computer Science Society*, SCCC '11, pages 210–218, Washington, DC, USA, 2011. IEEE Computer Society.

[4] Aaron Carroll and Gernot Heiser. An analysis of power consumption in a smartphone. In *Proceedings of the 2010 USENIX Conference on USENIX Annual Technical Conference*, USENIX-ATC'10, pages 21–21, Berkeley, CA, USA, 2010. USENIX Association.

[5] N. Davies, M. Langheinrich, R. Jose, and A. Schmidt. Open display networks: A communications medium for the 21st century. *Computer*, 45(5):58–64, May 2012.

[6] Patrick Seeling. An overview of energy savings potentials through mobile forwarding proxy framework. *Int. J. Ad Hoc Ubiquitous Comput.*, 16(4):260–267, September 2014.

[7] Patrick Seeling. Caching proxying for mobile users. In *Vehicular Technology Conference (VTC Spring), 2013 IEEE 77th*, pages 1–5, June 2013.

[8] "M. Leech, M. Ganis, Y. Lee, R. Kuris, D. Koblas, and L. Jones". Socks protocol version 5. RFC 1928 (Proposed Standard), Mar 1996.

[9] K. Kouzoubov. Java socks proxy. Available: http://sourceforge.net/projects/jsocks/, 2011.

[10] Sunghwan Ihm and Vivek S. Pai. Towards understanding modern web traffic. In *Proc. of the ACM SIGCOMM Internet Measurement Conference (IMC)*, pages 295–312, Berlin, Germany, November 2011.

[11] M. Butkiewicz, H.V. Madhyastha, and V. Sekar. Characterizing web page complexity and its impact. *Networking, IEEE/ACM Transactions on*, 22(3):943–956, June 2013.

[12] Niranjan Balasubramanian, Aruna Balasubramanian, and Arun Venkataramani. Energy consumption in mobile phones: A measurement study and implications for network applications.

In *Proc. of the ACM SIGCOMM Internet Measurement Conference (IMC)*, pages 280–293, Chicago, IL, USA, November 2009.

[13] Feng Qian, Zhaoguang Wang, Yudong Gao, Junxian Huang, Alexandre Gerber, Zhuoqing Mao, Subhabrata Sen, and Oliver Spatscheck. Periodic transfers in mobile applications: Network-wide origin, impact, and optimization. In *Proc. of the International Conference on World Wide Web (WWW)*, pages 51–60, Lyon, France, April 2012.

[14] Yu Xiao, Pan Hui, Petri Savolainen, and Antti Ylä-Jääski. Cascap: Cloud-assisted context-aware power management for mobile devices. In *Proc. of the International Workshop on Mobile Cloud Computing and Services (MCS)*, pages 13–18, Bethesda, MD, USA, June 2011.

[15] F. Sailhan and V. Issarny. Energy–aware web caching for mobile terminals. In *Proc. of the International Conference on Distributed Computing Systems (ICDCSW)*, pages 820–825, Vienna, Austria, July 2002.

[16] Huaping Shen, Mohan Kumar, Sajal K. Das, and Zhijun Wang. Energy–efficient data caching and prefetching for mobile devices based on utility. *Mobile Networks and Applications*, 10(4):475–486, August 2005.

[17] B.N. Thyamagondlu, V.W. Chu, and R.K. Wong. A bandwidth–conscious caching scheme for mobile devices. In *Proc. of the IEEE International Congress on Big Data (BigData Congress)*, pages 78–85, Santa Clara, CA, USA, June 2013.

[18] Patrick Meenan. How fast is your web site? *ACM Queue*, 11(2):60–70, March 2013.

[19] G. P. Perrucci, F. H P Fitzek, and J. Widmer. Survey on energy consumption entities on the smartphone platform. In *Proc. of the IEEE Vehicular Technology Conference (VTC Spring)*, pages 1–6, Budapest, Hungary, May 2011.

[20] Patrick Seeling. Caching proxying for mobile users. In *Proc. of IEEE Vehicular Technology Conference (VTC)*, pages 1–5, Dresden, Germany, June 2013.

[21] Troy A. Johnson and Patrick Seeling. Desktop and mobile web page comparison: Characteristics, trends, and implications. *IEEE Communications Magazine*, 52(9):144–151, September 2014.

[22] Serge A. Krashakov, Anton B. Teslyuk, and Lev N. Shchur. On the universality of rank distributions of website popularity. *Computer Networks*, 50(11):1769–1780, August 2006.

[23] Giuseppe Anastasi, Marco Conti, Enrico Gregori, and Andrea Passarella. Performance comparison of power–saving strategies for mobile web access. *Performance Evaluation*, 53(3–4):273–294, August 2003.

[24] Paul Barford and Mark Crovella. Generating representative web workloads for network and server performance evaluation. *SIGMETRICS Perform. Eval. Rev.*, 26(1):151–160, June 1998.

[25] Zhichun Li, Ming Zhang, Zhaosheng Zhu, Yan Chen, Albert Greenberg, and Yi-Min Wang. Webprophet: automating performance prediction for web services. In *Proceedings of the 7th USENIX conference on Networked systems design and implementation*, NSDI'10, pages 10–10, Berkeley, CA, USA, 2010. USENIX Association.

[26] Inc. Cisco. Cisco visual networking index: Forecast and methodology, 2012âĂŞ2017. Technical report, may 2013.

[27] Tom Callahan, Mark Allman, and Vern Paxson. A longitudinal view of http traffic. In Arvind Krishnamurthy and Bernhard Plattner, editors, *Passive and Active Measurement*, volume 6032 of *Lecture Notes in Computer Science*, pages 222–231. Springer Berlin Heidelberg, 2010.

[28] Michael Butkiewicz, Harsha V. Madhyastha, and Vyas Sekar. Understanding website complexity: measurements, metrics, and implications. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, IMC '11, pages 313–328, New York, NY, USA, 2011. ACM.

[29] Narendran Thiagarajan, Gaurav Aggarwal, Angela Nicoara, Dan Boneh, and Jatinder Pal Singh. Who killed my battery?: analyzing mobile browser energy consumption. In *Proc. of the International Conference on World Wide Web (WWW)*, pages 41–50, Lyon, France, 2012.

[30] httparchive.org. Http archive (beta), July 2013.

[31] Henri Theil. *Statistical decomposition analysis. With applications in the social and administrative sciences*. North-Holland Pub. Co, Amsterdam, 1972.

[32] S. Niida, S. Uemura, and H. Nakamura. Mobile services. *Vehicular Technology Magazine, IEEE*, 5(3):61–67, 2010.