# Introduction

The increasing prevalence of mobile devices has resulted in a significant escalation of mobile data traffic amounts.

The increased Internet connectivity of mobile devices has, in turn, led to a point where web access is mainly performed with the help of personal mobile devices, rather than desktop or fixed computers, see, e.g.

[1].

Web page complexity can be measured in terms of the number of objects contained, their type, or object lifetimes.

For desktop web pages,

While an evaluation of interactions and web pages that are presented after interaction takes place provides additional insights, evaluations noted that typically, landing pages exhibit the highest levels of complexity {}.

Recent evaluations indicate that the type of object retrieved to display a mobile web page has significant impacts on the energy required for rendering and resulting battery taxation, see, e.g., {}.

While most web pages in the past were

Recent trends not only indicate that the complexity of web pages is steadily increasing, but that there is a strong correlation between desktop versions and mobile versions of landing pages, see, e.g., [2].

Furthermore, the number of web page objects has dramatically increased and follows a general linear trend allowing for basic approximation of future demands on the delivery of mobile web pages.

This work focuses on the description of the simulation of

This is an important feature for future mobile service providers, as an identification of bottlenecks can greatly be facilitated and the potential for mobile content delivery be ubocked. Especially for

The remainder of this paper is structured as follows.

In the following section, we briefly describe the main dataset used for evaluation of mobile service landing web pages.

We subsequently describe the

Data Set Description

We intitially describe the underlying dataset for our evaluation. The dataset is for an October 15, 2013 evaluation of the most popular mobile service landing pages and is publicly provided by httparchive.org. This dataset was gathered by making “clean” requests, i.e., considering not prior data cached, by web browser agents, which initially store the resulting interaction with the remote server in a HTTP Archive (HAR) format \cite{}. An evaluation of web pages and requests is presented by httparchive.org as an aggregated database set that can be dowloaded. As described by the providers of the dataset, timing considerations within have to be viwed with caution; however, the web page object characteristics themselves are not prone to such consideraitons.

The particular data set under consideration originally contains invalid links and other, non-positive responses, which are filtered for the processing in the context of this paper. The October 15, 2013 dataset for mobile web page requests contains xx page requests with a total of xx object requests as starting point of the analysis.

Correlations

As our remaining approach focuses on the approximation of mobile web page characteristics through simulation, we initially evalaute the correlation between the sizes and expiration ages as main characterisitcs. We initially evalaute the correlations for all web pages, which results in a linear correlation factor of 0.017 and is in line with findings presented in \cite{}. Next, we evaluate the correlation in interdependencies with popularity rank and number of objects (as measure of complexity) as provided in Table .

|  |  |  |
| --- | --- | --- |
|  | Top 500 pages | |
|  | Complexity | Popularity |
| Objects/Expiration | -0.036 |  |
| Objects/Sizes | -0.024 |  |
| Rank/Objects | 0.043 | 0.150 |
| Rank/Page Size | 0.038 | 0.113 |
| Expirations/Sizes |  | 0.011 |

We note that no significant correlations can be observed, idnepdently of creating a subset of pages based on complexity or popularity. Furthermore, we note that in most evaluations, the coefficient of correlation remains close to zero, indicating a lack of linear relationship between expiration and size. The oonly indicator that can be derived from the valuation presented in Table~\ref{} is that increased popularity of a web page from the dataset shows a slight tendency to result in more objects and sizes.

Binning of data

We accmodate common mobile communication scenarios by aggregating object expiration ages by the number of seconds that objects are allowed to remain in cache. For bin sizes ranging from 15 minutes to whole days, the coefficient of correlation remains around 0.017. Similar results can be observed when evaluating the cache expiration boundaries and related object sizes over other time frames.

Evaluation of Mobile Web Page Objects

In this section, we evaluate the possibilities of modeling the main web page object characteristics, expiration age and size, across all objects from the most popular mobile landing web pages, contained in the httparchive.org dataset.

Expiration Ages

The distribution of expiration ages, which we also described in [bytebynight] is very long--tailed with a number of significant interruptions at logical time intervals, such as hourly, daily, etc.

To capture these properties, we employ a piecewise approach, using the distribution of the web page elements throughout the dataset and the respective expiration time characteristics they fall into. Specifically, we consider a time frame below one week, due to the high likelihood of additional requests and other impacts beyond that time frame being of less importance in practice~\cite{}.

**ExpAges (results in overallvalues excel sheet)**

N = number of expAges to generated

For x = 0; x<N;x++

x= Random(0,289355) //Generate random value between 0 and 289355

If x < 122004:

expAge = 0

ElIf x < 156484:

expAge = Generate expAge value from fisk distribution (c=0.88945,loc=1,scale=2600.7)

ElIf x < 160381

expAge = Generate random expAge between [36001, 42000]

ElIf x < 162203

expAge = Generate random expAge [43201,79200]

ElIf x < 176796

expAge = Generate random expAge [79201,86400]

ElIf x < 177367

expAge = Generate random expAge [86401,165600]

ElIf x < 179332

expAge = Generate random expAge [165601,172800]

ElIf x < 179788

expAge = Generate random expAge [172801,252000]

ElIf x < 181320

expAge = Generate random expAge [252001,259200]

ElIf x < 182440

expAge = Generate random expAge [259201,424800]

ElIf x < 183139

expAge = Generate random expAge [424801,432000]

ElIf x < 183985

expAge = Generate random expAge [432001,597600]

ElIf x < 199734

expAge = Generate random expAge [597601,604800]

Else

Value over 604,800. Only concerned with values less than or equal to week. Generate a different expAge (generate another random number and try to map it to the appropriate expAge range again).

//Output expAge

Response Sizes

**respSizes (results in overallvalues excel sheet)**

Distribution follows a weibull distribution(a= 0.57264, c = 0.50556451275314185, scale = 8887.5)

N = number of respSizes to generated

For x = 0; x<N;x++

respSize = weibull (a= 0.57264, c = 0.50556451275314185, scale = 8887.5)

//Output respSize

Simulation of Mobile Web Pages

Ran Simulation for 2000 Rounds (each round consisted for simulated 4775 pages)

**Page Level Algorithm**

N = number of pages to simulate

For x=0;x<N;x++:

expAges = [] //Empty List

respSizes = [] //Empty List

//Determine whether or not page will contain expAges = 86400

Y = randomvalue(1,4775)

If Y < 3046:

Large\_expAge = True //ExpAges CAN be >= 86,400 (can also still be < 86400 too)

Else:

Large\_expAge = False //ExpAges can’t be >= 86,400

//Determine initial proportion of expAges that will be set to zero

PropInitialZeros = **generateProportionZeros() – from commag paper**

**//**Determine number of responses for page

numResponses = **generateNumResponses()** //numResponses must be > 1

//Multiply proportion of zero-value expAges requests by number of requests to get number of //requests to initially set expAge to zero for.

numZeroExpAges = Math.ceil(numResponses \* PropInitialZeros)

For r=0;r<numZeroExpAges;r++ //Put in initial proportion of zero values expAges into list

expAges.append(0)

For s = 0;s< numResponses – numZeroExpAges;s++ //Generate rest of expAges into list

expAges.append(**generateExpAge(**Large\_expAge**)**)

For t = 0;t< numResponses – numZeroExpAges;t++ //Generate respSizes into list

respSizes.append(**generateRespSize()**)

//Output average of expAges list for average expAges of page

//Output average of respSizes list for average respSizes of page

**generateNumResponses()**

return Math.Ceiling(gengamma(alpha=3.1119,k=0.6167,scale=8.2912,loc=0.17238))

**generateExpAge (boolean Large\_expAge)**

expAge = **determine\_bin\_expAge(**random\_value(0,289355))

if(Large\_expAge == True):

while expAge > 604800: //Not concerned with expAges > 1 week, redraw value

expAge = **determine\_bin\_expAge**(random\_value(0,289355))

else:

while expAge >= 86400: //Page expAge must be < 86400 (large\_expAge is false)

expAge = **determine\_bin\_expAge**(random\_value(0,289355))

return expAge

**determine\_bin\_expAge(int x)** //Maps the integer to the appropriate range to generate expAge within

If x < 122004:

expAge = 0

ElIf x < 156484:

expAge = Generate expAge value from fisk distribution (c=0.88945,loc=1,scale=2600.7)

ElIf x < 160381

expAge = Generate random expAge between [36001, 42000]

ElIf x < 162203

expAge = Generate random expAge [43201,79200]

ElIf x < 176796

expAge = Generate random expAge [79201,86400]

ElIf x < 177367

expAge = Generate random expAge [86401,165600]

ElIf x < 179332

expAge = Generate random expAge [165601,172800]

ElIf x < 179788

expAge = Generate random expAge [172801,252000]

ElIf x < 181320

expAge = Generate random expAge [252001,259200]

ElIf x < 182440

expAge = Generate random expAge [259201,424800]

ElIf x < 183139

expAge = Generate random expAge [424801,432000]

ElIf x < 183985

expAge = Generate random expAge [432001,597600]

ElIf x < 199734

expAge = Generate random expAge [597601,604800]

Else

Value over 604,800. Only concerned with values less than or equal to week. Generate a different expAge (generate another random number and try to map it to the

**generateRespSize()**

return weibull\_min(a=0.27,scale=700,loc=2.5) //Weibull\_min dist, aka Frechet right distribution

**generateProportionZeros()** //Generates the the proportion of responses that will be have value of zero

return normal(0.629075947,0.300892868) //Normal Distribution

Performance Analysis

Using the simulated mobile web pages in simulations

Using the linear predictor to simulate mobile web pages over time is simple, as our findings indicate that one needs to predominantly consider the average number of web page elements over time and only to a lesser extend their composition if the main focus of an evaluation is on web page sizes and chanching opportinties.

[1] Cisco, Inc., “Cisco Visual Networking Index: Global Mobile

Data Traffic Forecast Update, 2013–2018,” Cisco, Inc., Feb. 2014.

[2] T. A. Johnson and P. Seeling, “Desktop and Mobile Web Page Comparison: Characteristics, Trends, and Implications.,” *IEEE Communications Magazine*, 2014.