Good morning. I will introduce a paper included in INFOCOM2015 named What is wrecking your data plan? A measurement study of mobile web overhead.

My content includes five parts. Let me introduce abstract at first, then analyze the unnecessary overhead sources, after know what is unnecessary, the next step is find a way to detect them, so we propose a measurement framework. i will show the experimental results later and make discussion at last.

The growing popularity of smartphone and user demand for a rich web experience has result in the surge in cellular bandwidth requirements. While many websites have adapted to the new mobile habitat, they often take shortcuts to transition from their desktop to mobile version, many times carrying redundant content that is never used. So mobile users are often paying for certain undesirable content, in the form of their bandwidth costs.

In this paper, we study the composition and complexity of modern websites, from both desktop and mobile perspective, to identify the wasted bandwidth. We developed a crawler-based framework to perform analysis of the 100 000 popular sites and results show 23% or more of the content size is unnecessary, unused or redundant.

Before detecting unnecessary data, we should know what is unnecessary, right? So we establish the following principle to guide our overhead measurement. A piece of content is considered unnecessary overhead if it satisfied at least one of the following conditions. The first .. The second .. And the last .. We evaluate page resources against the three conditions and marked them as unnecessary if one of the conditions are satisficed.

Indeed, our work is related to performance measurement study, but with several key differences. Most significantly, we don’t consider timing and latency. Instead, we focus on the content size of assets transferred over the network. Our goal is just to identify total added bandwidth consumed by data that is redundant and costly to the user.

Modern web pages include a rich variety of content types, for our analysis, we focus on six major components of a website. Since our evaluation shows that on average, they make up to 95% or more of the total size of website.

The first is image overhead, they are major contributors to the total page size with contributions of 63% and 64% of the total page weight for desktop and mobile sites respectively. Even though several image compression techniques exist, they are often not utilized by web developers.

The second is unreachable code in JavaScript and CSS, many websites use the same code base for both desktop and mobile version that often results in code that remains included but never used.

The third is advertisements. They introduce additional bytes that must be downloaded for each website, which effectively represents an additional cost to the end users.

The fourth is unused font, web fonts are included on a page as an external font file that is downloaded by the browser, these font files include a large set of possible fonts, even though the site does not consume all of them.

The fifth is comments and whitespaces in HTML, JavaScript and CSS, they are important to web developer, but have no value to the end user.

The last is improper use of cookies, in many cases the server doesn’t require all the cookies in order to serve the requested content.

We built a customized framework used to crawl and capture all requests and responses, and subsequently analyze the raw data. Another major component in the framework is JSCover proxy, which is a tool that enables us to measure code coverage for JavaScript.

We analyzed each web page with respect to various metrics that measures the impact on data usage on desktop and mobile devices.

Some measurements were performed at the time the web page were crawled. For example, we calculate the content length at the time of crawling since HTTP headers sometimes report incorrect content length for a given response. We also measured the content size online across different MIME types.

Other measurements, such as those resulting from code coverage and image optimization, are done offline after the websites are crawled.

For JavaScript coverage, we used the JSCover tool to find unused JavaScript code. JSCover is an open source tool that you can find it on github. JSCover saves the resulting coverage report to disk and we are then able to parse these report to infer lines of code not covered.

For images and web fonts measurements, we use compression tools that optimize the files by applying lossless compression. And we also use compression tools to strip out whitespace and comments for JavaScript, CSS and HTML.

In order to measure advertisements, we first needed to detect what portion of each website represented an advertisement. We used popular filter list used by ad blocker, so we are able to extract the segments that are identified as advertisements to measure and analyze their size overhead.

While the focus of our work is mobile due to its higher per-byte data cost, the insights from this study can also equally benefit desktop websites design practices. Our measurements show that the average size of a mobile website is 923 kilobytes, and the desktop is 1741 kilobytes.

Previous studies have shown that images make up the majority of the total page weight on the average website. From fig 2, we also see the prominence of JavaScript code in both mobile and desktop sites. It is interesting to note that mobile websites use relatively more JavaScript code, which could give rise to concerns related to performance bottlenecks and overhead.

Since images are the majority contributors to the page weight, their impact on data usage is significant. Images take up about 63% and 64% of total page weight for desktop and mobile sites respectively. We found that entertainment websites use a larger volume of image and finance websites have the least average image size.

We found that three formats – GIF, PNG, JPG – tend to be the most popular.

Despite best practice recommendations, our analysis shows that images are rarely optimized by most websites.

In terms of reducing bytes, one intuition is that converting GIF to 8-bit PNG can result in reduced image sizes while still preserving quality. We used OptiPNG image compression tool to convert GIF to PNG, and we saw an average of 13% file size reduction. Next, we ran the compression tool on existing PNG images and found a significant reduction in file size at about 31% on average.

Considering image dimensions, we found that a significant number of images exceeded the most popular 640x960 screen resolution. So image size reduction can be further enhanced if the image quality is appropriately optimized for the target screen quality.

For analysis of code usage, based on code coverage analysis, we found that up to 33% of JavaScript code remains unused on mobile web pages and 28% on desktop. So we want to know what contributes to unused code? From our analysis, we noted that in many instances unused code is attributed to blocks of code from JavaScript framework libraries. In our mobile dataset, we found that 58% of sites made use of some form of the jQuery framework. The top 5 libraries we identified are shown in table.

For CSS, there are two main methods of reducing the size of CSS files. First, the CSS files can be compress similar to JS files by using tools. And second, we can analysis of CSS specificity rules to determine the unused CSS rules. On mobile sites, we found the average was 71% of the rules not used.

In summary, with regard to the total overhead for unused code (include JS and CSS), our results show that a total savings of 9.2% can be achieved on mobile websites, and 6% can be achieved on desktop.

We also quantify the contribution of advertisement content to the total weight of a page. We found that advertisements account for about 10% of the total page weight. It consume an average of 75 kilobytes on mobile web pages, and 153 kilobytes on desktop pages.

Recent trends point to increasing adoption of web fonts across the web. Web fonts consist of external font files downloaded by the browser and utilized on a website through CSS rules. We found that over half of all font files are not utilized by browser in rendering the landing page, meaning they present significant overhead in terms of page size. So we use the open source sfntly library from Google to apply compression to our font files. After using this tool, font file were reduced by 40.3% for desktop sites and 29.4% for mobile.

The aggregated measurement results show that websites carry a significant overhead of redundant data. From table, not surprisingly, images and JavaScript carry the most overhead of all components. Overall, our results show that on an average websites suffer from a substantial overhead of more than 23%, thus motivating the urgent need to employ data optimization techniques.

And there is a interesting thing I want to add, from fig 5, we see a CDF of the overhead for JavaScript, CSS and images on mobile sites according to sites’ ranking. The site ranking seem to have little bearing on the overhead measurement, meaning both higher and lower ranked sites have almost equal tendency to exhibit similar overhead measurement.

Ok, let’s make a discussion.

In many performance optimization, they focus on reducing latency by minimizing the number of HTTP requests. This often leads developers to concatenate several code files into one file, but without further consideration of overhead in the code. But in our study, we concentrate on content size to find overhead of the various websites.

While some optimization tools are available today, we find that such tools are not being utilized. There is a need for comprehensive, automated solution that would optimize website without solely relying on developers. The insight gained from our measurement study could be leveraged while developing such a system, so maybe, it is our future work.

In a word, the primary contribution of our solution is to remove unused and redundant data that is unnecessary for the functionality of the website.