ExploreBP: A Simulation Tool for Mobile Browser Energy Optimization

Jin Zhang[1], Xin Wei[1], Zhen Liu[1], Fangxin Liu[1], Tao Li[1,3], Tingjuan Lu[2], and Xiaoli Gong[1,3](✉)

[1] Nankai University, Tianjin, China
gongxiaoli@nankai.edu.cn
[2] IT Department, Chinese PLA 117 Hospital, Hangzhou, China
[3] State Key Laboratory of Computer Architecture,
Institute of Computing Technology, Chinese Academy of Sciences,
Beijing, China

Abstract. The browser is one of the most commonly used applications. Users tend to pursue a good user experience and care more about the performance of the browser, while ignoring the power consumption of the browser. This paper proposes a method to reduce the energy consumption of web browsing. In order to better quantify the user experience, this paper uses the first screen load time as the evaluation metric of user experience. First, according to the relationship between the network speed and the first screen load time, find the most suitable primary frequency at a specific network speed, and define the point as the balance point. When the primary frequency is greater than the primary frequency corresponding to the balance point, the first screen load time will almost never change. The balance points of different web pages are also different. Then adjust the CPU frequency according to the balance point of the webpage and the network speed, which can reduce the browser energy consumption and reduce the impact on the user experience. At the same time, this paper proposes a simulation tool ExploreBP, which is used to simulate the working state of the network speed and different web pages to find the optimal energy consumption configuration.

Keywords: Mobile web browser · Energy optimization · Web page loading · CPU frequency modulation

This work is partially supported by the National Key Research and Development Program of China (2018YFB1003405), the National Natural Science Foundation of China (61702286), the Natural Science Foundation of Tianjin, China (18JCYBJC15600), Open Projects of State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences (CARCH201604), and Special Funding for Excellent Enterprise Technology Correspondent of Tianjin, China (17JCTPJC49500).

H. Song et al. (Eds.): SIMUtools 2019, LNICST 295, pp. 248–257, 2019.
https://doi.org/10.1007/978-3-030-32216-8_24

1 Introduction

Smartphones are widely used, but energy consumption has always been a bottleneck in their development. Limited by the screen, processing power and battery life of the mobile phone [14], although the battery energy density has increased by 3 times [2] in the same period, the endurance of most smartphones has not been significantly improved.

The browser is the main application of smartphones. The survey shows that more than 73% of smartphone users frequently browse the web. Google even launched Chromium OS [5], which is an operating system that fully uses web applications. So the browser should be the goal of optimization.

The browser is a very complex application and various studies work on improve its performance. It has been proved that the main bottleneck in the performance of mobile browsers is CPU computing power rather than networking [12]. Wang et al. [15] found that the resource loading contributed the most to network latency. Nevertheless, the research on energy consumption is relatively insufficient. Bui et al. [1] proposed an application-assisted scheduling technique to reduce energy consumption. However, the scheduling technique only adjusts the thread based on the drawing speed, which doesn't fully consider the characteristics of the different processes of the browser and the load of the webpage.

The optimization of energy consumption for smart mobile devices should be combined with the user experience. The browser interacts directly with the user. We propose the first screen load time to measure the user experience. The first screen load time is the time consumed by the browser to display the first screen page. That is, the time when all the elements in the area that the user can see are loaded. As for the user experience, when the first screen area is full of content, the user can see the main content of the web page.

The first screen load time is closely related to CPU processing speed, network speed, and web content [13]. Previous studies have shown that both network and frequency may become bottlenecks. The network speed affects the download speed of the browser process, and the frequency affects the processing speed of the rendering process. Any slowdown can result in increased page load time, affecting the user experience and increasing power consumption.

This paper proposes a method for cooperative modulation of network speed and frequency. For a specific web page, we find the most suitable frequency for the current network speed and web page based on the first screen load time, which is called the balance point. Adjusting the frequency according to the balance point can reduce the energy consumption of web page loading and reduce the impact on the user experience, which can avoid the waste of resource and save energy.

In order to solve this problem, this paper proposes a simulation method. For different web pages, the optimal match of network speed and frequency is found by detecting the first screen load time under different network speeds and CPU frequencies automatically. So the goal of energy optimization can be find. According to this method, a simulation tool ExploreBP is designed to find the optimal match. And the top 50 sites in Alexa Top Sites Chinese have been tested.

The contributions of this paper are as follows: 1. We propose a method for energy optimization by adjusting the main frequency based on the network; 2. We design a simulation tool to test the network speed, main frequency and open speed; 3. The tool is implemented and experiments on 50 websites are performed.

2 Background

2.1 Chromium Browser Architecture

The architecture of Chromium browser [6] is shown in Fig. 1 [1]. Chromium includes a browser process and multiple rendering processes. The rendering process is executed on the Blink [4] rendering engine, which is parsed and executed by the JavaScript engine. The browser process maintains a network stack that receives web resources from the server and shares these web resources between all rendering processes. Every process in Chromium has multiple threads. Processes and threads are executed asynchronously. The browser process and the rendering process communicate with each other through the pipeline, and exchange data through the shared memory.

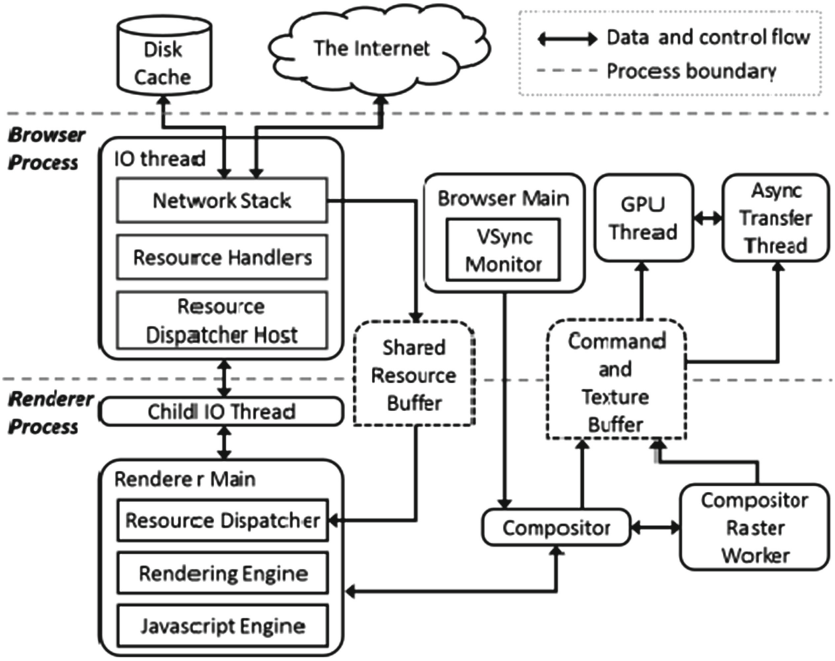


Fig. 1. Architecture of the Chromium browser

First, the browser process puts the received webpage resources into the shared resource buffer, and then the rendering process reads the data from the shared resource buffer to create a graphic layer corresponding to the web page. Because the render process running in the sandbox does not have the privileges of accessing to the GPU directly, the generated graphics data is loaded into the command and structure buffers through the synthesizer thread. The GPU thread processes the graphics data to produce the final web page image.

2.2 Webpage Loading Process

The browser loads the webpage [3] as a synchronization process that renders while parsing. As shown in Fig. 2 [3], the browser firstly parses the HTML file to build the DOM tree, and then parses the CSS file to build the rendering tree. After the rendering tree is built, the browser begins to lay out the rendering tree, which is to assign each node a geometric information that should appear on the display. The final step is to draw the nodes onto the screen.

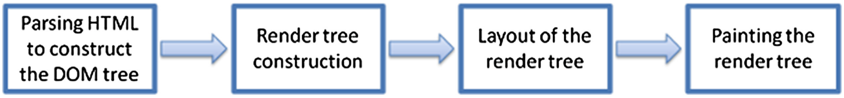

Parsing HTML to construct the DOM tree
Render tree construction
Layout of the render tree
Painting the render tree

Fig. 2. The process of loading a webpage by a browser

3 Experiment

The architecture of the browser and the loading process of the webpage are very complicated. It is impossible to express the whole process of webpage loading by formula, so it is impossible to find the CPU frequency that best matches the current network speed by formula derivation. We simulate the process of loading webpage and find the CPU frequency that best suits the specific network speed and the specific website based on the experimental measured data.

In order to ensure the accuracy of the experimental results, it is necessary to eliminate the interference factors in the process of loading webpage. Network speed and caching are the two main factors. We will clear the cache and disable the cache. Since the actual network speed may be unstable at each load, we use Web Page Replay [8] store the web page locally to simulate different network speeds and CPUFreq [16] to set different CPU frequencies. Then the performance analysis tool Trace-viewer [7] is used to record the page load time and the first screen load time.

3.1 Experimental Tools

WPR (Web Page Replay) WPR is a tool that enables web page to playback. It is composed of the DNS server and HTTP server running locally. It generally works in two modes: Record mode [13]: WPR acts as a proxy server, sends an HTTP(s) request to the server, and then records all the responses of the server. Eventually HTTP(s) requests, responses, web content, and network delays are stored in the local HTTP Archive. Replay mode [9]: WPR starts the simulated DNS server and web server locally on the device. The communication between the browser, the DNS server and HTTP server is hijacked by WPR. The browser can only communicate with WPR.

Trace-Viewer is a data analysis tool on the Chrome browser. It records the data of the browser and visualizes the recorded data. It saves the result as a JSON file.

CPUFreq is a lightweight CPU tuning tool on Linux that can select a core and adjust the frequency. It supports multiple CPU modes and can be set manually, but the range of modulation is limited.

3.2 Measurement

There are many measurement for webpage loading. The most common is the page load time. In the experiment, the first screen load time is used, beacuse it better reflects the user experience.

The page load time is the time spends by the browser to initiate a request until the page is fully loaded. All web resources are added to the DOM tree, and all images, scripts, links, etc. have been loaded.

The first screen load time is defined as the time taken by the browser to display the first screen page. This means that all elements in the visible area of the user are loaded.

3.3 Dataset and Variable Settings

Dataset. The top 50 websites selected from Alexa Top Sites Chinese [10] are used as test sites.

Network Speed Setting. We tested the first screen load time on six network types. In order to simulate the download speed and network delay in the real scene, the network speeds and delays of these six network types are set to the values in Table 1 [11].

Frequency Setting. We adjust the CPU frequency to 40%, 60%, 80%, and 100% of the initial frequency.

3.4 Experiment Environment

Host: The CPU's single core turbo frequency is 4.0 GHz, and the full core turbo frequency is 3.8 GHz. The memory size is 8 G. Virtual machine: The system is Ubuntu17.10, quad-core processor, and the memory is 1 G.

Table 1. Network speed and delay for six network types.

Network type	Download speed	Latency
Regular 2G	250 Kb/s	300 ms
Good 2G	450 Kb/s	150 ms
Regular 3G	750 Kb/s	100 ms
Good 3G	1 MB/s	40 ms
DSL	3 MB/s	2 ms
Regular 4G	4 MB/s	20 ms

3.5 Experimental Procedure

Turn off all applications not related to this experiment to prevent them from affecting the results of the experiment.

- *Step 1.* Set the CPU frequency using the CPUFreq tool. During the experiment, all four cores of the processor are set to the same frequency.
- *Step 2.* Open the Chromium browser and clear your browser's cache to ensure that the experimental results are not affected by the browser cache.
- *Step 3.* Open Web Page Replay and set it to Record mode. Then one of the URLs in the dataset will be entered to the browser. Web Page Replay will receive the HTTP request from the browser and send the request to the network. After the HTTP server responses, it sends the response to the browser, and then automatically saves all the records as archive.wpr file.
- *Step 4.* Set Web Page Replay to Replay mode, sp that the browser can only communicate with Web Page Replay at this time. Web Page Replay will reply to the browser's request using the saved archive.wpr.
- *Step 5.* Use Web Page Replay to set the download speed and delay of objects on the critical path. Objects not on the critical path will not affect the first screen load time and page load time.
- *Step 6.* Enter the test URL to the browser. Web Page Replay sends saved web pages to the browser. At this time, the Trace-Viewer is opened to obtain the first screen load time and the page load time, and the test results are recorded.

4 Analysis of Results

Limited by space, the experimental results only show the top three websites in the 50 websites, which are Baidu, Tencent and Taobao. These websites have large traffic and different characteristics, which are very representative.

4.1 Relationship Between Load Time and Frequency at a Fixed Network Speed

Figures 3 and 4 show the relationship between the load time and frequency of the three websites under DSL (Digital Subscriber Line) conditions. We can see that as the CPU frequency increases, the load time of the three websites is decreasing. When the frequency increases to a certain value, the page load time and the first screen load time almost don't change. It can be speculated that as the CPU frequency increases further, the load time will keep at a certain value.

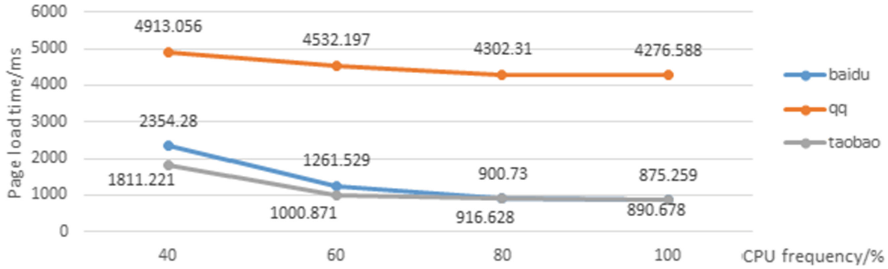


Fig. 3. Page load time of three websites varies with CPU frequency under DSL

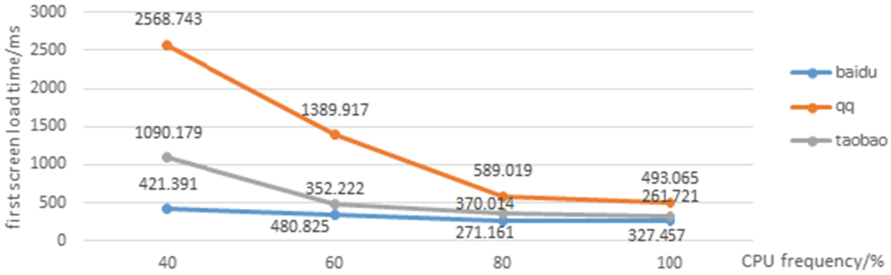


Fig. 4. First screen load time of three websites varies with CPU frequency under DSL

According to the architecture of the browser, the browser process firstly downloads the web resources from the network and puts it into the shared resource buffer. Then multiple rendering processes start processing the resources in the shared resource buffer. The increase of the CPU frequency means the increase of the processing speed of multiple rendering processes. So the processing speed is faster than the speed of loading the web resources into the shared resource buffer, and the resources in the shared resource buffer are rapidly reduced, causing multiple rendering processes to wait for the resource to load. From the above experimental results, we draw the following conclusions.

Conclusion 1: At a specific network speed, there is a threshold for CPU frequency. When the CPU frequency exceeds this threshold, the first screen load time and the page load time will not change.

4.2 Relationship Between Load Time and Network Speed at a Fixed Frequency

As shown in Fig. 5, we set the CPU frequency to 40% of the initial frequency, and measure the page load time of Baidu and the first screen load time of Tencent under six network speed conditions in Table 1. It can be seen that when the network speed is lower than 1MB/s, the page load time and the first screen load time change drastically with the network speed. When exceeding this value, the two load times change slowly.

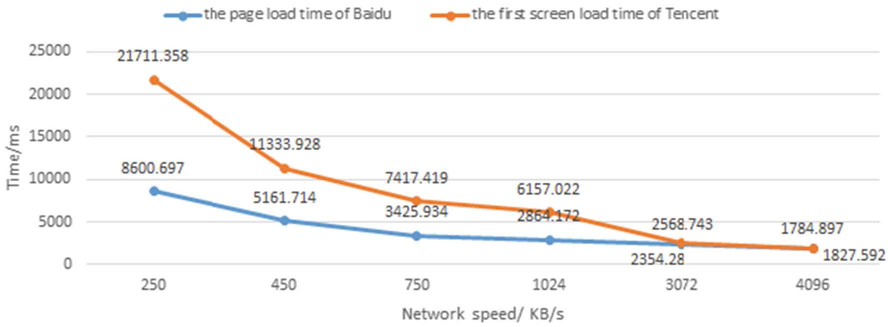


Fig. 5. Load time varies with network speed under 40% CPU frequency

This is due to the lower CPU frequency, which results in a lower processing speed for multiple rendering processes. When the network speed is higher than a value, the web resources loaded by the browser process into the shared buffer are always not processed in time. The shared buffer may overflow, while the browser process and the rendering process are busy. The browser may crash.

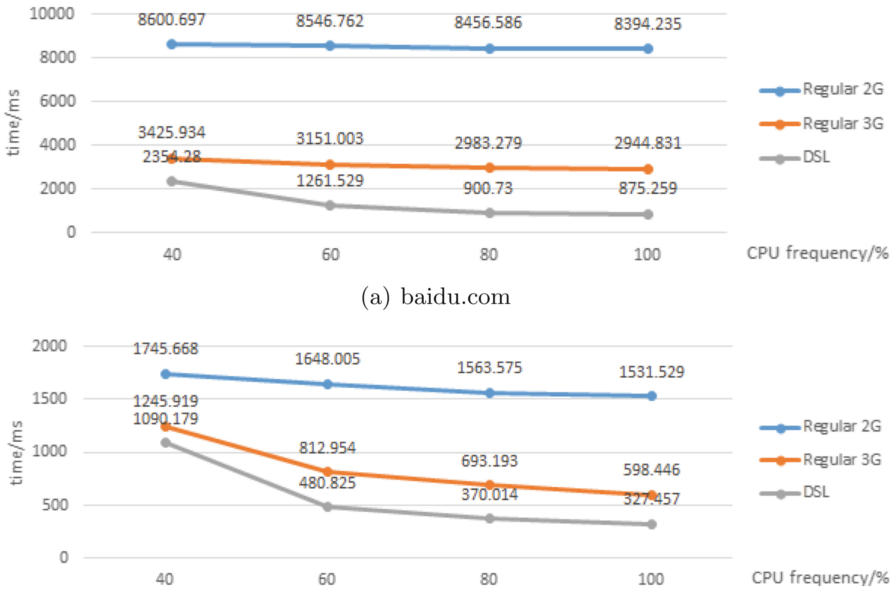
Conclusion 2: When the CPU frequency is low, there is also a threshold for the network speed. When the network speed exceeds this threshold, the page load time and the first screen load time decrease slowly.

4.3 Relationship Between Load Time and Frequency Under Three Network

From Fig. 6 we can see that whether Baidu or Taobao, with the increase of CPU frequency, the first screen load time and the page load time decrease gradually. And the faster the network speed, the greater the change in load time.

According to the browser kernel principle, when the network speed is low, the browser process downloads web resources from the network at a slow speed. So the resources placed in the shared buffer will reduce. Multiple rendering processes preempt the web resources in the buffer, and many rendering processes will be idle. From this we draw a conclusion.

Conclusion 3: At low network speeds, the first screen load time and the page load time are not sensitive to CPU frequency changes.



(a) baidu.com

(b) taobao.com

Fig. 6. Load time varies with CPU frequency under three network speeds

4.4 Network Speed and Frequency Coordinated Modulation Scheme

According to the conclusions 1, 2 and 3, we know that the page load time and the first screen load time are affected by the CPU processing speed and the network speed. The ideal state is that the high network speed corresponds to a high processor speed, and the low network speed corresponds to a low processor speed. That is, the CPU frequency matches the network speed.

According to the experimental results and the principle of browser loading webpage, we propose a browser energy optimization scheme: For a specific webpage, the most suitable frequency for the current network speed and webpage will be find based on the page load time. And this frequency is named as the balance point. The CPU frequency will be adjusted according to the balance point.

Based on this scheme, we design ExploreBP, a simulation tool for finding the balance point. We tested the top 50 sites in Alexa Top Sites Chinese and got the balance point under six network conditions. In theory, all websites will have such a balance point, and this method can be used to test more websites. When the browser loads the web page, the CPU frequency will be adjusted based on the balance point to reduce energy consumption while the browser is working.

5 Conclusion

This paper proposes a scheme for CPU main frequency modulation. For a specific web page, the most suitable frequency for the current network speed will be find based on the first screen load time. According to this method, a simulation tool ExploreBP for finding the optimal matching between network speed and CPU frequency is designed and experiments on the top 50 sites in Alexa Top Sites Chinese have been performed. In order to better quantify the user experience, this paper proposes using the first screen load time as an evaluation metric of the user experience.

The energy optimization scheme proposed in this paper is mainly for browsers, and in the future it can be extended from browsers to all applications that use client/server mode.

References

1. Bui, D.H., Liu, Y., Kim, H., Shin, I., Zhao, F.: Rethinking energy-performance trade-off in mobile web page loading. In: International Conference on Mobile Computing and Networking, pp. 14–26 (2015)
2. Casas, R., Casas, O.: Battery sensing for energy-aware system design. Computer **38**(11), 48–54 (2005)
3. Garsiel, T.: How Browsers Work. http://taligarsiel.com/Projects/howbrowserswork1.htm
4. Google: Blink Rendering Engine. http://www.chromium.org/blink
5. Google: Chromium OS. https://www.chromium.org/chromium-os/
6. Google: Google Chromium. https://www.chromium.org
7. Google: Trace-Viewer. https://github.com/google/trace-viewer
8. Google: Web Page Replay. https://github.com/chromium/web-page-replay
9. Google: Webpagereplaydiagram. https://github.com/chromium/web-page-replay/blob/master/documentation/WebPageReplayDiagram.png
10. Alexa Internet Inc.: Alexa top 50 Sites in China. www.alexa.com/topsites/countries/CN2018--4
11. Kayce Basques, G.: Analyze network performance. https://developers.google.com/web/tools/chrome-devtools/network-performance/
12. Nejati, J., Balasubramanian, A.: An in-depth study of mobile browser performance. In: International Conference on World Wide Web, pp. 1305–1315 (2016)
13. Sakamoto, K.: Time to First Meaningful Paint. https://docs.google.com/document/d/1BR94tJdZLsin5poeet0XoTW60M0SjvOJQttKT-JK8HI/view?hl=zh-cn
14. Tawalbeh, M., Eardley, A., Tawalbeh, L.: Studying the energy consumption in mobile devices. Procedia Comput. Sci. **94**, 183–189 (2016)
15. Wang, Z., Lin, F.X., Zhong, L., Chishtie, M.: Why are web browsers slow on smartphones? In: Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, pp. 91–96. ACM (2011)
16. Willkommen, H.: CPUFreq. https://www.brode.de/cpufreq