ACM Internet Measurement Conference 2021

Rapid Web-Page Loading with Preload Scanner

Jemin Ahn, jiwoong Won, Kyungtae Kang*

Hanyang University, Republic of Korea

Abstract

Despite being continuously advanced, conventional web page loading processes remain inefficient and prevent network devices from fully utilizing their computing capabilities

This study investigates the primary source of this inefficiency, known as script blocking, which occurs owing to browser pipelining. An advanced web page loading process is then proposed, in which the Preload Scanner in Chromium is modified such that it can identify correlations between web objects and pre-requests associated with them.

Experimental testing with example websites demonstrates an improvement in overall web page load times

Inefficiency of web page load

Reason

- Modern web browser use **pipelining** for improving web page load performance. It means **network transmission and script evaluation are executed simultaneously**
- Web browser request is below
  request and load HTML → Parsing HTML → request HTML content sequentially while DOM tree creating

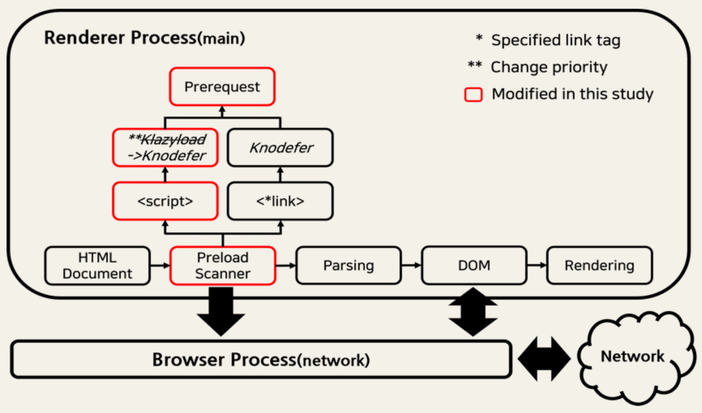
Problem

- HTML content must be executed sequentially to ensure that the result of DOM tree value is consistent. But **HTML contents loading sequence can not be guaranteed**
- **DOM tree generation is stopped** when the prior scripts are not loaded or evaluated

Preload Scanner

- Web browsers have several functions for improving web page load performance, preload scanner is one of them
- Preload scanner make and send pre-request some HTML contents(script, specified link tag content) before DOM tree creation start
- Preload scanner has 3 fetch parameter (*kNoDefer* / *kLazyLoad* / *kIdleLoad*)

Problem on the Process



- **Script** evaluation sequence affects the result of DOM tree, so the script should be executed in order and need to be loaded faster; for minimizing DOM creation blocking situation
- **Specified link tags** can usually be style-related files, contents that HTML writers want to load fast, etc. But these kinds of files do not occur DOM creation block
- Whole script files are preloaded with *kLazy Load* parameter, and specified link tags are preloaded *kNoDefer; **kNoDefer** have higher priority than kLazyload*
- We changed this sequence for **script loading priority to have the same priority of specified link tags** and got improved web page performance

Experiments

Setup

- OS : Ubuntu 18.04 LTS
- Browser : The official chromium source (https://chromium.googlesource.com/chromium/src/+/refs/heads/main/docs/linux/build_instructions.md)
- Target web sites : Top 10 popular web sites from Alexa TOP 50 (https://www.alexa.com/topsites/countries/KR)

How to do

- 50 times request with no cache condition for each site
- **Calculate average time duration between initial HTML document load completion time and completion of DOM tree**

Acknowledgement

This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.2020-0-01343) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (1711117678)

Result and Discussion

Site	The number of Prerequest	Changed Prerequest	Improvement	Average Improvement
F	16	0	-	
H	18	0	-	
A	12	1	7%	
C	22	2	12%	
D	72	1	12%	13.6%
E	13	2	23%	
J	21	4	14%	
B	22	10	21%	
G	34	10	32%	23.5%
I	24	11	16%	
			Total	17.12%

- **Result shows overall web page load performance enhanced except F and H site** (Because F and H have no change)
- Second group and third group was improved 13.6% and 23.5% each, and entirely improved 17.12% of DOM creation time
- E and I demonstrates that the number of changed prerequests is not always proportional to performance improvement
- When considering performance differences in the same group, the total number of prerequests and HTML document environments (type, size, etc.) also affect results.

Conclusion & Future Work

Conclusion

- We focused on inefficiency of the web page load process and modified preload scanner on the chromium browser
- **Average 17.12% of Web page load time was improved with simple modification**

Future Work

- Establish **more accurate and reasonable standards using machine learning classification**
- Performance evaluation with **various conditions** (different bandwidth, more web sites and count of requests etc.)

Contact Information

Jemin Ahn - Email: ahnjemin@hanyang.ac.kr

- Web : cpslab@hanyang.ac.kr

Kyungtae Kang - Email: ktkang@hanyang.ac.kr

- Web : ktkanglab.hanyang.ac.kr