March 09, 2017

Tunable Automatic Placement Finder for Third Party Content

Matthew Burriesci
Thomas Long

Follow this and additional works at: http://www.tdcommons.org/dpubs_series

Recommended Citation
http://www.tdcommons.org/dpubs_series/419

This work is licensed under a Creative Commons Attribution 4.0 License.
This Article is brought to you for free and open access by Technical Disclosure Commons. It has been accepted for inclusion in Defensive Publications Series by an authorized administrator of Technical Disclosure Commons.
TUNABLE AUTOMATIC PLACEMENT FINDER FOR THIRD PARTY CONTENT

In an automatic content slot placement system, third party content slots for displaying third party content can be automatically inserted into web pages in real time at the client side in response to a user request for content. Client side content placement is generally needed when a client device has not visited a page since the last time a cache was cleared. The automatic content slot placement system can provide many benefits, including optimizing the placement of content slots based on various goals of users, publishers and content providers, adapting the placement of the content slots to increase engagement, and making intelligent decisions regarding the placement, size, and format of the content slots. However, allowing the automatic content placement at the client side can sometimes lead to two outcomes. First, a content slot placement is found, but the user has scrolled past the placement. As a result, the user may never view the content item, reflow can occur, and the entire page may seem to shift down for the viewer. Second, the placement finder runs quickly, but never gives up the execution thread on the client device, thus leading to a lag-filled user session.

This paper introduces a technique that solves the above mentioned problems by providing tuning during the automatic content slot placement process. Specifically, the present technique can leverage a combination of a server and a client device to determine an optimal chunk size. A chunk size in the present technique can be defined as the execution time length for executing a chunk of code or instructions. The present technique can obtain a chunk size report from a client device and perform optimization processes to tune how long to hold the main execution thread based on the chunk size report. For example, if the report indicates that the current chunk size is above 200 milliseconds, the present technique can perform certain optimization processes such that the chunk size can be adjusted to below 200 milliseconds. By performing the tuning, the
present technique can minimize client device latency, thereby reducing the time needed for completing other actions. Furthermore, the present technique can minimize the number of times required to restart the entire process (e.g., due to user scrolling), thereby minimizing the number of cycles used to place a content item on the client device. The present technique can also minimize the latency to within a certain band determined to not be noticeable by the end user.

According to the present technique, data can be sent from a client device to a server in order to optimize execution decisions for later third party content requests. For example, “gen204” pings of chunk size can be sent from the client device to the server. The present technique can also leverage device characteristics in optimizing execution decisions. For example, a server can obtain information of a client device, such as device type, CPU clock speed, RAM size, RAM utilization, number of open applications on the client device, and browser type, to determine the expected execution time of a chunk of execution. The present technique can further utilize the visibility of a tab or browser window to determine the optimal chunk size. For example, in the case of a non-visible window or that a phone is running while the screen is off, a larger chunk size can be determined as an optimal chunk size. The present technique can be used to place any kind of third party content items, including advertisements, content recommendations, third party video resources, sign-in badges for social networks, and paid placements for goods, services or content. In addition to third party content, the present technique can be used for placing first party content.

Figure 1 is a block diagram depicting an exemplary environment 1 for implementing the present technique. The environment 1 includes at least one data processing system 2, one or more content providers 7, one or more content publishers 8, one or more client devices 9, and a network 10. The data processing system 2 can include at least one content placement component
3, at least one script provider component 4, at least one tuning analysis component 5, and at least one database 6.

The data processing system 2, the content placement component 3, the script provider component 4, and the tuning analysis component 5 can include one or more processors, servers, computing devices, memory, logic arrays, circuitry, software or hardware modules, logic elements, or digital logic blocks configured to communicate with the database 6 and with other computing devices (e.g., the content provider 7, the content publisher 8, and the client device 9) via the network 10. The memory of the data processing system 2, the content placement component 3, the script provider component 4, and the tuning analysis component 5 can store machine instructions that, when executed by the one or more processors, cause the one or more processors to perform one or more of the operations described herein. The content placement
component 3, the script provider component 4, and the tuning analysis component 5 can be or be within separate devices, or can be or be within one device.

The network 10 can include one or more of any type of computer networks such as the Internet, cellular network, WIFI network, WiMAX network, mesh network, Bluetooth, near field communication, satellite network, or other data network that facilitates communications between the data processing system 2, the content provider 7, the content publisher 8, and the client device 9. The network 10 can also include any number of computing devices (e.g., computer, servers, routers, network switches, etc.) that are configured to receive and/or transmit data within the network 10. The network 10 can further include any number of hardwired and/or wireless connections.

The content provider 7 can refer to, or include, an advertiser or other providers of content items, such as online documents, blogs, media or advertisements. The content provider 7 can establish an advertisement campaign with advertisements and advertisement selection criteria, such as keywords and geographic location. The content publisher 8 can refer to or include a web site operator, such as an entity that operates a web page. The web site operator or content publisher 8 can include at least one web page server that communicates with the network 10 to make the web page available to the client device 9.

The client device 9 can include, for example, mobile computing devices, mobile telecommunications devices, smartphones, personal digital assistants, laptop computers, notebooks, tablet computers, smart watches, or wearable devices. The client device 9 can include a display such as a liquid crystal display, light emitting diode (LED) based display, organic light emitting diode based display, bitmap display, pixel display, electronic ink display, or other display configured to visually output content including text, characters, strings, symbols,
images, or multimedia content provided by a data processing system 2. The content provider 7, the content publisher 8, and the client device 9 can include one or more processors and memory. The memory stores machine instructions that, when executed by the processor, cause the processor to perform one or more of the operations described herein.

The content placement component 3 can receive a request for content from the client device 9. The requested content can be third party content, such as an advertisement. The third party content can also be a sign-in badge for a social network, or a paid placement for goods, services or content. The content placement component 3, responsive to the request, can select a third party content, for example, from the database 6. The content placement component 3 can also make a request to the script provider component 4 for a script. The script provider component 4 can be configured to retrieve the script from the database 6 and provide the retrieved script to the content placement component 3 for transmitting to the client device 9. The script can be transmitted to the client device 9 separately or together with the third party content item.

The script can include computer-executable instructions, such as instructions in JavaScript, HyperText Markup Language (HTML), Extensible HyperText Markup Language (XHTML), Extensible Markup Language (XML), Cascading Style Sheets (CSS), among others. The script can be executed by a processor of the client device 9 to dynamically select a content slot on an information resource (e.g., a web page) at the client device and display the third party content in the selected content slot. The selection of the content slot can include identifying a Document Object Model (DOM) tree of the information resource, identifying DOM elements in the DOM tree, determining multiple placement locations on the information resource, and
selecting a location among the multiple placement locations for displaying the third party content item.

The script executed by the processor of the client device 9 can obtain a chunk size report or log during the automatic content slot placement process at the client device 9. As discussed herein above, a chunk size in the present technique can be defined as the execution time length for executing a chunk of code or instructions. In other words, a chunk size in the present technique can be a chunk of code execution time. The script executed by the processor can check the currently elapsed length of time to determine the chunk size. For example, the script executed by the processor can check each time a set of instructions is completed for the elapsed time since the last time the main thread was given up.

The script executed by the processor can cause the client device 9 to transmit a chunk size report back to the data processing system 2. For example, the chunk size report can be transmitted using “gen204” pings or other asynchronous messages. The data processing system 2 can be configured to determine the optimal execution chunk size. The tuning analysis component 5 of the data processing system 2 can receive the chunk size report and perform tuning analysis. For example, if the chunk size in the chunk size report is above a determined threshold, optimizations can be performed. For instance, if the chunk size in the chunk size report is 120 milliseconds while the determined threshold is 100 milliseconds, parameters for how long to lock the main thread can be modified such that the main execution thread will not be locked for longer than the determined threshold of 100 milliseconds. In some implementations, code for content slot selection and content placement can be split or can be merged into a set of contiguous code based on the reported chunk size and determined threshold. The optimized code can then be used to dynamically select content slots and place content at the client device 9. By
optimizing the code, the main thread used for content placement at the client device 9 can be
given up at a more proper time, thus avoiding the problems described herein above.

The optimization process can also be based on device characteristics of the client device
9. For example, data indicating device characteristics can be transmitted from the client device 9
to the data processing system 2. Device characteristics can include device type, CPU clock
speed, RAM size, RAM utilization, number of open applications at the client device, and
browser type. Network characteristics, such as how long it takes to get information from the
client device 9 to the data processing system 2, can also be used in the optimization process. The
present technique can modify the code for optimization based on the device characteristics and
network characteristics.

By performing tuning, the data processing system 2 can minimize client device latency
and reduce the time needed for completing other actions at the client device 9. The data
processing system 2 can also minimize the number of times required to restart the entire process
and minimize the number of cycles used to place a content item on the client device. The data
processing system 2 can minimize the latency to within a certain band determined to not be
noticeable by the end user. For example, the data processing system 2 can determine that users
may notice latency if the main thread is held up for 50 milliseconds. Accordingly, the data
processing system 2 performs tuning such that the main thread will not be held up for more than
50 milliseconds. The data processing system 2 can utilize the visibility of a tab or browser
window to determine the optimal chunk size. For example, if the data processing system 2
determines that the user is working on something else and the web page for rendering the content
item is running in the background, the data processing system 2 can perform tuning such that the
main thread can be held up longer to get the rendering completed. The technique as described in
this paper can also be utilized for first party content placement. For example, the data processing system 2, the content provider 7, or the content publisher 8 can perform the tuning functions as described herein during first party content placement.

The present technique can solve the problems described herein above and achieve several advantages. By performing tuning in the automatic content slot placement process, the automatic placement function may not be forced to find a placement that may end up above the user’s viewport by the time the content is rendered. Thus, more efficient use of end client device cycles can be achieved. More efficient use of server cycles can also be achieved by not making requests for third party content in the event that the user will be finished with the page before a third party item can be returned. When tuning is allowed, the automatic placement finder has the time it needs to find a placement, without leading to excessive user lag. By not causing lag or reflow, better user experience can result.
Abstract

This document describes a technique for performing tuning in automatic content slot placement process. A data processing system can send a script to a client device for automatic content slot placement. The script can be executed by the client device to obtain a chunk size report and to transmit the chunk size report to the data processing system. The data processing system can analyze data in the chunk size report and perform tuning based on the analysis. Parameters for how long to hold the main execution thread at the client device can be modified. The data processing system can also perform tuning based on device characteristics and network characteristics associated with the client device.