EXTRACTING PRECISE ACTIVITIES OF USERS FROM HTTP LOGS

Kiyotaka Takasuka[1], Kazutaka Maruyama[2], Minoru Terada[3] and Yoshikatsu Tada[4]
[1]*Graduate School of Information Systems, The University of Electro-Communications*
1–5–1 Chofugaoka, Chofu, Tokyo, Japan
[2]*Information Technology Center, The University of Tokyo, 2–11–16 Yayoi, Bunkyo, Tokyo, Japan*
[3]*Department of Information and Communication Engineering, The University of Electro-Communications*
1–5–1 Chofugaoka, Chofu, Tokyo, Japan
[4]*Graduate School of Information Systems, The University of Electro-Communications*
1–5–1 Chofugaoka, Chofu, Tokyo, Japan

Keywords: HTTP request, Browsing history, User profile, Filtering.

Abstract: Browsing histories are often used to build user profiles for browsing supports and personalizations. But, the browsing history also contains HTTP requests generated concomitantly with user activity(concomitant request), which must be removed in order to build correct user profiles. Current filtering methods are based on rather simple characteristics of requests such as the extension of the file name or reported content types. We invent a more efficient filtering method based on other characteristics such as the intervals of requests and the referer relations of requests. In this paper we analyze these characteristics in real web transactions and evaluate their usefulness on filtering.

1 INTRODUCTION

As the number of the Internet users grows, the amount of information on web has explosively swelled. The phenomenon is well known as *information explosion*.

Many researchers have been tackling the problem from various aspects, such as information retrieval, recommendation and extraction. We focus on the data preprocessing from the aspects of browsing supports and personalizations. The browsing history is often used to build the user profile at these researches. The following can be used as the source of the browsing history:

1. the browsing history itself,
2. the access log recorded by the web server,
3. the access log recorded by the proxy server, and
4. the access log recorded by the network sniffer.

The source #2, #3, and #4 above collect many extra HTTP requests in contrast to #1. The browser records only HTTP requests which explicitly occurs by the user. We call the requests *base requests*. On the other hand, the access log recorded by the web server, proxy server or network sniffer includes extra requests which implicitly are occurred by the browser after loading the obviously requested web page, such as images embedded in the web page, icons, css files and javascript files. We call the requests *concomitant requests*. In the point of view of building the user profile, the browser's history is suitable because it includes only explicit users' activities. If users provided their own history in some way, such as by the web browser extension, precise activities could be collected.

The access log is also available for this purpose with the elimination of the concomitant requests. But it is difficult because the number of concomitant requests is huge against the number of the base requests. This problem is specially difficult when the access log recorded by the proxy server or the network sniffer is used.

It is necessary to remove concomitant requests when these access logs are used as the data source. Traditional methods are to restrict the extension of the file name in URLs and the content-type included in the HTTP response. However, these methods are not enough when requests to advertisements or XML files are included in the access log. Many requests to advertisements don't have the extension of the file name in URLs, their content-types of the response are the same as base requests. These remaining concomitant requests affect the user profile. Therefore, the

Takasuka K., Maruyama K., Terada M. and Tada Y.
EXTRACTING PRECISE ACTIVITIES OF USERS FROM HTTP LOGS.
DOI: 10.5220/0001840403410346
In *Proceedings of the Fifth International Conference on Web Information Systems and Technologies (WEBIST 2009)*, page
ISBN: 978-989-8111-81-4

method of removing concomitant requests and identifying base requests from the access log recorded by the web server, proxy server or network sniffer is required.

2 PURPOSE

Our goal is to get rid of concomitant requests from the access log recorded by the web server, proxy server or network sniffer, to extract base requests from all the log. In this paper, we describe

1. the classification of requests,
2. the proposal of a filtering method using a timing at which requests were generated, and
3. th implementation and evaluation of the proposed filtering method.

3 RELATED WORK

Many researches use the access log to build the user profile for the browsing support. The log wrote down by the web or proxy server is used in many case.

In the case of the proxy server(J.Wang, Z.Chen, L.Tao, W.-Y.Ma, and L.Wenyin, 2002), a content-type included in the HTTP response header is available for the filtering. The header identifies whether a response is in text format or not. However,the removal by this method is not enough. The access log recorded by the proxy includes concomitant requests to advertisements and frames. These remaining requests affect the user profile.

To make a user profile, the access log recorded by the web server is used(Yunjuan Xie, Vir V. Phoha, 2001). In this case, the access log is cleaned in the preprocessing to remove noises. The filter removes concomitant requests by checking the suffixes of URLs using the black list of the extension. A request is removed as the concomitant request if the request have the extension which is included in the list. But this method doesn't remove correctly too.

The access log recorded by the web server is targeted in the data mining. (Yong Zhen Guo, Kotagiri Ramamohanarao and Laurence A. F. Park, 2007) (Ramakrishnan Srikant, Yinghui Yang, 2001). The filter removes requests to images, extracts requests using the white list of the extension. A request is treated as the base request when the list includes the extension of its file name. After all, these cases are not enough too.

4 THE ANALYSIS OF HTTP LOG

We analyzed the access log recorded by the proxy server. In this access log, access to 7 sites which we had selected were recorded. The selected 7 sites are very popular and standard on web. We got several knowledges through an analysis of access to these 7 sites.

4.1 Classification of the Requests

We classify the request as follows.

1. Base request
2. Concomitant request
3. Static request
4. Periodical request
5. Interaction-induced request

A web page generally consists of one base page and a lot of concomitant objects. A request to the base page is a base request. The base request is the one to the URL included the anchor which the user just clicked. A request to the concomitant object is a concomitant request. The concomitant object is the one except the base page gotten to display the web page on browser. An example of the base page and concomitant objects is shown in figure 1.

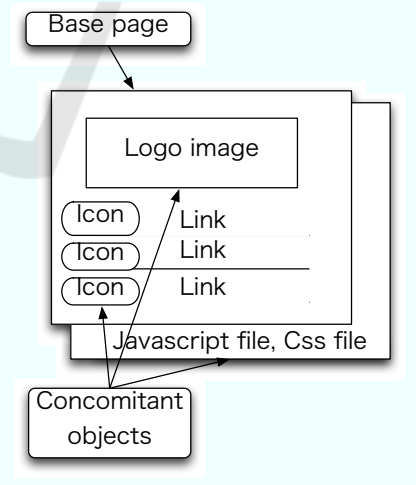

Figure 1: An example of a base page and concomitant objects.

Time relation between the base request and its concomitant requests is shown in figure 2. A first base request occurs by a click which the user performs actively. There is latency from the base request to the

arrival of the response, concomitant requests are generated after the response arrives, the web page is rendered on browser. The user clicks a link he wants to browse. Then a next base request is generated. If the user opens several tabs at the same time under tab browsing, it is thought that base requests are generated at the same time.

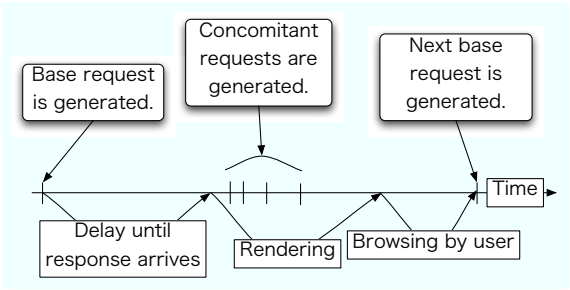

Figure 2: An example of a time relation between a base request and its concomitant requests.

We had viewed the top page of the 7 sites for 5 minutes. The number of concomitant requests generated every access are shown in table 1. Only one base request causes many concomitant requests

We classified concomitant requests to three classes based on characteristics. We describe the details of each class in the following.

4.1.1 Static Request

Many concomitant requests occur generally after a base request. Many requests classified as this class are necessary to display base page. For example, many images displayed on the web page, the css file which have the structure of the web page and the javascript file which have the action of the web page.

The static request have follow three characteristics.

1. Generated in a short period of time from base request
2. The number of requests is inherent in the web page, same for every access unless the web page is updated.
3. Many requests can be traced to the base request using the referer in the HTTP request header.

All static requests were generated at 0.818 [sec] in the Yahoo's top page. All of 53 concomitant requests were classified as the static request. 52 of these requests had the referer to the base page. One concomitant request, which was a request to the favorite icon image, didn't have a referer.

4.1.2 Periodical Request

The request on this class is automatically generated by Ajax and Adobe Flash to refresh some part of the web page periodically. The periodical request have follow two characteristics.

1. Regularly generating,
2. The number of generating depends on the display period

The purpose generating the periodical request is to renew regularly the specific images and strings of characters. Therefore regularly generating is the largest characteristic, many requests on this class get images and XML files.

4.1.3 Interaction-induced Request

The user's interaction makes Ajax and Adobe Flash generate the concomitant request on this class. Characteristics which the interaction-induced request have are the following.

1. Caused by the user interaction.
2. The number of generating depends on the amount of the user interaction.

The interaction-induced request occurs when an image changes by the user interaction. Great many interaction-induced requests were generated as a result of the interaction continuously in Google Maps.

5 PROPOSED METHOD

We propose a method removing the static request in addition to the traditional method using the extension and content-type.

The proposed method have two features:

1. Using the time interval from the base request to the concomitant request,
2. Reconstructing the base-concomitant relation based on the referer to tackle the browsing in parallel using multi tabs.

While the base request is caused by user's action, the concomitant request is generated by the browser. The user's action is slower than the browser's one. Therefore the time interval from the base request to each concomitant request is especially short. This interval is shorter than the time interval between base requests. The proposed method uses this difference of the time interval from the base request. Our proposal assesses a request of which the period is shorter than a threshold as the concomitant request.

Table 1: The number of concomitant requests when top pages of 7 sites are opened. A "request" is omitted in the table. For example, a "Concomitant" means the concomitant request.

	Yahoo	Google	Google image	Google map	Nicovideo[1]	Youtube	Wikipedia
Concomitant	53	6	1	1653	88	95	26
Static	53	6	1	14	86	46	26
Periodical	0	0	0	0	2	49	0
Interaction-induced	0	0	0	1639	0	0	0

If the user opens multiple new tabs at the same time, then more than one base requests occur simultaneously, and concomitant requests generated from different base requests are mixed in the access log. In order to identify the concomitant requests, we need to find out the trees of the derivations of the reqeusts; each concomitant request was yielded from one of the base requests. The referer header in the request helps us to do it, and the feature 1 above can be applied to the case of the tab browsing.

6 PERFORMANCE MEASURES

We introduce the performance measures to evaluate our proposal. We use a precision and recall used in information retrieval. These values are defined as follows:

$$Precision = \frac{|\,Extracted \cap Browser\,|}{|\,Extracted\,|} \quad (1)$$

$$Recall = \frac{|\,Extracted \cap Browser\,|}{|\,Browser\,|} \quad (2)$$

$$F\text{-}measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \quad (3)$$

where *Extracted* is a result set of the filtering, *Browser* is a set of base reqeusts, *Precision* is the accuracy of the extraction, *Recall* is the ratio between the base requests which the extracted set includes and the set of all base requests. *F-measure* is the value integrated the precision and recall.

7 EXPERIMENTS

We evaluate the accuracy of the proposed method by applying it to the access log recorded by the proxy server in the single tab browsing. There are two experimental data sets, the access log of restricted browsing and comparatively unrestricted browsing. At first we investigate the upper performance limit of our proposal using the access log of the restricted browsing in section 7.1, the practical performance of the proposed method using the access log of the unrestricted browsing in section 7.3.

7.1 Restricted Experiment

We apply the proposed method to the access log of the restricted browsing in order to make our proposal the most effective. Finding out the upper performance limit facilitates the evaluation of the performance in the section 7.3. A comparative method restricts the content-type to "text/html" only.

7.1.1 Experimental Data Set 1

We accessed to the web page as the effects on the user and network performance are as invariable as possible. All of the access to the web page were generated from the machines with the same settings and place. The machines for web browsing and the proxy server exist at a LAN.

We browsed web pages selected at random in a site, had 5 minutes interval between base requests. The number of web pages we browsed in each site was about 10 pages. 5 minutes interval is the necessary time to generate all of static requests. In this experiment, the web browsing was performed on single tab to make the base-concomitant relation obvious. Target sites of the browsing were restricted in 6 sites, the number of browsed web pages was 60 pages. We use Firefox2.0 as the browser and Squid2.6[2] as the proxy server.

The number of base requests was 60 because the number of browsed web pages was 60. The number of concomitant requests was 1567. We call an access log used in this experiment an experiment log. The characteristics of 6 sites selected as the target of this experiment are shown in table 2.

7.1.2 Result of the Restricted Experiment

Performance measures of the proposed method are shown in table 3. We set a threshold in order to remove the concomitant request as 5 [sec].

[1]This site is a popular video sharing service in Japan. http://www.nicovideo.jp/

[2]http://www.squid-cache.org/

Table 2: Websites selected in the target of the restricted experiment.

Site	Classification	Adobe Flash	Periodical requests	# of images
Yahoo	Search engine	used	none	many
Google	Search engine	none	none	few
Google Image	Image Search	none	none	medium
Nicovideo	Video hosting website	used	used	many
Youtube	Video hosting website	used	used	many
Wikipedia	Encyclopedia on web	none	none	few

Table 3: The evaluation of filtering performance using experiment log. A "method" is omitted in the table. For example, A "Traditional" means the traditional method. A "request" is omitted too.

	Result of filter	# of base	# of concomitant	Precision[%]	Recall[%]	F-measure
Initial condition	1627	60	1567	3.8	100	6.0
Traditional	133	60	73	45.1	100	62.2
Proposed	236	60	176	25.4	100	40.5
Combined	84	60	24	71.4	100	83.3

The precision of the proposed method is 25.4 [%], this value is less than the one of the traditional method. But, performance measures are improved by combining the proposed method with the traditional method. This indicates that our proposal is able to remove concomitant requests which the traditional method can't remove. The proposed method and traditional method respectively work as making up for deficiencies in each.

We analyzed actually a result of the filter. Many concomitant requests which the filter can't remove were the periodical request and interaction-induced request. The combined method can remove the static request which the proposed method targeted.

7.2 Practical Experiment

We evaluate the practical performance measures of the proposed method and traditional one. A subject browsed less restricted than the restricted experiment.

7.2.1 Experimental Data Set 2

We selected a male graduate student as the subject. He was in his early 20s, belonged to the information-related department. The subject performed the web browsing in the same environment as the restricted experiment. He browsed under follow two restrictions for about 30 minutes.

1. The single tab browsing
2. Browsing mainly on each websites used in the restricted experiment

The number of browsed web pages was 113 pages. So base requests were 113. On the other hand, concomitant requests were 3704. We didn't restrict strongly websites he browsed so that we aimed to evaluate more practically than the restricted experiment. Therefore 40 out of 113 base requests differ from websites used in Section 7.1, In addition, the set of base requests contained a few of requests to the same URL, the set had respectively two URLs browsed two times and three times.

7.3 Result of the Practical Experiment

We evaluated the traditional method and combined method changing a threshold every 1 [sec] from 1 [sec] to 5 [sec]. The result is shown in table 4.

The precision is improved by adding the proposed method to traditional method compared with the traditional method only. We focus on the case in which threshold is 1 [sec], because the recall doesn't decrease untill this threshold. In this threshold, the precision is improved about 25 [%] compared with the traditional method only.

The precision is less than the one of the restricted experiment. To browse in more practically environment, intervals between base requests were short. Actually, in the usual browsing many user shift to next page immediately. So, a threshold needs to be less than the restricted experiment because the recall is decreased in the same threshold as the restricted experiment by the incorrect removal of base requests.

Table 4: The evaluation of the filtering in more practically browsing. A “request” and “method” are omitted in the table.

	Result of filter	# of base	# of concomitant	Precision[%]	Recall[%]	F-measure
Initial condition	3817	113	3704	3.0	100	5.8
Traditional	283	113	170	39.9	100	57.0
Combined (1[sec])	175	113	62	64.6	100	78.5
Combined (2[sec])	168	111	57	66.1	98.2	79.0
Combined (3[sec])	108	57	51	52.8	50.4	51.6
Combined (4[sec])	105	54	51	51.4	47.8	49.5
Combined (5[sec])	85	42	43	49.4	37.2	42.4

8 CONCLUSIONS

In this paper, we classified each request recorded in the access log. We assumed that we would extract only the requests to the URLs included the anchors which the user just clicked from the access log. In addition, we proposed the filter to remove especially the static request and evaluated the filter. We performed the evaluation using experiment log and more practical log to analyze how traditional method and proposed method had worked.

As a result, the proposed method can improve the performance to remove the concomitant request by adding to traditional method. We conclude that proposed method and traditional method work as making up for deficiencies in each.

9 FUTURE WORKS

We hope to explore to direction as follow:

Evaluating in Multi Tab Browsing. HTTP logs used in this paper were recorded by the single tab browsing. But, in recent web browsing users browse usually by multi tab. So, it is necessary to evaluate in the multi tab browsing.

Removing the other Two Kind of Request. The proposed method targets at the static request. It is impossible to remove the periodical request and interaction-induced request by the proposed method. Therefore a method to remove these requests is needed.

Extending Target Websites. Experiments in this paper targeted at specific websites. This make the comparison between performances easy. In the future work we hope to extend the target website to the other and make a filter independent of websites.

REFERENCES

J.Wang, Z.Chen, L.Tao, W.-Y.Ma, and L.Wenyin (2002). Ranking user’s relevance to a topic through link analysis on web logs. In *Proceedings of the 4th international Workshop on Web information and data management*, pages 49–54.

Ramakrishnan Srikant, Yinghui Yang (2001). Mining web logs to improve website organization. In *Proceedings of the 10th International conference on World Wide Web*, pages 430–437.

Yong Zhen Guo, Kotagiri Ramamohanarao and Laurence A. F. Park (2007). Personalized pagerank for web page prediction based on access time-length and frequency. In *Proceedings of the 2007 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 687–690.

Yunjuan Xie, Vir V. Phoha (2001). Web user clustering from access log using belief function. In *Proceedings of the 1st international conference on Knowledge capture*, pages 202–208.