Dynamic Web Cache Management

Benjamin Yen
The University of Hong Kong, benyen@business.hku.hk

Culture Wang
The University of Hong Kong, culturewang.hkuccc@gmail.com

Dynamic Web Cache Management

Benjamin Yen, The University of Hong Kong, Hong Kong, benyen@business.hku.hk
Culture Wang, The University of Hong Kong, Hong Kong, culturewang.hkuccc@gmail.com

ABSTRACT

Web navigation has been the key issue for information retrieval in e-commerce. Information caching is critical for navigation subject to resource constraints and performance requirement. The research on caching originates from data access to computer memory, to database (e.g. multimedia database), to client/server architecture, and recently to Web navigation. The information access for caching normally is assumed the fixed size of data unit. In this research, we first generalize caching problem for Web navigation by considering information structures. The caching criteria also takes into account Web structure, data usage, and navigation patterns. The preliminary result shows the proposed dynamic caching approach, New Semantics-Based Algorithm (NSA), outperforms the common caching functions and can be applied to broader application domains. Some implications and future directions are discussed in the conclusion.

Keywords: Web Cache, Web Navigation, Website Design.

INTRODUCTION

Cache is normally deemed to be fully available in Web navigation. When the users browser the page visited before, the page is assumed available without the needs to download again. With the growing size of websites and number of visitors, resource constraints and performance requirement are taken into account for Web management. Web cache is essentially crucial for Web navigation. The research on Web navigation can be classified into several areas – principles of web usability [1-3], web personalization and recommendation to enhance the web usability [4-7], techniques to identify usage patterns [8-11], and studies on effectiveness and efficiency of information retrieval [12-15]. The followings are the summary of literature review on cache management.

(1) Most commonly used traditional algorithms. In order to reduce data traffic and loading time by implementing web proxy cache, various cache replacement algorithms have been proposed. Least Frequently Used (LFU) Algorithm and Least Recently Used (LRU) algorithms, which focus on frequency and recency respectively, are two most well-known examples. Some algorithms combine these two to improve the cache hit rate, e.g. Least Recently Frequently used algorithm [16] and Segmented Least Recently Used algorithm [17].

(2) Size based caching replacement algorithms. For web proxy cache, the size of the objects and pages is an important factor for the performance of caching algorithms. SIZE algorithm focuses on this issue and evict the largest page. There are other algorithms considered size and other factors. Least Recently Used – MIN (LRU-MIN) algorithm evicts page consider both recency and size, and size adjusted popularity aware Least Recently Used algorithm [17] takes into account size, recency and frequency. Another extension of SIZE algorithm is Greedy-Dual-Size algorithm, without considering frequency, proved better than most traditional algorithms. Its extension Greedy-Dual-Size-Frequency algorithm [19] also includes frequency factor.

(3) Loading time based caching replacement algorithm. Another important benefit of cache is to reduce loading time for internet user. Latency estimated algorithm [20] aims to estimate time for server connection and data transmission and to minimize loading time for web users. Hybrid algorithm [23], based on latency estimation, considers aging factor counting time interval since last access. MIX [21] algorithm takes into account size, frequency, recency and loading time, and sets weights to each factor.

(4) Caching replacement algorithms to predict future behavior. Instead of focusing on page eviction based on past accesses, some research projects estimate which cache object will be visited again in the nearest future. Taylor Series Prediction algorithm [22] predicts the time for a page to be visited in the nearest future by time interval between each access. Some approaches adopt computational intelligence, e.g. machine learning [23], to predict the page will be frequently accessed.

(5) Structural aware caching replacement algorithm. Properties can be associated with Web pages to reflect the corresponding importance level. Least-Frequently-Internal-Request algorithm [24] assumes that the more internal requests (e.g. video, image, download file and links) to other pages, the more likely it will be visited. Some algorithms [25] take advantage of hotlist pages by keeping them in cache to optimize performance. Most of the algorithms mentioned above do not consider the relationship among the objects in the decision of page eviction. In reality, users surf the Internet by successively clicking through links on the webpages. The relationship among cache objects also influence performance matrix for cache algorithms. The Semantics-Aware Replacement Algorithm [26] finds popular pages from mining tools to and decides the page eviction based on the distance to these pages and visit frequency of the pages in cache.

DYNAMIC CACHING APPROACH

The effectiveness of Web navigation is subject to the availability of cache. It is normally assumed that the cache is fully available that there is no need to reload the webpages already visited. In other words, the case represent the situation that cache size is extremely large. The other example is there is no cache at all that all pages need to be reloaded once they are visited no matter how many times they have been visited. Both cases above are the extreme cases that are relatively less sophisticated than the case of limited cache that involves decision of eviction in cache.

Web cache problem is different from traditional computer cache issue from various aspects, such as data size, information structure, and problem objectives.

(1) Data size. The size of webpages vary - information pages (e.g. pdf files; normally leave pages), portal pages (i.e. web pages with only hyperlinks), and hybrid pages.

(2) Information structure. The structure is based on the hyperlinks among pages; it can be tree or graph in general.

(3) Problem objectives. In addition to the traditional criteria (i.e. hit rate and byte hit rate), the objectives may include information volume, advertisement, etc.)

The proposed dynamic caching is basically based on two commonly considered factors, i.e. frequency and recency, as for other caching methods as well. However, the concept "recency" is changed to the expectancy. We uses "popularity" to represent the frequency of page visit in the past and "accessibility" to reflect the potential recency for both the page is accessed in the past and to access in the future. In addition to recency and frequency, the Web structure is another critical factor to be considered.
The fundamental concept is to prioritize the pages for caching on the "distance" to the currently visited page (or "current page" in short).

(1) Past Pages. For the pages visited previously (i.e. the pages on the path from the root to the current page), the pages closer to the current page (i.e. more recently accessed page) the higher the priority. This is to reflect the easiness to "go back".

(2) Future Pages. For the pages to be possibly visited later (i.e. the pages that the current page possibly connects to), the pages closer to the current page (i.e. more accessible pages) the higher the priority. This is to reflect the accessibility in the navigation.

(3) In the early stage of navigation, it is more likely moving forward to explore than going back. However, once the navigation goes deep in the website, there is higher tendency going back than moving forward.

Take into account popularity (i.e. frequency), it is also important for priority setting. Intuitively, the more popular pages are more possibly visited. One of the structure related properties is the connectiveness. Ibn the following, "connected" means that there is a path between the pages and all the pages on the path are cached already.

(1) For those cached page, they are given the priority if they are connected from the current page, more closely connected, higher priority.

(2) For those cached page, they are given the priority if they connect to the popular pages cached, more closely connected, higher priority.

DESIGN OF EXPERIMENT

Overview of Caching Algorithms

When the user visits a web page, the cache algorithm will first search whether the page is in the cache. If yes, a cache hit occurs and the page is accessed from cache; otherwise, the algorithm will check whether there is enough space to cache the requested page. If no, the algorithm will follow the criterion to choose the page(s) to evict. The five algorithms, including the proposed dynamic caching approach, used for the experiment are discussed as follows.

(1) Least Recently Used Algorithm (LRU). The algorithm evicts pages which has not been visited for the longest time.

(2) Least Frequently Used Algorithm (LFU). The algorithm chooses the page that least frequently referenced in cache to evict.

(3) Size Adjusted and Popularity Aware Least Recently Used Algorithm (LRU-SP) [17]. The cache memory is partitioned into multiple sections based on the value of (size / number of times the page has been referenced) defined as ratio T. The pages in each section are queued based on the LRU algorithm. When the user visits the web page, the LRU-SP algorithm checks

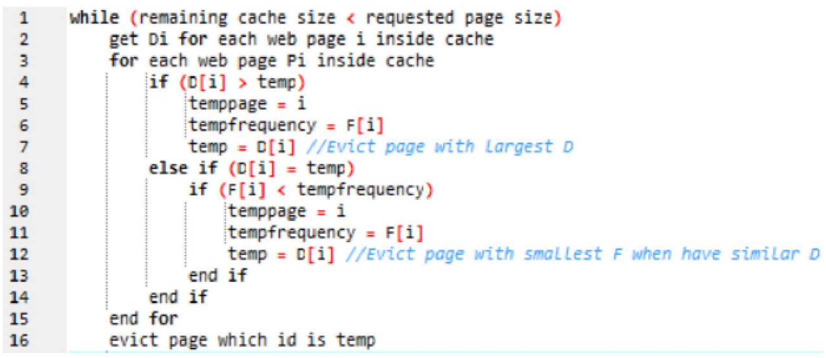
whether the page is inside cache memory or not. If yes, firstly increase the number of times it has been visited by 1. The page will be shifted to different section based on the updated ratio T until it is evicted from the section that has the largest ratio T in LRU order.

(4) Semantics-Aware Replacement Algorithm (SACS) [26]. The algorithm considers the structure of the web site and a list of frequently visited pages. It assumes if a page is closer to those popular pages, it is more likely to be visited in the future. It first compare how far the cached pages to those popular pages in depth value (i.e. the shortest distance in between). The page with larger depth value will be evicted first. For the same depth value, the page are less frequently accessed is evicted first.

(5) Dynamic Caching Approach - New Semantics-Based Algorithm (NSA). The algorithm considers the structure of the website and assume if the cached page is closer to the currently visited page, it will be more likely to be visited later. The pages with the same distance from the currently visited page, the less frequently visited page will be evicted first.

Dynamic Caching Approach - New Semantics-Based Algorithm (NSA)
The major difference between SACS approach and the proposed NSA approach is SACS focuses on the distance between the cached pages to "popular" pages and NSA also considers the distance between the currently visited page and the cached pages. The algorithm keeps track of access frequency and size of each page cached, the size of remaining cache available, hit rate and byte hit rate, etc.

Upon receiving a new request for page access, beside book-keeping update, the algorithm first checks whether the page is in the cache. If yes, a cache hit occurs; otherwise, the algorithm will check there is enough space to cache the page. If no, the algorithm will follow the criterion to choose the page(s) to evict. Since user navigate the website page by page following the hyperlinks structure, the pages cached closer to the current page visited are more likely accessed later. The algorithm will evict the page with longest distance from the current page. If there are more than one candidate, then evict the one with lowest frequency. The eviction process continues till there is enough memory for the newly accessed page. The algorithm is shown in Figure 1.



```
while (remaining cache size < requested page size)
    get Di for each web page i inside cache
    for each web page Pi inside cache
        if (D[i] > temp)
            temppage = i
            tempfrequency = F[i]
            temp = D[i] //Evict page with Largest D
        else if (D[i] = temp)
            if (F[i] < tempfrequency)
                temppage = i
                tempfrequency = F[i]
                temp = D[i] //Evict page with smallest F when have similar D
            end if
        end if
    end for
    evict page which id is temp
```

Figure 1: The eviction algorithm

System Design
An offline simulation platform is developed in C++ and DEV-C++ version 5.11 to simulate the how the five algorithms mentioned earlier work. Algorithm selection and cache size setting are flexible. Access log file and website structure serve as the main input information to simulate cache system, eviction process, and performance evaluation. The structure of simulation system is shown in Figure 2.

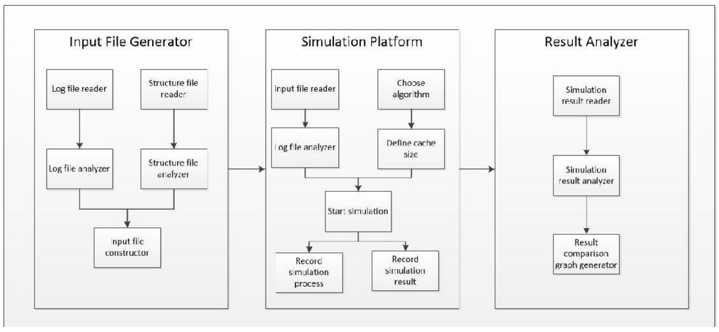


Figure 2: Structure of Simulation System

The performance metrics concerned in the simulation are *hit rate* and *byte hit rate*. Hit rate represents the percentage of the total requests are handled by cache; byte hit rate measures the number of bytes that transferred from cache as a percentage of total number of bytes of all the requests. These metrics are widely used in various cache replacement algorithms for the comparison of performance. Invalid user access or inaccessible objects, which would have resulted in cache misses, are actually not taken into account for hit ratio or byte hit ratio calculation, which will not affect the accuracy and the reliability of the simulation result. Please refer to Appendix for the detail.

Experiment Data

The input data for the simulation contains two sections: the website structure file which displays the structure of the source website and the access log file which contains user access history. The source website for simulation is from an education institute in Hong Kong (http://www.fbe.hku.hk), which contains 29,645 web objects and approximately 900,000 links among the nodes. Besides the structure information, the structure file also covers the size of each nodes as well as their loading time.

The access log file includes all user access logs from March 20th to March 27th, 2016. The log file contains 51,995 user accesses made over that period of time and each entry has user information including IP address of user, time when user made the request, users' target URL and other information as well. To improve the effectiveness and efficiency of simulation, the system generates a standardized input with all the essential information for simulation, including user IP-address, the time of the request, object id which user requests for, the loading time of the requested object and its size. Please refer to Figure 3 in Appendix about simulation procedure.

RESULT AND DISCUSSION

In this section, the simulation result comparison from the proposed dynamic caching approach (NSA) and other caching replacement algorithms are compared on hit rate (HR) and byte hit rate (BTR). The control group includes least frequently used algorithm (LFU), least recently used algorithm (LRU), the size-adjusted and popularity-aware least recently use algorithm (LRU-SP) [17], and semantics-aware replacement algorithm (SACS) [26].

Comparison with LRU and LFU

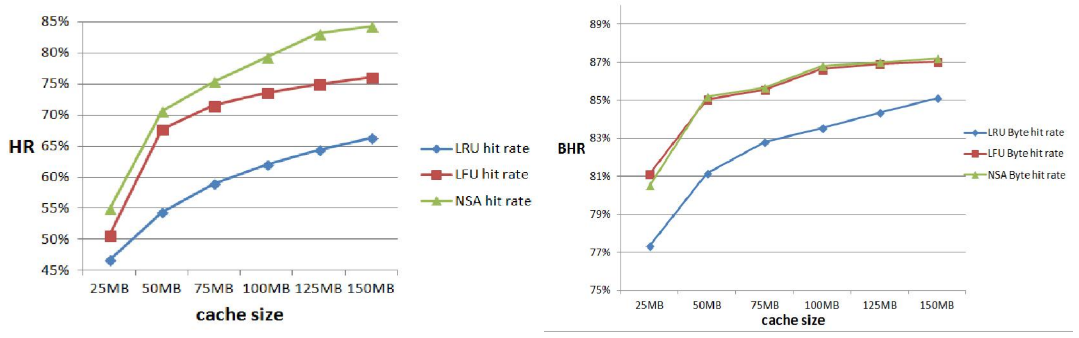


Figure 3: Comparison between proposed approach and LRU/LFU

Figure 3 shows the LRU algorithm performs not as well as others. The possible explanation is that the popular pages lead to the higher importance for frequency than recency. Though the proposed NAS algorithm has a better performance in BR but does not show a significant improvement for BHR, which is possibly due to few large size files with high frequency.

Comparison with LRU-SP

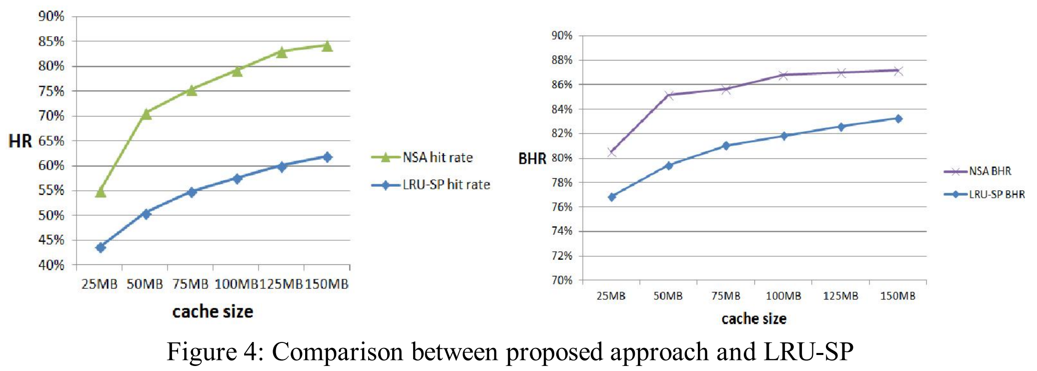


Figure 4: Comparison between proposed approach and LRU-SP

Though LRU-SP algorithm take into account both frequency and recency, it does not consider the relationship among pages. The proposed NSA algorithm outperform in both BR and HBR as shown in Figure 4.

Comparison with SACS

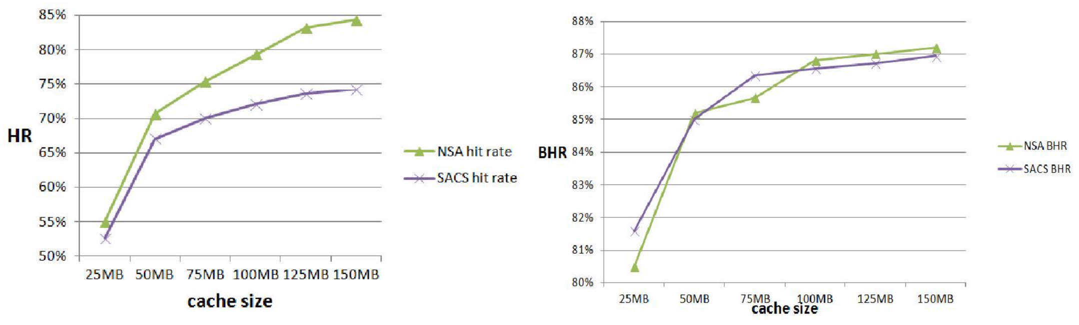


Figure 5: Comparison between proposed approach and SACS

In comparison with SACS, NSA perform better for HR and the gap grows with the increase in cache size. However, there is no significant difference for BHR as shown in Figure 5.

Result Summary

From the comparison between the proposed approach and other methods, it is found that, in general, it perform well for HR, but not particularly better than other for BHR. There are two possible reasons behind: (1) the proposed NSA approach is only the preliminary model of the overall approach described in the section "DYNAMIC CACHING APPROACH". The further development will be carried for better result; and (2) the website (see Figure 6) used for the simulation has a simple navigation map which connects to most of the entry pages and reduces the impact of depth value.

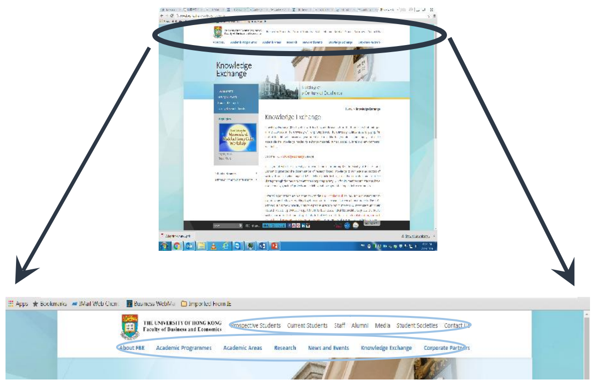


Figure 6: Navigation map in webpages of experiment

CONCLUSION

The study proposes a Web caching approach by taking into account the site structure which is essential for Web navigation. The proposed dynamic caching approach is based on both frequency and recency incorporating with site structure. The experiment result shows the proposed approach basically outperform the other common approaches, i.e. least frequently used algorithm (LFU), least recently used algorithm (LRU), size-adjusted and popularity-aware least recently use algorithm (LRU-SP) and semantics-aware replacement algorithm (SACS), on both hit rate (HR) and byte hit rate (BTR). Besides the result discussed in the previous section, here are some findings.

(1) Sensitivity. The simulation result largely depends on three types of sensitivities – structure sensitivity, data sensitivity, and time sensitivity. Structure sensitivity refers to the website design and organization. As shown in Figure 6, the page design indeed will impact on navigation. The structure issue here is correlated to usability study. Data sensitivity refers to the navigation patterns to reflect how users access the information on web sites. Among all the web pages, only one tenth of them were accessed. In addition, if the page accesses are further classified to portal pages (i.e. mostly links to other pages) and content pages (e.g. the documents and images), the data access might reveal different access patterns. Time sensitivity reveal the criticality of time window of test data. The page access is not uniformly distributed on time line. Some accesses follow the patterns in cycles of various lengths which might not be explicit in the simulation and experiment.

(2) Cache size. The cache size is another decision factor for cache performance. There will be many internal (e.g. structure) and external (e.g. usage) factors to decide suitable cache size. The size should be dynamically adjusted based on needs. How to decide and adjust the cache size will be interesting extension.

(3) Eviction Criteria. The eviction criteria discussed in the simulation is only the preliminary model. The complete model

takes into account both access frequency in the past and potential access in the future based on the structure to reflect the frequency (or popularity) issue. At the same time, both the slack time for last access and lead time for potential future access are considered to decide the weights to reflect the recency (or accessibility) issues. The extended model will be tested for data of longer time window.

There are several limitations and possible future directions as follows.

(1) The experiment data used in this study is from an educational institute within the time window of one week. The number of user access in that period of week is more than fifty thousand. However, it might not be adequate to show the general situation and access patterns. In addition, though the structure size of the website is not small (i.e. around thirty thousand pages), all pages have a navigation map at the header as shown in Figure 7. The navigation map direct the user to the different categories of pages. This might not be common for other websites in general. The larger size of data set should be considered for experiment in the future. In addition, the data mining and business analytics methods can be applied to explore the possible access patterns. Furthermore, the proposed approach should be applied to various types of websites to validate the generality.

(2) The methods in the experiment for comparison were chosen as the typical examples to benchmark the performance of the proposed approach. There are indeed many other approaches, such as computational intelligence based method, for Web caching problems. The algorithm in the experiment is the preliminary model of the proposed approach. The next step is to fully implement the proposed approach and to include more representative approaches in the experiment for comparison.

(3) The proposed method can also be extended to include the size for the decision of caching and eviction. The model also can be tailored for individual users or servers in conjunction with the global caching strategy. In addition, the proposed approach can be the applied to the navigation application, such as navigation guidance.

ACKNOWLEDGEMENT

The research was partially supported by GRF grant (719409E) from the Research Grants Council, HKSAR and HKU-Fudan Joint Research Fund (JRF) 2016. We would like to thank Mr. Chris Cheung for his help with the case study of real world website.

REFERENCES

[1] Nielsen, J. Alertbox, http://www.nngroup.com/articles/, 2016.

[2] Schaik P. and Ling, J. An Experimental Analysis of Experiential and Cognitive Variables in Web Navigation. *Human–Computer Interaction*, 27(3): 199-234, 2012.

[3] Spiliotopoulos, T., Papadopoulou, P., Martakos D. and Kouroupetroglou, G. *Integrating Usability Engineering for Designing the Web Experience: Methodologies and Principles*, IGI Global, 2010.

[4] Eirinaki, M., Louta, M.D. and Varlamis, I. A Trust-Aware System for Personalized User Recommendations in Social Networks. *IEEE Transactions on System, Man, and Cybernetics: System*, 44 (4): 409-421, 2014.

[5] Ho, S.Y. and Bodoff, D. The Effects of Web Personalization on User Attitude and Behavior: An Integration of the Elaboration Likelihood Model and Consumer Search Theory, *MIS Quarterly*, 38(2), pp. 497-520, 2014.

[6] Li, S. and Karahanna, E. Peer-based Recommendations in Online B2C e-Commerce: Comparing Collaborative Personalization and Social Network-based Personalization, *Proceedings of IEEE 45th Hawaii International Conference on System Sciences (HICSS)*, pp.733-742, Hawaii, USA, 2012.

[7] Shirgave, S., Kulkarni, P., and Borges, J. Semantically Enriched Web Usage Mining for Personalization. International Journal of Computer, Electrical, Automation, *Control and Information Engineering*, 8(1), 249-257, 2014.

[8] AI-Yazeed, N.M.A., Gadallah, A.M. and Hefny, H.A. A Hybrid Recommendation Model for Web Navigation, *Proceedings of IEEE 7th International Conference on Intelligent Computing and Information Systems (ICICIS'15)*, pp. 552-560, 2015

[9] More, S. Modified Path Traversal for an Efficient Web Navigation Mining. *Proceedings of IEEE International Conference on Advanced Communication Control and Computing Technologies (lCACCCT)*, pp. 940-945, 2014.

[10] Awa, M.A. and Khan, L.R. Web Navigation Prediction Using Multiple Evidence Combination and Domain Knowledge. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 37(6): pp. 1054–1062, 2007.

[11] Rafailidi, D. and Nanopoulos, A. Modeling Users Preference Dynamics and Side Information in Recommender Systems. *IEEE Transactions on System, Man, and Cybernetics: System*, 46(6), pp.782-792, 2016.

[12] Aggarwal, S. Van Oostendorp, H. and Indurkhya, B. Automating Web-Navigation Support Using a Cognitive Model, *Proceedings of ACM the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)*, No. 19, Thessaloniki, Greece, 2014.

[13] Chen, M. and Yu, Ryu. Facilitating Effective User Navigation through Website Structure Improvement, *IEEE Transactions on Knowledge and Data Engineering*, 25 (3), pp. 571-588, 2013

[14] Levene, M. *An Introduction to Search Engines and Web Navigation*, John Wiley & Sons, 2010.

[15] Yen, B.P.-C. and Wan, Y.-W. Design and Evaluation of Improvement Method on the Web Information Navigation – A

Stochastic Search Approach. *Decision Support Systems*, 49(14-23), 2010.

[16] Li, D., Choi, J. and Kim, J.H. LRFU (Least Recently/Frequently Used) Replacement Policy: A Spectrum of Block Replacement Policies, Technical Report SNU-CE-AN-96-004, Department of Computer Engineering, Seoul National University, July 1996.

[17] Cheng, K. and Kambayashi, Y. A Size-Adjusted and Popularity-Aware LRU Replacement algorithm for Web Caching. *The 24th Annual International Conference of Computer Software and Applications*. 1997.

[18] Cao, P. and Irani, S. Cost-aware WWW proxy caching algorithms. *USENIX Symposium on Internet Technologies and Systems. Monterey*, California, USA. 1997.

[19] Cherkasova, L. Improving WWW Proxies Performance with Greedy-Dual-Size-Frequency Caching Policy, Computer Systems Laboratory HPL-98-69 (R.1), *HP Computer System Laboratory*. November 1998.

[20] Wooster, R.P. and Abrams, M. Proxy Caching That Estimate Page Load Delay, *Journal of Computer Networks and ISDN Systems*, 29(8-13), pp. 977-986, September 1997.

[21] Niclausse, N., Liu, Z., and Nain, P. A New Efficient Caching Policy for the World Wide Web, *Proceedings of the Workshop on Internet Server Performance (WISP)*, pp. 119–128, Madison, WI, USA, June 1998.

[22] Waleed Ali, Siti Mariyam, Shamsuddin. Intelligent Dynamic Aging Approaches in Web Proxy Cache Replacement, *Journal of Intelligent Learning System and Application*, 7, pp. 117-127, 2015.

[23] Yang, Q., Zhang, H. H., and Zhang, H. Taylor Series Prediction: A Cache Replacement Policy Based on Second-Order Trend Analysis, *Proceedings of the 34th Hawaii International Conference on Systems Sciences (HICSS)*.Hawaii, USA, 2001.

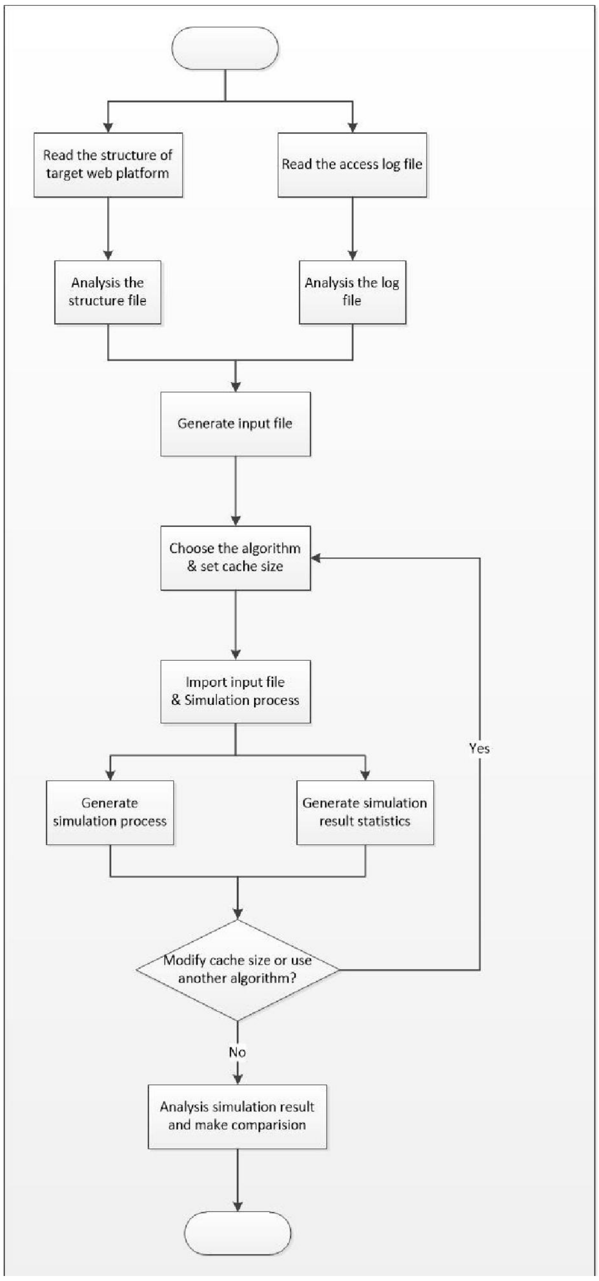
[24] Sarhan, A., Elmogy, A. M., Ali, S. M. A New Web Cache Replacement Approach based on Internal Request Factor, *International Journal of Computer Science and Network Security*.15(3), pp. 73-73, March 2015

[25] Tewari, G., Hazelwood, K. Adaptive Web Proxy Caching Algorithms, Computer Science Group, TR-13-04, Harvard University Cambridge, Massachusetts, 2013.

[26] Negrão, A.P., Roque, C., Ferreira, P. and Veiga, L. An Adaptive Semantics-Aware Replacement Algorithm for Web Caching, *Journal of Internet Services and Applications*, 6(4), Springer, 2015.

APPENDIX

Activity Diagram of Simulation Procedure


Read the structure of target web platform
Read the access log file
Analysis the structure file
Analysis the log file
Generate input file
Choose the algorithm & set cache size
Import input file & Simulation process
Yes
Generate simulation process
Generate simulation result statistics
Modify cache size or use another algorithm?
No
Analysis simulation result and make comparision