DE GRUYTER OPEN

Print ISSN: 1312-2622; Online ISSN: 2367-5357
DOI: 10.1515/itc-2017-0004

Improving the Performance of an Information Retrieval System through WEB Mining

V. Sathiyamoorthi

Abstract. *It is generally observed throughout the world that in the last two decades, while the average speed of computers has almost doubled in a span of around eighteen months, the average speed of the network has doubled merely in a span of just eight months!.In order to improve the performance, more and more researchers are focusing their research in the field of computers and its related technologies. World Wide Web (WWW) acts as a medium for sharing of information. As a result, millions of applications run on the Internet and cause increased network traffic and put a great demand on the available network infrastructure. The slow retrieval of Web pages may reduce the user interest from accessing them. To deal with this problem Web caching and Web pre-fetching are used. This paper focuses on a methodology for improving the proxy-based Web caching system using Web mining. It integrates Web caching and Pre-fetching through an efficient clustering based pre-fetching technique.*

1. Introduction

Internet technology has provided a lot of services for sharing and distributing information across the world. Among all the services, World Wide Web (WWW) plays a significant role. The slow retrieval of Web pages may subside the interest of users from accessing them. Therefore, in the present day Internet world, the speed of data transfer plays a vital role. Hence the speed of information retrieval from the Web must be addressed efficiently. To deal with this problem Web caching and Web pre-fetching are the two techniques used. Among these two, proxy-based Web caching has been widely used to reduce the network traffic by caching frequently requested Web pages (Pallis et al. 2008). In this thesis, Web usage mining is used to optimize existing proxy-based Web caching system for better performance.

A. Research Needs

At present, there are many factors that are affecting the performance of Web such as high cost of bandwidth, broken bandwidth and latencies, ever-increasing network distances, bandwidth demands continue to amplify. Hence Web caching plays a predominant role in improving the performance of the Web in the present web scenario.
Based on the above discussion, the research objectives can be framed as follows. The main objectives of a research work is for improving the performance of proxy-based Web caching system by integrating the Web pre-fetching system into Web caching system. This integrated system is arrived at after attaining the following objectives, namely:

1. To prepare the data using data preprocessing techniques such as data cleaning so as to apply data mining tasks on the data.
2. To experiment the optimized Web caching with clustering-based Web pre-fetching technique using sample datasets.
3. To identify and experiment Web caching replacement algorithms.
4. To integrate the two proposed works: i) clustering-based Web pre-fetching and ii) Web caching.
5. To test the optimized Web page retrieval speed that has reduced the latency in accessing a Web page.

B. Methodologies

Some of the methodologies that help to increase the performance of Web are Web caching and Web pre-fetching. Web caching is a technique to store as many Web pages in a location that are nearer to the client. It might be at the client side or at a proxy server or at the server side. Web pre-fetching is used to preload Web pages into the cache before the actual request arrives. As a standard process, Web caching technique makes use of temporal locality and Web pre-fetching technique exploits the spatial locality of Web objects (Teng et al. 2005). When combined these techniques may complement each other. In this background, a proxy-based Web caching and pre-fetching can be more efficient since it reduces the latency incurred while accessing Web pages.

Web caching and Web pre-fetching schemes have been presented in (Podlipnig and Boszormenyi 2003, Teng et al 2005, Balamash and Krunz 2004). They state that integration of these two techniques would perform better. An additional improvement to traditional cache replacement policies used in the proxy server's cache is explained in Pallis et al (2008) which are based on the clustering-based pre-fetching scheme using dependency graph. In this work, the authors have used traditional algorithm to measure the performance of pre-fetching technique.Caching and pre-fetching have often been studied as separate tools for reducing the latency observed by the users in accessing the Web. Less work has been done on integration of caching and pre-fetching techniques. Kroeger et al (1997) have studied the combined effect of caching and pre-fetching on end user latency. Lan et al (2000) have proposed a Rule-Assisted Pre-fetching in Web server caching. Yang et al (2004) have proposed a method for Mining Web Logs to obtain a prediction model and then using the model to extend the well known GDSF caching policy.

C. Problem Addressed

The problem with Web pre-fetching is, to identify the pages that are to be pre-fetched and then to be cached. This is forced by the fact that there are wide spectrums of users, and each one of them has their own preferences. Hence this research work tries to solve the above problem using Modified Adaptive Resonance Theory1 (MART1), a variation of ART1 algorithm, by clustering the users based on their access patterns. Deficiency in generation of cluster prototype vector and the similarity measure used is the challenges with traditional ART1 based clustering technique.

The problem associated with clustering-based pre-fetching is that users may not request some of these pre-fetched objects. In this case, the Web pre-fetching increases the network traffic as well as Web server load and hence lead to reduced bandwidth utilization (Huang et al. 2008). This makes traditional replacement algorithms end with reduced network performance and increased bandwidth consumption. Hence the need for an efficient integration of Web pre-fetching and Web caching to overcome the above said limitations (Pallis et al. 2008).

2. Proposed System

The proposed work integrates Web caching and clustering-based pre-fetching technique using MART1. This work also introduces two replacement policies for better bandwidth utilization and to improve network performances. Due to the grouping of users, the task of going into the individual preferences is avoided. A clustering-based pre-fetching technique, namely MART1 has been proposed and compared with traditional ART1 technique. The MART1 would provide better inter-, intra-clusters distance and produce highly homogeneous clusters than the traditional ART1.

A cache replacement policy has been considered in the second work. This work proposes two different cache replacement policies namely Modified Least Frequently Used (MLFU) and Pre-fetching based Modified LFU (MPLFU) to address the issues while integrating clustering-based Pre-fetching technique with Web caching. Both MLFU and PMLFU provide better performance by reducing the number of objects to be pre-fetched and increases the byte hit rate thereby improving the bandwidth utilization. The MLFU combines the benefits of frequency, recency, popularity and the size of a Web document in removal policy while PMLFU updates priority dynamically after assigning the priority based on whether it is: i) an actual request or ii) a pre-fetching request. This automatic update of priorities enables the efficient utilization of cache both for actual and pre-fetching requests.

3. Related Work

Podlipnig and Boszormenyi (2008) have presented an overview of various replacement algorithms. They conclude that Greedy Dual-Size (GDS) outperform when cache size is small. Martin (1996) has discussed that SIZE outperforms than Least Recently Used (LRU) and several Least Frequently used (LFU) variations in terms of different performance measures such as hit rate and byte hit rate. Their experiments have not considered the object frequency in decision making process. Web pre-fetching or pre-loading is a technique which pre-fetch Web pages into the cache before even the actual request arrive. There are two approaches namely,

i. **Short-term Pre-fetching:** Where Web pages are pre-fetched into the cache by analyzing recent Web cache access history (Chen et al. 2003).

ii. **Long-term Pre-fetching:** Where the probability of accessing Web pages are identified and pre-fetched by analyzing the global access pattern (Heung et al. 2009).

Both Web caching and Web pre-fetching schemes were presented in Podlipnig and Boszormenyi 2004, Teng et al. 2005. It is stated that integration of these two techniques would perform better. An additional improvement to traditional cache replacement policies used in the cache of proxy server is explained in George et al. 2008. This is based on the clustering-based pre-fetching scheme using dependency graph. The authors state that a graph-based approach to cluster the Web pages would be more effective. It is pruned by a threshold value. The authors (Jyoti et al. 2008) also have arrived at an approach that predicts the page access before user accessing them. They have used higher order Markov model for predicting next user request. Most of the work discussed above will predict only one object at a time which will increase network traffic when the number of users gets increased. To overcome this drawback, it is decided to propose a clustering-based pre-fetching technique using MART1 algorithm.

4. WEB Usage Mining

Log files are raw text files which contain information about the user's access to a Web site. It keeps track of information like who accessed, what was accessed from and when accessed a Web site. *Figure 1* shows the example of actual log file generated by the proxy server. Various fields included in this file are: time stamp of the request, time required to process the request in millisecond, IP address of the machine requesting the object, response code, requested item size in bytes, type of method used, requested object name, identity information, redirection information, whether the request was redirected to another server and content type.

A. Data Preparation

The raw proxy server log files are unsuitable for access pattern analysis. This log file requires efficient preprocessing to remove irrelevant, inconsistent and incomplete data from the proxy server log file for analysis. It is important to remove all requests from the Web proxy log file that are not explicitly requested by the user.

```
1168300926.602 285938 103.7.55.59 TCP_MISS/504 1663
GEThttp://204.95.60.12/servlet/StorageGuard/update/updateclientversion=2.1andversion=99.99andlanguage=en
uandoem=vsgandbannerDate=05/01/2010 - TIMEOUT_DIRECT/204.95.60.12 text/html
1168300927.853 1250 50.141.5.120 TCP_DENIED/407 1995 GET
http://cdn5.tribalfusion.com/media/261216.gif - NONE/- text/html
1168300928.348 1746 151.33.90.119 TCP_MISS/404 333 GET
http://info.ddcd.jp/ddcd3_info/fujitsu/Fujitsu_JPN_CD_News(421).txt - DIRECT/210.174.185.15 text/html
1168300928.351 1750 151.33.90.119 TCP_CLIENT_REFRESH_MISS/404 333 GET
http://info.ddcd.jp/ddcd3_info/fujitsu/Fujitsu_JPN_CD(421).txt - DIRECT/210.174.185.15 text/html
1168300928.354 1752 151.33.90.119 TCP_CLIENT_REFRESH_MISS/404 333 GET
http://info.ddcd.jp/ddcd3_info/fujitsu/Fujitsu_JPN_CD_BLN(421).txt - DIRECT/210.174.185.15 text/html
```

Figure 1. Snapshot of sample proxy server log

Identification of user from a log file is a critical task in most web usage mining applications. Most of the log files provide only the computer IP address and the user agent for user identification. For Web sites requiring user registration, the log file also contains the user login that can be used for the user identification. In this work, to identify frequent users and frequent pages for pre-fetching, an individual IP address is identified as a user.

For each unique IP address identified in user identification process, the page identification process constructs tuple of the form $\{IP_i, Pages_j\}$. This helps in identifying the set of pages visited by the particular user from a particular machine. The most frequent visitors are identified and stored in a vector – users $\{u_1, u_2, u_3 \ldots u_n\}$. The most frequent pages visited are identified and stored in a vector – pages $\{p_1, p2 \ldots p_m\}$. In the access pattern entries a_{ij} indicates the number of times user i has visited the page j. The vector gives information about the preferences of each user visiting the site. *Figure 2* shows the Web access pattern.

$$U \begin{bmatrix} & P & \\ a_{11} & a_{12} & a_{13} \ldots\ldots a_{1m} \\ a_{21} & a_{22} & a_{23} \ldots\ldots a_{2m} \end{bmatrix}$$

Figure 2. Web Access Pattern

5. Optimization of WEB Cache Performance through Clustering Based Pre-Fetching Technique Using Mart1

The architecture of the proposed system includes the following modules:

1. *Preprocessing component* to extract relevant fields from Web log file.
2. *Feature Extractor* to extract the access pattern from the log file.
3. *Clustering component* to apply MART1 algorithm for grouping users based on the access pattern.
4. *Pre-fetcher* module to identify the pages to be pre-fetched.

The MART1 approach differs from ART1 in the following ways:

- Use of binary addition on the input pattern with winning column of the top down matrix instead of the usual multiplication.
- Using binary addition introduces a new problem – i.e., all the bits in the centroid of the cluster become '1'. To overcome this issue, the centroid (top down weight of the winning column) is chosen by performing the test between the current input pattern and all the input patterns of the users belonging to that cluster.
- The similarity between two binary vectors A and B is calculated as follows

$$DAB = S / |B|,$$

where S refers to Number of 0's in A whose corresponding bit in B is 1. DAB is the distance between A and B, a conditional probability that defines object dissimilarity, that is, if A and B are similar then it should be assigned minimum DAB value.

A. Pre-Fetching

The steps involved in pre-fetching are given below.

- After the training of MART1 network, if a particular user request arrives, the previous access pattern of the user is searched from the database. If the access pattern is found, then it is fed into the MART1 network to identify the user cluster.
- After identifying the cluster, the output is the prototype of the cluster. Based on this prototype, the pages are pre-fetched and cached. Once cached, they can be accessed at much higher speed.

6. Results and Discussions

The datasets for testing the proposed system have been obtained from National Laboratory of Applied Network Research (NLANR) project that provides dataset for

Table 1. Testing Datasets and it Preprocessing Details

S.No.	Data source Name	Size of the Data source	Size of the Data source after preprocessing	No. of unique users	No. of unique pages	No. of frequent users	No of frequent pages
1	sv[1].sanitized-access.20070109	76.9 MB	1.18 MB	53	4157	44	986
2	bo2[1].sanitized-access.20070109	76.0 MB	0.957 MB	99	5376	82	417
3	ny[1].sanitized-access.20070109	66.5 MB	1.19 MB	53	4576	46	797
4	uc[1].sanitized-access.20070109	78.2 MB	1.35 MB	90	5046	75	372

researchers and encourages research on Web caching. The datasets and its preprocessing details are shown in *table 1*. The performance of the MART1 clustering algorithm is observed better, than ART1 clustering algorithm in terms of the average inter- and intra-cluster distance. *Figure 3* depicts the difference in intra-cluster distance between ART1 and MART1. It is seen that the intra-cluster distance for the MART1 algorithm is zero since the center of each cluster is formed based on binary addition. Thus, it pre-fetches all the frequent pages corresponding to all the frequent users belonging to that cluster. The difference in inter-cluster distance by ART1 and MART1 is shown *figure 4*. From the graph, it is inferred that use of MART1 algorithm does not affect the inter-cluster similarity. The average inter- cluster distance shows that 97.5 percent of the pages pre-fetched by the clusters is different. *Figure 5* and *figure 6* compares the hit rate without pre-fetching, pre-fetching using ART1 and pre-fetching using MART1 for different cache sizes. From the graph, it is observed that the average hit rate increases as it moves from schemes ART1 to MART1.

7. Novel Approaches for Integrating WEB Pre-Fetching and WEB Caching System

Most of the existing pre-fetching techniques employ single object pre-fetching technique, which is handled by traditional cache replacement algorithms. However, using the clustering-based pre-fetching technique, multiple objects are pre-fetched that users may not request some of these objects but the server load is increased. So to overcome these problems, efficient integration of Web pre-fetching and caching is challenged. The proposed replacement policies are compared with LFU, LRU, First in First out (FIFO), GDS and Greedy- Dual- Size-Frequency (GDSF) for hit rate and byte hit rate. For the GDS and GDSG algorithms the cost function considered is one. Cache size considered, starts from 1 MB to 5 MB, because the size of the testing dataset is smaller than 1.5 MB.

In order to evaluate the performance, cache size has chosen smaller than testing datasets. *Figure 7* shows the performance of MPLU algorithm in terms of hit rate under varying cache size. It is understood that MLFU performs much better than all other cache replacement policies. Graph plots show the performance of proposed work of MLFU in terms of hit rate with different cache size in dataset shown in *table 1*. To increase the byte hit rate of LFU policy, MPLFU policy has been proposed. Graph plots in *figure 8* shows the byte hit rate of MPLFU policy compared with LRU and LFU policies. From *figure 8*, it is seen that MPLFU algorithm provides better byte hit rate than LFU and LRU algorithm.

Graph given below gives the performance of MPLFU policy in terms of hit rate under varying cache size. From this it is observed that MPLFU policy provides higher byte hit rate. Hence network performance gets improved.

Figure 9 and *figure 10* shows the comparison of two proposed work MPLFU and MLFU polcies. These two policies are compared against hit rate and byte hit rate. It is inferred that MPLFU policy provides higher byte hit rate and low hit rate whereas MLFU policy provides mostly higher hit rate and byte hit rate. Hence it saves the bandwith. While considering hit rate of MPLFU policy it provides only a moderate hit rate than LFU policies. However, MPLFU policy improves network performance by yielding higher byte hit rate. Hence it provides better bandwidth utilization.

8. Conclusion and Future Work

The speed of information retrieval is very important while using the Internet. With the proposed system, the information providers can provide information at a faster rate to satisfy and retain their visitors. In this approach, frequently requested Web pages by the users are tracked and identified, in addition to the integration of the Web pre-

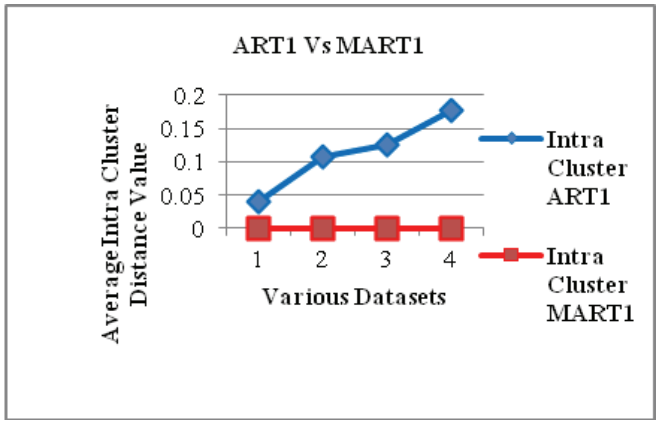

Figure 3. ART1 Vs MART1 intra-cluster distance

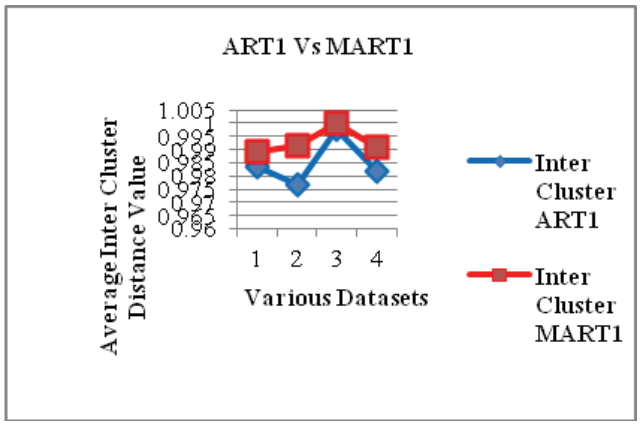

Figure 4. ART1 Vs MART1 inter-cluster distance

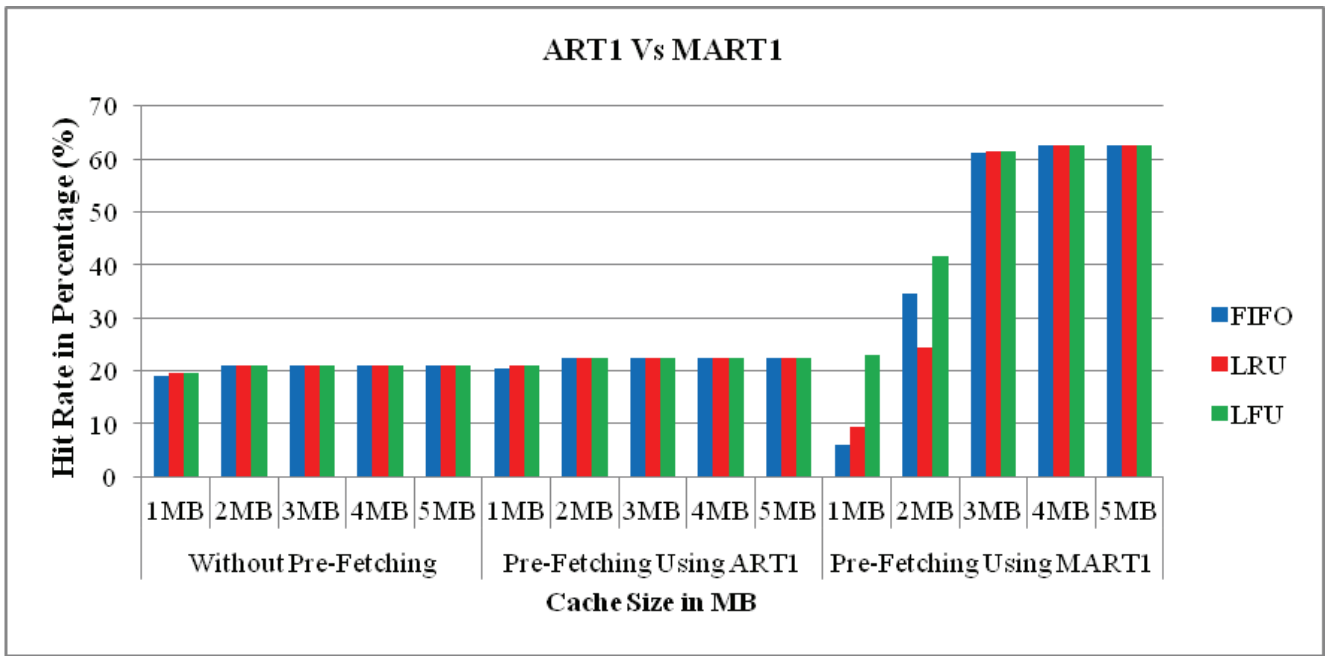

Figure 5. ART1 Vs MART1 using Data Set (ny[1].sanitized-access.20070109)

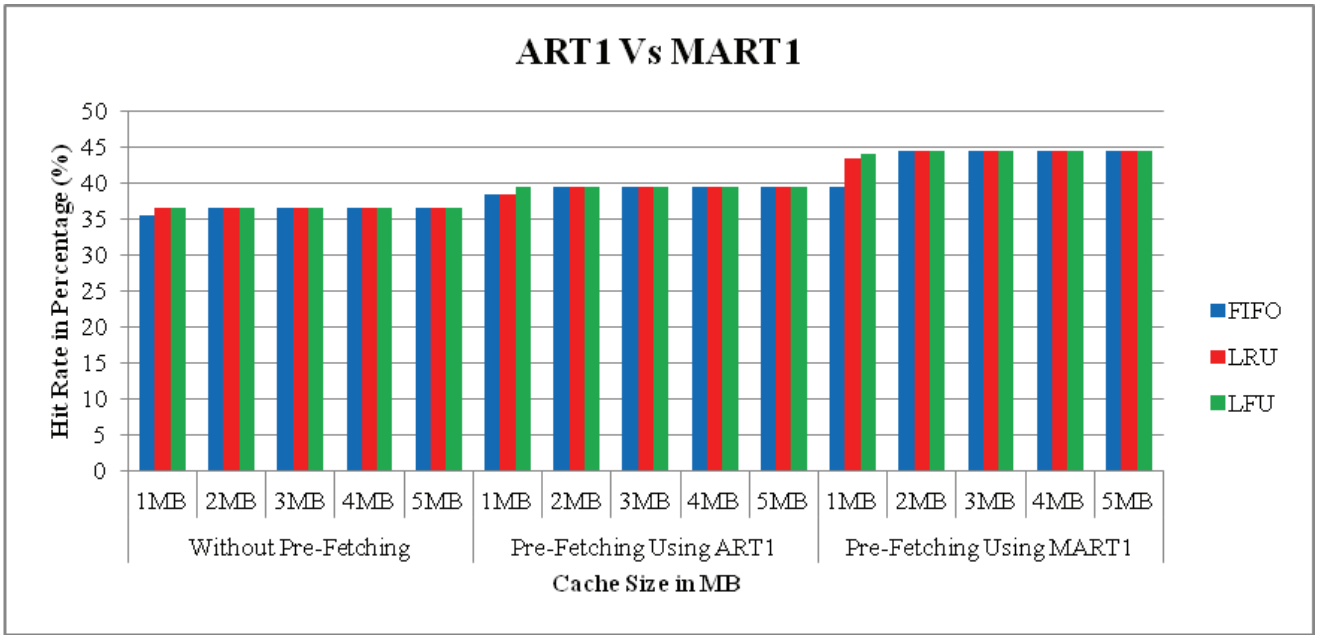

Figure 6. ART1 Vs MART1 using Data Set (bo2[1].sanitized- access.20070109)

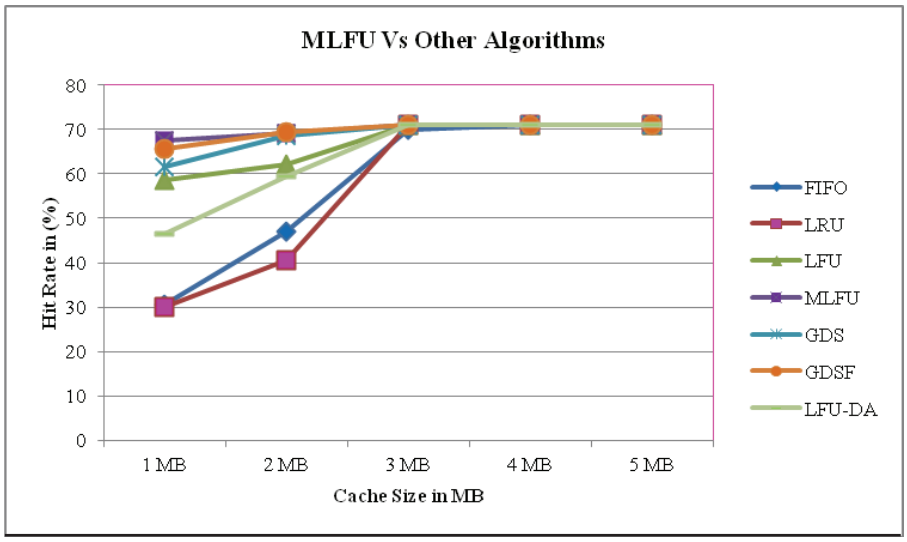

Figure 7. Comparision of Hit Rate by FIFO, LRU, LFU and MLFU on Dataset (sv[1].sanitized-access.20070109)

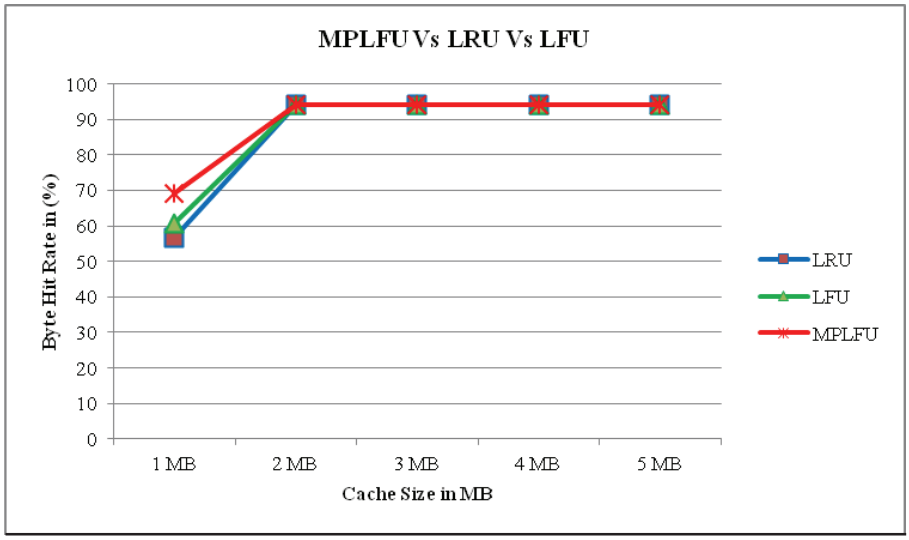

Figure 8. Byte Hit Rate of LRU Vs LFU Vs MPLFU on dataset (bo2[1].sanitized- access.20070109)

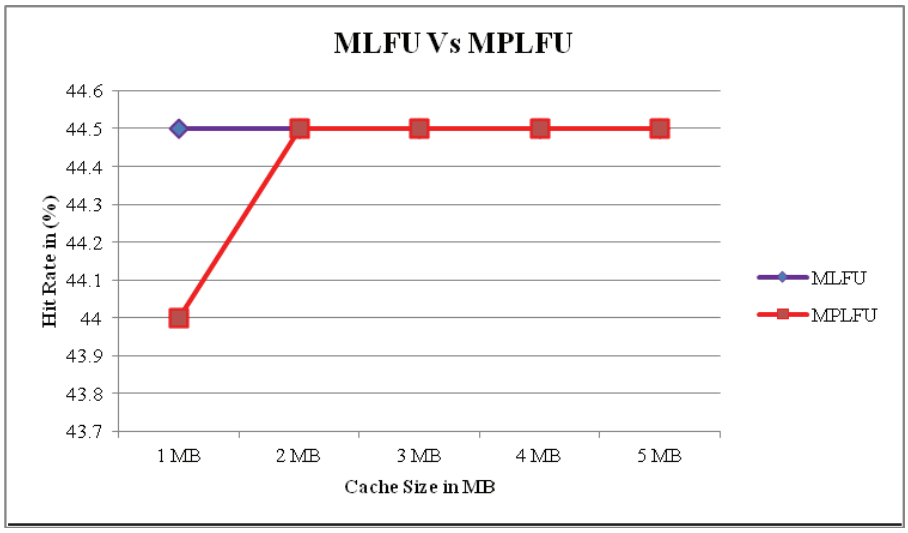

Figure 9. Comparison of Hit Rate in MLFU and MPLFU on dataset (uc[1].sanitized-access.20070109)

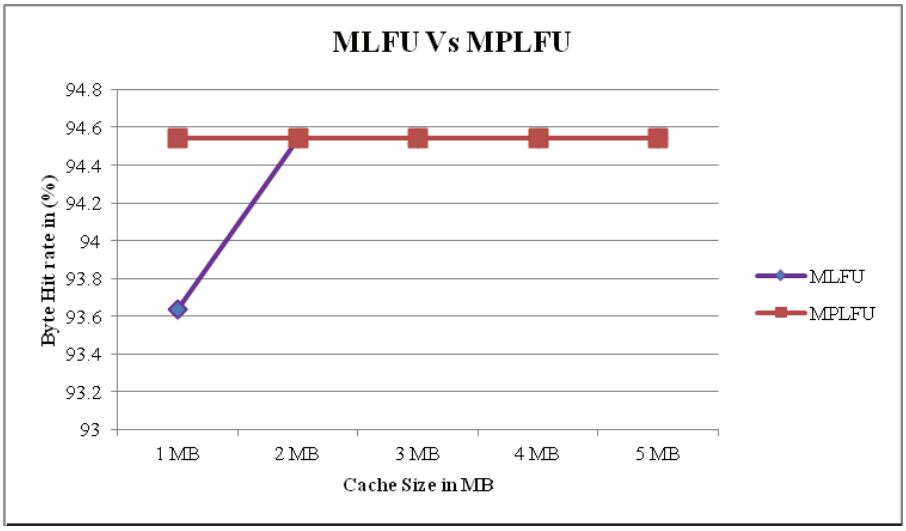

Figure 10. Comparison of Byte Hit Rate in MLFU and MPLFU on dataset (uc[1].sanitized-access.20070109)

fetching scheme and Web caching to achieve performance improvement for the proxy-based Web cache.

It is evident that the proposed cache replacement policies suit the proposed system for improved network performance. It is expected that the user datasets containing the privacy information should not be exposed to the outside world. Therefore, privacy preserving data mining techniques can also be applied in order to hide personal information about the users. Further, it is also possible to apply the evolutionary optimization technique in MART1.

References

1. Chen, Y., L. Qiu, W. Chen, L. Nguyen and R. H. Katz. Efficient and Adaptive Web replication using content clustering. Selected Areas in Communications, IEEE Journal on 21(6), 2003, 979-994.

2. Teng, W., C. Y. Chang, and M. S. Chen. Integrating Web Caching and Web Pre-fetching in Client-side Proxies. – *IEEE Transactions on Parallel and Distributed Systems,* 16, 2005, Issue 5, 444-455.

3. Podlipnig, S. and L. Boszormenyi. A Survey of Web Cache Replacement strategies. – *ACM Computing Surveys (CSUR)*, 35, 2003, 4, 374-398.

4. Pallis, G., A. Vakali and J. Pokorny. A Clustering-Based Pre-Fetching Scheme on A Web Cache Environment. – *ACM Journal Computers and Electrical Engineering*, 34, 2008, Issue 4.

5. Jyoti, P., A. Goel, A. K. Sharma. A Framework for Predictive Web Pre-fetching at the Proxy Level Using Data Mining. – *IJCSNS*, 8, 2008, No. 6, 303-308.

6. Arlitt, M. F. and C. L. Williamson. Trace-Driven Simulation of Document Caching Strategies for Internet Web Servers. – *J. of Simulation*, 68, 1997, 23-33.

7. Heung, K. L., S. A. Baik and E. J. Kim. Adaptive Pre-fetching Scheme Using Web Log Mining in Cluster-based Web. – *ICWS*, 2009, 1-8.

8. Feng, W., S. Man and G. Hu. Markov Tree Prediction on Web Cache Pre-fetching. Software Engineering, Artificial Intelligence (SCI), Springer-Verlag Berlin Heidelberg, 2009, 105-120.

9. Huang, Y. F. and J. M. Hsu. Mining Web Logs to Improve Hit Ratios of Pre-fetching and Caching. – *Knowledge-Based Systems*, 21, 2008, 1, 62-69.

10. Rangarajan, S. K., V. V. Phoha, K. Balagani, R. R. Selmic and S. S. Iyengar. Web User Clustering and its Application to Pre-fetching Using ART Neural Networks. – *IEEE Computer,* 2004, 1-15.

Manuscript received on 08.10.2016

Contacts:
Dr ***V. Sathiyamoorthi,*** *Associate Professor/CSE*
Sona College of Technology, Salem-5, Tamilnadu, India.
e-mail: Sathyait2003@gmail.com