I.J. Information Engineering and Electronic Business, 2016, 5, 60-67
Published Online September 2016 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijieeb.2016.05.08

A Novel Probability based Approach for Optimized Prefetching

Arvind Panwar
Northern India Engineering College, Delhi
Email: Arvind.nice3@gmail.com

Achin Jain and Manish Kumar
Bharati Vidyapeeth College of Engineering, Delhi
Email: {achin.jain, manish.kumar}@bharatividyapeeth.edu

Abstract—As the World Wide Web carries on to grow up rapidly in size and popularity, web traffic and network bottlenecks are more important issues in the networked world. The continued enhancement in demand for items on the World Wide Web causes severe overloading in many sites, network congestion, delay in perceived latency and network bottleneck. Many users have no patience in waiting more than a few seconds for downloading a web page, that's why Web traffic reduction system is very necessary in today World Wide Web for accessing the websites efficiently with the facility of existing networks. Web caching is an effective method to improve the performance of the World Wide Web but in today's World Wide Web caching method alone is not enough because of World Wide Web has grown quickly from a simple information-sharing mechanism to a rich collection of dynamic objects and multimedia data. The web prefetching is used to improve the performance of the proxy server. Prefetching predict web object that is expected to be requested in the near future and store them in advance, thus the response time of the user request is reduced. To improve the performance of the proxy server, this paper proposed a new framework which combines the caching system and prefetching technique and also optimize the prefetching with the help of probability. In this paper, we use the dataset for the experiment which is collected from ircache.net proxy server and give the result with the comparison of other technique of prefetching.

Index Terms—Prefetching, Web Caching System, Probability, Web Mining.

I. Introduction

It is beyond doubt that the new sudden increase of World Wide Web has transformed not only the authority of computer-related sciences, but also the lifestyles of people and the economies of the countries. The web server is the only system who enables any kind of web movement. Since its beginning, the web server has always taken the form of daemon procedure. The web server takes HTTP request, process the request and respond back with the requested file. As web services are more and more becoming popular, network gaming, latency, and server overfilling has become major problems. To defeat these problems, a lot of efforts are being made continuously to enhance the web performance.

The web caching is acknowledged as one of the efficient techniques to improve the server bottleneck performance and to decrease the network traffic, thus reducing network latency. In our work, we use the proxy caching system. The fundamental idea is to cache requested objects on the proxy server so that they don't have to be fetched again from the web server. Although web cache systems decrease the network and I/O bandwidth utilization, they still experience from a low hit rate, out of date data and incompetent resource management. [1] Shows that an inefficient web cache management caused a major news website crash, also called the Slashdot effect. Many researchers currently working on the web caching system, web cache is used to improve the performance of the proxy server, but web cache is not enough for the today world wide web because world wide web has grown quickly from an easy information-sharing system presenting only static text and images into a rich collection of dynamic and interactive services, such as video/audio conferencing e-commerce and distance learning. Web caching system is working on the past accessed objects; if we can predict the object which is expected to be requested in the near future and store them in advance thus we can reduce the response time of the user request. This type of caching system is called the preaching of the web objects, but prefetching is alone is not enough, that's why many researchers proposed the new method to integrate the web caching and prefetching together to improve the performance the server or to reduce the latency.

Web prefetch methods defeat the drawback of a web cache system through pre-processing web objects before a user request comes. Web prefetch methods predict future requests through web log file analysis and prepare the predictable requests before getting it. Evaluate with web cache schemes, the web prefetch method focal point of the spatial place of objects when present requests are related to previous requests. When we use the web prefetching method, it increase the bandwidth utilization

and decrease or hide the latency due to bottleneck at the web server. But the web prefetching method should be very careful as a wrong scheme of preaching can cause major network bandwidth bottlenecks rather than decreasing the web user latency.

In this paper, we are focusing on the use of the proxy server to improve the performance of the network so we focus on the proxy server cache memory or proxy server caching system. In computer networks, a proxy server is a server (a computer system or an application) that works as a mediator for requests from clients looking for resources from other servers. A client sends request to the proxy server for the object, such as a file, connection, web page, or other resource presented from a different server and the proxy server assesses the request as a way to make simpler and control its complexity. Today, most proxies are web proxies, facilitating access to content on the World Wide Web. Web proxy caching acts as key responsibilities to enhance the web performance by maintaining web objects that are expected to be visited again in the proxy server. This proxy caching assists in decreasing the supposing latency, reducing network bandwidth utilization. The space assigns to a cache memory is very limited in proxy server; space must be used very effectively. The cache replacement algorithm is the heart or central system of web caching, thus the designing of effective replacement algorithm is very important for the caching method to achieve the target. The most replacement algorithm is not well-organized enough and may experience a cache pollution problem (Cache pollution means that a cache contains objects that are not frequently visited) because they consider only one factor i.e. Past access of the objects and ignores the other factor which reduces the effective cache size and negatively affects the performance of the web proxy caching system. Many researchers proposed some web proxy caching method have attempted to merge some aspects which can increase the performance of web proxy caching for making decision about caching, but this is not an easy task, because one aspect in a particular environment may be more important in other. To determine which object is re-visited is still a big challenge faced by the web proxy caching method. In other words, here is a big question about the caching web objects that which one should be cached and which one should be replaced to improve the hit ratio and reduce the network traffic. Web proxy server, web proxy log files which record all the activities of the every user and behavior of the user. By analyzing the proxy server log file, we can predict the future of a user and can get the next access web object by mining the log file. By storing the web object in caching before the requested, we can achieve the prefetching and to improve the performance of the proxy caching system we integrate the prefetching system with web caching.

II. Related Works

Caching and prefetching: Web caching is used to reduce the network traffic by caching web pages at the proxy server level. Now a day's caching alone is not sufficient because of World Wide Web has evolved rapidly from a simple information-sharing mechanism [21]. The browser renders resources in between two consecutive resource downloads. However, during this period, the wireless interfaces consume energy doing nothing useful [22]. The Today World Wide Web is need of every person in this world to get & share the information and to use the huge warehouse of resources available online. World Wide Web carries on growing up rapidly in size and popularity, web traffic, and network bottlenecks are more important issues in the networked world. To access the resources, web users experience the response time delay, in order to several seconds. Sometimes such type of response time delay is unacceptable. The reason for this type of delay is a World Wide Web causes severe overloading in many sites, network congestion, and network bottleneck. Many users have no patience in waiting more than a few seconds for downloading a web page, that's why Web traffic reduction system is very necessary in today World Wide Web for accessing the websites efficiently with the facility of existing networks. Web caching is an effective method to improve the performance of the World Wide Web but in today's World Wide Web caching method alone is not enough because of World Wide Web has grown quickly from a simple information-sharing mechanism to a rich collection of dynamic objects and multimedia data. The web prefetching is used to improve the performance of the proxy server. Prefetching predict web object that is expected to be requested in the near future and store them in advance thus the response time of the user request is reduced. To improve the performance we integrate the caching system and prefetching system together.

However, the benefits of caching make smaller as Web documents become more dynamic [1]. Prefetching (or proactive caching) goal is to defeat the limitations of passive caching by proactively fetching documents in expectation of ensuring demand requests. Several studies have demonstrated the effectiveness of prefetching in addressing the limitations of passive caching (e.g., [2, 3, 5, and 6]). The Prefetched documents may include hyperlinked documents that have not been requested yet as well as dynamic objects [7, 4]. Out of date, cached documents may also be updated through prefetching. In principle, prefetching systems need forecast the documents that are most probable to be visited in the near future and determining how many documents to prefetch. Most research on the web prefetching is focused on the prediction point of view. In many of these studies (e.g., [9]), a fixed-threshold-based approach is used, whereby a set of candidate files and their accessibility probabilities are first determined. Among these candidate files, those whose access probabilities exceed a certain prefetching threshold are prefetched. Other prefetching schemes involve prefetching a fixed number of popular documents [8].

Teng et. al [10] proposed the Integration of Web Caching and Prefetching (IWCP) cache replacement

policy, which considers both demand requests and prefetched documents for caching based on a normalized profit function. The work in [11] focuses on prefetching pages of query results of search engines. In [12], the authors proposed three prefetching algorithms to be implemented at the proxy server: (1) the hit-rate-greedy algorithm, which greedily prefetched files so as to optimize the hit rate; (2) the bandwidth-greedy algorithm, which optimizes bandwidth consumption; and (3) the H/B-greedy algorithm, which optimizes the ratio between the hit rate and bandwidth consumption. The negative impact of prefetching on the average access time was not considered. Most of the above work relies on prediction algorithms that compute the likelihood of accessing a given file. Such computation can be done by employing Markovian models [13].

III. Web Proxy Server Caching Algorithms

Conventional & traditional caching system: This section reviews conventional proxy caching system such as SIZE, Least-Recently-Used (LRU), Greedy-Dual-Size (GDS), Least-Frequently-Used (LFU) and Greedy-Dual-Size-Frequency (GDSF). In this paper, we use the terms traditional and conventional proxy caching policies interchangeably to express about the most common Web proxy caching policies. Most Web proxy servers are still based on the conventional caching policies. These conventional policies are suitable in traditional caching like CPU caches and virtual memory systems, but they are not efficient in the Web caching field. This is because they consider just one factor in order to make caching decisions and ignore the other factors that have more impact on the efficiency of the Web proxy caching [14]. Table 1 shows the comparison of the conventional web caching system.

Table 1. Conventional Web Caching System

Policy	Brief description	Advantages	Disadvantages
LRU	Remove least recently used data	Useful in uniform data such as memory cache	Web object size and downloading time ignore
LFU	Remove least frequently data	Simple	Web object size and downloading time ignore, obsolete web object may store
SIZE	Remove large object first	High cache hit ratio, store small web objects	Byte hit ratio low, small object store even not in use
GDS	Each object have key value in cache, object with lowest key value is remove	Hit ratio high, remove the weakness of SIZE	Byte hit ration low, not consider previous frequency of web objects
GDSF	Modified GDS by including the frequency factor into key value	Hit ratio high, remove the weakness of GDS	Byte hit ration low, not consider future access of web objects

Intelligent web caching system: Although there are many studies of Web caching, research using learning machine techniques in the web caching is still new. Recent studies have shown that the intelligent approaches are more efficient and adaptive to Web caching environments compared to other approaches. More details about intelligent Web caching approaches are given in [15]. In [16], the client-side cache has been divided into two caches namely short-term cache and long-term cache. In NNPCR [17] and NNPCR-2 [18], back-propagation neural network (BPNN) has been used for making cache replacement decision. An integrated solution of BPNN as a caching decision policy and LRU technique as a replacement policy for script data object has been proposed by Farhan (2007) [19]. Foong et al. (1999) [20] proposed a logistic regression model (LR) to predict the future request. In Table 2 show comparison of some intelligent web caching scheme.

Table 2. Intelligent Web Caching System

	Brief description	Feature of dataset	Limitations
ICWCS [33,10]	LRU is used with Neuro-fuzzy system (ANFIS) in cache replacement.	Timestamp, frequency, delay time	• Long time in training process • BHR not good • Client side implemented
NNPCR [2] & NNPCR-2 [4]	According to BPNN cache replacement decision is make	Frequency and size	• BPNN training take long time • Need extra computation
Intelligent web caching architecture[11]	Combination of LRU(replacement) and BPNN(decision)	Bandwidth, script size, number of hits	• Limited to VB.Net script • Similar to SIZE • Suffer cache pollution
Logistic regression in an adaptive Web cache [24]	The author proposed a logistic regression model (LR) to predict the future request.	Time since last access, Frequency within a backward looking window, size and type.	The objects with the lowest re-access probability value are replaced first regardless of cost and size of the predicted object

Proposed framework: Web caching is used to decrease the network traffic flow by caching web pages at the proxy server level, but nowadays caching only is not enough because the World Wide Web has grown quickly from a simple information-sharing device with dynamic and multimedia data. To expand the performance researcher shows that the arrangement of prefetching with the caching approach is good. In this paper, we give a new framework of the arrangement of prefetching and caching techniques to expand the performance of proxy servers. This section describes the architecture of the proposed framework as shown in fig 1. Following subsection describes the component of the proposed system.

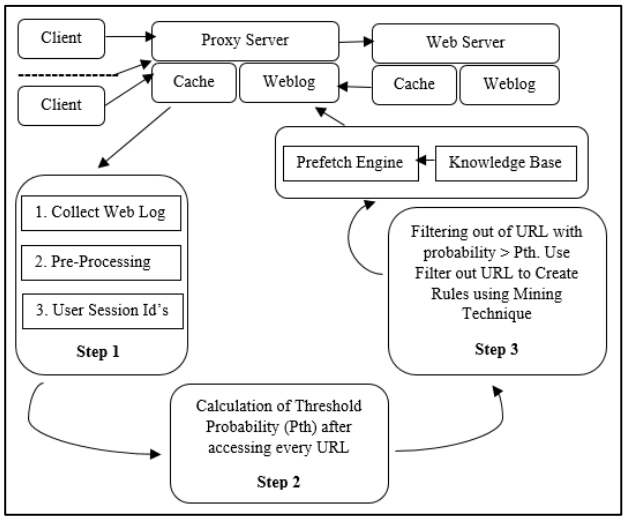

Fig.1. Proposed system framework

Fig 1 shows proposed framework for prefetching. This framework is divided into four parts. Part one contains four tasks, namely collect web log, pre-processing, user session identification and notes all requests with their number of call. Web user visits many web sites time to time and spent random quantity of time between various visits. To deal with the user browsing behavior, we should analyze the proxy server log file. In fussy, the web proxy access log is an in order file with one user access data per line. Web proxy log files, make available information about the actions performed by a user from the moment the user logs.

Preprocessing is defined as removing all the irrelevant and noise data from our actual data. Preprocessing stage is divided into four phases, i.e., Data Extraction, Data Cleansing, Data Transformation, and Data Reduction. In our proposed approach during the preprocessing phase, we carried out the cleansing task to filter out all the unwanted entries from the proxy log data. We use the proxy log explorer tool to preprocess the log record of the proxy server. For our work we have filtered out all the log entries which have status code other than '200', therefore only the requests that are fulfilled are analyzed to make further rules.

After preprocessing the web log data, we identify the user session and separate all data according to the session. In our proposed approach we collect all the request of the user according to season and their number of accesses for example in session s1 user sends the request google.com five times and yahoo.com ten times and in session s2 user sends request google.com eight times and yahoo five times note all such data in excel sheet session wise.

In the second part of the approach we calculate the conditional probability of all requests according to the session. A conditional probability is the probability that an event will occur, when another event is known to occur or to have occurred. If the events are A and B respectively, this is said to be "the probability of A given B". It is commonly denoted by P (A|B). In this approach the conditional probability is used in support of two URL when exist in the same session. Given two URL in same session A=google.com and B=yahoo.com with P (B) > 0, the conditional probability of A given B is defined as the quotient of the joint probability of A and B, and the probability of B:

$$P(A \mid B) = \frac{P(A \bigcap B)}{P(B)} \quad (1)$$

After calculating probabilities of all request, session wise, then the next task to calculate the threshold probability which is used in the prefetching and optimization for the work. To estimate threshold value calculate the average or mean probability for all the URL and set that mean value as threshold value.

In the third part of the approach we filter out all the log data or URL which has the probability less than the threshold value because these URL are not use full for mining and have played a less role in prefetching that's why filter out all such types of data. The next task of part third is using the data for data mining, which have the probability greater than the threshold value. Here in this framework we use "*Enhancing the Performance of Web Proxy Server through Cluster Based Prefetching Techniques*" approach for the prefetching which is a cluster based prefetching approach proposed by S.Nanhey et.al. In the last part of this approach provide a knowledge base and engine. The knowledge base is a warehouse of extracting rules which have been derived by applying the data mining on the web log data of proxy servers. These rules are used in the prefetching and sending recommendation by the recommendation engine to the proxy server using generated rule.

IV. Experimental Work

In the experimental work, we have divided the complete process in two phases i.e., Training Phase and Testing Phase. In the training phase the aim is to train our proposed system to find out the Threshold Value which will be used during Testing Phase.

Training Phase: To carry out the experimental work in this paper we have considered 5 different URL and measure the frequency of occurrence for five sessions of each URL which is shown in table 3.

Table 3. Frequency of URL Session Wise

Session	U1	U2	U3	U4	U5
S1	23	45	33	22	29
S2	15	34	45	12	24
S3	54	76	23	29	44
S4	35	23	21	41	23
S5	28	17	40	36	13

Table 3 shows the individual occurrence of URLs in different sessions, but to monitor the association among the URLs we have found how many times different URLs are occurring together in the same session. For those in

this article we have analyzed different combination of the selected URLs for the experiment. After that conditional probability of each combination is calculated in the training period, which is used to calculate the Threshold Probability value for the testing phase of the experiment. Conditional Probability for each combination of URLs is calculated using the formula given by the equation below and the results are shown in Table 4.

Table 4. Conditional Probability of URL Combination

Prob/Session	S1	S2	S3	S4	S5
P(U1|U2)	0.61	0.8	0.85	0.8	0.68
P(U1|U3)	0.695	0.733	0.611	0.428	0.928
P(U1|U4)	0.913	0.8	0.388	0.942	0.785
P(U1|U5)	0.565	0.733	0.574	0.542	0.714
P(U2|U3)	0.466	0.705	0.236	0.73	0.823
P(U2|U4)	0.311	0.323	0.263	0.913	0.647
P(U2|U5)	0.466	0.411	0.223	0.826	0.705
P(U3|U4)	0.454	0.466	0.521	0.904	0.325
P(U3|U5)	0.454	0.355	0.826	0.952	0.25
P(U4|U5)	0.863	0.75	0.551	0.317	0.305

Threshold Probability is calculated using the equation given below.

$$P_{th} = \frac{\sum P(A|B)}{No.ofInputs} \quad (2)$$

Testing Phase: In the testing phase we have used the same approach as we consider for Training Phase and calculated the conditional probability for each URL combination.

Table 5. Conditional Probability of URL Combination (Testing Phase)

Threshold Probability	URL Pair	Conditional Probability
0.6092	P(U1|U2)	0.74782
0.6092	P(U1|U3)	0.67925
0.6092	P(U1|U4)	0.7661
0.6092	P(U1|U5)	0.62595
0.6092	P(U2|U3)	0.59441
0.6092	P(U2|U4)	0.79158
0.6092	P(U2|U5)	0.52682
0.6092	P(U3|U4)	0.53454
0.6092	P(U3|U5)	0.66771
0.6092	P(U4|U5)	0.67612

From the fig. 2 graph it is clear that values that lies below the trend file in the threshold probability value calculated during the training phase are irrelevant and we will not consider those URL combinations in the generations of Rules which will help in effective prefetching.

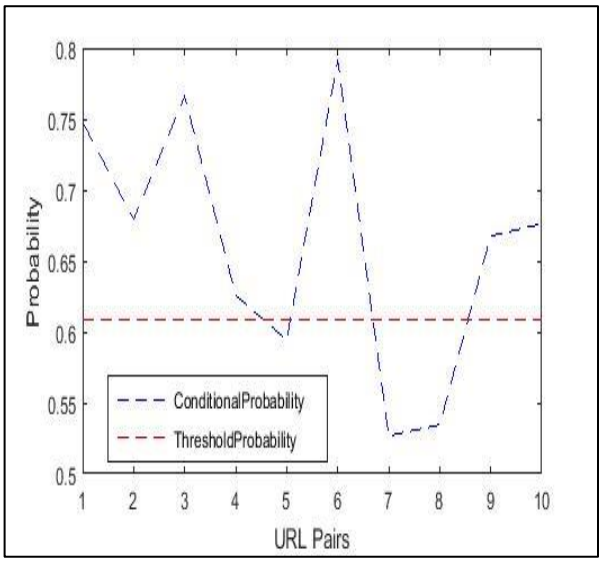

Fig.2. Threshold Graph of URL Probability

After calculating threshold value filter all the URLs according to thershold. We use proxycache.net proxy server data for our experimental work. The data file contains approximately 1500 rows after preprocessing. In prefetching all complete data is used for the rule generation, but in optimize prefetching only the URL which satisfy thershold level is used for the rule generation. Details of data are given below.

- Before filter data: 1500 rows
- After filter data: 695 rows

In the next phase of work apply any pattern discovery technique for the rule generation for the prefetching in this work we apply a cluster based approach for prefetching in which firstly form cluster using K-Mean algorithm then apply the Apriori algorithm for the rule generation.

ExampleSet (695 examples, 2 special attributes, 1094 regular attributes)

Table Index	Construction	Type	Block	Statistics
1100	id	integer	single_value	avg = 348 +/- 200.774
1101	cluster	nominal	single_value	mode = cluster_0 (564),
9	Unique IPs = 27.109.138.21	integer	single_value	avg = 0.754 +/- 0.431
10	Unique IPs = 57.25.111.203	integer	single_value	avg = 0.053 +/- 0.225
11	Unique IPs = 99.143.68.131	integer	single_value	avg = 0.040 +/- 0.197

Fig.3. Meta Data View

To form the cluster, in this work used Rapid miner5 tool. All the screen shot given in this section is generated by rapid miner. Fig 3 shows the Meta data view of the cluster in which all the statistics are given to use in form cluster.

In this work we form five clusters. Fig 4 shows the text view or a cluster model view which shows how many rows are in the data set or used rows to form clusters. Total number of rows is 695 is used to form the clusters text view of the model also describe the number of rows in a particular cluster.

Cluster Model Clustering

Cluster Model

```
Cluster 0: 564 items
Cluster 1: 78 items
Cluster 2: 7 items
Cluster 3: 31 items
Cluster 4: 15 items
Total number of items: 695
```

Fig.4. Cluster model view

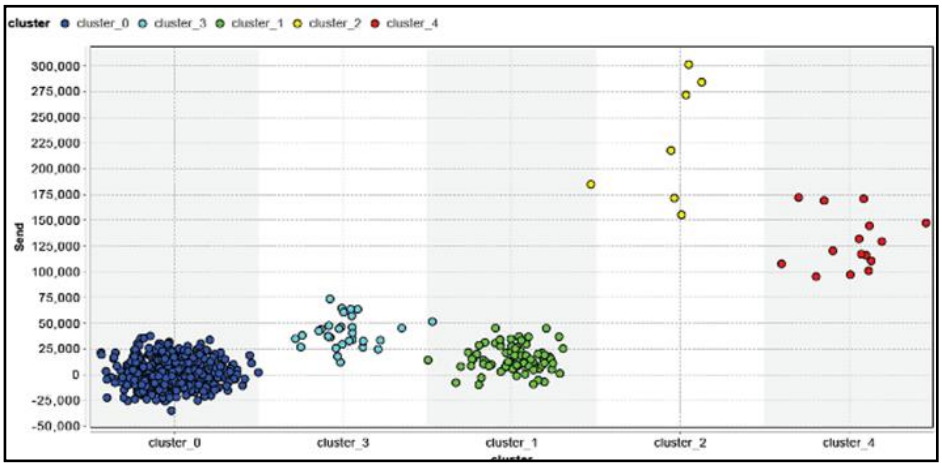

Fig.5. Cluster View

```
Unique IPs=99.143.68.131 (Unknown,Unknown,United States) 20 ==> cluster=cluster_4 2
Unique IPs=99.143.68.131 (Unknown,Unknown,United States) 20 ==> cluster=cluster_3 2
Unique IPs=55.209.150.180 (Fort Huachuca,Arizona,United States) 20 ==> Request=http://
Unique IPs=55.209.150.180 (Fort Huachuca,Arizona,United States) 20 ==> Request=http://
Unique IPs=27.109.138.213 (Unknown,Unknown,Macau) cluster=cluster_6 35 ==> Request=htt
Unique IPs=57.25.111.203 (Unknown,Unknown,Europe) 21 ==> Request=http://www.google.ca/
```

Fig.6. ARM View

Fig 5 shows the cluster view of the data. It is clear from the cluster view that cluster one has the maximum data and cluster four have the minimum number of rows in the cluster. After cluster the data in different cluster next step apply ARM to generate the rule for prefetching.

Fig. 6 shows snapshot of the best rule generated by weka3.6 tool. It is not possible to show all rules here therefore, we give only an appropriate snapshot. These rules are used in prefetching. Form these rules; we take only those data which cross the threshold level of the support. We have taken threshold value of support is 0.3 and used these values in prefetching. After getting the rules we store all rules in the knowledge based database. Knowledge base is useful for the recommendation engine which gets the data for prefetching according to the rule or knowledge and stores that data at proxy cache which is useful to reduce the network traffic and increase the cache hit. In next section we show the results of the experimental work.

V. Result Analysis

In this section, result analysis has discussed on the basis of experimental work in which we have tested out the hit and byte ratio for two predefined schemes i.e., LRU and LFU. Simulations have been carried out by implementing the proposed framework in C using MATLAB. In the experiment we have checked normal scenario without prefetching and then implemented our prefetching approach and tested out the hit and byte hit ratio.

In the first experiment we have compared our approach using LFU, LRU algorithm for page replacement. Fig. 7 shows the effect of hit ratio on different values of cache size used in all the cases LFU, LRU and OPTP. In all the cases, as the cache size increases, the hit ratio also increases accordingly. But from the Fig. 7 It is clear that our approach works better than LFU, LRU without prefetching.

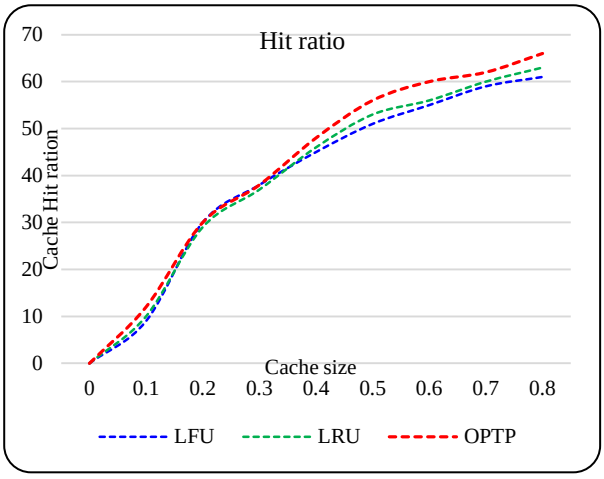

Fig.7. Cache Hit Ratio

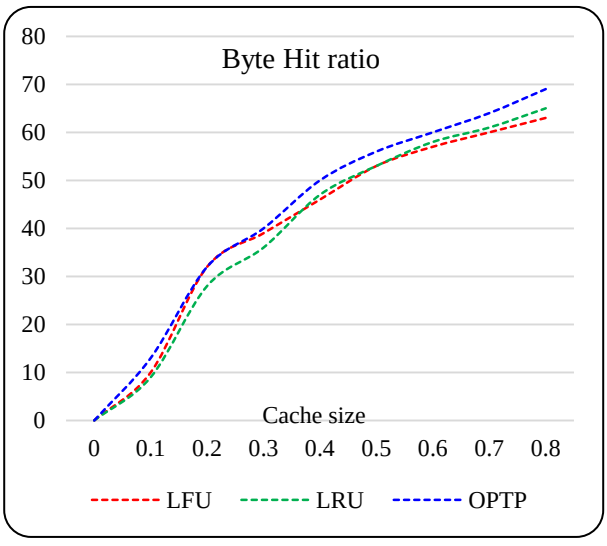

Fig.8. Byte Hit Ratio

Fig.8 shows the comparison between byte hit ratio and cache size in case of LFU, LRU with and without perfecting. It gives a similar result as in Fig. 7. Web cache is a temporary storage memory for web documents. The copies of web documents passing through the proxy server are stored in the web cache, therefore next time the request can be fulfilled directly from the web cache (present in proxy server). After studying the behavior of users, server can prefetch the documents requested by users.

VI. Conclusions

In this paper, we have proposed a framework for optimization of the web prefetching and prediction of web requests of users and accordingly, prefetching the content from the server. The dataset is used for the experimental work which has collected from ftp://ircache.net. The proposed framework optimizes the prefetching by reducing the load approximately 45% and improves performance of web proxy servers using web usage mining and prefetching scheme which is clear in the result section. By using this proposed framework hit ratio is improved up to 2% to 4% as shown in the result section. Designing of the proxy server and implementation of the proposed framework is under the future scope.

References

[1] F. Douglis, A. Feldmann, and B. Krishnamurthy. Rate of change and other metrics: a live study of the World Wide Web. In *Proceedings of USENIX Symposium on Internet Technologies and Systems*, 1997.

[2] A. Bestavros. Using speculation to reduce server load and service time on the WWW. In *Proceedings of the 4th ACM International Conference on Information and Knowledge Management*, pages 403–410, 1995.

[3] M. Crovella and P. Barford. The network effects of prefetching. In *Proceedings of IEEE INFOCOM Conference*, pages 1232–1239, 1998.

[4] S. Schechtera, M. Krishnanb, and M. Smithc. Using path profiles to predict http requests. In *Proceedings of the 7th Internationa World Wide Web Conference*, pages 457–467, April 1998.

[5] A. Venkataramani, P. Yalagandula, R. Kokku, S. Sharif, and M. Dahlin. The potential costs and benefits of long-term prefetching for content distribution. In *The Sixth Web Caching and Content Distribution Workshop*, 2001

[6] I. Zukerman, D. Albrecht, and A. Nicholson. Predicting users' requests on the WWW. In *Proceedings of the 7th International Conference on User Modeling*, pages 275–284, 1999

[7] A. B. Pandey, J. Srivastava, and S. Shekhar. Web proxy server with intelligent prefetcher for dynamic pages using association rules. Technical 01-004, University of Minnesota, Computer Science and Engineering, January 2001.

[8] E. Markatos and C. Chronaki. A top-10 approach to prefetching on the web. In *Proceedings of the INET Conference*, 1998.

[9] V. Padmanabhan and J. Mogul. Using predictive prefetching to improve World Wide Web latency. In *Proceedings of the ACM SIGCOMM Conference*, pages 26–36, 1996.

[10] W.-G. Teng, C.-Y. Chang, and M.-S. Chen. Integrating web caching and web prefetching in client-side proxies.

IEEE Transactions on Parallel and Distributed Systems, 16(5):444– 455, May 2005.
[11] R. Lempel and S. Moran. Optimizing result prefetching in web search engines with segmented indices. *ACM transactions on Internet Technology*, 4(1):31–59, February 2004.
[12] B. Wu and A. D. Kshemkalyani. Objective-optimal algorithms for long-term Web prefetching. *IEEE Transactions on Computers*, 55(1):2–17, 2006.
[13] M. Deshpande and G. Karypis. Selective Markov models for predicting Web page accesses. *ACM Transactions on Internet Technology*, 4(2):163–184, May 2004.
[14] C.C. Kaya, G. Zhang, Y. Tan, V.S. Mookerjee, An admission-control technique for delay reduction in proxy caching, Decision Support Systems 46 (2009) 594–603.
[15] W. Ali, S.M. Shamsuddin, A.S. Ismail, A survey of Web caching and prefetching, International Journal of Advances in Soft Computing and Its Applications 3(2011) 18.
[16] W. Ali, S. Shamsuddin, Intelligent client-side web caching scheme based on least recently used algorithm and neuro-fuzzy system, in: W. Yu, H. He, N. Zhang (Eds.), Advances in Neural Networks — ISNN 2009, Springer, Berlin/Heidelberg, 2009, pp. 70–79.
[17] J. Cobb, H. ElAarag,Web proxy cache replacement scheme based on back-propagation neural network, Journal of Systems and Software 81 (2008) 1539–1558.
[18] S. Romano, H. ElAarag, A neural network proxy cache replacement strategy and its implementation in the Squid proxy server, Neural Computing and Applications 20 (2011) 59–78.
[19] Farhan, Intelligent Web Caching Architecture, Faculty of Computer Science and Information System, UTM University, Johor, Malaysia, 2007.
[20] A.P. Foong, H. Yu-Hen, D.M. Heisey, Logistic regression in an adaptive Web cache, IEEE Internet Computing 3 (1999) 27–36.
[21] Monti Babulal Pal," Enhancing the Web Pre-Fetching at Proxy Server using Clustering," International Journal of Technology Research and Management ISSN (Online): 2348-9006 Vol 1 Issue 1 March 2014
[22] Mohammad Ashraful Hoque,"Poster: Extremely Parallel Resource PreFetching for Energy Optimized Mobile Web Browsing" research gate CONFERENCE PAPER ·SEPTEMBER 2015

Authors' Profiles

Arvind Panwar , working as Assistant Professor in Northern India Engineering College, Delhi (Affiliated to Guru Gobind Singh Indraprastha University, Delhi) in the Information Technology Department. He received his M.Tech(Computer Science and Technology) & B.Tech(Computer Science) from the Guru Gobind Singh Indraprastha University, Delhi. He has rich experience teaching B.Tech students and has published more than 8 Research Papers in International Journals and Conferences. His area of interest includes Web Usage Mining.

Achin Jain, working as Assistant Professor in Bharati Vidyapeeth College of Engineering, New Delhi (Affiliated to Guru Gobind Singh Indraprastha University, Delhi) in the Information Technology Department. He received his M.Tech(Computer Science and Technology) & B.Tech(Information Technology) from the Guru Gobind Singh Indraprastha University, Delhi. He has rich experience teaching B.Tech students and has published more than 6 Research Papers in International Journals and Conferences. His area of interest includes Web Usage Mining, Web Attacks.

Manish Kumar, working as Assistant Professor in Bharati Vidyapeeth Institute of Computer Application and Management, New Delhi (Affiliated to Guru Gobind Singh Indraprastha University, Delhi) in the Information Technology Department. He received his M.Tech(Computer Science and Technology) degree from the Guru Gobind Singh Indraprastha University, Delhi. His area of interest includes Web Mining, VANET, Ad hoc Networks.

How to cite this paper: Arvind Panwar, Achin Jain, Manish Kumar,"A Novel Probability based Approach for Optimized Prefetching", International Journal of Information Engineering and Electronic Business(IJIEEB), Vol.8, No.5, pp.60-67, 2016. DOI: 10.5815/ijieeb.2016.05.08

