A Framework for Predictive Web Prefetching at the Proxy Level using Data Mining

Jyoti Pandey[1] , Amit Goel[2] , Dr. A K Sharma[3]

[3] Prof. and Head, [1, 2] Lecturer
Deptt. Of Computer Engineering,
YMCA Institute of Engineering, Faridabad, India

Abstract
The intelligence of adapting to the needs of specific users has become a potential research area for web technologies amidst the problems like heterogeneous network connectivity, real-world distances, and congestion due to unexpected network demand. Web caching, one such adaptations, is a technology that helps to reduce the network usage and server loads and improve average latencies experienced by the user. Web latency would be further significantly reduced if proxy or Web server softwares could make predictions about the pages that a user is most likely to request next, while the user is viewing the current page, and prefetch their content. This technique is known as predictive web preftching. Both these techniques, along with the use of data mining mechanisms have been used in this paper for optimizing the access time for accessing web pages from a web server. In the proxy server, the proposed data mining based service would run in background mode. The service computes the web pages likely to be requested by the user, considering their past web access log history, using association rules and thus optimizing the access time.

1. Introduction

Owing to increasing network bandwidth and computing power, the usage of internet has grown at the breakneck rate especially in the area of videoconferencing, gaming and online education. Over the years, at the same time, there has been a significant increase in the number of users who access Internet through high speed DSL; but still the access latencies

perceived by them is high. Therefore, reducing significantly web latency assumes importance for the Internet service providers who desire to increase the web surfing speed.

Web latency can be reduced either by pushing the bandwidth further though at the expense of higher costs or by implementing better technological solutions like introduction of caching. Caching is the automatic creation of temporary copies of information residing on computers other than host servers in order to make this information readily available to people around the world. Caching prevents traffic jams when users wish to access a web site by enabling the temporary storage of digitized materials closer to the user. Otherwise, users have to go all the way to the publisher's server to get the content that he or she wants.

Caching can be introduced at the various levels such as client, proxy, and server [2, 3 and 4]. Effective client and proxy caches reduce the *client perceived latency*. In fact, the required page desired by the user, if had been fetched earlier is made available from the local cache instead of importing it from the web server. This further reduces the server load and the number of packets traveling across the network, thereby reducing the traffic. However, the benefits gained through caching can be limited when the web resources tend to change very frequently e.g. when a web-site contains dynamic pages then even if its local copy is available in the cache, it would be of no use to the clients.

Prefetching of web pages is another potential research area that can significantly reduce the web access latency. It refers to the process of deducing a client's future requests for web objects and getting those objects into the cache before it is explicitly requested. The major advantage of using prefetching is that it prevents bandwidth under-utilization. However without a carefully designed prefetching scheme, it may happen that several already transferred web pages might never be requested by the client. This would result in the bandwidth wastage.

In this paper, a web prefetching framework has been proposed that employs data mining techniques to predict web pages likely to be accessed by the user especially in an environment where users access the Internet through proxy servers. Moreover, it fetches the predicted web pages from web servers and loads them into the proxy server's cache while the user may be busy in performing some other tasks. Therefore, subsequently whenever the user desires a web page then if prefetched, the web page is delivered to the user immediately.

1.1 Related Work

The existing predictive prefetching algorithm can be categorized into two families: Dependency Graph (DG) and Prediction by Partial Match (PPM). DG holds the patterns of accesses [7, 8] while PPM adopts a scheme from the text compression domain [5, 6, and 9].

Manuscript received June 5, 2008.
Manuscript revised June 20, 2008.

Both these methods suffer from two major drawbacks, i.e. they both consider only single past visit of the user and they do not consider patterns corresponding to higher dependencies.

These two drawbacks have been overcome by the higher order *PPM* algorithms. These algorithms make use of *constant maximum value* for considering orders of the patterns. However, fixing this value involves a serious drawback. To calculate the patterns from the transaction logs to the orders determined by this *constant maximum value* requires the maintenance of the large set of rules and this involves high costs. Moreover palpanas and mendelzon [5, 6] does not provide any mechanism for the determination of this maximum value.

Cooley et al [11] defined web usage mining as the process of using data mining techniques on the data requested by the users from the www.

Klemm et al [10], support the fact that there is no need to make any modifications either in the current existing web protocols or in the existing web browsers, if the prefetching module runs as a proxy in the browsers.

Krishnan et al [12] introduced the use of path profiles for predicting HTTP requests. These requests can be used in predicting a user's next step. For example, upon learning that the majority of users go to page *c* after going from page *a* to page *b*, an intelligent Web designer may find it beneficial, for both the user and the server, to place a direct link from page *a* to page *c*.

Jacobson et al [6] present a technique called proxy initiated prefetching and a prediction algorithm based on PPM whereby its implementation forces the modification of the browsers at the client-side.

Nanopoulas et al [13] also introduced the prefetching schemas wherein they used the association rules for predicting user access sending hints from the server side to the clients.

Pei et al [14] talk about the strong regularities in the www surfing done by the user.

Chen, Cooley and Pie [15, 16 and 17] provide various data mining algorithms for the path traversal patterns showing how the data is to be prepared before mining data from www and how to efficiently mine the access patterns from the web logs.

Manopaulos et al [24] describe the methodology for mining patterns from the graph traversals.

A critical look at the available literature indicates that several efforts have been made in isolation in the areas of prefetching and caching. In this paper, a framework has been proposed which clubs the fruits of both the caching and prefetching at one place. Further, to extract the benefits to the larger extent, the framework is proposed to be implemented at the proxy server level. Apart from predicting the user's behavior for the next page's access, the focus of this framework will also be on improving the relevance of the requested pages. For this, Pandey et al [20] introduced a new methodology based on the page rank algorithm and the association rules to improve the relevance of the web pages that are extracted from the world wide web.

2. Proposed Architecture

Internet is a client server architecture wherein a client sends his request for a resource over the www to a server. The server responds by serving the request. The session involves the exchange of messages and protocols. However, due to exponential increase of www, there are a large number of clients that interact with servers through millions of networks connected with each other leading to a significant increase in the www latency and traffic on the net.

Since a proxy server sits between a Web browser and a web server, it is a potential tool that can be suitably employed to reduce the www latency i.e. it can intercepts all requests to the web server to see if it can fulfill the requests by itself. If not, then only it may forward the request to the web server. In fact, the proxy servers can be employed to achieve two main purposes:

Reduce latency: A Proxy server saves the results of all the requests from various clients for a certain amount of time. For instance, consider a case where both users X and Y access the www through a proxy server. Let us assume that user X requests for a certain web page say Page 1. Sometime later, user Y also requests the same page. Instead of forwarding the request to the web server where page 1 actually resides, which can be a time-consuming operation, the proxy server simply returns this page from its cache where all the downloaded pages are retained before being over written by new arrivals. Since proxy server is often on the same network as the user, this is a much faster operation, thereby reducing the perceived latency to some extent.

Filter Unwanted Requests: Proxy servers can also be used to filter unwanted requests. For example, a company might use a proxy server to prevent its employees from accessing a specific set of Web sites.

The www latency can be further reduced if the behavior of the user can be predicted and accordingly the predicted pages are prefetched and stored temporarily in the cache of the proxy server. As soon as the user asks for a page, the request can be fulfilled if the requested page is available in the cache.

In this work, a prediction engine called Prediction Prefetching Engine (PPE) has been proposed that processes the past references to deduce the probability of future access for the documents accessed so far. In fact, it resides on the proxy server as shown in Fig. 1. The component wise working of the various components of PPE is given below:

2.1 Proxy server log

The records of all the users/clients that send requests to the server are kept on the web logs which are used to form the mineable warehouse [17].

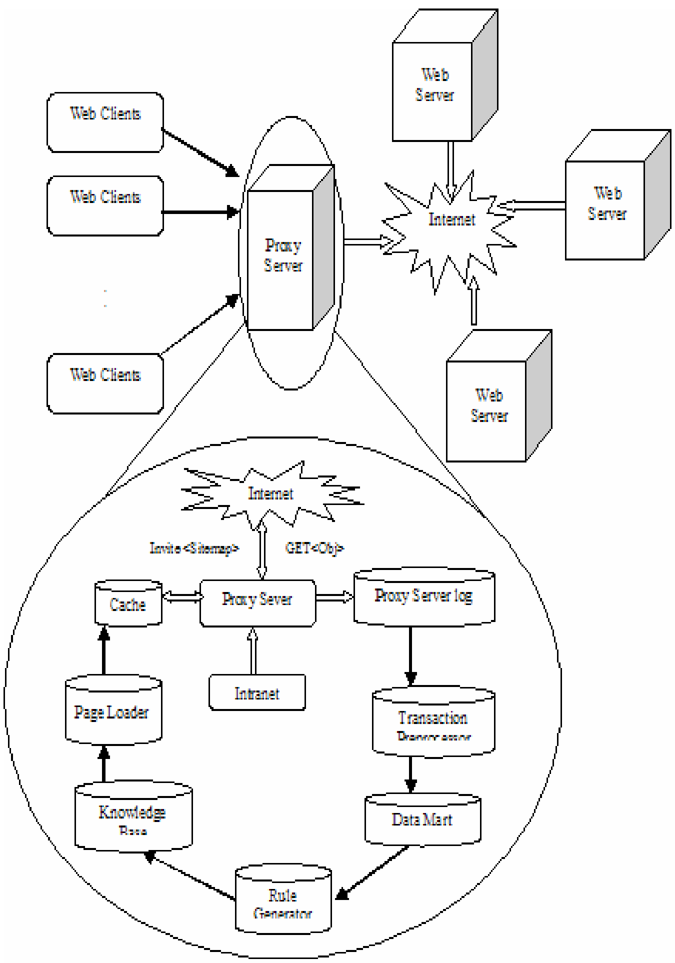

Fig.1 Framework for prefetching the documents at the proxy side

These warehouses are used to track the user activity. The user activity is tracked and recorded in these warehouses in the form of logs maintained by the proxy server. Since a proxy server sits between the client and the web server, it is comparatively easy to manage than the web server logs especially when it has to maintain the log for limited clients.

2.2 Transaction Preprocessor

Since in the proposed work, the log is being maintained on the proxy side, the transaction preprocessor operates on these proxy logs to accomplish preprocessing.
Preprocessing involves the following tasks:

Reduction of Search Space: The foremost is to reduce the search space for mining which is done by cleaning the proxy log for any unwanted records. It may include clearing the log from the irrelevant items like the image files (GIF and JPEG) and java script files (JS) etc as these do not contribute for the patterns relevance.

User and Session Identification: A user session is defined as the sequence of requests made by the single end user during a visit to a particular site. Within a single session, a user may follow links to several pages that belong to the similar pattern but during the same session, it may also be possible that the user might visit some other pages that do not belong to the same pattern i.e. user session may contain the documents belonging to patterns while others that do not and the documents are interleaved in the session.

Path completion: It forms the next step of preprocessing within a session. It is necessary as it may happen that some of the important accesses are not recorded in the log due to use of cache. After all the preprocessing is done, the cleaner version of the proxy log is formed called Data mart.

There are few other parameters like the length of user access sequences, the dependencies between the accesses, and the existence of a page accesses inside the sequences that do not belong to patterns but on the contrary depend on the contents of the document or the structure of the web site. In small web sites, this impact may be small because of the limited navigational alternatives. But in large web sites, this impact is quite significant.

2.3 Data Mart

Data mart acts as a database on which various data mining operations [ ] operate for generating the rules.

2.4 Rule Generator

It extracts the information from the data mart and applies the various data mining operations e.g. association rules, sequential patterns etc. to generate the rules for prediction.

2.5 Knowledge Base (KB)

The various rules formed by the rule generator forms the part of this base.

2.6 Page Loader

For a given request made by the user, page loader consults the rules of the KB and if the user's requested pages exist in the heads of the rules, then the pages present in the body of those particular rules are prefetched. For instance, the k^{th} entry in the knowledge base may have the following format:

$\mathbf{R_k}$: $\mathbf{D_i \Rightarrow D_j}$; if document D_i has been requested then prefetch document D_j.

Similarly, n^{th} entry in the knowledge base may have the following format:

$\mathbf{R_n}$: $\mathbf{D_j \Rightarrow D_k}$; if document D_j has been fetched then prefetch document D_k also.

This method follows the forward chaining in the knowledge base till the time no more rules can be fired. To prevent increase in the network traffic due to the chaining-activation process, all the prefetched documents are stored in the queue maintained in proxy cache as the DocRefQueue.

3. Transaction Processing Phases

The overall transaction processing can be broadly classified into three main phases as shown in Fig 2.

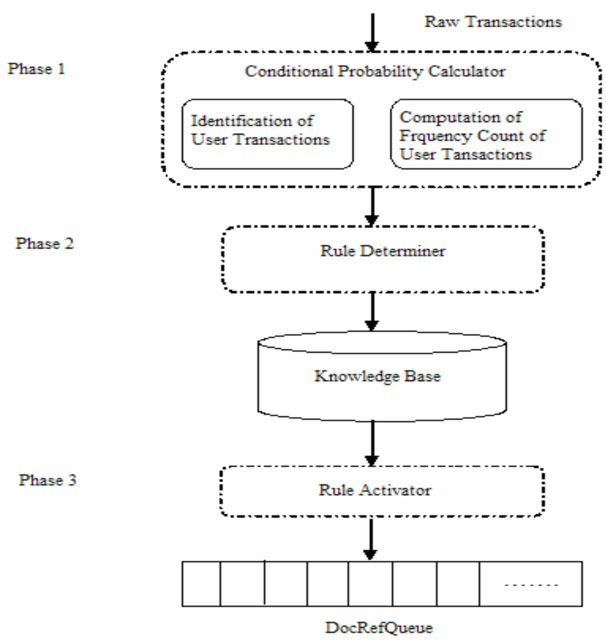

Fig. 2 Transaction Processing Phases

The step wise working of these phases is as follows:

3.1 Conditional Probability Calculator

In this phase the conditional probabilities of the user accesses are calculated. To perform this task, two subtasks need to be performed. They are the identification of the user transactions from the proxy server log and the computation of the frequency count.

a) Identification of the user transactions: The foremost thing for the determination of the user transactions is the identification of the user sessions from the log file. In the proposed framework, this work will be done by the transaction preprocessor component as shown in Fig. 1. The objective of user session is to separate independent accesses made by different users or by the same user at distant points in time [15, 16]. These user sessions are then decomposed into a number of maximal forward references, using MF algorithm described in [15]. This algorithm filters out the effect of backward references which are made only for navigational purposes.

b) Computation of frequency count: In Data Mining notation, given the set of user transactions, the frequency of an access sequence S is equal to the number of transactions T, for which S is contained in T or in other words, if S is the subsequence of T. The term frequency is also referred to support in data mining context. For the purpose of frequency counting, each user transaction is read and the frequency/support of each contained subsequence is increased by one [20]. Once, all the transactions are read, the calculation of corresponding probabilities and the formation of rules can be accomplished. The probability of an access sequence S (P(S)) is equal to frequency of the access sequence (fr(S)) divided by the total number of transactions, where fr(S) denotes the number of occurrences of S, i.e. its frequency.

3.2 Rule Determiner

Once the conditional probability of the user's accesses is calculated, the next step is to determine the rules. These rules will let know which pages are to be prefetched. To determine the rules, markov predictors will be used. E.g. if $S = \langle p_1, \ldots, p_n \rangle$ is a sequence of accesses (called a transaction) made by a user, then the conditional probability that the next access will be p_{n+1} is $P(p_{n+1} \mid p_1, \ldots, p_n)$. Therefore, given a set of transactions, rules of the form:

$$p_1, \ldots, p_n \Rightarrow p_{n+1} \quad (1)$$

(1) can be derived, where $P(p_{n+1} \mid p_1, \ldots, p_n)$ is equal to or larger than the user defined cut-off threshold value T_c. The left part of the rule is called the *head* and the right part is called the *body*. The body of the rule can also be any length larger than one. E.g. rules of the form

$$p_1, \ldots, p_n \Rightarrow p_{n+1}, \ldots, p_{n+m} \quad (2)$$

In this case, $P(p_{n+1},\ldots, p_{n+m} \mid p_1, \ldots, p_n)$, has to be larger than T_c.

The dependency of the forthcoming accesses on past accesses defines a *Markov Chain.* The number of past accesses considered in each rule for the calculation of the corresponding conditional probability is called the order of the rule. E.g. the order of the rule *A, B=>C* is 2.

The predictive web prefetching algorithm can be defined as a collection of 1, 2, …, n-order Markov Predictors. An n-order Markov predictor is defined to be a scheme for the calculation of conditional probabilities $P(p_{n+1},\ldots, p_{n+m} \mid p_1, \ldots, p_n)$ between document accesses and the determination of the rules of the form (2). The head of the each rule has a size equal to n and the body of each rule has the size equal to m.

The job of determining the rules is performed by the *rule generator* component which are then stored in the *knowledge base* component of the proposed framework as shown in Fig.1.

3.3 Rule Activator

After the determination of the rules of the form (2), the next requirement is for the activation mechanism. The job of the rule activator is to find the prefetched pages from the corresponding rules. It will match the user's request for the documents with the heads of the rules. If the suitable match is found, it will prefetch the documents found in the tail of the corresponding rule. This task is done by the page loader component of the proposed framework as shown in Fig. 1. The prefetched documents are stored in the DocRefQueue which is placed in the cache of the proxy server.

4. Conclusion

The Prefetching scheme used in the framework of Fig.1 makes use of Markov Predictors of higher orders in addition to that of first order. This means that they will allow for rules whose both head size and body size should be greater than one. The maximum order depends on the way users navigate and on the site's characteristics. Thus, it should vary and should not be considered as a constant value as it is taken in high-order PPM. Hence, a Prefetching scheme should be able to adaptively select the appropriate maximum value for the order which is tried in this proposed work.

References

[1] Stefan Saroju, Krishna P. Gommadi, Richard J. Dunn,Steven D. Gribble, and Henry M. Levy, "An analysis of intenet content delievery systems," *in Proceedings of the 5th sumposium on Operating Systems Design and Implementation*, Boston, USA, 2002

[2] J. Han and M. Kamber. *Data Mining, Concepts and techniques.* Morgan Kaufmann publishers, 2001

[3] P. Cao and S.Irani, "Cost Aware WWW proxy cachingAlgorithms," *Proc. 1997USENIX Symp. Internet Technologies and Systems(USITS '97),* pp. 193-206, jan 1997

[4] J.Shim, P. Scheuermann, and R. Vingralek, "Proxy Cache Algorithms: Design, Implementation, and Performance," *IEEE Trans. Knowledge and Data Eng.,* Vol. 11, Aug 1999.

[5] T. Palpanas and A. Mendelzon, "Web Prefetching using Partial Match prediction", *Proc. 4th Web Caching Workshop(WCW '99),* Mar 1999.

[6] L. Fan, P. Cao, W. Lin, and Q. Jacobson, "Web Prefetching between Low-Bandwidth Clients and Proxies: Potential and Performance", *Proc. ACM conf. Measurement and Modelling of Computer Systems,* June 1999

[7] J. Griffioen and R. Appleton, "Reducing File System latency using a predictive approach", *Proc. 1994USENIX Annual Technical Conf. ,* Jan 1995

[8] V. Padmanabhan and J. Mogul, 'Using Predictive prefetching to improve world wide web latency", *ACM SIGCOMM Computer Comm. Rev. ,Vol. 26,no.3,* July 1996

[9] K.M. Curewitz, P. Krishnan, and J.S. Vitter," Practical Prefetching via Data Compression", *Proc. ACM Conf. Management of Data (ACM SIGMOD '93),* June 1993

[10] R. Klemm, "WebCompanion: A friendly Client Side Prefetching Agent", *IEEE Trans. Knowledge and Data Eng., Vol.11,* no.4, July/August 1999

[11] B. Mobasher, R. Cooley, and J. Srivastava, " Automatic Personalization based on Web Usage Mining", *Communications of ACM, 43(8),pages 142-151,* 2000

[12] S. Schechter, M. Krishnan, and M.Smith, "Using Path Profiles to predict HTTP requests", *in the 7th World Wide Web Conf.,* pages 457-467, Brisbane, Australia, 1998

[13] A. Nanopoulas, D. Katsaros and Y. Manolopaulos, "Effective prediction of Web User accesses", *Proc. of WEBKDD '01,* San Francisco, CA, 2001

[14] B. Huberman, P. Pirolli, J. Pitkov, and R. Lukose, "Strong Regularities in World Wide Web surfing", *Science,Vol. 280,*pp. 95-97, Apr. 1998

[15] M.S. Chen, J.S. Park, and P.S. Yu," Efficient Data Mining for Path Traversal Patterns," *IEEE Trans. Knowledge and Eng., Vol.10,* n0.2, pp.209-221, Apr. 1998

[16] R. Cooley, B. Mobasher and J.Srivastva, "Data Preparation for mining World Wide Web Browsing Patterns,"*Knowledge and information Systems(KAIS), Vol. 1, no.1,* pp. 5-32, Feb. 1999

[17] J. Pei, J.Han, B. Mortazavi, and H. Zhu, "Mining Access Patterns efficiently from Web Logs," *Proc. Pacific- Asia Conf. Knowledge discovery and Data Mining(PAKDD'00),* Apr. 2000

[18] A. Nanopoulas, and Y. Manolopaulos," Finding Generalized Path patterns for web Log Data mining," *Proc. East European Conf. Advances in databases and information systems,* Sept. 2000

[19] B.Lan, S. Bressan, B.C. Ooi, and Y. Tay, "Making web servers pushier," *Proc. Workshop Web Usage Analysis and User Profiling,* 1999.

[20] J. Pandey, Ashok, Kumar, A. Goel, “A novel approach for Extracting Relevant Web pages from WWW using Data Mining,” *Proc. Common Ground Publishers,* Australia, 2006

[21] R. Aggarwal, and R. Srikant, “Fast Algorithms for Mining Association Rules”, *Proc. 20th Conf. Very Larger Databases (VLDB’94),* Sept. 1994

[22] H. Mannila, H. Toivonen, and I. Verkamo, “Discovery of Frequent Episodes in Event Sequences”, *Data mining and Knowledge Discovery,* Sept. 1997

[23] R. Aggarwal, and R. Srikant, “Mining Sequential Patterns”, *Proc. IEEE conf. Data Eng.,* Mar. 1995

[24] A. Nanopoulas, and Y. Manolopaulos,” Mining patterns from Graph traversals,” *Proc. Of Data and Knowledge Eng.,* June 2001

[25] O. R. Zaïane, *Web Usage Mining for a Better Web-Based Learning Environment*. Department of Computing Science University of Alberta Edmonton, Alberta, Canada, 2001.

[26] R. Cooley, B. Mobasher, and J. Srivastava, *Data Preparation for Mining World Wide Web Browsing Patterns*. Journal of Knowledge and Information Systems, (1), 1999.