IRACST - International Journal of Computer Science and Information Technology & Security (IJCSITS), ISSN: 2249-9555
Vol. 2, No.6, December 2012

Web Page Prefetching Mechanism: A Study

Meghna Khatri
Deptt. Of Computer Science Engineering
U.I.E.T., M.D.U.
Rohtak, India

Abstract— Web users are facing the problems of information overload and drowning due to the significant and rapid growth in the amount of information and the number of users. Prefetching techniques try to predict the next set of files/pages that will be requested, and use this information to pre-fetch the files/pages into the server cache. This greatly speeds up access to those files, and improves the users' experience. In this paper, a survey of the web page prefetching mechanism is provided.
Keywords- prefetching mechanism, web page.

I. INTRODUCTION *(HEADING 1)*

It is indisputable that the recent explosion of the World Wide Web has transformed our way of doing work. The single most important piece of software that enables any kind of Web activity is the Web server. Since its inception the Web server has always taken a form of a daemon process. If the World Wide Web is to be approached from a client-server view then, as the name suggests, Web server is the server part of the scheme and a browser is the client. In a typical interaction a user will request a file from a server either by clicking on a link or typing the request in manually. The browser translates it into an HTTP request, connects to the proper server, sends the request and waits for a reply. Meanwhile the Web server has been waiting for requests. It accepts the connection from the client, parses the HTTP request and extracts the name of the file. The server then gets the file from its cache or from its disk, formats an HTTP reply that satisfies the request and sends it to the browser. The browser then closes the connection. Access to disk is much slower than access to memory. Just as in the case of OS file systems, caching techniques are used in Web servers to reduce disk accesses. One difference is that Web server file accesses are all reads due to the nature of the application. In this context the cache is a collection of files that logically belong on the disk but are kept in memory for performance reasons. Great efforts are being made to address these problems and improve Web performance.

A popular technique to reduce web latency is web page prefetching. Web Pre-fetching, which can be considered as "active" caching, builds on regular Web caching and helps to overcome its inherent limitation. It attempts to guess what the next requested page will be. For regular HTML file accesses, pre-fetching techniques try to predict the next set of files/pages that will be requested, and use this information to pre-fetch the files/pages into the server cache. This greatly speeds up access to those files, and improves the users' experience. To be effective however, the pre-fetching techniques must be able to reasonably predict (with minimum computational overheads) subsequent web accesses.

Web pre-fetching builds on web caching to improve the file access time at web servers. The memory hierarchy made possible by caches helps to improve HTML page access time by significantly lowering average memory/disk access time. However, cache misses can reduce the effectiveness of the cache and increase this average time. Pre-fetching attempts to transfer data to the cache before it is asked for, thus lowering the cache misses even further. Pre-fetching techniques can only be useful if they can predict accesses with reasonable accuracy and if they do not represent a significant computational load at the server.

II. BACKGROUND

Web Pre-fetching, which can be considered as "active" caching, builds on regular Web caching and helps to overcome its inherent limitation. It attempts to guess what the next requested page will be. For regular HTML file accesses, pre-fetching techniques try to predict the next set of files/pages that will be requested, and use this information to pre-fetch the files/pages into the server cache. This greatly speeds up access to those files, and improves the users' experience. To be effective however, the pre-fetching techniques must be able to reasonably predict (with minimum computational overheads) subsequent web accesses.

This section introduces current techniques for web page pre-fetching mechanism. Existing pre-fetching approaches can be classified as client-side, proxy-based or server side.

A. Client-Side Prefetching

In the client-side approach, the client determines pages to be prefetched and request them from the server. A key drawback of this approach is that it typically requires modifications to the client browser code or use of a plug-in, which may be impractical. Furthermore, it may double the required bandwidth, actually resulting in deteriorated performance. For example, in the worst-case, the pre-fetcher will repeatedly request files that the user never wants to see. Therefore, the number of requests to the server will double without any benefit to the user. Finally, maintaining cache coherency in client-side pre-fetching approaches is expensive. Cache coherency deals with the following issue. If a file in cache has changed on the server the new version of the file needs to be presented to the user instead of the stale cached version. This requires checking with the server on the state of the file(s) in the cache (possibly through a special protocol). As a result there is increased complexity on the client and the server side, as well as increased traffic between the two.

B. Proxy Prefetching

The proxy-based prefetching approach uses an intermediate cache between the server and a client . This proxy can request files to be prefetched from the server, or the server can push some files to the proxy. Both of these schemes increase the required bandwidth. Furthermore, like client-side schemes, maintaining cache coherency in proxy-based schemes is expensive. This overhead gets even more significant when multiple levels of proxy caches are employed.

One advantage of client and proxy side prefetching is that they separate the HTTP server part from the caching part thus allowing greater geographic and IP proximity to the client. For example, placing a proxy cache next to or inside of an organization's subnet means that the data a user requests will have far fewer IP hops. These schemes are also better suited for user-pattern tracking algorithms. In particular, the client-side mechanism is dedicated to a particular user and spends all its time trying to follow what the user might want. By the same token a proxy cache dedicated to a particular organization will do a good job following that organization's preferences. Another advantage is that requests from multiple servers can be cached.

C. Server-Side Prefetching

In server-side approaches, the entire prefetching mechanism resides on the Web server itself. These approaches avoid the problems mentioned above. There is no increase in the bandwidth, as no files that haven't been requested will be sent to the client. Furthermore, maintaining cache coherency in this case is almost effortless. Proxy-based caches and client-side prefetching mechanisms require additional messaging and protocols between the cache and the HTTP server for cache coherency. This overhead can become expensive in terms of wasted bandwidth. There is no complicated protocol and no extra messaging outside the server in case of server-side schemes.

As the file system in this case is either local or mounted, all the messaging is within the server and does not require external bandwidth. Furthermore, the OS file system guarantees access to the latest copy of a file, and provides excellent and easy to use mechanisms to check file attributes such as creation and modification times and dates, to assist in maintaining cache coherency. Another distinction with the client-related schemes is that client-side prefetching makes

Identify applicable sponsor/s here. *(sponsors)*

decisions on which files to prefetch based on the particular user's preferences, whereas in the server-side prefetching, decisions are based on the document popularity, and more than one client can benefit from it.

A server-side prefetching approach based on analyzing server logs and predicting user actions on the server side is presented by Su et. al. [1]. Tracking users on a server, however, is quickly becoming impractical due to the widespread use of web proxies. The proxy either presents one IP address to the server for a large group of users, or it cycle through some set of IP addresses according to its load-balancing scheme. Both cases render a single user identity moot.

III. LITERATURE REVIEW

A. *Study of Prefetching Strategies*

This section corresponds to the study of various strategies involved till now for prefetching web pages.

Jose Borges and Mark Levene[2] propose a dynamic clustering-based method to increase a Markov models accuracy in representing a collection of user web navigation sessions. The method makes use of the state cloning concept to duplicate states in a way that separates in-links whose corresponding second-order probabilities diverge. In addition, the new method incorporates a clustering technique which determines ancient way to assign in-links with similar second-order probabilities to the same clone.

Siriporn Chimphlee [3]present a rough set clustering to cluster web transactions from web access logs and using Markov model for next access prediction. Using this approach, users can effectively mine web log records to discover and predict access patterns. He performs experiments using real web trace logs collected from .the servers. In order to improve its prediction ratio, the model includes a rough sets scheme in which search similarity measure to compute the similarity between two sequences using upper approximation.

Silky Makker and R.K Rathy[4]proposes a bracing approach for increasing web server performance by analyzing user behavior, in this pre-fetching and prediction is done by pre-processing the user access log and integrating the three techniques i.e. Clustering, Markov model and association rules which achieves better web page access prediction accuracy; This work also overcomes the limitation of path completion i.e. by extracting web site structure paths are completed, which helps in better prediction, decreasing access time of user and improving web performance.

B. *Study of Different Models for Prefetching*

This section corresponds to the study of various algorithm which has been used in the various stages of web page prediction process i.e. V. Padbanabham and J. Mogul [5] use N-hop Markov models predicted the next web page users will most likely access by matching the user's current access sequence with the user's historical web access sequences for improving prefetching strategies for web caches.

R.R. Sarukkai [6] used first-order Markov models to model the sequence of pages requested by a user for predicting the next page accessed. A "personalized" Markov model is trained for each individual and used for predictions in user's future request sessions. In practice, however, it is very expensive to construct a unique model for each user respectively, and the problem gets even worse when there exist thousands of different users within a big Web site.

F. Khalil[7] introduces the Integration Prediction Model (IPM) by combining Markov model, Association rules and clustering algorithm together. Then, the prediction is performed on the cluster sets rather than the actual sessions. The IPM integration model is based on the different constraint. The web user sessions first are divided into a number of clusters using k-means clustering algorithm and cosine distance measure. Then, an integration model computes Markov model prediction on the resulting clusters. This

algorithm improves the state space complexity because Markov model prediction carried out on the particular clusters as opposed to the whole data set. In the case of state absence in the training data or where, the state prediction probability is not marginal, Association rules are examined more states than Markov model by looking at more history. Lastly, if a new page is presented, the cosine distance is calculated and identifies an appropriate cluster that a new web page should belong to. The integration model has been proved through the experiments that improve the prediction accuracy. Moreover, implementing the prediction model on the clusters achieves better results than on the non-clustered data. Although, a web page access prediction performance was improved, however, it can be seen that their integrated algorithm has a complicated procedures and must repeatedly employ in order to increase their prediction performance.

S. Chimphlee [3] presented a Hybrid Markov Fuzzy Models (HyMFM) that are obtained by integrating the advantages of all three prediction models: Markov model, Association rules and Fuzzy Adaptive Resonance Theory (Fuzzy ART). HyMFM algorithm was developed for the web user sessions clustering by proposing the new sequence representations and the new similarity measures in incremental learning of Fuzzy ART control structure. A web user session was represented into the transition matrix representation, referred to as session matrix, which is constructed based on a transition matrix of a first Hybrid Markov model. Both elements fit well into the design of this thesis and the clustering task which the web user sessions are treated as order sets of accesses. Consequently, the new similarity measures were developed to enable the application of Fuzzy ART clustering. This study defined two new similarity measures: Matrix norm similarity and Matrix distance similarity. These measures alleviate the overestimation problem in Fuzzy ART algorithm which use the city-block distance metric as the similarity between input and prototypes. Thus, the web user sessions were clustered into groups with similar patterns in during the training phase and when it is confronted by a new input, it produces a response that indicates which cluster the pattern belongs to and then HyMM applied to each cluster.

IV. CONCLUSION

Web page prefetching reduces users' perceived latency but it also increases network traffic. Though pre-fetching adds no extra traffic to network but sometimes burstiness of individual sources is increased, leading to increased average queue sizes in network switches. Some other negative network effects due to pre-fetching include unknown cache-ability, server overhead, side-effects of retrieval and user activity conflation. All these give way to improved in this approach and scope of more research in this field.

REFERENCES

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] was the first . . ."

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [4]. Papers that have been accepted for publication should be cited as "in press" [5]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [6].

[1] Brian Amento, Loren Terveen, and Will Hill." Does "authority" mean quality? Predicting expert quality ratings of web documents". In the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pp. 296–303, 2000.

[2] J. Borges and M. Levene, " A clustering based approach for modeling user navigation with increased accuracy" Proceedings of the Second International Workshop on Knowledge Discovery from Data Streams

(IWKDDS) in conjunction with PKDD 2005, Porto, Portugal, Outubro,2005

[3] Siriporn Chimphlee, Naomie Salim, Mohd Salihin Bin Ngadiman, Witcha Chimphlee, Surat Srinoy," Rough Sets Clustering and Markov model for Web Access Prediction",2006.

[4] Silky Makkar and R. K. Rathy,"Web Server Performance Optimization Using Prediction Prefetching Engine"International Journal of Computer Applications, Volume 23– No.9, June 2011.

[5] 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

[6] R. R. Sarukkai, "Link prediction and path analysis using Markov chain",proc. of the 9th International World Wide Web Conference on Computer networks, 2000.

[7] F. Khalil, J. Li and H. Wang," Integrating recommendation models for improved web page prediction accuracy". Proceedings of the 31th Australasian Computer Science Conference, (ACSC'08), Wollongong, NSW, pp: 91-100,2008.