Proceedings of the 2nd ACM Workshop on Performance and Architecture of Web Servers, (PAWS-2001)

Coordinated Data Prefetching by Utilizing Reference Information at Both Proxy and Web Servers *

Xin Chen and Xiaodong Zhang
Department of Computer Science
College of William and Mary
Williamsburg, VA 23187-8795
{xinchen, zhang}@cs.wm.edu

Abstract

Existing prefetching techniques rely on server-based, proxy-based, or client-based reference access information. Although Web servers may provide accurate access information, our studies show that significant communication overhead can be involved by sending unnecessary reference information to clients or/and proxy servers. Our study also shows that prediction accuracy of proxy-based prefetching can be significantly limited without input of Web servers. We propose a *coordinated proxy-server prefetching technique* that adaptively utilizes the reference information and coordinates prefetching activities at both proxy and web servers. We have carefully considered the trade-off between the accuracy of server information and the high cost of obtaining the information in prefetching method designs. In our design, the reference access information stored in proxy servers will be the main source serving data prefetching for groups of clients, each of whom shares the common surfing interests. The access information in the web server will be used to serve data prefetching only for data objects that are not qualified for proxy-based prefetching.

Conducting trace-driven simulations, we have quantified the limits of both server-based and proxy-based prefetching, and evaluated the proposed coordinated proxy-server prefetching technique to show its effectiveness. We show that both hit ratios and byte hit ratios contributed from coordinated proxy-server prefetching are are up to 88% higher than that from proxy-based prefetching, and they are comparable to the ratios from server-based prefetching with a difference of 5%.

1 Introduction

Web prefetching based on the reference access information in Web servers and/or proxy servers is becoming important and demanding for two reasons. First, accessing proxy caches through a local area network is much faster than accessing web servers through global area networks. The study in [12] divides the total web access latencies into *internal latencies* in local area networks and *external latencies* in global area networks. Based on case studies, the authors show that the percentage of external latencies in total access latencies are dominantly high. Thus, an effective prefetching for clients based on reference access information can significantly reduce the external latencies.

Second, the number and types of web servers have increased and will continue to increase dramatically, providing services to a wider range of clients. Thus, the number of unique file objects cached in client browsers will continue to increase. The study in [2] gives a comprehensive evaluation of the changes in web client access patterns based on the traces collected from the same computing facility and the same nature of the user population separated by three years. Their study shows that the hit ratios are significantly reduced compared with the data three years ago. One reason for this, we believe, is that access variations have increased as more and more web servers are emerging. A recent study in [20] shows that increasing proxy cache size is no longer the only effective way for performance improvement, because it is mainly beneficial to the commonly shared data objects among clients. Prefetching popular objects for individual clients becomes as important as prefetching commonly shared data objects. To achieve this goal, access reference information stored in Web servers will be the main source for making prediction decisions.

Existing prefetching models are either server-based or proxy-based. In a server-based environment, prefetching decisions are made based on the reference access information provided by Web servers. Although Web servers may provide accurate access information, frequent communications between clients/proxies and servers are required, and unnecessary communications can be conducted. In a proxy-based environment, prefetching decisions are made based on the reference access information in proxy servers. Since proxy servers can only predict commonly shared data files among clients, the prediction accuracy can be significantly limited without the input of global Web servers.

We propose an efficient data prefetching technique, called *coordinated proxy-server prefetching*, that adaptively utilizes the reference information and coordinates prefetching activities at both proxy and web servers. We have carefully considered the trade-off between the accuracy of server information and the high cost of obtaining the information in prefetching method de-

*This work is supported in part by the National Science Foundation under grants CCR-9812187, EIA-9977030, and CCR-0098055.

signs. In our design, reference access information stored in proxy servers will be the main source serving data prefetching for sizable groups of clients, each of whom share the common surfing interests. Access information in the web server will be used to serve data prefetching only for the object files that are not qualified for proxy-based prefetching.

In this study, we will address the following three questions: (1) How important to data prefetching is the reference access information in Web servers? (2) Under what conditions is the prefetching accuracy of proxy-based technique sufficient? (3) Can we effectively integrate both server-based and proxy-based techniques to achieve the goal of retaining the prefetching accuracy of the server-based model and minimizing the communications between proxy and Web servers?

2 Evaluation Methodology

Our study and performance evaluation are based on trace-driven simulations. The evaluation environment consists of Web traces, a simulated Internet environment with clients, proxy servers, and Web servers. We will discuss the selected Web traces and our simulation model in this section.

2.1 A PPM structure for data prefetching

In data compression community, researchers have developed several context models to use m proceeding symbols to determine the probability of next symbol (an m order Markov model). Prediction by Partial Match (PPM) (see e.g. [5]) is one such context model that has been actively used in Web predictions. In a Web prediction model, multiple Markov models are used to store historical URL paths. Each Markov model may partially represent a client session. The PPM model structure is represented by a tree in which each Markov model with multiple URL predictors is a branch. The length of each branch varies. Variable orders can be used in each branch to make predictions.

There are two representative approaches to build a PPM prediction model. The first approach is to widely create branches of predictors from the historical URL files. The PPM tree is given a fixed height so that the length of each branch is not allowed to grow beyond the height. This fixed height is used to limit the number of predictors in order to save the storage space. The advantage of this approach is that the model can be easily and regularly built with a low complexity. This approach has been used in several prefetching prototypes and systems (see e.g. [18], [8]). We use this PPM structure in our study. The other approach is to reduce the space requirement by only storing long branches with frequently accessed URL predictors, which is called LRS (keeping the Longest Repeating Subsequences) [15].

The prefetching process is initiated after accessing a requested object at a level of the Internet caching system. The prefetching process searches the PPM tree rooted by this object in a proxy or in a server. Each object in a PPM tree is marked by a relative access probability that is defined as the ratio between the access frequency to this element and the frequency to the root object. For example, if the root object has been accessed 10 times, and an object in the PPM tree has been accessed 3 times, the relative access probability of this object is 30%. The relative access probability is a variable value to adjust the prediction accuracy in both proxy and Web servers. In our study, we set 30% as the pre-determined threshold for an object to be a qualifier for either proxy-based or server-based prefetching, and found it is a representative value. If no such objects in the PPM trees rooted by the requested object file is found in the proxy, the prefetching decision will be made by the Web server.

2.2 Traces

The Boeing Company has six proxy servers in its Puget Sound firewall. They collected anonymized logs from the proxies by using an anonymizer tool (log2anon) and made these logs available in [4]. We used the trace files of five days from March 1 to 5, 1999. The first 5 proxies are in a DNS round robin configuration for load balancing. The sixth is running a newer version of software from the others, and is used as a parent for internal proxy cache testing. Certain particular web sites and ftp traffic are directed to this proxy. Since we are interested in tracing accesses of each client, we have used the first 5 traces in our experiments. After sorting the 5 traces in its time order, we have merged 5 traces into a large file for each day. Table 1 presents the characteristics of the 5 merged Boeing traces for 5 days.

One challenge in our study is the lack of correlated server traces and proxy traces accessed by clients. In practice, the available traces are either proxy-based or server-based. Therefore, we have created pseudo server traces and scaled-down proxy traces from the obtained Boeing proxy traces. In order to provide a practical basis for using the pseudo server traces, we have compared the access patterns of these traces with standard server traces.

2.2.1 Scaled-down proxy traces

We have created multiple proxy trace files with different numbers of clients accessing the proxy. We call them scaled-down proxy traces. The numbers of clients we used are 50, 200, 500, 1,000, 5,000, and 10,000. The clients are randomly selected in the proxy file for each pseudo proxy trace. Since each proxy trace file is a portion of the original Boeing proxy trace file, the trace file contains realistic proxy access reference information. Table 2 presents the characteristics of the 6 scaled-down proxy trace files collected from the day trace files on March 4 1999.

2.2.2 Pseudo server traces

We have created two types of pseudo server traces: *global pseudo server traces*, and *individual pseudo server traces*. For a given scaled-down proxy trace, its global pseudo server trace consists of traces accessing the servers listed in that proxy trace by all the clients who do not use the proxy.

In each of the five merged Boeing proxy trace files, we identified several popular servers that are frequently accessed by clients to form individual pseudo server trace files. Such an individual pseudo server trace is created by collecting all access references to the identified server. We use three invariants to determine whether such a trace is sufficiently realistic to represent a server trace: (1) distinct request ratio (the percentage of distinct requests in the total number of requests), (2) distinct file access ratio (the percentage of distinct files in the total number of distinct server files), and (3) concentration ratio of references (the percentage of requests concentrating on 10% of popular files). In

	# requests	# objects	# clients	# servers	# successful requests	total transferred Bytes	average # requests/minute
Day1	22,529,006	6,356,818	66,117	147,577	19,477,134	148,249,338,973	15,645
Day2	22,135,881	6,291,157	66,661	147,054	19,160,640	150,005,649,738	15,372
Day3	23,409,805	6,288,377	65,322	145,988	20,488,491	158,082,426,460	16,256
Day4	22,187,580	6,183,673	66,358	142,338	19,316,066	152,748,081,589	15,408
Day5	21,005,234	6,123,227	59,361	141,869	18,305,089	148,026,687,533	14,586

Table 1: Merged Boeing proxy traces of 5 days.

	proxy 1	proxy 2	proxy 3	proxy 4	proxy 5	proxy 6
Total Clients	50	200	500	1,000	5,000	10,000
Total Requests	13,566	68,985	154,527	341,619	1,859,045	3,068,912
Successful Requests	12,275	60,613	133,517	302,856	1,607,145	2,653,288
Transferred MBytes	123	360	1,559	2,080	13,353	20,745
Number of Servers	647	1,692	3,482	7,004	24,486	34,179
Number of Files	9,484	321,343	74,958	157,717	679,876	995,879

Table 2: Selected scaled-down proxy traces.

practice, a regular Web server's distinct request ratio, distinct file access ratio, and concentration ratio of references are below 3%, around 30%, and around 90%, respectively [1]. Among numerous pseudo server trace files, we are able to select many traces where the three invariants are very close to that of regular Web servers. We selected 6 of them for our trace-driven simulations. Tables 3 and 4 presents the characteristics of the 6 individual pseudo server trace files and 6 global pseudo server trace files collected from the day trace files on March 4 1999. The global pseudo traces of global servers 1, 2, 3, 4, 5, and 6 are 558, 163, 88, 40, 10, and 6 times larger than the proxy traces of proxies 1, 2, 3, 4, 5, and 6, respectively.

Other factors may not be considered as important as the three invariants, such as the number of clients. In practice, the number of clients connected to a specific server can be dynamically and timely changed.

Individual server traces are available in the public domain. We did not select any of them in our study because we are not able to distinguish accesses between clients and proxy servers.

2.3 Trace-driven Simulations

We have built a simulator to construct multiple Web servers connected to clients through a large proxy. A standard prefetching PPM model ([5]) is built in both proxy and servers. The server or proxy makes prefetching decisions based on the PPM models by sending both requested and prefetched data to the proxy or the targeted clients. The PPM models are dynamically maintained and updated based on historical data of a time interval. In our study, the models are updated daily.

Following network parameters are set in the simulation:

- A Local Area Network (LAN) is constructed between clients and the proxy server. It is a high bandwidth and low latency network with a round trip time, $RTT < 1ms$.
- A Wide Area Network (WAN) is constructed between the proxy server and Web servers. It is a high bandwidth and high latency network with a round trip time $RTT \approx 90ms$.

We use persistent TCP connection and pipelining. Using default values of the current version 2.0 of the Apache HTTP server as references, we set a fixed holding time of 15 seconds for each persistent connection, and set a maximum number of requests of 100 per connection.

We assume each client has a browser cache of 10 MBytes using LRU replacement policy. The disk cache size of the proxy is 16 GB, and LRU replacement policy is used. We only prefetch objects whose data sizes are less or equal to 50 KBytes, in order to reduce overhead caused by inaccurate predictions.

3 Limits of Proxy-Based Prefetching

We first investigated the conditions for a proxy-based prefetching technique to be effective. In other words, we want to find out under what conditions, proxy-based prefetching is not effective and needs help from Web servers. Since server-based prefetching provides high accuracy, we use client hit ratios obtained from server-based prefetching as the baseline for performance comparisons. A relative hit ratio for a given proxy and Web server is defined as the ratio between a hit ratio obtained by a proxy-based prefetching and a hit ratio obtained by a server-based prefetching. The relative hit ratio presents the relative prefetching ability of a proxy-based prefetching compared with the server-based prefetching.

Making prefetching predictions at the proxy for the 6 individual servers, we have observed the changes of relative hit ratios by increasing the number of clients to access the proxy (see Figure 1). Since each group of clients is randomly selected, the selected clients can be both active and inactive, which is close to reality. Moreover, we repeated each experiment with an individual server 20 times and presented the average results. This study indicates that proxy-based prefetching ability, measured by the relative hit ratios, increases as the number of clients served by the proxy for accessing a particular server increases. For example, as the number of clients increases to 16 and 64, the average relative hit ratios increase to 59% and 79%, respectively, for accessing each of the six servers from the clients.

	server 1	server 2	server 3	server 4	server 5	server 6
Total Requests	648,534	243,462	210,131	182,541	157,466	140,473
Successful Requests	647,277	233,216	208,059	181,531	133,854	139,459
Transferred MBytes	1,815	1,204	1,112	875	542	555
Total Clients	21,769	15,193	3,086	2,149	1,810	3,582
Number of Files	5,810	4,185	3,003	3,637	5,130	1,549
Distinct Request Ratio	0.28%	0.40%	0.59%	0.86%	1.75%	0.43%
Distinct File Access Ratio (%)	30.98	22.32	41.00	43.11	45.79	39.12
Concentration ratio (%)	93.79	87.40	93.22	92.43	87.74	90.94

Table 3: Selected pseudo server traces.

	global server 1	global server 2	global server 3	global server 4	global server 5	global server 6
Total Requests	7,951,627	11,270,833	13,462,240	14,865,900	18,126,906	18,902,854
Successful Requests	6,850,784	9,866,199	11,718,589	12,897,176	15,689,774	16,414,926
Transferred MBytes	38,886	59,204	76,752	82,450	114,371	119,795
Total Clients	63,054	64,769	65,182	65,517	65,987	66,068
Number of Files	1,505,796	1,969,177	2,464,367	2,794,933	3,818,797	4,136,209
Number of Servers	647	1,692	3,482	7,004	24,486	34,179

Table 4: Global pseudo server traces.

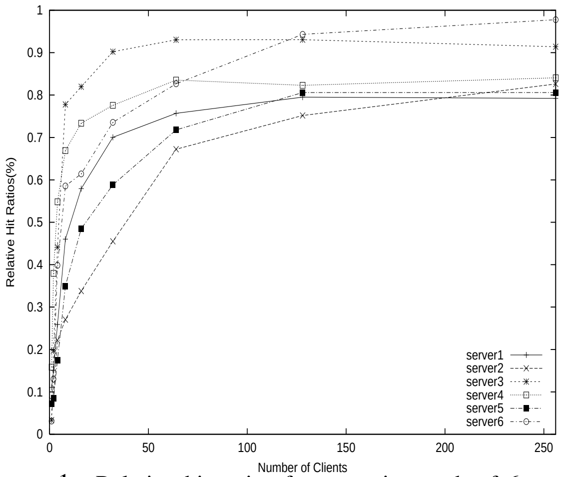

Figure 1: Relative hit ratios for accessing each of 6 servers through the proxy where the prefetching is made in the proxy.

As we have shown, the ability of proxy-based prefetching to improve hit ratios of a particular server is affected by the number of shared requests to the server through the proxy. The next important question we want to ask is about the shared request distributions to different servers through the proxy in a global Internet environment. Studying the distributions, we are able to evaluate the effectiveness and limits of proxy-based prefetching.

The distributions presented in Figure 2 are measured by the cumulative density functions (cdf) for the six proxy traces with 50, 200, 500, 1,000, 5,000, and 10,000 clients respectively. For the 50 client proxy server (proxy 1), 97% of the requests consist of 16 or less shared clients to Web servers. On the other hand, for the 10,000 client proxy (proxy 6), 44% of the requests consists of 16 or fewer shared clients to Web servers. The average percentage of the requests consisting of 16 or fewer shared clients going through the proxy to Web servers is 59%.

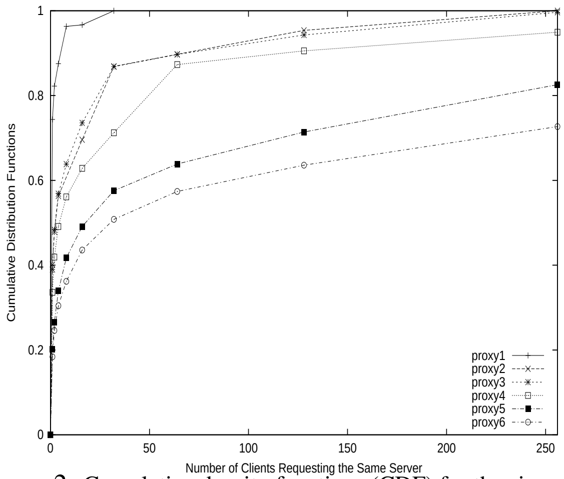

Figure 2: Cumulative density functions (CDF) for the six proxy traces with 50 (proxy 1), 200 (proxy 2), 500 (proxy 3), 1,000 (proxy 4), 5,000 (proxy 5), and 10,000 (proxy 6) clients respectively.

In practice, the number of clients served by proxy 1 or proxy 6 is either too small or too large. If we use the 1,000 client proxy (proxy 4) as a regular case, we find that more than 62% of the requests consist of 16 or fewer shared clients to Web servers, which is very close to the average percentage (59%).

The distribution study indicates that in a normal proxy server with 1,000 clients, proxy-based prefetching can reasonably satisfy less than 40% of the requests by achieving approximately a 60% relative hit ratio. However, for more than 60% of the requests, proxy-based prefetching may not be sufficiently effective. In order to improve the access hit ratios, we should adaptively rely on server-based prefetching for those requests that do not have a sufficient number of shared clients.

4 Coordinated Proxy-Server Prefetching

One effective approach to overcome the limits of both server-based and proxy-based prefetching techniques is to develop a coordinated proxy-server prefetching technique. Since proxy servers may be located between servers and clients, Web server traces may not exactly reflect client access behavior. Our coordinated prefetching technique would effectively address this concern. Our objective is to adaptively utilize the reference information at both proxy and Web servers. Figure 3 presents the outline of the coordinated proxy-server prefetching technique. Upon a client request of an object file, if the request hits in the browser, the object will be accessed locally. The client will also inform the proxy about the access to the object file, and initiate the coordinated proxy-server prefetching process. If the request misses in the browser, the request will be forwarded to the proxy, and the coordinated proxy-server prefetching process is initiated. Starting in the proxy, the coordinated proxy-server prefetching process will handle the following four different cases in the proxy:

1. if the object **exists** and prefetching information related to the object is **available** in the proxy;
2. if the object **does not exist** and prefetching information related to the object is **not available** in the proxy;
3. if the object **exists** and prefetching information related to the object is **not available** in the proxy; and
4. if the object **does not exist** and prefetching information related to the object is **available** in the proxy.

A proxy-based prefetching procedure will handle case 1, while a server-based prefetching procedure will handle case 2. We propose the proxy-server-based prefetching procedures (I) and (II), defined below, to handle case 3 and case 4, respectively. The 4 procedures involved in the coordinated proxy-server prefetching process are shown in Figure 3.

server-based prefetching procedure:

(a) The proxy forwards the client request to the server, and asks the server to prefetch related objects based on its PPM model.
(b) The server sends the requested object to the proxy along with a list of URLs of predicted objects.
(c) The proxy stores the object, and sends it to the client along with the list of URLs of predicted objects.
(d) After a local searching, the proxy sends a selected list of URLs of predicted objects to the server.
(e) The server sends the selected objects to the proxy. Meanwhile, the client sends a selected list of URLs of predicted objects to the proxy after a local searching.
(f) The proxy sends the selected objects to the client.

proxy-based prefetching procedure:

(a) The proxy sends the requested object to the client, along with a list of URLs of predicted objects.
(b) After a local searching, the client sends a selected list of URLs of predicted objects to the proxy.

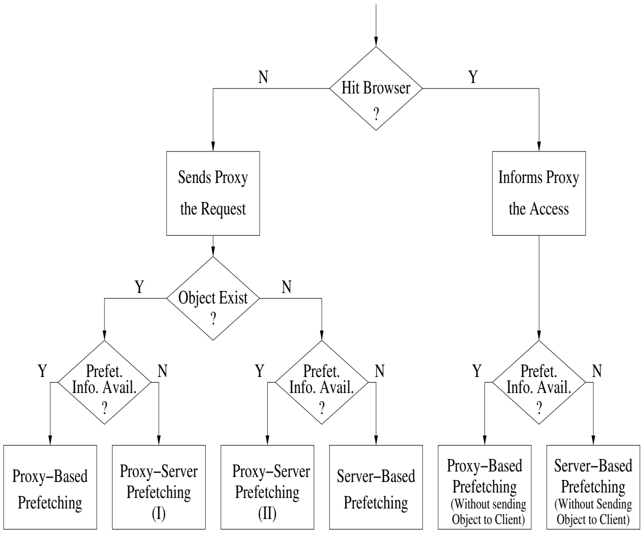

Figure 3: The coordinated proxy-server prefetching system design.

(c) The proxy sends the selected objects to the client.

proxy-server-based prefetching procedure (I):

(a) The proxy sends the object to the client, and asks the server to prefetch related objects based on its PPM model.
(b) The server sends a list of URLs of predicted objects.
(c) The proxy sends the list of URLs of predicted objects to the client.
(d) After a local searching, the proxy sends a selected list of URLs of predicted objects to the server.
(e) The server sends the selected objects to the proxy. Meanwhile, the client sends a selected list of URLs of predicted objects to the proxy after a local searching.
(f) The proxy sends the selected objects to the client.

proxy-server-based prefetching procedure (II):

(a) The proxy forwards the client request to the server, along with a selected list of URLs of predicted objects based on the local prefetching information and a local search.
(b) The proxy sends the selected list of URLs of predicted objects to the client.
(c) The server sends the selected objects to the proxy. Meanwhile, the client sends a selected list of URLs of predicted objects to the proxy after a local searching.
(d) The proxy sends the selected objects to the client.

5 Performance Evaluation

We have evaluated the coordinated proxy-server prefetching technique by measuring its communication reductions over the server-based prefetching technique, and by comparing its hit ratios with that of server-based and proxy-based prefetching techniques. We also have evaluated the overall effectiveness of the coordinated proxy-server prefetching technique based on reductions of the number of packets connecting to servers and the number of prefetching decisions made by servers.

5.1 Comparisons of Hit and Byte Hit Ratios

Figure 4 presents hit ratio comparisons among proxy-based, coordinated proxy-server, and server-based prefetching techniques using the six proxy traces and their pseudo global server traces. There are two parts of the contribution to hits ratios: the browser caching and prefetching. Figure 4 shows that the hit ratios contributed from proxy-server-based prefetching are significantly higher than those from proxy-based prefetching, and they are comparable to the hit ratios from server-based prefetching for all the 6 proxy traces. For example, using the trace file of proxy 1 with 50 clients, the hit ratio contributed from coordinated proxy-server prefetching is 76% higher than that of proxy-based prefetching, but only 6% lower than that of server-based prefetching. Taking another example, using the trace file of proxy 6 with 10,000 clients, the hit ratio from coordinated proxy-server prefetching is 88% higher that of proxy-based prefetching, but only 4% lower than that of server-based prefetching. The average hit ratio from coordinated proxy-server prefetching is 85% higher than that of proxy-based prefetching, and 5% lower than that of server-based prefetching.

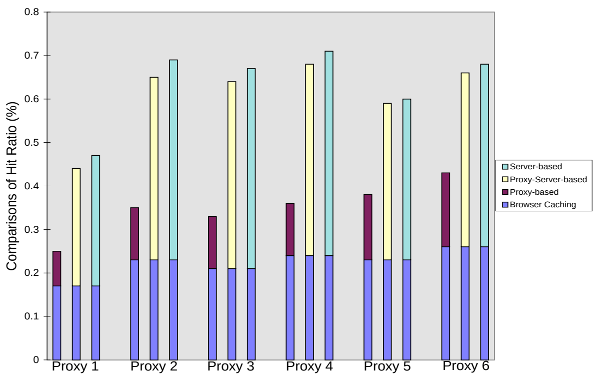

Figure 4: Comparisons of hit ratios among the proxy-based, coordinated proxy-server, and server-based prefetching techniques using the 6 proxy traces.

Figure 5 presents byte hit ratio comparisons among proxy-based, coordinated proxy-server, and server-based prefetching techniques using the six proxy traces and their pseudo global server traces. Again, we divide the contributions into two parts to the byte hit ratios: from proxy caching and from prefetching. We show that the byte hit ratios are consistent with their hit ratio performance.

5.2 Reductions of Individual Web Server Traffic and Computing

The persistent connection of TCP in HTTP version 1.1 [10] is used in servers in our study. In a server-based prefetching system, a prefetching process will start from a server connection by

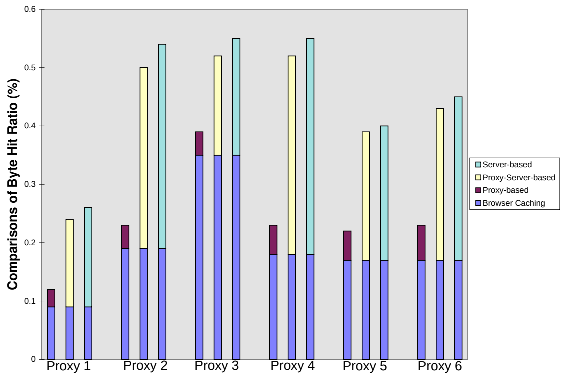

Figure 5: Comparisons of byte hit ratios among the proxy-based, coordinated proxy-server, and server-based prefetching techniques using the 6 proxy traces.

a client or a proxy after a hit or a miss to a requested object in the browser or the proxy. The server will then either send the object along with a list of URLs of predicted objects related to the requested object, if the requested object is missed in the browser or proxy, or just send the list of URLs of objects, if the requested object is hit in the browser or proxy. After receiving the list, the client makes a local search to provide a selected list of URLs of objects that are not available in the browser or the proxy. This list will be sent to the server again. The server will send the selected objects to the client or proxy. The server will be contacted twice by short messages, called packets, from the client or proxy. We compare the numbers of packets used for the server connections between the server-based prefetching and the coordinated proxy-server prefetching. The packet reduction rate is calculated by

$$R_{packets} = \frac{P_{server} - P_{coordin}}{P_{server}},$$

where P_{server} is the number of packets generated from server-based prefetching, and $P_{coordin}$ is the number of packets generated from coordinated prefetching.

The left figure in Figure 6 shows packet reduction rates of the coordinated proxy-server prefetching technique over the server-based prefetching system for each of the six individual server traces as the number of clients increases. The reduction rate proportionally and significantly increases as the number of clients increases. For example, as the number of clients increases to 256, the packet reduction rates of the coordinated proxy-server prefetching technique for server 1, server 2, server 3, server 4, server 5, and server 6, are 85%, 94%, 93%, 94%, 89%, and 94%, respectively.

The coordinated proxy-server prefetching technique moves some of the prefetching decisions to the proxy, so that the number of prefetching decisions made in Web servers is reduced. As a result, the computing overhead in servers for prefetching can be effectively reduced. The overhead reduction rate is calculated by

$$R_{overhead} = \frac{D_{server} - D_{coordin}}{D_{server}},$$

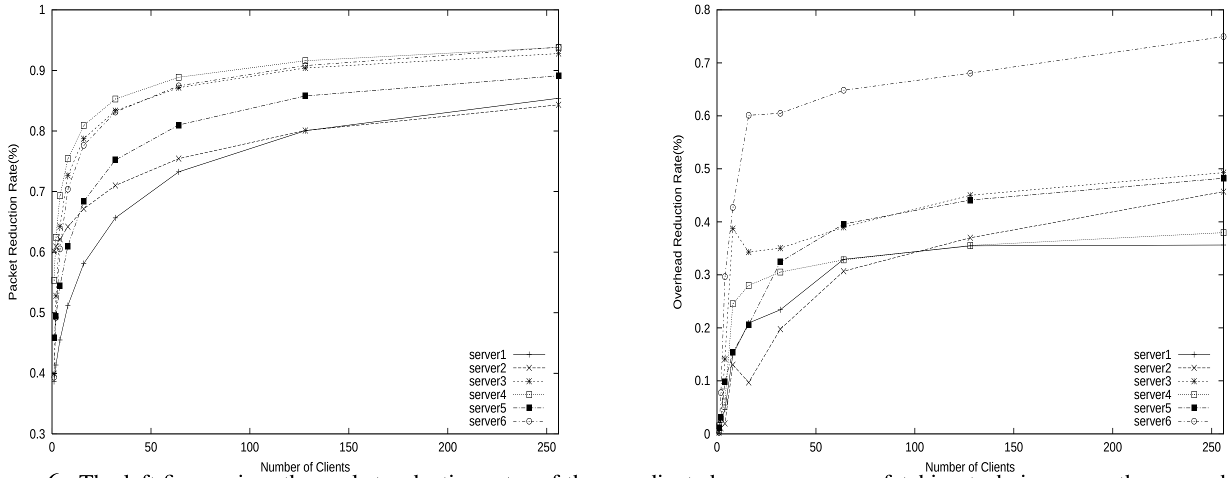

Figure 6: The left figure gives the packet reduction rates of the coordinated proxy-server prefetching technique over the server-based prefetching technique. The right figure gives the number of server prefetching decision reduction rates.

where D_{server} is the number of decisions made by server-based prefetching, and $P_{coordin}$ is the number of decisions made by coordinated prefetching.

The right figure in Figure 6 shows the overhead reduction rates of the coordinated proxy-server prefetching technique over the server-based prefetching system for each of the six individual server traces as the number of clients increases. We show that the overhead reduction rate increases as the number of clients increases. The overhead reduction to server 6 is significant. For example, as the number of clients increases to 256, the packet reduction rates of the coordinated proxy-server prefetching technique for server 1, server 2, server 3, server 4, server 5, and server 6, are 36%, 46%, 49%, 38%, 48%, and 75%, respectively.

5.3 Reductions of Global Web Server Traffic and Computing

We have shown that the hit ratios and byte hit ratios of coordinated proxy-server prefetching are close to server-based prefetching using six proxy traces and their pseudo global server traces. Using the same trace-driven simulation, we have compared the numbers of packets received by servers, and the numbers of prefetching decisions made by servers for both techniques in Figure 7 and Figure 8, respectively. The trace-driven simulations show that as the number of clients served by the proxy increases, the difference of the numbers of packets sent to servers between the two techniques increases. For example, the packet differences are 221,582 representing a 26% reduction from the server-based prefetching and 364,584 representing a 25% reduction from the server-based prefetching for proxy 5 and proxy 6, respectively. The comparisons of the number of prefetching decisions made by servers follow similar patterns. For example, the number of prefetching decision differences are 307,087 representing a 45% reduction from server-based prefetching and 471,969 representing a 41% reduction from server-based prefetching for proxy 5 and proxy 6, respectively.

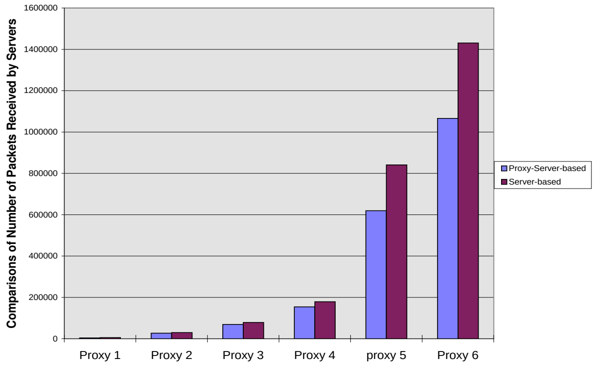

Figure 7: Comparisons of the number of packets received by servers between the coordinated proxy-server, and server-based prefetching techniques using the 6 proxy traces and their pseudo global servers.

6 Other Related Work

Other prefetching methods can be divided into three types: server-initiated, client-initiated, and cooperative prefetching between servers and clients. In server-initiated prefetching, servers always push their popular documents to the clients. The work reported in [3] uses service proxies (agents) to serve for multiple home servers, and the mapping between home servers and service proxies can be many-to-many. Markatos and Chronaki [13] propose a Top-10 approach that combines the servers' active knowledge of their popular documents with client access profiles. Web servers regularly push their most popular documents to web proxies, and proxies then push those documents to the active clients.

In client-initiated prefetching, the information observed by the client side is used to determine the prefetching contents. A simulation study on accesses to two popular UCLA sites is reported to present a client-initiated prefetching software [11]. Wcol [19]

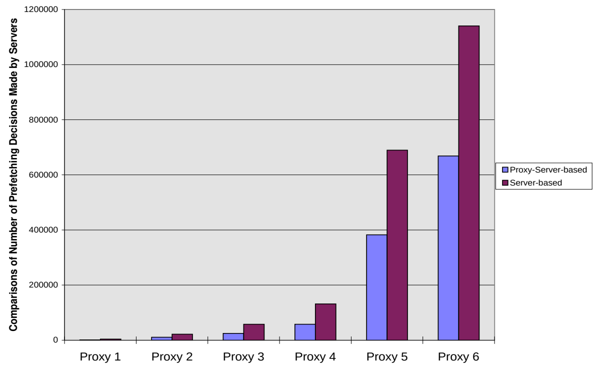

Figure 8: Comparisons of the number of prefetching decisions made by servers between the coordinated proxy-server, and server-based prefetching techniques using the 6 proxy traces and their pseudo global servers.

is a proxy software that parses HTML and prefetch documents, links and embedded images.

Cohen et. al. [6] propose an end-to-end approach to improving web performance by collectively examining servers and proxy. The server groups the relative files together, and the proxy requests selected files. The server and client/proxy communicate with each other through data piggybacking. Duchamp [7] presents a new implementation of a server-based prefetching system that collects the knowledge of popular URLs for clients to initiate prefetching. Foygel and Strelow consider a hierarchical structure for prefetching in proxy [9].

Using the first-order of Markov models and speculations based on statistical information of servers, other researchers have developed prefetching methods (see e.g. [3], [14], [17], and [16]). Researchers have also proposed hint-based web clients and servers that can load/store the predicted data objects for future use. (see e.g. [6] and [12]).

7 Conclusion

In this study, we have investigated the issues of coordinations between proxy-based prefetching and server-based prefetching. We show that the reference access information of servers is important to data prefetching, but should be utilized effectively. Setting a prefetching threshold in proxy, we are able to ensure an acceptable prefetching accuracy of proxy-based prefetching to significantly reduce unnecessary traffic between the proxy and servers for exchanging reference access information in servers. Our trace-driven simulations show that we have effectively integrated and coordinated both server-based and proxy-based techniques to achieve the goal of retaining the prefetching accuracy of the server-based model and minimizing the communications between proxy and Web servers.

Acknowledgement. Technical discussions with Li Xiao have been helpful for this work. We appreciate Bill Bynum for reading the paper and for his suggestions. The comments from anonymous referees are constructive to improve the quality and readability of the paper.

References

[1] M. Arlitt and C. L. Williamson, "Web server workload characterization, the search for invariants", *Proceedings of 1996 ACM SIGMETRICS Conference*, Philadelphia, PA May 1996, pp. 126-138.

[2] P. Barford, A. Bestavros, A. Bradley, and M. Crovella, "Changes in web client access patterns: characteristics and caching implications", *World Wide Web Journal*, 2(1):15-28, January,1999.

[3] A. Bestravos, "Using speculation to reduce server load and service time on the WWW", *Proceedings of the 4th ACM International Conference on Information and Knowledge Management*, (CIKM'95), Baltimore, Maryland, 1995.

[4] Boeing log files. ftp://researchsmp2.cc.vt.edu/pub/boeing/

[5] J. G. Cleary and I. H. Witten, "Data compression using adaptive coding and partial string matching", *IEEE Transactions on Communications*, Vol. 32, No. 4, pp. 396-402, 1984.

[6] E. Cohen, B. Krishnamurthy, and J. Rexford, "Improving end-to-end performance of the web using server volumes and proxy filters", *Proceedings of the ACM SIGCOMM Conference*, 1998, pp. 241-253.

[7] D. Duchamp, "Prefetching hyperlinks", *Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems* (USITS'99), October 1999.

[8] L. Fan, P. Cao, W. Lin, and Q. Jacobson, "Web prefetching between low-bandwidth clients and proxies: potential and performance", *Proceedings of ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems*, May 1999, pp. 178-187.

[9] D. Foygel and D. Strelow, "Reducing Web Latency with Hierarchical Cache-based Prefetching", *Proceedings of the International Workshop on Scalable Web Services*, (in conjunction with ICPP'00), Toronto, Ontario, Canada , August 2000.

[10] Hypertext Transfer Protocol – HTTP/1.1, www.w3.org/Protocols/rfc2616/rfc2616.html.

[11] Z. Jiang and L. Kleinrock, "An adaptive network prefetch scheme", *IEEE Journal on Selected Areas of Communication*, Vol. 17, No. 4, 1998, pp. 358-368.

[12] T. M. Kroeger, D. D. E. Long, and J. C. Mogul, "Exploiting the bounds of web latency reduction from caching and prefetching", *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, Monterey, California, April 1997.

[13] E. P. Markatos and C. E. Chronaki, "A top-10 approach to prefetching on the web", Technical Report, No. 173, ICS-FORTH, Heraklion, Crete Greece, August 1996.

[14] V. N. Padmanabhan and J. C. Mogul, "Using predictive prefetching to improve World Wide Web latency", *Computer Communication Review*, 1996, pp. 22-36.

[15] J. Pitkow and P. Pirolli, "Mining longest repeating subsequences to predict world wide web surfing", *Proceedings of the 1999 USENIX Technical Conference*, April 1999.

[16] S. Schechter, M. Krishnan, and M. D. Smith, "Using path profiles to predict HTTP requests", *Proceedings of the 7th International World Wide Web Conference*, Brisbane, Australia, 1998.

[17] R. R. Sarukkai, " Link Prediction and Path Analysis Using Markov Chains", *Proceedings of the 9th International World Wide Web Conference*, 2000.

[18] T. Palpanas and A. Mendelzon, "Web prefetching using partial match prediction", *Proceedings of Web Caching Workshop*, San Diego, California, March 1999.

[19] Wcol Group, "Www collector: the prefetching proxy server for WWW", URL: http://shika.aist-nara.ac.jp/products/wcol/wcol.html, 1997.

[20] L. Xiao and X. Zhang, "Exploiting neglected data locality in browsers", *Proceedings of the 10th International World Wide Web Conference*, (WWW10), Hong Kong, May 1-5, 2001, (an extended abstract).