A Review on Recent Web Pre-Fetching and Caching Technique

Varun Kumar¹ Nidhi Seth²

¹Research Scholar ²Assistant Professor

¹,²Department of Computer Science & Engineering
¹,²JMIT, Radaur, Haryana, India

Abstract—Due to the fast development of internet services and a huge amount of network traffic web caching and prefetching are the most popular techniques that play a key role in improving the Web performance by keeping web url that are likely to be visited in the near future closer to the client. Web caching technique work integrated or independently with the web prefetching. The Web caching and prefetching techniques are complement each other since the web caching exploits the temporal locality for predicting revisiting requested url, while the web prefetching utilizes the spatial locality for predicting next related web object of the requested Web url. This paper surveys some principles and existing web caching and prefetching approaches. The intelligent and conventional web caching techniques are investigated and discussed. Furthermore, Web prefetching techniques are summarized and classified with comparison limitations of these approaches. This paper also discusses some studies that take into consideration impact of integrating both web caching and web prefetching jointly. This paper explains about the various prefetching and caching techniques, how these technique predict the web object to be pre-fetched and what are the issues challenges involved when these techniques are applied.

Key words: Web caching, Web prefetching, URL, Prediction

I. INTRODUCTION

Web caching is a well-known strategy for improving the performance of Web-based system by keeping Web objects that are likely to be used in the near future in location close to user. The Web caching mechanisms are applied at three levels: client level, proxy level and original server level[5,6]. Significantly, proxy servers play the key roles between users and web sites in lessening of the response time of user requests and saving of network bandwidth. Thus, for achieving better response time, an efficient caching technique can be built in a proxy server.

The cache replacement is the core or heart of the web caching; consequently, the design of efficient cache replacement algorithms is crucial for caching mechanism achievement. So, cache replacement algorithms are also called web caching algorithms[7]. Because of limited space of cache, an intuitive mechanism is required to manage the Web cache content properly. The conventional caching policies are not efficient in the Web caching since they consider just one factor and ignore other factors that have impact on the efficiency of the Web caching. In these caching policies, most popular objects get the most requests, while a large segment of objects, which are stored in the cache, are never requested again. This is called cache pollution problem. Therefore, many Web cache replacement policies have been proposed attempting to get good performance. Hence, the difficulty in determining which ideal web objects will re-accessed is still a big challenge faced by the existing Web caching techniques. In other words, what Web objects should be cached and what Web objects should be replaced to make the best use of available cache space, better hit rates, decrease network traffic, and reduce loads on the original server[3,4].

Unfortunately, the cache hit ratio is not improved much with caching schemes. despite with a cache of infinite size, the hit ratio are still limited only at the range from 40% to about 50%, regardless of the caching scheme [8,9,10]. This is because most people browse and explore the new web pages trying to find new information. In order to improve the hit ratio of cache, Web pre-fetching technique is integrated with web caching to overcome these limitations. Web prefetching is fetching web pages in advance by proxy server/client before a request is send by a client/proxy server. The major advantage of using web prefetching is reduced latency. When a client makes a request for web object, rather than sending request to the web server, it may be fetched from a pre-fetch area. The main factor for selecting a web prefetching algorithm is that its ability to predict the web object to be pre-fetched in order to reduce latency. Web prefetching exploits the spatial locality of web pages, i.e., pages that are linked with current page will be accessed with higher probability than other pages. Web prefetching can be implemented in a web environment as between clients and web server, between proxy server and web server and between clients and proxy server[11]. If it is implemented between web server and client, it is helpful in decreasing user perceived latency, but the problem is that it will increases network traffic. If it is implemented between web server and proxy server, can reduce the bandwidth usage by prefetching only a specific number of hyperlinks. If it is implemented between clients and proxy server, the proxy starts feeds pre-fetched web objects from its cache to the clients so there won’t be extra internet traffic. This paper tells about the different prefetching techniques, how they predict the web object to be pre-fetched and what are the issues involved in these techniques.

The remaining parts of this paper are organized as follows. Some principles of Web caching are presented in Section 2. The existing works of traditional and intelligent web caching techniques are also discussed in Section 2. Section 3 describes types and approaches of Web prefetching and surveys some of the representative techniques for each approach. Section 4 elucidates some studies that discussed integration of web caching and web prefetching together. Finally, Sections 5 concludes the paper.

II. WEB CACHING

Web caching is one of the most successful solutions for improving the performance of Web-based system. In Web caching, the popular web objects that likely to be visited in the near future are stored in positions closer to the user like client machine or proxy server. Thus, the web caching helps
in reducing Web service bottleneck, alleviating of traffic over the Internet and improving scalability of the Web system[2].

A. Basic Types of Web Cache:

1) Browser Cache:
It is located in the client. The user can notice the cache setting of any modern Web browser such as Internet Explorer, Safari, Mozilla Firefox, Netscape, and Google chrome. This cache is useful, especially when users hit the “back” button or click a link to see a page they have just looked at.

2) Proxy Server Cache:
It is found in the proxy server which located between client machines and origin servers. It works on the same principle of browser cache, but it is a much larger scale. Unlike the browser cache which deals with only a single user, the proxies serve hundreds or thousands of users in the same way. When a request is received, the proxy server checks its cache. If the object is available, it sends the object to the client. If the object is not available, or it has expired, the proxy server will request the object from the origin server and send it to the client. The object will be stored in the proxy's local cache for future requests.

3) Origin Server Cache:
Even at the origin server, web pages can be stored in a server-side cache for reducing the need for redundant computations or database retrievals. Thus, the server load can be reduced if the origin server cache is employed.

III. WEB CACHING ALGORITHMS

Cache replacement policy plays an extremely important role in Web caching. Hence, the design of efficient cache replacement algorithms is required to achieve highly sophisticated caching mechanism. In general, cache replacement algorithms are also called Web caching algorithms.

<table>
<thead>
<tr>
<th>Policy</th>
<th>Brief description</th>
<th>Advantage</th>
<th>Disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRU</td>
<td>The least recently used objects are removed first.</td>
<td>Simple and efficient with uniform size objects, such as the memory cache.</td>
<td>Ignores download latency and the size of web objects</td>
</tr>
<tr>
<td>LFU</td>
<td>The least frequently used objects are removed first.</td>
<td>Simplicity</td>
<td>Ignores download latency and size of objects and may store obsolete web objects indefinitely.</td>
</tr>
<tr>
<td>SIZE</td>
<td>Big objects are removed first</td>
<td>Prefers keeping small web objects in the cache, causing high cachet hit ratio.</td>
<td>Stores small web objects even if these object are never accessed again. -- Low byte hit ratio.</td>
</tr>
<tr>
<td>GD-size</td>
<td>It assigns a key value to each object in the cache. Consequently, the object with the lowest key value is replaced when cache space becomes occupied</td>
<td>Overcomes the weakness of SIZE policy by removing objects which are no longer requested by users.</td>
<td>Not take into account the previous frequency of web objects</td>
</tr>
<tr>
<td>GDSF</td>
<td>It extends GDS algorithm by integrating the frequency factor into of the key value K(\left( p \right) = L + F \left( p \right) \times CM)</td>
<td>Overcomes the drawback of GD-size</td>
<td>Not take into account the predicted accesses in the future</td>
</tr>
</tbody>
</table>

A. Replacement Policy Categories:

<table>
<thead>
<tr>
<th>Category</th>
<th>Brief description</th>
<th>Available replacement policies</th>
<th>Representative policy</th>
<th>disadvantage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recency-based policies</td>
<td>These policies use recency factor as the primary factor to remove Web object</td>
<td>LRU, LRU- threshold, LRU*, LRU-hot, LRU- LSC, SB-LRU, SLRU, HLRU, Pitkow/Recker, EXP1, value- aging, generational replacement.</td>
<td>LRU</td>
<td>not consider frequency, size and download latency of Web objects</td>
</tr>
<tr>
<td>Frequency-based policies</td>
<td>These policies use object popularity (or frequency count) as the primary factor to remove Web object</td>
<td>LFU, LFU-Aging, LFU-DA, Window-LFU, swLFU, AgedswLFU, a-Aging, HYPER - G.</td>
<td>LFU</td>
<td>not consider recency, size and download latency of Web objects</td>
</tr>
<tr>
<td>Size-based policies</td>
<td>These policies use object size as the primary factor for removing Web object.</td>
<td>SIZE, LRU min, partitioned caching, PSS, CSS, LRU-SP</td>
<td>SIZE</td>
<td>not consider recency, frequency and download latency of Web objects</td>
</tr>
</tbody>
</table>
A Review on Recent Web Pre-Fetching and Caching Technique

IV. PREFETCHING

Web prefetching is another very effective technique, which is utilized to complement the Web caching mechanism. The web prefetching predicts the web object expected to be requested in the near future, but these objects are not yet requested by users. Then, the predicted objects are fetched from the origin server and stored in a cache. Thus, the web prefetching helps in increasing the cache hits and reducing the user-perceived latency.

Table 2: replacement policies Categories

| Function-based policies | These policies associate each object in the cache with a utility value. The value is calculated | GD-Size, GDSF, GD*, PGDS, server-assisted cache replacement, GD-SIZE | Choosing appropriate weights of factors is difficult task |

A. Types of Prefetching Based On Location:

The prefetching techniques can be implemented on server, proxy or client side. The client-based prefetching concentrates on the navigation patterns of a single user across many Web servers. On another hand, the sever-based prefetching concentrates on the navigation patterns of all users accessing a single website. The proxy-based prefetching concentrates on the navigation patterns of a group of users across many Web servers.

Table 3: Types of prefetching based on location

<table>
<thead>
<tr>
<th>Prefetching Location</th>
<th>Data for Prediction Model</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>Historical and current user requests</td>
<td>Easy to partition user session and realize personalized prefetching.</td>
<td>– Not Share Prefetching Content Among Users. – Needs A Lot Of Network Bandwidth.</td>
</tr>
<tr>
<td>Proxy</td>
<td>Proxy log and current user requests</td>
<td>– Reflects Common Interests For A Group Of Users. – Shares Prefetching Content from Different Servers among Users.</td>
<td>– Not Reflect Common Interests For A Single Website From All Users.</td>
</tr>
<tr>
<td>Server</td>
<td>Server log and current user requests</td>
<td>– Records Single Website Access Information From All Users And Better Reflect All Users’ Common Interests.</td>
<td>– Not Reflect Users’ Real Browsing Behavior. – Difficult To Partition User Session. – Needs Additional Communications Between Clients And Servers For Deciding Prefetching Content.</td>
</tr>
</tbody>
</table>

V. WEB PRE-FETCHING TECHNIQUES

A. Domain Top Approach:

(Domain1 ∪ Domain2 ∪ ... ∪ DomainN)

Seung Won Shin et al. proposes a domain top approach for web prefetching, which combines the proxy’s active knowledge of most popular domains and documents[11], [13]. In this approach proxy is responsible for calculating the most popular domains and most popular documents in those domains, then prepares a rank list for prefetching.

B. Dynamic Web Prefetching:

In dynamic web pre-fetching technique [12], each user can keep a list of sites to access immediately called user’s preference list. The preference list is stored in proxy server's database. Intelligent agents are used for parsing the web page, monitoring the bandwidth usage and maintaining hash table, preference list and cache consistency. It controls the web traffic by reducing pre-fetching at heavy traffic and increasing pre-fetching at light traffic. Thus it reduces the idle time of the existing network and makes the traffic almost constant. A hash table is maintained for storing the list of accessed URLs and its weight information [12], [13].

Depending upon the bandwidth usage and weights in the hash table, the prediction engine decides the number of URLs to be pre-fetched and gives the list to pre-fetch engine for pre-fetching the predicted web pages. After pre-fetching, the proxy server keeps the pre-fetched web pages in a separate area called pre-fetch area.

Fig. 1: Structure of rank list for Domain Top Approach

<table>
<thead>
<tr>
<th>web page</th>
<th>web page</th>
<th>web page</th>
</tr>
</thead>
<tbody>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>web page</td>
<td>web page</td>
<td>web page</td>
</tr>
<tr>
<td>Top domain</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
C. Link Pre-Fetching:
A web page provides a set of pre-fetching hints to the browser and after the browser finishes loading the page, it starts pre-fetching specified documents and stores them in its cache. When the user visits one of the pre-fetched documents, it can be served up quickly out of the browser’s cache. Fisher et. Al proposed a server driven approach for ink prefetching [14]. In this approach browser follows special directives from the web server or proxy server that instructs into pre-fetch specific documents. This mechanism allows servers to control the contents to be perfected by the browser. The browser looks for either HTML <link> tag or an HTTP Link: headerTag to pre-fetch the subsequent links. The Link: header can also be specified within the HTML document itself by using a HTML <meta>tag[16]. When the browser is idle, it observes these hints and queues up each unique request to be pre-fetched.

D. Top10 Approach:
Evangelos P. Markatos et al. proposes a top 10 approach to prefetching on the web, in which the server calculates the list of most popular documents[19]. This approach is easy to implement in an acient server architecture. It considers a list of access for predicting the web object, not the client characteristics on the web.

E. Model Based Predictive Pre-Fetching:
Yang et. al proposed a model based predictive prefetching, in which an integrated web-caching and web-pre-fetching model is used[18,19]. The prediction model used in this is based on the statistical correlation between web objects. The prediction model is time based, prediction window represents some specific time period than number. The algorithm constructs a logical graph called correlation graph, which shows the correlation between web objects and pre-fetch web objects that are highly correlated to a currently requested object. They developed an integrated caching and prefetching algorithm, Pre- GDF. This algorithm is based on the algorithms GD- Size [13] and its enhancement GDSF [7]. The key components in the algorithm are replacement manager, pre-fetching agent, prediction queue and cache.

F. Adaptive Pre-Fetching Scheme:
Adaptive pre-fetch scheme is developed to adapt user’s browsing history and habits [15]. Jia and et. al proposed an adaptive pre-fetch scheme, in which the number of files to be pre-fetch depends on user access history and network conditions. This scheme consists of two modules: prediction module and threshold module. The prediction module predicts the access probability of each file. Files whose access probabilities are greater than or equal to the threshold are only prefetched. Chen and et. al [7] proposed an adaptive pre-fetch scheme, in which dynamically adjust the prefetch aggressiveness in web servers and uses a threshold to adjust the aggressiveness of prefetching. Fagni and et. al [7] proposed an approach for boosting the performance of search engine by exploiting the spatial and temporal locality present in the stream of processed queries.

G. Semantic Prefetching:
"Semantics", hidden in web documents. From certain point of view, the semantics of web document is already considered in history-based prediction. In that case, this semantics is derived from user interest assuming that users passing the same URL-graph are interested in the same thing semantically.

They do not consider real semantics of document, however. As semantic prefetching we understand prefetching based on preferences of past retrieved documents in semantics, rather than on the chronological relationships between URL accesses. Semantically based prefetching tries to extract a semantic description of a document and asks server to provide pages with similar semantics, with the same so called "semantic locality". Based on the document semantics, this approach is capable of prefetching documents whose URLs have never been accessed [15].

H. Measuring the Performance of Pre-Fetching:
To judge the success of a pre fetch system and to tune the parameters used, the performance of the system must be measured[17]. The following criteria can be used to do this:-
1) Usefulness of Predictions/Pre-fetches: the percentage of fetched pages that had already been predicted is pre-fetch.
2) Accuracy of Predictions/Pre-fetches: The percentage of predicted or pre-fetched pages that were later actually requested by the user.
3) Practical accuracy of predictions: the probability that one of the received predictions was correct.
4) Coverage: the percentage of actual fetches which were preceded by the predictions.
5) Network traffic Increase: The volume of network traffic with pre-fetching enabled / the volume of traffic without pre-fetching.

VI. CONCLUSION
On a large scale research of web prefetching is going on in various directions. In this paper survey on principles and some the existing web caching and prefetching approaches. Web caching and prefetching are two effective approach for Web service bottlenecks, lessen traffic over the Internet and improve scalability of the Web system. Combination of the web caching and the web prefetching improve the performance compared to single caching. In this paper different web prefetching techniques and other directions of web prefetching are studied and discussed Firstly, we have reviewed principles and existing works of web caching. This includes the conventional and intelligent web caching. Secondly, types and categories of prefetching have presented and discussed briefly. The web prefetching scheme focuses on the properties of spatial locality of web objects. Finally, this review paper has presented some studies that discussed integration of web caching and web prefetching together. Top 10 domain technique is better than other technique.

VII. ACKNOWLEDGEMENT
I would express my deep sense of regard to our Head of Department Mr. Vivek Sharma and to Mrs. Nidhi Seth, AP
CSE Department for regular inputs and for constantly guiding. I am obliged to the staff members of CSE department, JMIT for the valuable information provided by them as and when necessary.

REFERENCES


[12] Sandhaya Gawade and Hitesh Gupta, "Review of algorithms for prefetching and caching” 2012 IJARCCSE


