A Web Prefetching Model Based on Content Analysis

O Kit Hong, Fiona Robert P. Biuk-Aghai
Faculty of Science and Technology
University of Macau
{csb.fiona|fst.robert}@umac.mo

Abstract
Web-accessible resources have rapidly increased in number and are being widely used. However, network latency and server overload often cause slow response times. Common solutions include increasing bandwidth and caching, but delays caused by overloaded servers still exist and hit rates of caches are usually limited to no more than 50%. Prefetching of Web pages offers an alternative solution, and several prefetching models already exist. Most of these models, however, prefetch Web pages based on the profile of an average user and fail to take individual differences and preferences into account. This paper presents a new prefetching model which analyses the content of Web pages for each user and builds up both a profile of individual preferences and a site-wide knowledge base of accessed Web content. The model recognizes temporally related individual preferences and adapts to changes in preferences by adopting a recency-before-frequency selection algorithm.

1 Introduction
Web-accessible resources have developed at a rapid rate and become part of our life. However, when accessing non-local sites Web users often experience response delays in the order of several seconds to several minutes, so-called Web Latency. While one may think that increased network bandwidth should help reduce Web latency, the factors causing it are numerous and could include the speed of the server or the client computer, the propagation delay, or the inefficient use of the network connection by the HTTP 1.0 protocol.

A common approach for tackling Web latency is caching. However, research suggests that the hit rate in the caching proxy is usually only 30-50%, i.e. 50-70% of user requests are for pages not previously accessed for which caching does not help, no matter how good the caching algorithms are.

An alternative and pro-active approach is prefetching—fetching the Web pages that the users are likely to access next while viewing the currently displayed page; then if the users do request one of the prefetched pages it will already be in the cache and can be served immediately. Web latency is, in this way, hidden from the users. Several prefetching models already exist. According to their system architecture and the location at which prefetching
is initiated, they can be classified into (1) Server-Initiated Prefetching; (2) Client-Initiated Prefetching; (3) Proxy-Initiated Prefetching; and (4) Hierarchical Prefetching [1].

The prediction in most of these models is based on statistics on the access path or pattern of Web pages. In Server-Initiated Prefetching, the access statistics are kept at the server side. With the knowledge of its own Web pages, the server can provide prediction of its Web pages, however not of other Web servers. From the client’s point of view, prefetching is not constantly possible and can not be performed across servers. Furthermore, the summarized access pattern is a general user access pattern and may not correctly reflect the access patterns of each individual user. In Statistical Client-Initiated Prefetching, the access statistics are maintained at each client computer. As it is based on the past access patterns, prediction for non-accessed pages is impossible. An alternative is Proxy-Initiated Prefetching, where access statistics are kept at a proxy server. Since the proxy server can have knowledge of the access patterns of a group of users, the representation becomes less general and more typical as compared to server-initiated prefetching. Further, prediction across servers is made possible and from the client’s point of view, prediction for non-accessed pages is possible to some degree. As Web pages are prefetched based on statistical prediction, some prefetched Web pages may not be requested by the client at all, thus wasting the bandwidth for prefetching these pages. However, as these pages are cached in the proxy server, they may still have a chance to serve other clients. In this way, proxy-initiated prefetching is more efficient than statistical client-initiated prefetching. In Hierarchical Prefetching there are usually two levels of proxies. A local proxy serves a user or a small group of users while a remote proxy serves a larger group. Access statistics are utilized on both of them.

Besides these models which utilize access statistics are Deterministic Client-Initiated Prefetching and Lookahead Proxy-Initiated Prefetching. The former model allows users to specify a list of URLs they access regularly and prefetches them during less busy periods. The latter model simply prefetches a predefined number of Web pages that are linked to by the currently viewed Web page.

However, a shortcoming with all of these models is that they fail to take into account that individual users have their own specific interests and usually have a specific objective when accessing Web pages. Analyzing the content of the Web pages they have already visited can help to gain an understanding of these interests, which can be used to predict what they are likely to request next. This paper will introduce a new prefetching model which is based on content analysis. An overview of the model and underlying technologies are given in section 2, while experimental results are presented in section 3, and a number of conclusions are made in section 4.

2 Overview of Prefetching Model

2.1 System overview

The proposed model can be classified into the category of proxy-initiated prefetching, as it integrates the prefetching and caching functions in the proxy server. Users are registered with the prefetching proxy server, and the server maintains two profiles for each user, a user access log and a periodical user interest profile. The user access log records a specific user’s access history, while the periodical user interest profile records the user’s
interests at certain time periods, such as at different hours of the day. At the beginning of a particular user’s Web session, the user’s current interests are deduced from these two profiles by considering both frequency and recency factors. The deduced result is maintained for the duration of the Web session as the user’s current interest register.

When a user accesses a Web resource, the user request is routed to the proxy server which will resolve it with the cache if possible, or directly with the remote Web server if the page does not exist in the cache. The newly requested Web page is saved in the cache and its content is analyzed. Content descriptors are extracted and saved in the corresponding user access log and a site-wide global information file—a knowledge base of all accessed Web content of a site. Moreover, the information in the user’s current interest register is adjusted immediately with descriptors from the newly-accessed page.

Next, the links included in the Web page are collected. For each linked page, content descriptors are obtained either from the global information file (if the page was already previously accessed) or from the anchor text of the referencing link. These descriptors can now be compared with the information in the current interest register to determine which linked pages match the user’s current interests. The \( n \) Web pages that match the user’s interest most closely are then prefetched, where \( n \) is an adjustable parameter. The prefetched Web pages are then cached in turn and their content descriptors are extracted and saved in the global information file. However, no prefetching prediction will be performed on any of the prefetched Web pages until they are actually requested by the user.

The information in the user’s current interest register is adjusted with the feedback from the periodical user interest profile at the beginning of each time slot (e.g. once an hour). It will then contribute back to the periodical user interest profile at the end of the time slot.

### 2.2 Keyword extraction

User interests are captured from Web pages according to the keywords extracted. However, traditional methods of automatic keyword extraction are not suitable for a Web environment as they tend to be domain-specific and do not scale well to large numbers of documents, while the Web contains documents from unrestricted subject areas and a virtually unlimited number of documents.

Fortunately, what is going to be analyzed is an HTML document, and the structure of such a document is available through HTML tags. In an HTML document, one can easily tell whether a term appears in the title, one of the six headings or whether it is emphasized by using underlining, italics or bold characters. The terms appearing in the title, header, or that are emphasized in the text are usually more important than the other terms in a document. Further, the text placed in an anchor tag from a referring Web page is usually the best descriptor of the content of a Web page since it is placed by the authors, who by definition have the best knowledge of the content of the Web page. Related applications have also made successful use of this approach, such as Cutler et al.’s Web-based Information Retrieval System [2].

Six classes of HTML tags are considered: (1) title, (2) header, (3) anchor, (4) font size, (5) emphasized (e.g. strong, emphasized, bold, underline, italic tags), and (6) table caption class. An importance factor is assigned to each class of tags. Keywords are then extracted by following method: (1) extract the terms from the anchor tag of the referring Web page;
(2) locate the other five classes of tags within the Web page and extract the embedded
terms; (3) remove the stop words (e.g. the, of, etc.) from the extracted terms and stem
the remaining terms; (4) assign a proper weight to each extracted term. The weight \( w_t \)
of term \( t \) will be calculated as \( w_t = \sum_{i=1}^{k} t f_i \cdot I f_i \), where \( t f_i \) is the term frequency of \( t \)
embedded in tag \( i \) in a Web page, and \( I f_i \) is the importance factor of tag \( i \). If a term is
embedded by more than one tag, the weight of the different tags is accumulated.

The resulting weighted terms, called content descriptors, are represented by a vector
of terms. Vectors of Web URLs are of the form \( URL = (U t_1:U w_1, U t_2:U w_2, \ldots, U t_i:U w_i) \), where \( U w_i \) is the weight, or importance, of term \( U t_i \) assigned to the Web page.
The user interest register, \( INT \), is also a vector which is of the form \( INT = (I t_1: I w_1, I t_2: I w_2, \ldots, I t_k: I w_k) \), where \( I w_k \) represents the weight, or importance, of term \( I t_k \) that represent the current interest of a user. The similarity computation measuring
the similarity between a Web page vector and a user interest vector is done by the cosine
measure [3]:

\[
SIM(INT, URL) = \frac{\sum_{i=1}^{k} I w_i U w_i}{\sqrt{\sum_{i=1}^{k} I w_i^2 \cdot \sum_{i=1}^{k} U w_i^2}}
\]

The similarity function returns a figure in the range 0...1. It can be compared with a
threshold or alternatively with other Web pages to determine which pages to prefetch.

**2.3 User interest analysis**

Psychological research on human memory suggests that the strongest predictors of recall
accuracy are frequency (how often the item was seen), recency (how recently the item
was seen), and spacing (the gap between item presentations). Furthermore, the results
show that as the frequency or the recency of document accesses increases, the probability
of access on the current day increases exponentially. Recency also proved to be a much
better predictor than frequency, according to Pitkow [4].

Instead of simple statistics, a set of deflating factors \( DFV \) similar to the following will
be incorporated in the prediction of the current user interest.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>&gt; 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>( d_j )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( DFV(d_j) )</td>
<td>1.0</td>
<td>0.6</td>
<td>0.38</td>
<td>0.27</td>
<td>0.23</td>
<td>0.19</td>
<td>0.17</td>
<td>0.15</td>
<td>0.14</td>
<td>0.11</td>
</tr>
</tbody>
</table>

The unit for \( d_j \) is days, i.e. on day 0 (the current day), the deflating factor is 1, while
it decreases day by day thereafter. Modifying these factors will determine the importance
of past interests in determining the user’s current interest.

When calculating the current user interest at the beginning of a Web session, the sys-
tem reads records from the user access log, finds out the elapsed time since the last access,
decreases the weights of content descriptors by applying corresponding deflating factors
and summing them accordingly. Finally, the \( n \) content descriptors with the highest weight
are placed in the current interest register.

During a Web session, for each Web page requested by the user, its content descriptors
are used to update the current interest register. In order to reflect the changing access
patterns of a user, another temporal deflating factor is introduced. The weight of the
content descriptors in the register are decreased by this deflating factor before merging
with the content descriptors from the new Web page.
In practice, many people recall items on a periodic basis. For instance, reading an online newspaper at 9:00 am and searching for entertainment information at 5:00 pm. The interest analysis based on frequency and recency factors alone can not capture these periodic interests. This is why the use of the periodical user interest profile is incorporated in the interest analysis. At the end of each time slot, the value of the current interest register is recorded down into the corresponding record (time slot). If there was a value before, the weight of the content descriptors will be decreased according to the time elapsed by applying the corresponding deflating factor mentioned above, and then merged with the new ones. At the beginning of the next time slot, this information can be used to adjust the value of the current interest register in turn. Deflating factors are applied to the content descriptors read from the periodical user interest profile in a similar fashion.

2.4 User identification

In order to maintain user profiles, the proxy server needs to be able to identify each individual user and maintain the user profiles correctly. A possible approach is by providing an entry point or registration page for users who want to use the prefetching service, and let them specify their preferences (e.g. aggressive or conservative prefetching; personal interests) as well as their IP address.

A Web session has to be identified as well so that the correct user profiles can be identified and the appropriate initial values for the current interest register can be deduced. For users who have exclusive use of a given IP address, such as typical office users, the proxy server can simply identify the user by IP address (after having done a one-time initial registration). For mobile users, however, or for users of shared workstations it will be necessary to identify themselves through an authentication page hosted by the proxy server. Once authenticated, the proxy server will maintain the users’ information appropriately for the remainder of the session.

3 Experimental Results

3.1 Keyword extraction

A simple HTML parser was developed in Perl, using the word stemmer from Lovins [5]. With this parser, a number of Web pages taken from the fields of entertainment, science, commerce, news, personal, etc. were analyzed. The results obtained were quite satisfactory: several ten to several hundred keywords were extracted from a Web page, depending mainly on the size of the Web page. However, since the terms that can identify the contents well are usually the content descriptors that have the highest weight, only the terms with the highest weight are used in the similarity computation.

3.2 Similarity computation

Experiments on similarity computation were carried out for a group of Web pages using the content descriptors extracted by the HTML parser mentioned above. Results show that the Web pages with related contents have a higher similarity (up to 0.559) while the other non related Web page have a similarity of 0 to each other. The results agree with
the expected outcome. Experiments were also carried out on a personal Web cache, the results show that the user interest and the Web page has an inclusion relationship, that is user interest usually contains more content descriptors than a particular Web page. The prefetching threshold cannot be too high (e.g. not higher than 0.4); alternatively the Web page that has a higher similarity among the others can be prefetched.

4 Conclusions

This paper has presented a new Web prefetching model, integrating methods which have proven effective and efficient. Most of today’s prediction algorithms are based on simple access statistics and fail to incorporate document content as well as recency factors. While access statistics build up knowledge of past facts, it can not discover new knowledge based on these facts, and its ability to adapt to changing user access patterns is low. The prediction approach introduced in this paper, on the other hand, attempts to overcome this limitation.

References


O Kit Hong, Fiona is a computer officer in the Computer Service Bureau of the University of Macau. She graduated in Computer Engineering from Hua Qiao University, Fujian, China in 1989, and is currently a part time M.Sc. student in Software Engineering at the Faculty of Science and Technology, University of Macau.

Robert P. Biuk-Aghai is a lecturer in the Software Engineering Programme of the Faculty of Science and Technology, University of Macau. He received an M.Sc. degree in Information Systems from the University of London, U.K. in 1990, and is currently pursuing a Ph.D. degree in Computing Sciences at the University of Technology, Sydney. He is member of the IEEE Computer Society and the ACM.