Object Prefetching Using Semantic Links

Alexander P. Pons
University of Miami

Abstract

To date the most common means of gaining access to the Internet continues to be via dial-up modem connections. These slow communication channels significantly affect the rendering of the majority of web pages. Higher speed communications channels can alleviate rendering latency but based on the web page's content, delays still are incurred. The technique of web object prefetching can expedite the presentation of web pages by utilizing the current web page's view time to acquire the web objects of future web pages. The proposed Semantic Link Prefetcher utilizes information associated with the current web page's hyperlink set to predict which web objects to prefetch during the limited view time interval of the current web page. The results presented in this paper show that this proposed method can be effective in improving a web browser's cache-hit percentage, while significantly lowering web page rendering latency.

ACM Categories: H.3.3, H.3.4

Keywords: Web-Prefetching, Semantic Links, Web-Application

Introduction

One of the major issues affecting the use of the World-Wide-Web (WWW) and the Internet consists of web page rendering delays often incurred by web clients. These delays are mostly attributed to the transmission of web pages consisting of a large amount of multimedia, and less with the computational performance of the server and client systems (Cherkasova et al., 2003). A web site's composition is typically a collection of web pages that address an underlying topic, service, and/or application. These web pages are linked, structured, and organized through hyperlinks as a web-application that provides the required functionality to customers, who seek to gain greater access to information, services, or products. The composition of these web-applications varies significantly in content type and use of multimedia, which prolongs the rendering latency associated with aggregate object retrievals.

The availability of high-speed communication to the general public adversely affects the rendering time of most web pages. Despite the increase in the adoption of broadband Internet connections, which more than doubled from 9.1 percent (9.9 millions) in September 2001 to 19.9 percent (22.4 millions) in October 2003, over one-third of Internet (38.6 million) users access the Internet through dial-up

connections (U.S. Department of Commerce, 2004). Of the users that use these high-speed services, about 9.3 million customers are DSL subscribers and 12.6 million customers are cable modem subscribers. Although the number of broadband subscribers increased, a PEW report (Horrigan, 2003) suggests that growth may be moderating, in fact nearly three in five (57%) dial-up users say they have no interest in having a faster connection at home, while 38 percent say they would like to upgrade to broadband. According to the PEW report (Horrigan, 2004), the 40/60 ratio of dial-up users who do not want broadband versus those that do remained stable since October 2002. Since, the number of dial-up users is expected to remain relatively the same; modem speeds will continue to affect user attitudes (Nielsen, 1999a). Although the modem speed of users' connections in the six months between two GVU surveys nearly doubled, increasing from 28Kbps and 33Kbps in early 1998 (GVU, 1998a) to 56Kbps in late 1998 (GVU, 1998b), complaints about connection speeds only decreased about nine percentage points, from about 65% to 56% of all respondents (GVU, 1998c).

To most Internet users, accessing web-applications seems slow, a perception worsened as web page component makeup increases in file size and number. These web page objects (graphics, pictures, audio, and video) are received from the origin or proxy server on demand, when the user navigates to the linked web page and the web browser's cache does not contain these objects. The main problem with this scenario is that the user must wait for the requested web page's objects to be retrieved, only after the page is selected, which contributes to the display latency of the web page.

Consider an individual accessing an instructional web site that covers various aspects of a specific topic. The web pages are rich in multimedia to effectively depict the information and contain several hyperlinks to supporting or expanding web pages. Using an on-demand approach, the individual must wait to view the selected web page until its entire set of context objects (text and multimedia) are received at the client system, starting from the time the hyperlink is clicked and ending when the last web page object arrives. Although higher communication speeds alleviate such latency, delays exist and affect to some extent the individual's inconvenience in web site browsing. For dial-up connections these delays are extensive, and while these delays are reduced for cable modem or DSL connections, they are still noticeable and can be improved.

A potential improvement would be to anticipate to which web pages the user will navigate in the future, and acquire their content prior to the user's request. This technique is known as web object prefetching. During the current web page's viewing time, the next anticipated web page's objects are retrieved. Prefetching augments the operation of web browser caching in that if an object is not found in the browser's cache, a successful prefetch would have acquired the object already, and made it available to the browser, thereby eliminating the delay from the user's perspective. The ideal prefetcher offers a browser all the objects it needs (coverage) and only the objects it needs (accuracy). The use of any type of prefetching can contribute to an increase in network congestion as additional web page objects are sent to a client's system. This may be exacerbated by higher-speed and shared communication mediums, but can be alleviated by utilizing threshold values to limit a prefetcher's activities and reduce the total number of prefetched web objects according to characteristics of the transfer medium. According to our empirical results, the delays affecting web page rendering was improved across all communication channel speeds using web page object prefetching, which translates to faster and more effective web site browsing. In this paper, we propose a prefetch technique that reduces the rendering delay of web pages by increasing a browser's cache hit percentage. This novel technique adopts the theory of semantic links to the area of web object prefetching in order to exploit the semantic association among structurally linked web pages.

The remainder of this paper is organized as follows. The next section reviews some important related work concerning web-prefetching techniques. Then, we discuss the proposed semantic prefetcher and follow up with a section that highlights the approach with an example. Then, the next two sections introduce the experiment design and explain the performance comparison results, respectively. Finally, we provide concluding remarks.

Related Work

In the past ten years a significant amount of work has been conducted to enhance the performance of web page loading and presentation. These efforts have focused on various techniques related to the delivery process of web pages, such as client-side, server-side, and hybrid client/server methods, as well as the usage of proxy servers. These techniques have supported the concept of web page prefetching in many forms by providing methods that have increased the speed of web page delivery. The

list of related work is divided into two parts, consisting of historical-based and context-based techniques. The following seven techniques are historical-based:

- Padmanabhan and Mogul (1996), Bestavros (1995; 1996), and Albrecht et al. (1999) introduce a server-initiated prefetching approach, wherein the web server maintains per-client usage statistics, and determines a page request interdependency graph-based Markov model. When a client requests a page, the server sends along with the page the names of the most likely subsequently accessed pages, and leaves the initiative for prefetching to the client.
- Markatos and Chronaki (1998) and Hine et al. (1998) combine a server's knowledge of its most popular pages (their Top-10) with client access profiles. A client determines how much it will prefetch from the list and when it will start prefetching.
- Cunha and Jaccoud (1997) use a prefetch model that focuses on the client being active in gathering usage information and making prefetch decisions. Their approach uses a mathematical model that combines link categorization and a Markov model.
- Pons (2003; 2004) uses a Markov model that is downloaded to the client's system that becomes personalized to the client's distinct behavior. Each web-application sends an initial Markov model to the client that learns and on subsequent web-application accesses can more accurately predict prefetch objects.
- Duchamp (1999) indicates that a collaborative effort between the client and server works best for accurate prefetching. The server uses a Markov model built from client data to dispense information to clients, allowing them to perform prefetching according to their own needs by using different algorithms.
- Fan et al. (1999) proposes a prediction algorithm based on the Prediction by Partial Matching (PPM) studied by (Vitter & Krishnan, 1996) that demonstrates a relationship between data compression and prediction. They construct an m-order Markov and consider a prediction depth of more than one. The model predicts not only the next page, but also which pages will be requested after that. The m-order predictor uses the context of the past m references to predict the next set of prefetch pages for the client.
- Yang and Zhang (2001) present an integrated architecture, in which a certain amount of caching space is reserved for prefetching, such that the prefetching systems operates in a seamless manner with the existing Web caching systems to increase retrieval performance. Web caching and document prefetching is integrated using a prediction model built by mining the frequent paths of the aggregate behavior of users from Web logs.

The following research differs from the previously mentioned work in that it suggests the utilization of the content of the current web page to decide which links on the page to consider for prefetching. While the previous predictive techniques look at past actions as a basis to predict the future, these techniques base their link selection on the hypertext characteristics of the current web page. These approaches can make predictions of actions that have never been taken by the user, and can make predictions that reflect current user interests.

- Ibrahim and Xu (2000) describe a neural net approach to prefetching, using clicked links to update weights on anchor text keywords for future predictions. The link ranking of a page is determined as a score computed through an artificial neuron.
- Davison (2002) proposes predicting a user's next action based on analysis of the content of the pages recently requested by the user. Predictions are made using the similarity of a model of the user's interest to the text in and around the hypertext anchors of recently requested web pages.
- Zhuge (2003) presents a high-level Single Semantic Image among versatile resources using and defining semantic links between these resources. There are various types of semantic links defined to connect resources and to determine the manner of their use. The application model employed in his work consists of an Active e-Document Framework and a resource-browser that accepts keywords or the relationship between topics to retrieve corresponding documents. In order to improve the quality of the search result set, these relationships are associated with the different semantic link types between the documents. This concept has been expanded in (Zhuge, 2004) to include the retrieval of related images using semantic links, since these semantic links reflect the semantic relationships between images. As in the Active e-Document

Framework, Zhuge has derived reasoning rules from the semantic links to enhance retrieval based on the established semantics links. Zhuge et al. (2004) present the design and implementation of the Semantic Link Network (SLN) Builder and the Intelligent Semantic Browser. The SLN-Builder is a software tool that enables definition, modification, and verification of semantic links, and can tie semantic link definitions into XML descriptions. The Intelligent Semantic Browser is used to visualize the semantic link network between objects and to retrieve semantically related objects.

Our research combines both the historical-based and context-based techniques for web object prefetching. The approach proposed in this paper is similar to the methods that use the content of the current web page to decide which links on the page to consider for prefetching, except that we have combined the semantic link information (explicitly embedded) with past user behavior actions (hyperlink access frequency) to determine the user's most likely next action. The semantic links are established during web-application design, according to the relationship that exists between the web pages, and are combined with Markov link access statistics during web page retrieval to further examine similar semantically linked web pages, and identify and retrieve future web page objects.

Semantic Link Prefetcher

We define the Semantic Link Prefetcher (SLP), as a component of the client's web browser that interprets the semantic links in the current web page to determine which future web page objects to prefetch. When a web page is retrieved by the browser, it may include a set of hyperlinks that the client can follow. From this set of hyperlinks, the browser will identify which links it should pursue for object prefetching. By utilizing semantic links, the browser is afforded information concerning the meaning of the links, and as such, through established precedence rules, it accesses those most likely to be selected by the user. The semantic information is a fixed element of the link that is dynamically augmented at the server with click frequency statistics. In other words, if more that one link is associated with the same meaning or significance, the browser will select the links that represent the most often selected link. In the case that two or more links share the same meaning and statistic, then the relative order of the link on the web page will determine which one to consider first for object prefetching. Therefore, the semantic link information dominates in selecting the next web page objects. The use of frequency statistics can be replaced by any other statistic or held constant that infers an order for identical semantic links on a web page. We selected the access frequency statistic to compare the SLP to the 1-Markov model, but other measures could have been used. The statistical information is added to XML tags, including the names of prefetch objects within the web page before sending the page to the client. Thus, each web page hyperlink has a meaning, frequency, and list of potential prefetch objects.

The semantic link information is a static component of a web page that is associated with the page's hyperlinks during the web site's design. Therefore, the web page designer must not only determine a web page's composing hyperlink structure (their relative position), but must also add a semantic type to each hyperlink with XML tags. It is the explicit knowledge of the web page's content that facilitates the location and type of these hyperlinks during the design of the web site. The effort involved in assigning a meaning to a hyperlink is very low (similar to naming a hyperlink) and becomes even less when supported in web design tools. The web server maintains frequency access statistics on the number of times a hyperlink is selected with a set of hyperlinks for each web page.

In Figure 1, the operations of the SLP are presented. When the client's system request a web page, the server sends the web page to the client system including the semantic/statistic/object data associated with every link found in that web page. Once, the web page is rendered at the client's system, and until a link is clicked, there exists an opportunity to prefetch objects from web pages linked from this web page. During the web page view time the SLP utilizes the link data to initiate object prefetching. First, the SLP Link Analyzer uses the XML tag information to prioritize which objects to prefetch from a generated ordered list of links; secondly, the SLP checks the browser's cache and SLP prefetch area for the presences of these objects; and lastly, if the objects are not found, otherwise the SLP request these objects and stores them in the SLP prefetch area, anticipating a browser request for the objects, in which case the objects are transferred to the browser's cache.

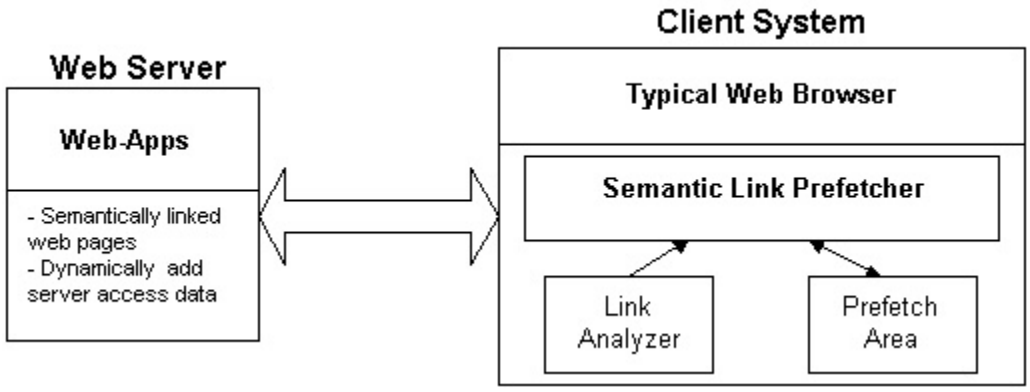

Figure 1. Prefetcher System Scheme

In general, prefetching is effective in reducing rendering latency of most web-applications, but in particular certain types of web-applications have the most benefit to gain. Of these, web-applications that consist primarily of static content will benefit considerably, while the ones that consist of mostly dynamic content would only profit in circumstances where the content can be associated to a client. When this is not possible, the performance of the prefetcher is negatively affected for unqualified dynamic content. Although these types of dynamic web-applications exist, some client-specific dynamic content can be anticipated through cache cookies, query URLs and dynamic content, which augments our basic model.

Semantic Link

Throughout the Internet hyperlinks have been the connecting components between myriads of web pages. A hyperlink associates two web pages in a directed and type-less manner; as such, all hyperlinks found on a web page possess the same meaning without distinction. Although the position of the hyperlink and surrounding text may suggest some meaning, it is too imprecise and potentially misleading to extract the relative importance or association of the hyperlinks to the referenced web page. A natural extension of the hyperlink to connect web pages is the semantic link. A semantic link is different from a hyperlink in that a semantic link represents a pointer with a type or meaning directed from one web page (predecessor) to another web page (successor). A semantic link reflects a certain type of semantic relevancy between two web pages. Table 1 shows the semantic link types defined by (Zhuge, 2003), ordered according to their relative semantic strength (descending). In addition, Zhuge also describes heuristic rules for connecting different types of links and for determining their semantic precedence. These semantic links are defined to establish an association between web documents for searching and retrieval, which extends naturally to web prefetching. However, this does not preclude that other semantic link types could be defined to amplify the meaning between these associations. Therefore, the semantic links presented in Table 1 are a basic set that could be expanded in future research.

The heuristic rules are not significant for the purpose of prioritizing semantic links in a web page, and therefore are not covered in our work; but the semantic precedence of the links is pertinent in establishing this technique's potential applicability to prefetching.

Link Type	Definition
Sequential link (seq)	The predecessor web page should be browsed or used before the successor web page.
Similar-to link (sim)	The semantics of the successor web page are similar to those of its predecessor web page.
Cause-effective link (ce)	The predecessor web page is the cause of its successor web page; the successor is the effect of its predecessor.
Implication link (imp)	The semantics of the predecessor web page imply the successor web page.
Subtype link (st)	The successor web page is a part of its predecessor web page.
Instance link (ins)	The successor web page is an instance of the predecessor web page.
Reference link (ref)	The successor web page is a further explanation of the predecessor web page.

Table 1. Semantic Link Types

The order relationship that exists between these semantic links is:

ref < *ins* < *st* < *imp* < *ce* < *sim* < *seq*,

such that the right most type reflects a stronger relationship between the two documents than the left most type. A *ref* type is similar to a type-less hyperlink that links two web pages together without any specific meaning, while a *seq* type conveys a very strong correlation between the two web pages.

When a web page contains more than one semantic link of the same type, then the statistic associated with the link type augments the priority of the link within the type. For example, consider a web page that contains five semantic links, three are *imp* links and two are *st* links as {(st1(30), st2(40), st3(30)} and {(imp1(25), imp2(75)}. Each set type contains elements of the form **type#(%)**, where the type is one of the previous semantic link types, the # indicates the relative position of the link within the web page for this link type, and the % is the frequency with which these links within the link type have been selected. The server is responsible for recording and maintaining the statistic information of a web page, and dynamically adjusting the web page's semantic links as the page is being sent to the client machine. Therefore, the browser would first attempt to retrieve the objects (not already in the browser's cache) associated with the *st* type links, but since there are three identical meaning links, the server statistics for these link types would provide a distinction between the links in the *st* type. In the case that all objects from st2(40) already had been retrieved and stored in the client's cache, the SLP would use the relative position between st1(30) and st3(30) to retrieve the objects associated with the st1(30), since they both share the same frequency statistic.

When semantic links of different types have frequency statistics that contradict the semantic information, adjustments to the semantic link type must be made. For example, consider these two semantic links, st(100) and sim(10), based on semantic the sim link is given higher priority, but with regard to frequency statistics the st link is of higher priority. This normally occurs when a link type has been improperly specified and requires adjustment. In most cases, manual adjustment can be performed, but generally, an automated method would be more efficient. As an example, on a periodic basis, the server can utilize a clustering algorithm to migrate links from one type to another type according to page link statistical information. At the most basic and simplistic level, the algorithm uses the frequency values to cluster web page links into as many clusters as there are link types on the web page. Any links not clustered into their semantic type are transformed into the cluster they reside and the link's frequency value adjusted to the cluster's centroid value. This dynamically moves a link up or down in the hierarchy as certain thresholds are reached. There are other means of reclassifying or verifying link types other than using access frequency and clustering, such as content analysis, which will serve as the focus of ongoing research.

A Semantic Link Prefetcher Example

The concepts covered in section 3 are illustrated through an example web-application consisting of six web pages denoted as {A,B,C,D,E,F}, with each web page referencing a set of objects. Figure 2 depicts the web-application structure containing the web pages and their respective hyperlink references.

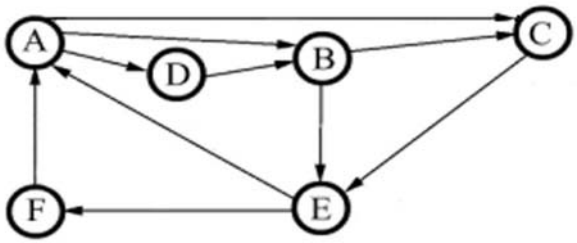

Figure 2. Web-Application Structure.

The web page's characteristics are shown in Table 2. The columns indicate the referenced web pages with the first row containing the web page's compositional objects. The additional rows show the semantic/statistic/location link information connecting a row (referencing) web page to a column (referenced) web page with the symbol, ---, which signifies no association between the web pages. The subscripted objects are identified with a numerical value (object location), its referencing web page, and a number in parentheses that depicts the size of the object in kilobytes. In the computations that follow, we assume that the client is using a standard 56.6k communication channel that transfers 7075 bytes per second when accessing the web-application. In addition, we assume that the time associated in retrieving the text content of the web pages is of negligible consequence compared to the time required for their compositional objects. It should be noted that our assumption of a constant rate 56.6k communication channel is used for demonstration of the proposed techniques without loss of generality, since network errors and other factors can affect the persistence of maintaining a constant communication rate.

Web Page	A	B	C	D	E	F
	$o_{1A(4)}$ $o_{2A(5)}$	$o_{1B(3)}$ $o_{2B(5)}$ $o_{3B(1)}$	$o_{1C(7)}$ $o_{2C(10)}$ $o_{3C(5)}$ $o_{4C(8)}$	$o_{1D(30)}$ $o_{2D(40)}$	$o_{1E(3)}$	$o_{1F(5)}$
A	---	**st1(25)**	**imp1(100)**	**st2(75)**	---	---
B	---	---	**st1(100)**	---	**imp1(100)**	---
C	---	---	---	---	**ref1(100)**	---
D	---	**seq1(100)**	---	---	---	---
E	**ref1(50)**	---	---	---	---	**ref2(50)**
F	**seq1(100)**	---	---	---	---	---

Table 2. Web Page Characteristics

When the user views the current web page "A" and the communication channel is available, the prefetcher will determine that the objects associated with web page "C" should be retrieved first, followed by the objects from web page "D," and then the objects from web page "B," since the *imp* link is of higher precedence than the other two *st* links, and st2 has a greater request value than st1. Therefore, the prefetcher starts the acquisition of objects $o_{1C(7)}$, $o_{2C(10)}$, $o_{3C(5)}$, $o_{4C(8)}$, $o_{1D(30)}$, $o_{2D(40)}$, $o_{1B(3)}$, $o_{2B(5)}$, and $o_{3B(1)}$, until the user clicks on a link to an actual web page. During the user view time of web page "A" (A_{vt}), a number of objects can be retrieved. Consider the potential number of objects that can be prefetched during the A_{vt} with a 56.6k communication channel. Assume that these objects do not currently reside in the browser's cache or in the prefetch area. If they did, they would be removed from the prefetch list. If the A_{vt} is 14.14 seconds, then all objects from web pages "C" and "D" will be prefetched. A shorter A_{vt} of 4.25 seconds retrieves all objects of the most likely next web page "C." Even an A_{vt} as small as 1.0 second retrieves at least one object for web page "C," thus reducing the page's rendering time. Without prefetching, the communication channel is idle, awaiting the user's hyperlink click during the time interval A_{vt}. If the user hyperlinks to web page "C," it would require 4.25 seconds of object download time (assuming objects are not in the browsers cache) before the web page is regarded as rendered. However, using prefetching, the channel is active during the A_{vt}, and acquires objects for web page "C" to render it without delay for A_{vt} longer than 4.25 seconds, as all objects for web page "C" would have been prefetched.

Semantic Link Prefetcher Analysis

Our SLP analysis consists of evaluating the advantages of using semantic link prefetching in comparison to the most common Markov type prefetching approaches. We selected the 1-history Markov model to compare to the SLP, and to establish a performance baseline, since a Markov-based prefetching technique is comparable to the SLP in computing complexity and resource consumption. Other higher order n-Markov models would gain in performance in a similar manner as our results indicate for the 1-Markov model, except that greater memory and processing requirements would be necessary. Comparison to other context-based methods would require extensive processing and memory usage at the client system to build and perform a neural network, or analyze the text of web pages, while the SLP uses minimal resources because semantics are built into the web page hyperlinks, and are not derived for each web page at the client's system. These methods require significant processing for each web page to determine which hyperlinks to prefetch, thus making it difficult to assess their relative performance.

The SLP combines semantic links and clickstream data (1-Markov) to deliver a solution which integrates link meaning with historical data. Initially, historical based prediction techniques perform poorly, since they require a learning period before becoming more accurate. The use of semantic links alleviates this problem by establishing priorities for which links to peruse first for prefetching in the absence of historical data. Once the historical data has accumulated, it is utilized to distinguish between identical semantic links for prefetching, and to dynamically augment semantic links prior to being integrated and transmitted as a static component of a web page. The following section discusses the manner in which a web-application Markov model is developed and maintained at the server-side to supply access frequency statistics to the semantic links.

Web-Application Markov Model

A 1-Markov model is maintained and managed at the server for each web-application build according to the clickstream sequences of web-application clients. Consider the web-application depicted in Figure 2 and the following three users' clickstreams.

User 1 { **A, B**, C, E, F, **A, B**, C }
User 2 { **A, D**, B, C, E, **A, D** }
User 3 { **A, D**, B, E, F, A}

The access patterns for User1, User2 and User3 comprise the sequence of web page accesses for each user and indicate that web page "A" is followed by a reference to web page "B," and so on. Using these users' references, a Markov model is built that approximates the references with transition frequencies. All transitions from node X to node Y in the diagram are assigned a value that represents the fraction of all references X that are followed by a reference Y. The arc values are computed as:

$P(Y \mid X) = Pr(Y = y \mid X = x)$.

The following definitions and notations are used to define the Markov model for a web-application and have been adopted from (Pons, 2003):

Let S be the set of all web pages composing a web-application.

Let T_x be the set of all the web pages that can be hyperlinked from the current web page x, where $x \in S$ and $T_x \subseteq S$.

$|A|$ is the number of members in set A.

F_x is the set of probabilities f_y associated with all possible hyperlinks from web page x to web page y, where $y \in T_x$ and $0 < f_y \leq 1.0$.

Now, applying these definitions and notations to the following model construction algorithm, a complete 1-Markov model is produced and maintained as more clickstream sequences are made available. The results of the algorithm are subsequently illustrated using the previous user clickstream data to generate the access frequencies stemming from node "A".

```
i = 1
While i < |S|  do
   x ← S_i
   For each member in T_x
      Compute f_y from client access patterns as P(y|x)
   i ← i + 1
End
```

Consider node "A" from the diagram, T_A = {B, C, D}. There are three transitions that commence from node "A," which has $F_A = \{0.40_B, 0.0_C, 0.60_D\}$ as hyperlink probabilities from web page A. The pattern "AB," "AC," and "AD" occurs 0.40, 0.0, and 0.60 percent of the time, respectively. The Markov model uses one previous web page reference to predict the next web page reference of a user that accesses the web-application and follows the same hyperlink sequence. When the user views web page "A," the model would predict that web page "B," "C" or "D" would be the next referenced web page, and would issue prefetch requests for the objects associated with each of these web pages.

SLP Evaluation Activities

The approach undertaken requires the following activities:

- ***Web-application identification:*** To evaluate the performance of the SLP, a web-application was selected that contains static and dynamic web pages. While the content of the web-application is mostly static, it does contain client-specific dynamic content and unqualified dynamic content. Only static web pages were used in the study, since unqualified dynamic web pages are difficult to use for predictions, and client information was inaccessible for client-specific dynamic web pages. The incorporation of these dynamic web pages in our study lowers cache-hit percentages, but since we are conducting a relative comparison, its affects are diminished. The web-application is an informational web site that provides its members and guests (restricted) access to current and past information on e-business. The web-application was selected mostly due to its largely static web page composition, and less on the service it provides. It has over 250 unique web pages, nearly 2,700 hyperlinks, and in excess of 2,200 unique objects that vary in size from 99 to 562K bytes and consist of banners, image maps, icons, and content specific images. The average web page contains 8.87 images per page, with a range of 1 to 25 images, and a standard deviation of 4.27. The average number of hyperlinks per web page is 10.70, with a range of 1 to 26 hyperlinks, and a standard deviation of 4.49. Furthermore, each hyperlink has been associated with a link type through a manual inspection, according to the semantic meaning between the current web page and its associated next web pages in order to evaluate the SLP performance.
- ***Hyperlink and semantic association:*** The task of assigning a semantic type to each of the hyperlinks of the web-application required a significant effort by our research team, which would have been greatly reduced if done during the design of the web site, or conducted incrementally as web pages were added to the

site. The addition of the semantic links would require no more work then adding an attribute to a hyperlink, which could be facilitated using a supportive web design tool. Since the establishment of the semantic information is done offline, its introduction would not affect scalability as the web site grows in content. The process consisted of taking one web page at a time and identifying all of its hyperlinked web pages. Next, we had to interpret the content of each of these web pages (both referencing and referenced) and augment the referencing web page's hyperlinks with a corresponding semantic link type, based on the relative interpretation of the hyperlink web page relationships. As previously mentioned, the web-application consists of web pages with many hyperlinks. Assigning link semantics to these higher hyperlink web pages was at times a challenge, and required an in-depth understanding of the web pages. This could have been alleviated with hyperlink-semantic association during the development of the web-application, since the web-application designer would have instinctive knowledge of the respective meaning of the web pages.

- ***User clickstream data:*** To evaluate the performance of the SLP, actual user interaction with web-application is required. For each user accessing the web-application, their actual session clickstream is extracted from the web server's logs, converted into trace-runs (sequence of web pages), and stored in a database. These trace-runs represent the sequence of web pages the user accessed during a session, and will serve to calculate how well a prefetching method performs. When a prefetch method predicts the user's next web page, the trace-run information is utilized to record the success or failure of the web page prediction.
- ***Trace-driven web browser simulator:*** A trace-driven web browser simulator (Smith, 1994) was developed to utilize the collected trace-runs to reproduce client-web-application interactions. A trace-run is a sequence of web pages followed during a client visit to the web-application that includes the amount of view time for each web page. Each trace-run is used to determine the number of cache-object misses encountered using demand-fetching (no prefetching) and prefetching, to compute the performance of the prefetcher compared to demand-fetching. For each trace-run, the simulator aggregates the number of total objects requests, the number of cache misses, the number of bytes transferred, and the time required to transfer the objects from the origin server utilizing a selected communication channel speed (dialup modem connection 56.6Kbps, cable modem connection 1Mbps, and DSL connection 1.5Mbps). At the start of each demand-fetching and prefetching trace-run simulation, the cache is cleared to establish a comparative baseline. The process of clearing the cache allows for worst-case results so that we can focus on the performance of the prefetcher, avoiding any affects associated with caching characteristics such as size and replacement policy. The following algorithm depicts the process conducted to perform the simulation for each trace-run. Let TR be a trace-run consisting of a sequence of accessed web pages wp_i, where $1 \le i \le |TR|$ and $|TR|$ is the length of the trace-run. The functions viewTime and transferTime return the recorded view time in milliseconds and the time required to transfer the specified object in milliseconds according to the selected simulator communication channel speed.

```
i = 1
Clear Cache and Prefetch area
While i < |TR|  do
    x ← wp_i
    y ← prioritized list objects to prefetch for x,
    not in cache or prefetch area
    sumTime = 0
    k = 1
    While viewTime(x) > sumTime do
        If transferTime(y_k) + sumTime ≤
        viewTime(x) then
            prefetchArea ← y_k
        End
        sumTime ← sumTime + transferTime(y_k)
    End
    Compute aggregate bytes and time to
    transfer prefetch objects
    Compute cache hits based on wp_{i+1}
    Adjust coverage and accuracy metrics
    i ← i + 1
End
```

Results Metrics

The study uses the metrics of coverage and accuracy to characterize the effectiveness of the SLP. The coverage is the fraction of cache object misses used to compare the performance of both demand-fetching and object prefetching. The demand-fetching coverage is a measure of the number of objects requested and found in the client's system cache. The prefetcher coverage encompasses objects in the client's cache and prefetcher area during a client's session with the web-application. The coverage provides an evaluation of the prefetcher's effectiveness in satisfying future object requests. To

compute the coverage and accuracy of the proposed SLP, let O_{req} be the total number of objects requested, O_{cache} be the number of objects found in the browser's cache, O_{fp} be the number of objects prefetched and residing in the browser's prefetch area, and O_{hfp} be the number objects in the prefetched area that are used to satisfy cache misses. Then, the coverage for on-demand fetching and prefetching is given as $C_{df} = O_{cache} / O_{req}$ and $C_{pf} = (O_{cache} + O_{fp}) / O_{req,}$ respectively, and the cache hit increase percentage (HIP) is $100 * (C_{pf} - C_{df}) / (C_{df})$.

The accuracy is the fraction of the total prefetched objects that actually were used to satisfy object requests preventing cache misses, given as $100 * (O_{hfp} / O_{fp})$. The accuracy of the prefetcher is a measure of the wasted communication channel bandwidth and the memory storage to handle unused objects. The ideal prefetcher offers a browser all the objects it needs (coverage) and only the objects it needs (accuracy). Apart from these metrics, additional facets are evaluated, such as the average decrease of transferred bytes and the reduction in web page rendering time.

Empirical Study

Evaluating the effectiveness of the SLP requires information regarding a client's actual usage of the web-application to be compared to the predicted client behavior. Trace-run data are issued to the simulator according to the order in which the web-application was accessed during client visits. The combined client click count for the web-application is over 2 million, which ranges from trace-runs as low as one to those as high as 234 click counts. The simulation measurements of these trace-runs are accumulated into demand-fetching and prefetching miss counts, and are used to evaluate the object selection technique. For the simulation of each trace-run, we assume that the browser's cache no longer maintains the objects for the web-application's pages, thereby requiring origin server object requests that focus on the prefetcher's performance.

The study requires actual client clickstream data obtained from the server's data-log files associated with the usage of the web-application. From the data-log files, 92% of the client click sequences are within a 1 to 50 range. Trace-runs outside of this range are excluded from the study, since they are attributed to automated web tools (robots/spiders). Incorporating these trace-runs in the simulations would adversely affect the performance of the SLP, since these web tools ignore the link semantic information embedded in the web pages. Therefore, trace-runs supplied to the simulator range from 1 to 50 client clicks. These represent a client's navigation throughout the website and are issued to the simulator according to the order in which the client accesses the web-application.

In Table 3, the aggregate results of the simulation are given, which include the performance for various simulation communication channel speeds: dialup modem (dm), cable modem (cm), and DSL (dsl). The odd rows list the results comparing the effects of utilizing a 1-Markov prefetcher (MP) to demand-fetching, while the even rows compare the effects of utilizing the SLP to demand-fetching for these communication speeds. These values are for all trace-runs, independent of the length of the individual trace-runs and the values represent the effects on a per client visit to the web-application.

In addition, Table 4 compares the performance of the MP and SLP based on trace-run length (number of clicks per client visit) and the selected communication channels speeds utilizing five categories, 1-10, 11-20, 21-30, 31-40, and 41-50. For example, a trace-run that consists of 15 clicks would be in the 2nd category.

Prefetching Technique	Hit Increase Percentage	Byte Transfer Decrease	Time Saving (Sec)	Accuracy Percentage
1-Markov (dm)	22.12%	102495	14.66	40.42%
SLP (dm)	32.26%	103731	15.49	46.06%
1-Markov (cm)	77.35%	263153	2.12	49.15%
SLP (dm)	83.21%	265082	3.11	52.07%
1-Markov (dsl)	82.46%	273261	1.31	49.97%
SLP (dsl)	86.57%	274677	1.52	53.94%

Table 3. Web-Application Prefetching Comparison

Trace-run Length	1-10	11-20	21-30	31-40	41-50
1-Markov (dm)	20.41%	30.50%	35.45%	45.90%	46.56%
SLP (dm)	31.22%	35.33%	41.68%	52.92%	54.98%
1-Markov (cm)	76.50%	79.92%	87.37%	92.90%	94.09%
SLP (cm)	82.54%	85.02%	92.93%	96.54%	98.25%
1-Markov (dsl)	81.68%	85.09%	92.04%	96.54%	97.53%
SLP (dsl)	85.91%	88.59%	95.66%	98.72%	98.97%

Table 4. Hit Percentage Based on Trace-run Length

This provides an indication of the prefetcher's performance according to the length of the client's click sequence. Results of over 50 accesses per client are excluded as discussed previously. These uniformly formed categories allow the performance of the SLP to be analyzed with regard to the user's clickstream length, in an attempt to characterize any correlations between performance and clickstream length. In order to facilitate analysis, the table combines the performance results generated for the various communication channel speeds and clickstream categories.

Performance Comparison

The benefits of the SLP compared to the Markov prefetcher were investigated. The observed results reflect advantages to the SLP approach over the Markov method. These results are attributed to the effects of the SLP's capabilities to predict more precisely the user's link selections when initially accessing the web-application. In other words, the semantic links provide a significant benefit when there is a lack of historical information required to differentiate among a web page's hyperlinks for prefetching. This is an important factor as web-application clients can quickly become frustrated with the performance of the web-application. A t-test was conducted between the means of each trace-run for both the MP and SLP, which produced a significant difference between the trace-run cache-hit result means, the least being ($T = -18.08$, $p = 0.00$).

The results in Table 3 indicate that the cache-hit percent value increased almost 10%, 6%, and 4% with the SLP compared to the Markov method, and that the accuracy also is greater than with the Markov approach by almost 6%, 4%, and 3% for a dialup, cable modem, and DSL connection respectively. In addition, the SLP outperformed the Markov method across all speeds in decreasing the number of bytes transferred and in increasing accuracy, which led to an increase in the average time savings per client. The accuracy percentage of the SLP is greater than that of MP since the SLP initially matches a client's semantic web-application access patterns, while maintaining access statistics on an ongoing basis to adapt the semantic links as a greater amount of client historical behavioral information is obtained. From the table, the relative performance increase is greater for slower communication speeds than for higher speeds, although the semantic prefetching technique still outperforms at these higher speeds. Therefore, augmenting hyperlinks with semantic information can benefit the rendering speed of web pages at all communications speeds, with much more noticeable effects for the slower communications speeds that are still a common means to access the Internet for many users.

From Table 4, the SLP outperformed the MP for all communication channel speeds and trace-run length categories. The increasingly higher cache-hit percentages for longer trace-runs are attributed to the greater number of objects requested and found in the cache and prefetcher area during the client's session. The important observation is the relative performance increase of the SLP compared to the MP for all categories, but in particular for the 1-10 trace-run length, which are the most critical in affecting a client's perception of a web-application's performance during initial access.

These results indicate the potential improvement for web-applications that utilize the SLP compared to a pure Markov approach. The results also identify the significance of the semantic link in accurately estimating a client's future web page requests. The high values for the SLP in Table 3 are indicative of the time savings contribution that the link semantics have on prefetching. The advantage of the SLP is that semantic link information dominates in identifying which objects to prefetch, and when combined with frequency access statistics, distinguishes between identical link types within a web page. This is a significant, as the semantic links are capable of reducing the reliance on training as in the Markov model, which requires an initial set of client accesses to establish starting frequency access values. Overall, our performance results are higher than the performance results of the pure Markov model, as our prediction method combines semantic link information with corresponding Markov frequency access

information to more accurately predict, and to provide for higher cache-hit percentages.

Conclusion

Many of today's web-applications are composed of a multitude of multimedia components. Unfortunately, multimedia prolongs the latency of web page rendering that is generally associated with greater and larger object retrievals. This has adversely impacted the effectiveness of a web-application when it is accessed, in particular over slow communication channels. Prefetching techniques have demonstrated their potential in reducing the latency associated with web page rendering. The use of semantic links connecting documents has been shown to improve the performance of search engine accuracy. We have extended the theoretical foundation of semantic links to define a web page object prefetcher that utilizes semantic links as the basis for determining which set of objects to obtain while the client views the current web page. The use of semantic link information and link frequency statistics has been shown to be an effective approach to reduce the rendering delays associated with web-applications and slow communication channels. These reductions are possible through the use of a prefetcher that operates within the limits of a web-application's hyperlink domain.

We have conducted trace-driven simulations that indicate that the combination of these two components is an effective means of reducing the latency encountered during web page rendering. These results exhibit a significant increase in the cache-hit percentage, which indicates that objects are being acquired before they are requested. In addition, the prefetcher's accuracy is increased according to the client's access frequency, since the semantic links provide for initial predictions during the establishment of the frequency access values. Currently, we are investigating the use of various types of server-side statistics to promote or demote semantic links that will adapt to the overall access behavior of the web-application clients. In addition, we are considering possible adjustments to the basic set of semantic link types. The concept of semantic link prefetching with access information is an effective technique in predicting future web objects, since the meaning of these links provides a prioritization that is established during the design of the web-application. The cost associated with augmenting hyperlinks with semantic information is minimal when performed as part of the web-application design, as hyperlinks are added to a web page by specifying an additional hyperlink attribute, this will be further simplified as web design tools incorporate the ability to indicate a semantic label to a hyperlink in a manner as straightforward as designating a font color for a hyperlink. The SLP has shown its ability to reduce the latency associated with access to web-applications, which positively affects web users.

References

Albrecht, D. W., Zukerman, I., and Nicholson, A. E. (1999). "Pre-Sending Documents on the WWW: A Comparative Study," *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI-99)*, Stockholm, Sweden, Vol.2, pp. 1274-1279.

Bestavros, A. (1995). "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, pp. 403-410.

Bestavros, A. (1996). "Speculative Data Dissemination and Service to Reduce Server Load, network Traffic and Service Time in Distributed Information Systems," *Proceedings of the International Conference on Data Engineering (ICDE '96)*, New Orleans, Louisiana, pp. 180-187.

Cherkasova, L., Fu, Y., Tang, W., and Vahdat, A. (2003). "Measuring and Characterizing End-to-End Internet Service Performance," *ACM Transactions on Internet Technology*, Vol.3, No.4, pp. 347–391.

Cunha, C.R. and Jaccoud, C.F.B. (1997). "Determining WWW User's Next Access and Its Application to Pre-fetching," *Proceedings of the 2nd IEEE International Symposium on Computers and Communications (ISCC '97)*, Alexandria, Egypt, pp. 6-11.

Davison, B. D. (2002). "Predicting Web Actions from HTML Content," *Proceedings of the 13th ACM Conference on Hypertext and Hypermedia (HT'02)*, College Park, Maryland, pp. 159-168.

Duchamp, D. (1999). "Prefetching Hyperlinks," *Proceedings of the 2nd USENIX Symposium on Internet Technologies and Systems (USITS '99)*, Boulder, Colorado, pp. 127-138.

Fan, L., Jacobson, Q., Cao, P., and Lin, W. (1999). "Web Prefetching Between Low-Bandwidth Clients and Proxies: Potential and Performance," *Proceedings of the ACM SIGMETRICS'99 Conference*, Atlanta, Georgia, pp. 178-187.

GVU. (1998a). "Connection Speed," http://www.gvu.gatech.edu/user_surveys/survey-1998-04/graphs/technology/q1.htm (accessed March, 2004)

GVU. (1998b). "Connection Speed,"

http://www.gvu.gatech.edu/user_surveys/survey-1998-10/graphs/technology/q01.htm (accessed March, 2004)

GVU. (1998c). "Dissatisfying Experiences," http://www.gvu.gatech.edu/user_surveys/survey-1998-10/graphs/shopping/personal/q208.htm (accessed March, 2004)

Hine, J., Wills, C., Martel, A., and Sommers, J. (1998). "Combining Client Knowledge and Resource Dependencies for Improved World Wide Web Performance," *Proceedings of the 8th Annual Conference of the Internet Society (INET'98)*, Geneva, Switzerland, http://www.isoc.org/inet98/proceedings/1i/1i_1.htm (accessed April, 2004)

Horrigan, J. (2003). "Adoption of Broadband to the Home," Pew Internet & American Life Project, http://www.pewinternet.org/pdfs/PIP_Broadband_adoption.pdf. (accessed July, 2005).

Horrigan, J. (2004). "Broadband Penetration on the Upswing," Pew Internet & American Life Project, http://www.pewinternet.org/pdfs/PIP_Broadband04.DataMemo.pdf. (accessed August, 2005)

Ibrahim, T. I. and Xu , C. (2000). "Neural Net Based Pre-Fetching to Tolerate WWW Latency," *Proceedings of the 20th International Conference on Distributed Computing Systems (ICDCS2000)*, Taipei, Taiwan, Republic of China, http://ece.eng.wayne.edu/~czxu/paper/dcs2k-pf.pdf (accessed April, 2004)

Markatos, E. and Chronaki, C. (1998). "A Top-10 Approach to Prefetching the Web," *Proceedings of the 8th Annual Conference of the Internet Society (INET'98)*, Geneva, Switzerland, http://www.ics.forth.gr/carv/r-d-activities/wwwPerf/INET98_prefetch/paper.html (accessed April, 2004)

Nielsen, J. (1999). "User Interface Directions for the Web," *Communications of the ACM*, Vol.42, No.1, pp. 65-72.

Padmanabhan, V. and Mogul, J. (1996). "Using Predictive Prefetching to Improve World Wide Web Latency," *Proceedings of the ACM SIGCOMM '96 Conference on Communications Architectures and Protocols*, Stanford University, California, pp. 22-36.

Pons, A. (2003). "Web Application Centric Object Prefetching," *Journal of Systems and Software*, Vol.67, No.3, pp. 193-200.

Pons, A. (2004). "Improving the Performance of Client Web Object Retrieval," *Journal of Systems and Software*, Vol.74, No.3, pp. 303-311.

Smith, A. (1994). "Trace Driven Simulation in Research on Computer Architecture and Operating Systems," *Proceedings of New Directions in Simulation for Manufacturing and Communications Conference (SIM94)*, Tokyo, Japan, pp. 43-49.

U.S. Department of Commerce (2004). "A Notion Online: Entering the Broadband Age," Economic and Statistics Administration National Telecommunications and Information Administration, http://www.nita.doc.gov/reports/anol/NationOnLineBroadband04.htm (accessed August, 2005).

Vitter, S. and Krishnan, P. (1996). "Optimal Prefetching via Data Compression," *Journal of the ACM*, Vol.43, No.5, pp. 771-793.

Yang, Q. and Zhang, H. (2001). "Integrating Web Prefetching and Caching Using Prediction Models," *World Wide Web Journal*, Vol.4, No.4, pp. 299–321.

Zhuge, H. (2003). "Active e-Document Framework ADF: Model and Platform," *Information and Management*, Vol.41, No.1, pp. 87-97.

Zhuge, H. (2004). "Retrieve Images by Understanding Semantic Links and Clustering Image Fragments," *Journal of Systems and Software*, Vol.73, No.3, pp.455-466.

Zhuge, H., Jia, R., and Liu, J. (2004). "Semantic Link Network Builder and Intelligent Browser," *Concurrency and Computation: Practice and Experience*, Vol.16, No.14, pp.1453 -1476.

About the Author

Alexander P. Pons is Associate Professor of Computer Information Systems at the University of Miami. Dr. Pons has a Ph.D. in Computer Engineering with extensive industry experience. His research interests include database design, object-oriented modeling, real-time systems and Internet technologies.