

Semantic prefetching objects of slower web site pages

Alexander P. Pons

University of Miami, Coral Gables, FL 33146, United States

Received 31 December 2005; received in revised form 30 January 2006; accepted 3 February 2006

Available online 15 March 2006

Abstract

The structure of most web sites consists of a composition of web pages that require varying amounts of time to render. Typically, web pages with large amount content (text/images/code) require more time to render than web pages with a smaller aggregate content size. Most users expect this discrepancy among web page rendering times. In this paper, we will describe a semantic link prefetching technique that uses object bundling to expedite the rendering time of slower web pages, at the cost of extending the rendering speed of faster web pages within an established threshold value limit. Study results show that our approach enhances the overall rendering time of slower web pages with imperceptible time extension to other, faster rendering web pages.

© 2006 Elsevier Inc. All rights reserved.

Keywords: Semantic link; Web prefetching; Object bundling

1. Introduction

The structure and composition of web-applications can vary significantly in their content type, particularly in the amount of multimedia they contain, prolonging rendering latency associated with aggregate object retrievals. An inherited characteristic of these web-applications is the rendering latency variability associated with its web pages, with some pages loading quickly, and others requiring more time. Therefore, narrowing the rendering time differences between these web pages promotes web page rendering time consistency, but more importantly, will enhance the loading time of slower web pages at the cost of imperceptibly slowing down faster loading web pages. We propose a technique that semantically bundles objects from slower loading web pages with objects of faster loading web pages to prefetch these objects for the client's system prior to arriving at the slower loading web page. The method combines the hyperlink structure of the web-application with semantic link information to select which slower loading web page objects to bundle with faster load-

ing web page objects, limited through empirically established rendering delay threshold values. The use of semantic link information increases the accuracy of object selection beyond the simple use of the hyperlink structure of the web-application.

Fig. 1 depicts the basic concept of semantic links and object bundling. The relative loading speed of web pages A, B, and C is $A < C < B$, shown graphically as the number of image objects in each web page (assuming all images have the same size). In addition, assume that the semantic links $A \rightarrow B$ and $A \rightarrow C$ have the same semantic strength (same prefetch priority). When web page A is requested by a client, then one of the images of web page B is bundled with the requested page and sent to the client's system, thereby expediting the rendering of web page B when requested and evening out the rendering speeds of all of the web pages. If the semantic strength of the link $A \rightarrow C$ is stronger than the link $A \rightarrow B$, then the bundle will instead be formed with an image from web page C; since the association from web page A to C is stronger, it implies a logical order of accessing the linked web pages. Once, the user returns to web page A the process would bundle any objects not in the client's cache from web page B during the rendering of web page A.

E-mail address: apons@miami.edu

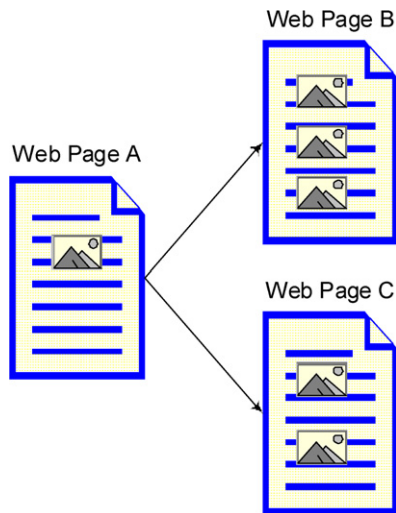


Fig. 1. Object bundling scheme.

The remainder of this paper is organized as follows. Section 2 reviews some important related work concerning Internet delays and web-prefetching techniques. Section 3 discusses semantic links. Section 4 covers the concept of object bundling. In Section 5 the proposed semantic link object bundling (SLOB) technique is presented. Section 6 provides an example that demonstrates the functionality and capabilities of the web object prefetcher. In Section 7, an empirical analysis is conducted to evaluate the performance gains using the SLOB technique. Section 8 provides a conclusion and future work directions.

2. Research on Internet-related delays

With the increasing popularity of the World-Wide-Web, the effects of slow computer response times mostly attributed to dial-up connections became salient to the general public. Notwithstanding the numerous advantages that the web provides to computer users, the slow speed of interaction has emerged as a serious hindering factor to web usage. Nielsen, a prominent usability engineer and web-design expert, has carried out several overviews of web sites to determine the most frequent design flaws and limitations to greater web usage. His survey of web sites of large corporations reveals that slow download times took first place with 84% of the sites being judged as too slow (Nielsen, 1999a,b). According to Nielsen's study, users are not sympathetic to waiting for web content; slow response times lead to lower levels of trust for the web site owner and always result in a loss of traffic to the site.

A large percentage of users connect to the Internet through a dial-up modem. Reports from a census conducted by Telecommunications Reports International (2000), show that an overwhelming number of subscribers of on-line services (93%) access the web through a dial-up modem (about 58 million users). Among the high-speed services, about 282,000 are DSL subscribers and 2.8 million

were cable-modem customers. Although, there has been an increase in the adoption of broadband connections to 9.3 million DSL subscribers and 12.6 million cable-modem subscribers, over one-third of Internet (38.6 million) users access the Internet through dial-up connections (US Department of Commerce, 2004). Now, according to a PEW report (Horrihan, 2003) the number of dial-up user is expected to remain about the same, which would continue to make dial-up connection speeds a concern.

2.1. Delay threshold of web pages

Many studies (Ramsay et al., 1998; Dellaert and Kahn, 1999; Jacko et al., 2000; Weinberg, 2000; Rose et al., 2001) have been conducted to determine the longest amount of web page rendering latency that web clients are willing to tolerate. Of importance to our work is to determine a threshold for extending the rendering delay of fast loading web pages. The studies conducted by Hoxmeier and DiCesare (2000) and corroborated by Galletta et al. (2004) show that any delays exceeding 12 s will adversely impact a web client's experience with the web page. These two studies are summarized as follows:

- Hoxmeier and DiCesare (2000) studied the effects of delay on user satisfaction, perceived power of the system, and intentions of repeated system use. The subjects browsed a simulated web environment and engaged in an information retrieval search task. The download delay was the independent factor with values of 0, 3, 6, 9, and 12 s. The results supported a significant relationship between satisfaction and delay, with satisfaction being highest in the 0-s delay condition. Satisfaction remained fairly constant throughout the 3- to 9-s range with a noticeable drop in the 12-s delay. The analysis of intentions of system reuse revealed a significant decrease in the 12-s category as well. Similarly, perceived power of the system was negatively correlated to the delay condition. However, user experience did not seem to have any effect on satisfaction in various delay conditions. The study supported the hypotheses that user satisfaction and intended system reuse in a browser setting are affected by the system response speed. It also revealed that in a simple search experimental subjects could consider a task environment delay of 12 s as a threshold value separating two patterns of user responses that differ in the level of acceptability of the wait.
- Galletta et al. (2004), the researchers extended the previous study by increasing the number of delayed conditions. Delays of 0, 2, 4, 6, 8, 10, and 12 s were studied under similar experimental setting. The goal of the study was to uncover a declining trend in performance and attitudes with increased delays, and to identify a critical delay length over which dependent measures exhibit significantly worse values. Consistent with the results of the first experiment, the data supported significant relation-

ships between delay and performance, as well as delay and attitudes. With increasing delay length from 0 to 6 s, performance and user attitudes toward the site showed a steady decline. In the range from 6 to 10 s, both dependent measures were observed to level off with only a slight decrease as the delay increased. At the point of 12 s another discontinuity in dependent measures occurred: performance as well as attitudes reached extremely low levels. This observation is in agreement with Hoxmeier and DiCesare (2000) who reported similar break at 12-s delay. The results suggest that under long delay conditions the cognitively simple search task ceased to be engaging and the subjects became unusually bored, and began to exhibit withdrawal behavior. None of the experiments provided information about the wait duration in the long conditions, nor was any feedback given to the subjects during the delay.

2.2. Improving web page delays

Many researchers have studied various methods to expedite web page loading and display to benefit dial-up user primarily, but also to a lesser extent to assist broadband users. The majority of these techniques are historical-based Markov model focusing on client-side, server-side, and hybrid client/server methods. While these techniques are effective once the client's behavioral patterns are defined, they require a client history that is often timely to develop and costly when the client alters their behavior. In contrast, context-based prefetch techniques use the content of the current web page to decide which links on the page to consider for prefetching based on the hypertext characteristics of the web page. Historical and content based techniques are described respectively as follows:

- These methods (Padmanabhan and Mogul, 1996; Bestavros, 1995, 1996; Albrecht et al., 1999) employ Markov models of page request interdependencies that vary according to the type and use of the statistics maintained to conduct predictions. The work of Markatos and Chronaki (1998) combines a server's knowledge of its most popular pages, a top-10 list of pages, with client access profiles. A different method (Hine et al., 1998) defines various client-browsing modes based on the number of client accesses to a server within a single client-browsing session to arrive at a prefetching scheme. The researchers (Cunha and Jaccoud, 1997) use a prefetch model that makes use of an active client that gathers usage information and makes prefetch decisions. The experiments conducted by Dunchamp (1999) indicate 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.
- These methods use the current web page context to make predictions of future client accesses without the

need of historical information. A neural net approach (Ibrahim and Xu, 2000) makes predictions according to weights established based on what "anchor text" have been clicked to rank web page link based on an artificial neuron computed score. A text analysis method proposed by Davison (2002) conducts content analysis of recently requested user web pages to predict the user's next web page request. The algorithm uses the text in and around the hypertext anchors of selected web pages to ascertain a user's interest in predicting the user's next actions.

3. Semantic link

Our technique uses semantic information associated with a web page's links to predict a user's next web page. The semantic information is explicitly embedded in a web page, providing more precise context-based information that improves on link significance using surrounding anchor text. The work conducted by Zhuge (2003) defines semantic links between resources to establish a high-level single semantic image among versatile resources. These semantic links establish relationship types between documents to improve on the quality of search result sets, where a resource-browser accepts keywords or topic relationships to retrieve corresponding documents. In Zhuge (2004) the theory, mathematics and formal structure of the semantic link network are comprehensively presented with various applications on the use of semantic links discussed.

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, i.e. they are context neutral. Although the position of the hyperlink and surrounding text may suggest some context, it is too imprecise and potentially misleading to extract the relative importance or association of the hyperlinks to the referenced web page. 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. The following semantic link types have been defined in Zhuge (2003):

- 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 is similar to that 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 implies 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.

The semantic link priority is: $\text{ref} < \text{ins} < \text{st} < \text{imp} < \text{ce} < \text{sim} < \text{seq}$, 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 link two web pages together without any specific meaning, while a seq type conveys a very strong correlation between the two web pages in that one follows the other in progression. These semantic link types serve as a basic set, which in future research can be expanded to include other link types.

When a web page contains more than one hyperlink of the same type, then the relative position of the links are used in establishing their priority. For example, consider a web page that contains five hyperlinks, three imp links and two st links as $\{\text{st}_1, \text{st}_2, \text{st}_3\}$ and $\{\text{imp}_1, \text{imp}_2\}$. Each set type (formed based on the same hyperlink type) contains elements of the form **type#**, where the type is one of the previous semantic hyperlink types and the # indicates the relative position of the link (numbered from top-to-bottom and left-to-right in increasing order) within the web page. Therefore, the browser would first attempt to retrieve the objects (assuming they are not already in the browser's cache or the prefetch area) associated with the st type links, but since there are three identical meaning links, the one encountered first would serve as the source for prefetching. In the case that all objects from st_1 had already been retrieved and stored in the client's cache, the prefetcher would use the relative position between st_2 and st_3 to retrieve the objects associated with st_2 next. This basic concept—augmented with object bundling and web page-threshold value—forms our technique for object prefetching. 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-application.

4. Object bundling

A web page is not considered “loaded” until all referenced objects are present and rendered at the browser. The number of distinct server connections that must be established for the objects in order to retrieve them at the client's system is another factor that impacts page rendering time, beyond object size. The work conduct by Wills et al. (2001, 2003) promotes and demonstrates that the transmission time of an object bundle is less than or equal to the transmission of the individual objects. Though the

aggregate object size remains the same in both cases, what changes is the time involved in establishing the connection for each individual object. Once the client system determines which objects it wants bundled and transmitted, it requests the object bundle from the server, at which time the server builds the object bundle and sends it to the client. The client machine then receives and unbundles the object in its prefetch buffer. The object bundling cost at the server-side and the unbundling of the objects at the client-side involves some computations, but given the processing power of modern systems, this is a minor effect when once considers the amount of time saved in opening and closing connections. An even greater benefit is realized with object bundling of secure web pages, which require additional processing in the form of authentication handshaking time for each individual connection. The technique of pipeline connection between a client and server would eliminate the need to establish multiple connections for each web page object as these objects can be retrieved through a persistence connection, but the technique is not universally support in most servers and routers. Although, object bundling is not absolutely necessary for our semantic prefetching technique, it could potentially reduce the time required to obtain the requested web page's objects, including objects marked for prefetch. Any reduction in the time to obtain prefetched objects is significant, since requesting unutilized objects translates to wasted bandwidth. Our object bundle definition varies from the previous works in that the object bundles are dynamically assembled at the server-side according to the client-side requested object list. The server combines the identified client objects together into a single file, maintaining their individual characteristics (URL, type, size, etc.) such that they may be identified and extracted at the client-side. The file format would contain the current web page's objects first, followed by any prefetch objects. By thus prioritizing these objects, rendering of the current web page can begin as soon as its objects are received, even while prefetch objects are being transferred.

5. Semantic link object bundling (SLOB) technique

While the basic approach to prefetching is similar in most of the previous studies, differences exist within the details of the methods. In this paper we present the novel concept of semantic link object bundling (SLOB) prefetching which employs object bundling through semantic link information. It is distinct from other approaches by improving the speed of slower loading web pages through faster loading web pages in using semantic link information to determine the order and objects to bundle together. The required information for the SLOB technique is statically incorporated into each web page during web page design and server-side augmented during web page downloading. During the design of a web page, semantic link information is incorporated into each of the page's hyperlinks through XML tags. The effort involved in assigning a semantic type

to a hyperlink is very low and becomes even less when supported in web-design tools. The server-side dynamic information consists of the objects (name and size) of the web pages associated to the current web page through the page’s semantic links. In addition, a download threshold value limiting the amount of object prefetching for the current page is embedded into the web page. This information can change throughout the lifetime of a web-application, making it more efficient to be automatically determined and incorporated into a downloading web page, since the composition of web pages can change without requiring modification to their semantic association. Therefore, during the transfer of a client request for a web-application page, an XML tag is added with a server-computed web page-threshold value (P_{tv}). The P_{tv} determines the slowest loading time for any page of the web-application. Therefore, if the currently requested web page’s loaded time is P_{lt} , then the prefetcher time value PF_t is computed as $P_{tv} - P_{lt}$.

The semantic links within the current web page specify the association type to page’s its hyperlinked web page, including all of the objects in these web pages. These objects are specified as $o_{X\#(size)}$, where X is the containing web page, $\#$ is the order in which the object is positioned in the web page (numbered from top-to-bottom and left-to-right in increasing order), and size is the number of bytes comprising the object (note that an object with $\#$ equal to 0 represents the text content of the web page).

Fig. 2 shows the respective prefetch information for the web pages found in Fig. 1. When the client navigates to web page “A”, its PF_t is determined according to the web page’s P_{lt} , computed from the size of the composing objects in the web page, in this case $o_{A0(size)}$ and $o_{A1(size)}$. Using the link type information from web page “A” and the objects found in the linked web pages, a proposed prefetch object list (P_{ol}) is built using link semantics and compositional objects. Assume that link type X and Y are st and imp, respectively. Then, the prefetcher would place the objects in web page “B” before the objects in web page “C” in the $P_{ol} = \{o_{B0}, o_{B1}, o_{B2}, o_{B3}, o_{C0}, o_{C1}, o_{C2}\}$. Assume

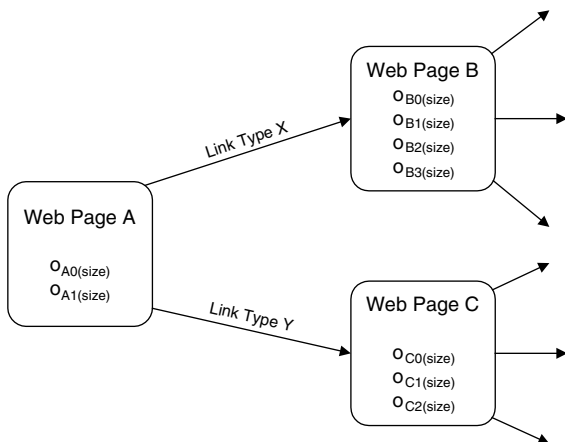


Fig. 2. Web page “A” semantic link structure.

o_{B0} and o_{C0} are negligible (size equal zero). Now, if o_{B1} , o_{B3} , and o_{C1} are chosen as prefetch objects, they will be converted into an object bundle and transmitted along with the requested web page “A” objects. An object in web page “C” would be prefetched over an object in web page “B” if the object already resides in the browsers cache or the prefetch area or its size would surpass the PF_t value, thereby allowing a web page “C” object to be used. The location of the object within the web page is used as a means of prioritizing the object for prefetching. This method does not preclude the use of other object prioritization criteria, such as object size, spatial position, and surrounding context, which is the focus of future research. Currently, we are researching these different criteria to determine their use in acquiring the highest benefit objects. Therefore, the link type establishes the main order of the objects, while their location serves as the minor order within the same web page objects.

6. A semantic link object bundling example

The concepts covered in Section 5 are illustrated through an example web-application consisting of six web pages, denoted $\{A, B, C, D, E, F\}$, with each web page referencing a set of objects. Fig. 3 depicts the web-application structure, with the semantic link and composing web page objects. The subscripted objects are identified with a numerical value (object location), referenced web page, and followed with a number in parentheses depicting 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 in accessing the web-application and that the server has established a P_{tv} of 4 s. These values are detected at the browser and downloaded with the current web page from the server respectively. In addition, the time associated in retrieving the text content of the web pages is represented through the definition of a text context object of each web page, such as for web page “A” is o_{A0} . 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 and will be further expanded in subsequent sections.

When the client requests web page “A,” we need to determine the value of PF_t in order to identify the amount of available time to conduct object prefetching. To determine the PF_t value, first the object from web page “A” is used to compute the P_{lt} value. Web page “A” contains three objects $o_{A0(1)}, o_{A1(3)}$ and $o_{A2(5)}$, for a combined size of 9 kilobytes, requiring $(9000/7075) = 1.27$ s; therefore, the prefetcher time value is $PF_t = P_{tv} - P_{lt}$ or $4 - 1.27 = 2.73$ s.

Once the PF_t value is computed, the prefetch object list P_{ol} is formed as $\{o_{B1(3)}, o_{B2(5)}, o_{B3(1)}, o_{D1(15)}, o_{D2(20)}, o_{C1(3)}, o_{C2(7)}, o_{C3(1)}, o_{C4(4)}\}$. 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 P_{ol} . Now,

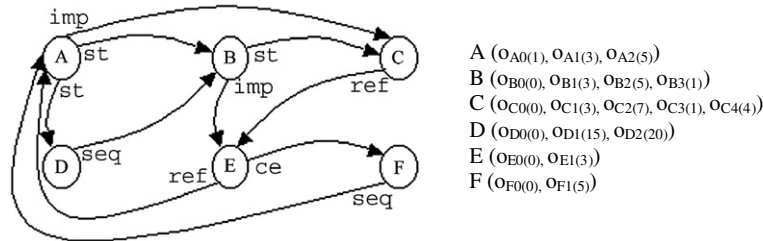


Fig. 3. Web-application structure.

using a simple greedy algorithm, the list is traversed from left-to-right selecting objects until the PF_t is reached without surpassing the value. From our example, the object sizes are converted to time values according to the channel's speed of 56.6k for $\{o_{B1(3)} = 0.4240, o_{B2(5)} = 0.7067, o_{B3(1)} = 0.1413, o_{D1(15)} = 2.120, o_{D2(9)} = 1.272, o_{C1(3)} = 0.4240, o_{C2(7)} = 0.9894, o_{C3(1)} = 0.1413, o_{C4(4)} = 0.5654\}$, selecting objects $o_{B1}, o_{B2}, o_{B3}, o_{C3}, o_{D2}$ with an accumulative time of 2.685 s. There is some wasted capacity of $2.730 - 2.685 = .045$ s for which a more complex object selection algorithms could better utilize; but with our simple greedy approach, various objects from different priority web pages have been selected which are semantically significant with respect to the currently requested web page. Once these prefetch objects are determined, a request is made to the web server to identify these objects. The server bundles the objects into a single unit and transfers it to the client for unbundling and storage in the prefetch area, with the anticipation that one of the predicted web pages will be navigated to next.

7. SLOB prefetcher analysis

The analysis focuses on measuring the coverage and accuracy of the SLOB prefetcher to evaluate its performance in selecting anticipated objects. These two measures provide an indication of the prefetcher's performance as coverage dictates whether objects requested are available in the prefetcher's buffer, while accuracy determines lost bandwidth associated with retrieved, but unused objects in the prefetcher's buffer. The ideal prefetcher offers a browser all the objects it needs (high coverage) and only the objects it needs (high accuracy). The coverage is the fraction of cache object misses when comparing both demand-fetching and object prefetching. The accuracy is the fraction of the total prefetched objects that actually were used to satisfy object requests and that prevented cache misses. In addition, the effects of object bundling are considered to assess its contribution to the technique of prefetching.

For the study, a typical web-application consisting of primarily static web pages (dynamic web pages were ignored) was chosen. The web-application offered its affiliated members current and past information on e-business. It has over 250 unique web pages, nearly 2700 hyperlinks and in excess of 2200 unique objects that vary in size from

99 to 562,996 bytes, consisting of banners, image maps, icons, and content specific images. The average web page contains 8.87 images per page, ranging from 1 to 25 images, with a standard deviation of 4.27. The average number of hyperlinks per web page is 10.70, ranging from 1 to 26 hyperlinks, with a standard deviation of 4.49. Furthermore, each hyperlink has been associated with a link type, according to the semantic meaning between the current web page and its associated next web pages, which is based on a set of design criteria to categorize the hyperlinks.

The approach undertaken requires the development of a web browser simulator in Fig. 4, the acquisition of trace data from web server logs to capture client web page requests, and the interpretation of the trace-driven simulation results. The trace-driven web browser simulator uses server-log data to reproduce client web-application navigation. Using the trace data, the simulator can determine the performance of the prefetcher in terms of the number of demand-cache object misses encountered with and without prefetching. The simulator uses the specified web page-threshold value, current web page load-time, semantic link information to make its prefetch object list, and then compares its selections to the actual users actions obtained from the web server logs. At the start of each client simulation run, the cache is cleared to establish a comparative baseline on which to base the prefetcher's performance. For each trace-run (a user clickstream) we assume that

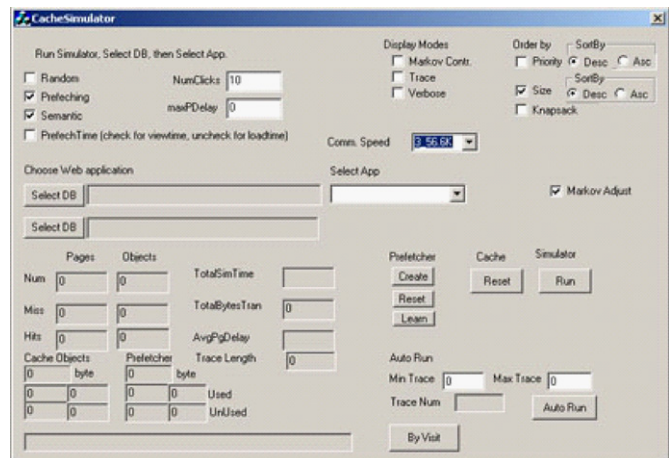


Fig. 4. Web browser simulator interface.

the browser's cache no longer maintains the objects for the web-application's pages, thereby requiring original server object requests. The simulator uses the time of the currently selected web page and the server threshold value to determine the available time in retrieving a set of predicted objects. It also aggregates various statistics, such as, 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 specified communication channel speed (dial-up connection 56.6 kbps, cable-modem connection 1 Mbps, and DSL connection 1.5 Mbps). In addition, an option for object bundling can be turned on, which incorporates the effects of object bundling in the computed statistics, using an object bundling factor empirically determined for each communication speed.

7.1. Empirical study

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 1–50 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 semantic prefetcher, since these web tools ignore the hyperlink 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.

Since the semantic prefetcher does not require historical data, it is not important to group trace-runs per client or IP; instead, the overall performance associated with all clients is ascertained. In Tables 1–3, the performance of the unbundled and bundled semantic prefetchers are shown respectively for dial-up modem, cable modem, and DSL communication channel speeds across a range of web page

delay threshold values. The tables show the increase in cache-hit percentage (coverage) and the amount of unused prefetch objects (accuracy) for all trace-run groups (aggregation associated with web page delays from 2 to 12 s). An aspect that is also investigated is the effect of varying the delay threshold value from 2 s up to 12 s, which is the maximum value tolerated by clients during web page rendering.

The performance of the semantic prefetcher when compared with demand-fetching is given in Table 1 when object bundling is not used. In addition, Table 2 further divides the aggregate performance of the semantic prefetcher into various categories. This provides an indication of the prefetcher's performance according to the length of the client's click sequence. Since the number of clicks for each client visiting the web-application varies significantly, we established five frequency access categories, consisting of 1–10, 11–20, 21–30, 31–40, and 41–50 trace-runs. Results of over 50 accesses per client are excluded as discussed previously. These uniformly formed categories allow the performance of the semantic prefetcher 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, delay threshold values and clickstream categories.

In Table 3, the performance of the SLOB is shown when semantic prefetching and object bundling are both used. The requested object bundle is composed of the current web page objects and the determined prefetch objects. In order to establish the object bundling factors supplied to the simulator for each communication channel speed, a series of object bundles were formed, transmitted and timed across the network. A total of 48 web pages were designed and divided into web pages consisting of individual images and bundled images. The individual web pages contain 2–25 images, while the bundled image web pages aggregate these images into a bundle corresponding to the number of images of the respective web page. The web server runs IIS 4.0, while the client uses a java applet to record the start and end times required to transmit the complete web page.

Table 1
Web-application client frequency access (unbundled)

Time (s)	Dial-up modem		Cable modem		DSL	
	Hit increase (%)	Unused prefetch (%)	Hit increase (%)	Unused prefetch (%)	Hit increase (%)	Unused prefetch (%)
2	3.43	12.71	38.77	15.03	42.65	14.22
3	5.60	10.48	42.41	14.21	46.66	13.66
4	7.66	12.18	44.84	13.87	49.94	13.69
5	10.43	11.53	47.15	13.59	53.03	13.54
6	14.15	11.57	49.29	13.64	55.42	13.16
7	15.40	11.49	52.33	13.53	56.40	12.62
8	20.14	12.43	53.56	13.49	59.10	12.26
9	21.71	12.73	55.30	13.24	59.81	12.11
10	24.05	12.73	55.82	12.88	66.64	11.79
11	25.28	12.72	56.42	12.56	70.03	11.63
12	27.21	12.77	58.99	12.36	78.24	11.37

Table 2
Web-application client frequency access

Time (s)		1–10		11–20		21–30		31–40		41–50	
		Hit increase (%)	Unused (%)	Hit increase (%)	Unused (%)	Hit increase (%)	Unused (%)	Hit increase (%)	Unused (%)	Hit increase (%)	Unused (%)
2	dm	1.62	30.66	2.75	9.91	5.99	11.24	2.80	5.88	3.01	9.50
	cm	26.54	32.85	37.19	10.37	43.31	7.86	36.97	21.38	35.33	13.89
	dsl	31.69	32.63	41.34	7.62	44.41	4.78	42.22	24.81	40.22	11.27
3	dm	3.81	24.91	4.45	8.37	11.40	6.90	4.39	3.71	5.11	7.89
	cm	31.48	32.54	40.97	7.65	44.41	5.37	42.22	24.03	39.78	11.41
	dsl	36.84	32.91	44.76	6.55	48.36	3.14	43.35	26.40	45.77	9.77
4	dm	4.78	26.61	5.90	10.67	14.71	9.10	5.85	5.61	6.57	9.14
	cm	33.87	32.75	42.92	6.79	46.90	2.78	42.22	27.70	43.57	10.13
	dsl	41.80	32.19	48.30	6.80	48.70	3.42	44.79	27.42	48.31	8.94
5	dm	6.36	25.71	8.34	9.42	17.74	7.25	7.62	5.14	9.91	9.43
	cm	37.75	32.91	45.57	6.62	48.36	3.36	43.35	26.29	46.36	9.41
	dsl	46.52	32.01	53.09	7.09	52.17	3.72	45.10	27.77	50.95	8.20
6	dm	8.13	25.20	11.90	9.93	21.03	6.58	9.20	6.24	12.18	9.90
	cm	41.08	32.33	47.21	6.77	48.61	3.50	44.79	26.99	47.79	9.01
	dsl	50.51	31.78	55.39	6.92	56.15	3.05	46.45	28.24	53.72	7.57
7	dm	9.49	25.24	12.67	9.66	22.86	6.07	10.62	6.73	13.27	10.47
	cm	44.93	31.87	52.44	6.75	51.10	3.50	45.10	27.99	50.10	8.44
	dsl	52.71	31.65	55.82	6.27	57.54	2.06	47.33	28.16	54.36	7.26
8	dm	11.68	27.47	17.85	9.63	27.10	8.31	15.38	6.51	17.84	11.94
	cm	47.40	32.00	53.67	7.13	52.92	3.65	45.89	27.79	51.46	8.15
	dsl	56.84	31.14	59.35	5.97	60.90	1.87	47.77	27.95	56.88	7.15
9	dm	13.88	27.43	18.64	9.97	28.25	9.27	16.96	6.70	19.75	12.46
	cm	50.07	31.85	55.31	7.01	56.15	3.16	46.45	28.12	53.57	7.70
	dsl	58.36	30.79	60.38	5.81	60.98	1.71	47.77	28.08	57.69	6.98
10	dm	15.42	27.83	20.84	10.86	29.81	8.88	20.43	6.80	21.70	13.07
	cm	51.50	31.72	55.57	6.61	57.54	2.55	46.45	28.22	53.93	7.44
	dsl	65.34	30.27	66.59	5.47	67.43	1.25	58.52	27.33	64.68	6.71
11	dm	16.12	28.47	21.98	10.99	31.44	9.26	21.38	6.90	22.78	12.00
	cm	52.84	31.63	55.82	6.24	57.54	2.01	47.33	28.16	54.39	7.25
	dsl	69.22	29.92	71.17	5.38	70.91	1.02	61.24	26.70	68.07	6.66
12	dm	16.96	28.93	24.49	10.52	33.24	10.15	23.22	6.37	24.10	11.98
	cm	56.66	31.22	59.35	6.06	60.90	1.98	47.77	28.04	56.76	7.18
	dsl	76.92	29.44	79.28	5.28	79.66	1.00	72.91	25.75	76.60	6.50

Table 3
Web-application client frequency access (bundled)

Time (s)	Dial-up modem		Cable modem		DSL	
	Hit increase (%)	Unused prefetch (%)	Hit increase (%)	Unused prefetch (%)	Hit increase (%)	Unused prefetch (%)
2	6.04	10.74	61.77	12.03	86.36	12.23
3	11.39	10.57	76.62	12.60	90.08	11.73
4	14.76	11.41	87.30	11.63	95.00	10.82
5	17.32	12.01	90.27	11.30	96.64	10.71
6	20.79	12.37	93.64	10.86	97.06	10.66
7	22.00	12.78	96.55	10.73	97.24	10.64
8	26.64	12.60	96.71	10.71	97.24	10.64
9	28.41	12.63	96.87	10.68	97.24	10.64
10	29.85	13.09	97.13	10.66	97.24	10.64
11	30.93	13.20	97.24	10.64	97.24	10.64
12	32.01	13.89	97.24	10.64	97.24	10.64

In order to focus strictly on the times associated with the transfer, the client and server machines were connected directly to avoid any delays attributed to network traffic.

Each web page was requested several times to compute an average time. Then, according to these results, an object bundling factor can be established. These values are con-

sidered optimal, since various physical characteristics and factors are not taken into account. While fluctuations in bandwidth performance, network traffic, and transmission anomalies can influence the computed object bundling factors, our results are consistent with the values published in the previously stated references on object bundling. The results indicate that the time for individual images increases as more images are retrieved, while for the bundled images the time is much more consistent, with slight increases due to the size of the aggregate image bundle.

7.2. Performance comparison

From Table 1, the results indicate that the semantic prefetcher outperforms demand-fetching for all communication speeds, while these unused prefetch ranged from 10.48% to 14.22% across all channel speeds and delay threshold values. For all communication channel speeds there was a continuous increase in cache-hit performance proportional to the increase in delay threshold value, from 3.43% for 2 s to a maximum of 27.21% for 12 s when using a dial-up modem, from 38.77% for 2 s to 58.99% for 12 s when using a cable modem, and from 42.65% for 2 s extending to 78.24% for 12 s when using DSL. Throughout all of these trials 85% or better of accuracy was maintained for all threshold values.

In Table 2, the aggregate results of Table 1 are divided in several trace-run length categories. From these categories, we can observe that the cache-hit percentage remains relatively consistent across all categories and communication channel speed. In addition, the effects of varying the delay threshold values correspondingly impacts the cache-hit percentage as the delays increase, since a greater number of objects are transferred to the client's systems with longer delays. A consequence of increasing the delay threshold values is that the accuracy of the semantic prefetcher is reduced, since both used and unused objects are retrieved at a greater quantity.

In addition, click sequences ranging between 1 and 30 clicks reveal a consistent increase in cache-hit percentage and a decrease in unused object percentage. The results in the range 31–40 clicks show degradation in the performance of the semantic prefetcher. This is attributed to two issues; first as the client hyperlinks to previous prefetched web pages, the semantic prefetcher will continue to retrieve objects that are unused, as the client does not subsequently access those web pages. Secondly, there was some incorrectly labeled semantic type hyperlinks discovered when analyzing the server log files after the fact. The results for 41–50 clicks are still impacted with the first issue discussed, but not with the problem of mis-typed semantic hyperlinks. Therefore, the performance of the semantic prefetcher is not significantly influenced by the length of the user's clickstream within the web-application. Improving these problems forms the focus of our current research in order to avoid over-prefetching and minimize the mis-labeling of semantic hyperlinks.

The use of object bundling improves the performance of object prefetching when object bundles are retrieved faster than individual objects. The approach of image bundling enhances the rendering time of predicted web pages, since the client obtains more of the requested objects during the prefetch period. From Table 3, the results indicate that the SLOB outperforms the semantic prefetcher for all communication speeds, although the accuracy of the SLOB is less, ranging from 10.64% to 15.03% across all channel speeds and delay threshold values. This was expected and verified in our study, since with object bundling a greater number of objects can be retrieved for the same amount of time used for semantic prefetcher (unbundled). For all communication channel speeds, there was a continuous increase in cache-hit performance proportional to the increase in delay threshold value, for dial-up modem from 6.04% for 2 s to a maximum of 32.01% for 12 s, cable modem from 61.77% for 2 s to 97.24% for 12 s, and DSL from 86.36% for 2 s extending to 97.24% for 12 s. An interesting observation is that the SLOB is saturated at threshold delay values of 10 and 6 s for cable modem and DSL speeds respectively. This implies that the SLOB has retrieved the maximum amount of objects possible, except for a few objects that were very large or outside the web-application's domain. The number and size of the images affect the time associated with unbundling, but with the processing speed of most client machines, this time becomes insignificant. These results indicate the potential improvement to web-applications using the SLOB technique which, despite its performance contribution to enhancing the client's web-application experience, requires substantially less recourses at the server and client sides to conduct web object predictions.

These results demonstrate the performance gains in using faster web pages to decrease the rendering delays of slower web pages using a semantic prefetcher. The results (cache-hit) depicted in the table increase as the channel's communication speed becomes faster, but can lower accuracy as more objects are retrieved from other less likely visited web pages. For faster communication speeds, we have found that accuracy is affected, but with additional gains in cache-hit performance. We are investigating various metrics to improve accuracy without significantly affecting cache-hit by using client-side information to establish prefetch limit thresholds.

8. Conclusion

The use of semantic links to connect documents has been shown to improve the performance of search engine accuracy. We have extended the theoretical foundation of semantic links in defining a web page object prefetcher that utilizes semantic links as the basis for determining which set of objects to obtain while the client requests the current web page. The use of semantic hyperlink information has been shown to be an effective approach to decrease the rendering delays associated with web-applications and slow communi-

cation channels. To offset the delays associated with web page rendering times, the concept of web page prefetching has been proposed. These prefetching techniques attempt to anticipate the hyperlink click of the client in order to retrieve these predicted web pages before the client selects them. When a prefetcher has high coverage and accuracy, the client's browser has the objects it needs and only the ones it will use. The proposed prefetching technique combines various methods to achieve these performance metrics. Through empirical trace-driven simulations, the combination of these components has been shown to be an effective method to reduce the delays associated with web page rendering. The cooperation between web-applications and browsers in prefetching web objects has the potential to improve a client's quality-of-service by reducing web page rendering time. Currently, we are investigating various metrics to improve the overall performance of the SLOB technique through the incorporation of client-side information. In addition, we are exploring the use of heuristics to avoid over-prefetching that occurs when a client returns to a previously prefetch web page. We are also investigating the opportunities of dynamically adjusting the initial semantic link information associated with the web-application during design, from web-application usage information recorded from a wide range of client web-application accesses.

References

- Albrecht, D.W., Zukerman, I., Nicholson, A.E., 1999. Pre-sending documents on the WWW: a comparative study. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99), Stockholm, Sweden. Morgan Kaufmann.
- Bestavros, A., 1995. Using speculation to reduce server load and service time on the WWW. In: Proceedings of the 4th ACM International Conference on Information and Knowledge Management (CIKM '95), Baltimore, MD.
- Bestavros, A., 1996. Speculative data dissemination and service to reduce server load, network traffic and service time in distributed information systems. In: Proceedings of the International Conference on Data Engineering (ICDE '96).
- Cunha, C.R., Jaccoud, C.F.B., 1997. Determining WWW user's next access and its application to pre-fetching. In: Proceedings of the 2nd IEEE International Symposium on Computers and Communications (ISCC '97).
- Davison, B.D., 2002. Predicting web actions from HTML content. In: Proceedings of the Thirteenth ACM Conference on Hypertext and Hypermedia (HT '02), College Park, MD, June 11–15.
- Dellaert, B.G.C., Kahn, B.E., 1999. How tolerable is delay? Consumers' evaluations of Internet web sites after waiting. *Journal of Interactive Marketing* 3 (1), 41–54.
- Dunchamp, D., 1999. Prefetching hyperlinks. In: Proceeding of the 2nd USENIX Symposium on Internet Technologies and Systems (USITS '99).
- Galletta, D., Henry, R., McCoy, S., Polak, P., 2004. Web site delays: how tolerant are users. *Journal of the Association for Information Systems* 5 (1), 1–28.
- Hine, J., Wills, C., Martel, A., Sommers, J., 1998. Combining client knowledge and resource dependencies for improved world wide web performance. In: Proceedings of the 8th Annual Conference of the Internet Society (INET '98), Geneva, Switzerland.
- Horrigan, J., 2003. Adoption of broadband to the home, Pew Internet & American Life Project. Available from: <http://www.pewinternet.org/pdfs/PIP_Broadband_adoption.pdf>.
- Hoxmeier, J.A., DiCesare, C., 2000. System response time and user satisfaction: an experimental study of browser-based applications. In: Proceedings of the Association of Information Systems Americas Conference, Long Beach, CA.
- Ibrahim, T.I., Xu, C., 2000. Neural net based pre-fetching to tolerate WWW latency. In: Proceedings of the 20th International Conference on Distributed Computing Systems (ICDCS 2000).
- Jacko, J.A., Sears, A., Borella, M.S., 2000. The effect of network delay and media on user perceptions of web resources. *Behavior and Information Technology* 19 (6), 427–439.
- Markatos, E., Chronaki, C., 1998. A top-10 approach to prefetching the web. In: Proceedings of the 8th Annual Conference of the Internet Society (INET '98), Geneva, Switzerland.
- Nielsen, J., 1999a. Who commits the 'Top ten mistakes' of web design? Alertbox. Available from: <<http://www.useit.com/alertbox/990516.html>>.
- Nielsen, J., 1999b. The Top ten new mistakes of web design, Alertbox. Available from: <<http://www.useit.com/alertbox/990530.html>>.
- Padmanabhan, V., Mogul, J., 1996. Using predictive prefetching to improve world wide web latency. In: Proceedings of the ACM SIGCOMM '96 Conference on Communications Architectures and Protocols, July.
- Ramsay, J., Barbesi, A., Preece, J., 1998. A psychological investigation of long retrieval times on the world wide web. *Interacting with Computers* 10 (1), 77–86.
- Rose, G.M., Lees, J., Meuter, M., 2001. A refined view of download time impacts on e-consumer attitudes and patronage intentions toward e-retailers. *The International Journal on Media Management* 3 (2), 105–111.
- Telecommunications Reports International, 2000. Macroview, Iconocast, August 17, 2000. Available from: <<http://www.iconocast.com/issue/20000817.html#macroview>>.
- US Department of Commerce, 2004. A notion online: entering the broadband age, economic and statistics administration national telecommunications and information administration. Available from: <<http://www.nita.doc.gov/reports/anol/NationOnLineBroadband04.htm>>.
- Weinberg, B.D., 2000. Don't keep your Internet customers waiting too long at the (virtual) front door. *Journal of Interactive Marketing* 14 (1), 30–39.
- Wills, C.E., Mikhailov, M., Shang, H., 2001. N for the price of 1: bundling web objects for more efficient content delivery. In: Proceedings of the Tenth International World Wide Web Conference, Hong Kong.
- Wills, C.E., Trott, G., Mikhailov, M., 2003. Using bundles for web content delivery. *Computer Networks* 42 (6), 797–817.
- Zhuge, H., 2003. Active e-document framework ADF: model and platform. *Information and Management* 41 (1), 87–97.
- Zhuge, H., 2004. *The Knowledge Grid*. World Scientific Publishing Co., Singapore.

Alexander P. Pons is an Associate Professor of Computer Information Systems at the University of Miami. He 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.