Available online at www.sciencedirect.com

SCIENCE DIRECT®

The Journal of Systems and Software 74 (2005) 303–311

The Journal of Systems and Software

www.elsevier.com/locate/jss

Improving the performance of client Web object retrieval

Alexander P. Pons [*]

Computer Information Systems, University of Miami, 421 Jenkins Building, Coral Gables, FL 33146, USA

Received 25 April 2003; received in revised form 27 February 2004; accepted 29 February 2004
Available online 30 April 2004

Abstract

The growth of the Internet has generated Web pages that are rich in media and that incur significant rendering latency when accessed through slow communication channels. The technique of Web-object prefetching can potentially expedite the presentation of Web pages by utilizing the current Web page's view time to acquire the Web objects of likely future Web pages. The performance of the Web object prefetcher is contingent on the predictability of future Web pages and quickly determining which Web objects to prefetch during the limited view time interval of the current Web page. The proposed Markov–Knapsack method uses an approach that combines a Multi-Markov Web-application centric prefetch model with a Knapsack Web object selector to enhance Web page rendering performance. The Markov Web page model ascertains the most likely next Web page set based on the current Web page and the Web object Knapsack selector determines the premium Web objects to request from these Web pages. The results presented in the paper show that the proposed methods can be effective in improving a Web browser cache-hit percentage while significantly lowering Web page rendering latency.

Keywords: Web prefetching; Markov model; Knapsack selector; Web application

1. Introduction

The majority of the Internet population accesses the World Wide Web via dial-up modem connections. Studies have shown that the limited modem bandwidth is the main contributor to latency perceived by end users. Today's Web browsers employ a demand-fetch technique in which Web page objects (graphics, pictures, audio and/or video) are acquired only when the user initiates a request for a page. When a browser navigates to a Web Page, it checks to see whether the page and its compositional objects are in the browser's cache. If not, these objects must then be retrieved from the origin or proxy server. Since browser caches sizes are limited and maintain the most recently accessed objects, it is likely that the current page's objects will have to be requested from the origin server. This demand-fetching approach typically increases a Web page's rendering latency, depending on the number and sizes of these objects and the speed of the communication channel.

The concept of Web object prefetching has the potential to expedite Web page rendering and increase utilize the communication channel, since the process makes use of the bandwidth that would otherwise be idle. The operation of object prefetching is complementary to the technique of browser object caching. For example, if a browser requests an object that is not in the cache, but the object is in the prefetch area, the browser avoids having to make a server request. Anticipating which objects will be referenced in the near future has been the focus of much recent research. In Pons (2002), we introduced a Multi-Markov Web application centric prefetching approach. In this paper we take our basic model and augment it with an object selection technique that extends our previous work to further improve performance. We define and subsequently elaborate on the Markov–Knapsack Prefetcher (MKP), which has the following properties:

- An Initial Web-application Markov Model (IWAM) is generated and downloaded to a client's machine.

[*] Tel.: +1-305-284-1960; fax: +1-305-284-5161.
E-mail address: apons@miami.edu (A.P. Pons).

0164-1212/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2004.02.030

The IWAM is a complete Web-application hyperlink domain bounded model in which the hyperlinks are the nodes and the edges are the transition frequency.

- During subsequent Web-application access the IWAM is personalized into a Web-application Markov Model (WAM) for the client.
- An IWAM exists for each Web application and is customized to a client's Web-application access patterns.
- The WAM predicts based the most likely next Web page based on the currently viewed Web page and acquires its objects during the idle communication channel time.
- The Knapsack technique identifies from among these probable Web page objects which one should be retrieved to minimize latency.

The remainder of this paper is organized as follows: Section 2 reviews some important related work concerning Web prefetching techniques. The Web-application centric Markov–Knapsack prefetcher is discussed in Section 3, and an example that highlights the approach is illustrated in Section 4. In Section 5, the experiment design is introduced with performance comparison results explained in Section 6. Finally, Section 7 provides a conclusion and suggests future research directions.

2. Related work

In the literature many prefetching methods have been proposed to predict the user's next action, from subsequence matching that use past sequences of actions to identify a possible next action for the current sequence, to single and multi-step Markov and Markov-like models that associate the possible next actions as states with probabilities from the current state. Other approaches use utilize user characteristics such as bookmarks and history to determine the user's behavior, while some use Web page content and semantic link information to make predictions for what will be requested next. These proposed methods consist of the following works.

Padmanabhan and Mogul (1996), Bestavros (1995, 1996) and Albrecht et al. (1999) proposed server-initiated approaches where the Web server maintains a Markov model of page request interdependency. These approaches vary in the type of statistics and use of the model to make predictions. In Padmanabhan, when a client requests a page, the server sends along with the page the names of the most likely subsequent accessed pages, leaving the initiative for prefetching to the client. Using trace-driven simulations, they achieved a 36% reduction in network latency at a cost of 40% increase in network traffic. They constructed the Markov model using the number of client page accesses to compute the weights. In Bestavros, a Time Markov model is used to presend pages to the client based on the currently requested page. The model is constructed from the behavior patterns of the general user population and selecting the presend page with the highest probability of being requested next. Betavros' trace-driven simulations showed that with an increase of 10% in bandwidth, a 23% reduction in page miss rate is possible. Albrecht builds on the approach taken by Bestavros and constructs a hybrid prediction model, which combines four Markov models and uses a decision-theoretic model for presending pages.

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 client determines how much and when items from the list are prefetched. Page prefetching occurs off-line, with experimental results suggesting that it manages to prefetch up to 60% of future requests, with less than a 20% increase in traffic. Hine et al. (1998) also combines client and server information to perform page prefetching. They extend the work done by Bestavros and define 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.

Fan et al. (1999) proposes a prediction algorithm based on the Prediction by Partial Matching (PPM) studied by Vitter and Krishnan (1996) that demonstrates a relationship between data compression (Mogul et al., 1997) 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. Results show that their technique reduces client latency up to 23.4%.

Cunha and Jaccoud (1997) use a prefetch model that focuses on the client being active in gathering usage information and making prefetch decisions. The client machine utilizes a Markov model with three types of links that indicate the manner in which objects were accessed. These links distinguish between objects accessed within a time window, objects embedded within other objects, and objects that are likely to be accessed close in time. Their approach uses a mathematical model that combines link categorization and a Markov model, which exceeds a hit rate of over 80% when the user's behavior fits the model. 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 using different algorithms.