Exploiting Neglected Data Locality in Browsers

Li Xiao
Department of Computer Science
College of William and Mary
Williamsburg, VA 23187-8795
lxiao@cs.wm.edu

Xiaodong Zhang
Department of Computer Science
College of William and Mary
Williamsburg, VA 23187-8795
zhang@cs.wm.edu

ABSTRACT

With an significant increase of memory and disk capacity in workstations and PCs, and with the fine improvement of web browser caching capability, users are able to enlarge the browser cache size for more frequent accesses to the cached data objects and to retain them in an organized manner for a longer period of time. However, the browser caches are not shared among themselves and the available locality in browsers are neglected in Web proxy caching. In this paper, we propose an enhanced caching technique, called "Browser-Aware Proxy Server", to exploit the neglected data locality in browser caches for further performance improvement. Conducting trace-driven simulations, we present three new findings and contributions: (1) The neglected data locality in browser caches is significant and can be utilized for improving caching performance. (2) The additional index file and the storage required by the browser-aware proxy server are justifiable for the performance gain. (3) We show that the browser-aware proxy server outperforms a browser-unaware proxy server by 21% and 40%, measured by the average hit ratio and byte hit ratio, respectively. We also show that web server access latency can be reduced to 35% with a slight increase of local traffic in the client side.

Keywords

Browser Locality, Web Browsers, and Web Caching.

1. INTRODUCTION

Proxy caching is an effective solution to quickly access and reuse the cached data on the client side and to reduce Internet traffic to web servers. A group of networked clients connects to a proxy cache server, where each client has a browser with its cache buffering popular and recently requested data objects. A standard web caching model built on such a system has the following data flows.Upon a web request of a client, the browser first checks if it exists in the local browser cache. If so, the request will be served by its own browser cache. Otherwise the request will be sent to the proxy cache. If the requested data object is not found in the proxy cache, the proxy server will immediately send the request to its cooperative caches, if any, or to an upper level proxy cache, or to the web server without considering if it exists in other browsers' caches. We believe there are three practical reasons for a proxy server to exclude this consideration. First, the browser caches are not shared and the dynamic status in each cache is unknown to the proxy server. Second, the possibility of a proxy cache miss which is a browser cache hit may have been considered low although no such a study has been found in literature. Finally, a browser cache was initially developed as a small data buffer with a few simple data manipulation operations. Users may not effectively retain the cached data with a high quality of spatial and temporal locality.

How much potential benefit can we gain in caching performance by exploiting the neglected locality? First, since browser caches are not shared among themselves, it is certainly possible that a data object is stored in one or more browser caches, but has been replaced in the proxy cache. Secondly, browsers provide a function for users to set the browser cache size. With the rapid increase of memory and disk capacity in workstations and PCs, and with the rapid growth of web applications, user browser cache size will tend to increase as time passes. Thirdly, with the aid of more functions provided by browser softwares, users will pay more attention to the organized browser cache data objects, and tend to keep them in the the cache much longer than to keep the unorganized data objects. Fourthly, new technologies have been developed to improve the browsing speed. Finally, the number and types of web servers have increased and will continue to increase dramatically, providing services to a wider and wider range of clients with more diverse interests. It is impossible for proxy caches to cover all multi-requested file objects of tremendous types even with a perfect cache replacement strategy, increasing the possibility that browser caches will keep file objects that have been replaced in proxy caches. In summary, the quality of spatial and temporal locality in browser caches has been and will continue to be improved.

2. BROWSER-AWARE PROXY SERVER

In this design, the proxy server connecting to a group of networked clients maintains an index file of data objects of all clients' browser caches. If a user request misses in its local browser cache and the proxy cache, the browser-aware proxy server will search an index file attempting to find it in a client's browser cache before sending the request to an upper level proxy cache or the web server. If such a hit is found in a client, the proxy will collect the file object from this client and then forward it to the requesting client.

In order to implement the browser-aware concept in a proxy server, we create a *browser index file* in the proxy server. This index file records a directory of cached file objects in each client machine. Each item of the index file includes the ID number of a client machine, the URL including the full path name of the cached file object, and a time stamp of the file, or the TTL (Time To Live). Since the dynamic changes in browser caches are only partially visible to the proxy server (when a file object is sent from the proxy cache to the browser), the browser index file will be updated periodically by each browser cache.

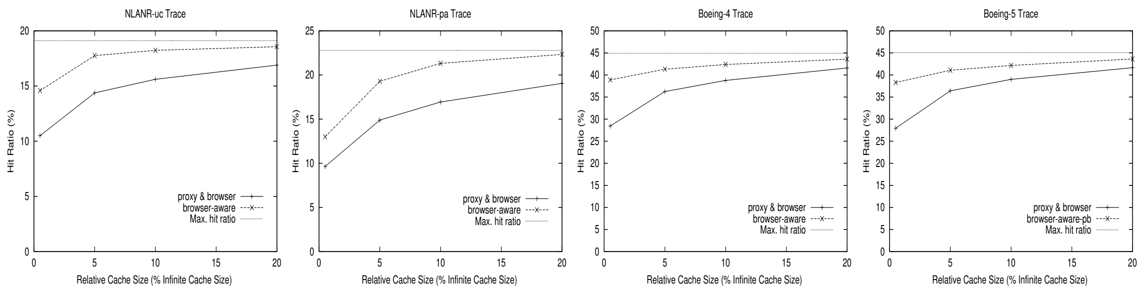

Figure 1: The hit ratios of the *browser-aware-proxy-server* and the *proxy-and-local-browser* scheme.

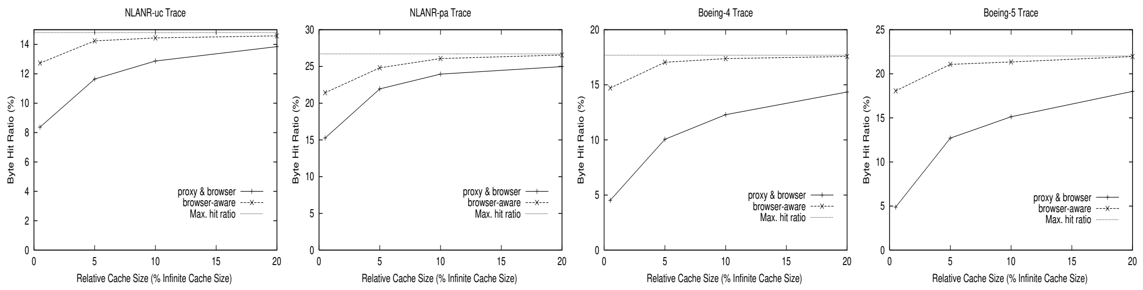

Figure 2: The byte hit ratios of the *browser-aware-proxy-server* and the *proxy-and-local-browser* scheme.

3. PERFORMANCE EVALUATION

We have used several web traces for performance evaluation: (1)NLANR traces [2]: We have used one day's trace from the "uc" proxy, and one day's trace from the "pa" proxy. (2) Boeing traces [1]: We have used one day's trace on March 4, 1999 and one day's trace on March 5, 1999 from the sixth proxy. We have built a simulator to construct a system with a group of clustered clients connecting to a proxy server. The cache replacement algorithm used in our simulator is LRU. We have implemented and compared the following web caching organizations using the trace-driven simulations: (1) *Proxy-and-local-browser*: each client has a private browser cache, and there is a proxy cache server for the group of client machines. If a request misses in its local browser, it will be sent to the proxy to check if the requested document is cached there. If it misses again, the proxy will send the request to an upper level server. (2) *Browser-aware-proxy-server*: this is the enhanced proxy caching technique presented in section 2. We have evaluated and compared the performance these two schemes using all traces.

Figures 1 and 2 present the hit ratios and the byte hit ratios of the two policies on the "NLANR-uc" trace, the "NLANR-pa" trace, the "boeing-4" trace, and the "boeing-5" trace, respectively. The average hit ratio improvements of the *browser-aware-proxy-server* for the four traces are 17.6%, 20.95%, 13.05%, and 12.6%, respectively. The average byte hit ratio improvements are 17.1%, 13.58%, 39.46%, and 39.94%, respectively. Our experiments also show that the *browser-aware-proxy-server* achieves an average latency reduction of 35%, compared with the *proxy-and-local-browser* scheme. The additional overheads of the browser-aware proxy cache come from (1) the data transferring time for the hits in remote browsers; (2) the update of the browser index file if the update is not conducted at a suitable time or conducted too frequently; (3) the space requirement of the proxy cache to store the browser index record. Our analyses show that the overheads are insignificant.

4. SUMMARY AND TECHNICAL ISSUES

We have proposed and evaluated the *browser-aware proxy server* to exploit neglected data locality in browsers. Our study shows that neglected locality in browsers is significant and should be utilized. We have shown that proxy caching performance can be further improved by exploiting browser cache locality with a simple software structure.

Besides exploiting data locality in browsers, our next objective is to effectively control and manage the data flows among the proxy and browser caches to make the proxy cache as the commonly shared place, and to make each browser mainly for individual usage. In order to incorporate these objectives to caching system designs and implementations, we must address two major technical issues. First, a client browser is not normally configured as a server. One solution for this is to add some simple server functions to a browser so that it is able to actively communicate with other clients and the proxy server. Another alternative is to let the proxy provide additional services on behalf of each browser. Second, the reliability and security of the browser data must be seriously considered. For example, the browser data files which have been modified by an owner client are not reliable for sharing among clients. In addition, the contents of browsers should not be visible among clients to preserve the privacy of each client. This concern can be addressed by requiring the proxy server to handle the data searching and transferring among the browsers.

5. REFERENCES

[1] Boeing log files.
ftp://researchsmp2.cc.vt.edu/pub/boeing/

[2] National Lab of Applied Network Research.
http://www.ircache.net/