

Site-based Approach in HTTP Proxy Design¹

K. Y. Wong

Department of Electronic Engineering
City University of Hong Kong
Email: kywong@ee.cityu.edu.hk

K. H. Yeung

Department of Electronic Engineering
City University of Hong Kong
Email: eeayeung@cityu.edu.hk

Abstract

Due to the explosive growth of the World-Wide Web, Web proxy becomes very important in today's Internet systems. Among these proxy servers, most of them will make cache decision based on document names. However, we propose in this paper a new direction in making these caching decisions, namely a site-based approach. What we mean by site-based approach is that a proxy only considers the site name of a document in making caching decisions. If a replacement has to be made, all the documents of a site will be deleted from cache at the same time. In this paper, we argue the feasibility of the site-based approach by comparing its caching performance to that of the general document based approach. Our simulation results show that caching based on site names only will not degrade the cache performance. But as discussed in this paper, caching based on site names only will bring a lot of benefits in designing fast proxy servers.

1. Introduction

The use of World-Wide Web [1] has grown dramatically in recent years and it is certain that this trend will continue in the coming years. In July 1997, the number of Internet hosts is 19,540,000 while there were only 213 hosts in August 1981 [2]. This growth in WWW traffic has led to traffic congestion in many parts of the Internet, which in turn increase the latency in accessing web documents. The use of World-Wide Web Proxy [3] that caches documents is an effective way to solve the problem. It can reduce the number of requests to popular Web sites. This in turn reduces the volume of network activities and the latency in retrieving a Web document. This calls for many research on proxy design.

Research on proxy has five main different areas of interest: searching, cache consistency, prefetching, cache replacement policy and multiple-proxy design. Searching process must be performed each time when a proxy receives a request from one of its Web client. A proxy first finds out whether the request can be served locally or

not. If the requested document is found in its local cache and is still fresh (i.e. up-to-date), it is sent to the client without contacting the corresponding remote Web server. To check the existence of a requested document, a unique-document list is usually stored in a proxy. For example the Squid proxy server version 2 [4] keeps a sorted linked list on memory, and the list is sorted according to the last access times of the documents. Every time a client request is received, each entry of the list will be compared until the requested document is found.

Cache consistency is the second area in proxy research. This kind of study is important because clients may receive stale data, which is outdated with respect to the master copy on the origin server. Every Web cache must somehow update pages in its cache so that it can give up-to-date pages. A representative method in keeping the cache consistent is the TTL (Time-to-live) approach [5]. In this approach, the proxy assigns a TTL attribute to each document in its cache. When one's TTL expires, the cached copy is considered to be stale, and the remote site should be contacted for the latest status of the copy.

Prefetching is a technique to reduce the latency of retrieving a Web document. A proxy prefetches Web pages that a user is likely to access soon by processing the page which he is viewing. To predict which page the user will select, the proxy server first figures out what hypertext links are contained in the current HTML document. The proxy then evaluates those links with some criteria, such as the document popularity, so as to determine whether a prefetching is performed or not. As mentioned in [6], by maintaining hit counts for related documents, the proxy may be able to predict which links are the most likely ones to be followed.

Cache replacement policy is used by a proxy server in managing the limited amount of storage space. A proxy tries to remove the cached documents that are likely not to be accessed in the near future in order to provide enough space for the new documents. The main objective of cache replacement policies is to optimize the cache space usage to obtain maximum cache hit rate. A commonly used algorithm for cache space management is the LRU

¹ This research is partly supported by the City University Research Grant numbered 9030719.

algorithm (Least Recently Used) [6]. It replaces the document that hasn't been accessed in the longest past.

Multiple-proxy design allows downstream proxies (proxies closer to the client) to get benefit from the cache of upstream proxies. Anawat Chankhunthod [7] has discussed the design and the performance of a hierarchical proxy-cache which is scaleable for various Internet information systems. It is believed that a good hierarchical proxy cache can significantly improve the performance of a proxy system.

Although the research on proxy has divided into so many areas, a major observation on the previous works is that caching decision is always made in a document-to-document basis. In the searching process, a list which stores each document's entry is searched. In keeping the cache consistency area, a TTL attribute is assigned to each document. In prefetching, each hyper links on a document will be evaluated individually. In cache replacement, each document in the cache will be evaluated to determine which one should be replaced. Although making cache decision in a document-to-document basis is the most natural approach, we believe that it is not the best one. In contrast, we propose² in this paper a new direction in making these caching decisions: to use a site-based approach. What we mean by site-based is that in making any caching decision, a proxy only considers the site which a document belongs to but not the document itself.

In the next section, we first briefly discuss the benefits of using the site-based approach. We then pass on to the focus of the paper, which is to prove that caching based on site names only will not decrease the cache hit rate of a proxy. We aim at providing enough evidence to show that it is not necessary to keep individual document statistic in a proxy. By only keeping site statistics, much more efficient proxy can be designed.

2. Site-based approach

In this section, an example will be given to illustrate how site-based approach can be used in cache replacement. Then the benefits of using the site-based approach in other proxy design areas will be discussed.

2.1. Illustration of site-based LRU

Figure 1 shows the link list management when document-based LRU and site-based LRU algorithms are used. The cache was empty initially, and is just full after serving requests 1 to 8 (the link list at this moment is shown in solid lines). At the moment shown in Figure 1, one or more documents in cache must be removed so that the document requested by request 9 can be fitted into the cache. Since two different approaches are adopted,

² The authors do not aware of any previous works on this approach.

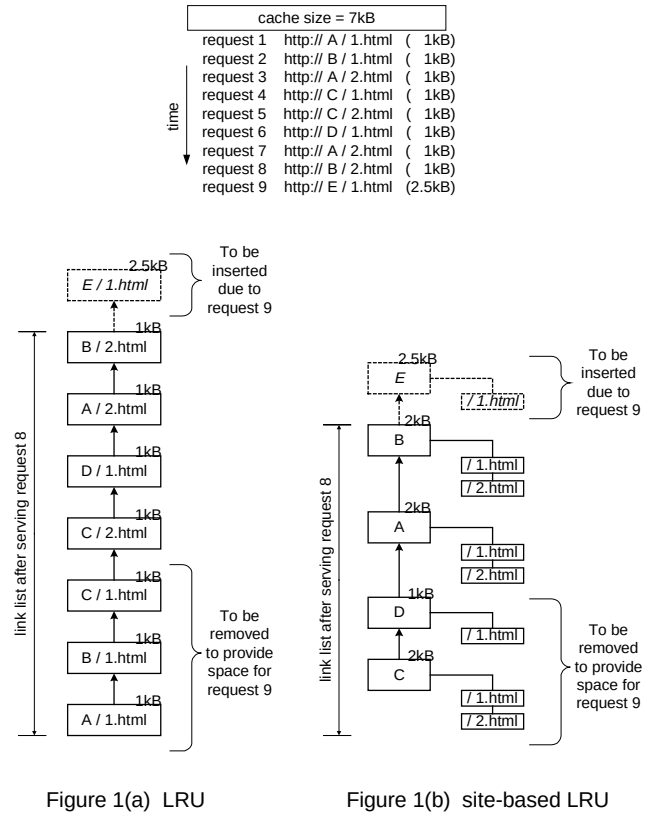


Figure 1. An example showing the link list management when (a) document-based LRU, and (b) site-based LRU, is used.

different link lists are formed and different documents will be removed for request 9. The operations are discussed as below.

LRU policy links the cached documents to a document list, as shown in Figure 1(a). When a new document is requested, it is inserted to the top of the document list. Every time a document in cache is accessed, it is also moved to the top of the list. As time goes by, the least recent documents will migrate to the bottom of the list. When the free cache space cannot fulfill the size of an incoming document (e.g., when request 9 arrives), the last objects in the list will be removed until the new document can be fitted into cache. In our example, the documents A/1.html, B/1.html and C/1.html are removed to free 3kB cache space for the new document E/1.html which has a size of 2.5kB.

When site-based LRU is used, the proxy links the site names to a site list, as shown in Figure 1(b). When a document of an unaccessed site is requested, its site name is inserted to the top of the site list, and the document will be stored under its site. Note that in site-based approach a document is kept in the cache, as long as its site name is in the list. Every time when a document is requested its corresponding site will be moved to the top of the site list.

As time goes by, the least recent sites will migrate to the bottom of the list. When the size of an incoming document is larger than the free space in cache (e.g., when request 9 arrives), the last site(s) in the list including all its documents will be removed until the new document can be fitted into cache. In our example, all the documents in sites C and D are removed so as to store the new document which is requested by request 9.

2.2 Benefits of using site-based approach

There are a lot of potential benefits when site-based design approach is used in a proxy. In the design of the searching algorithm, a site-based cache replacement policy can improve the searching speed significantly. Since the proxy will first scan a unique-site list, which is much shorter than a document list, and then search the requested document within that site. Consider a case that a proxy contains N cached pages from S sites. For a document-based proxy, there are N entries in the unique-document list. Thus, to find out a particular page in the cache we need an average number of $N/2$ matching operations. For a site-based proxy, there are S entries in the unique-site list, and on the average N/S pages in a site. Thus, to find out a particular page, the average number of matching is only $S/2 + (N/S)/2$. Consider the real access log which is used in our simulation study that $N=100,000$ and $S=5,000$, it takes on the average 50,000 matching for a search in the document list. In contrast, it only takes 2510 matching for a search in the site list. This corresponds to 20 times faster than the document-based approach. Although this is only a rough estimation, it is quite certain that the actual improvement will be significant.

In the research area of cache consistency, Pei Cao has shown that strong cache consistency should be maintained for the Web, with the invalidation protocol the right protocol for keeping strong cache consistency [8]. The invalidation protocol relies on the server to send out invalidation messages when a file is changed. Upon receiving an invalidation message, the proxy deletes its corresponding cached copy. If a proxy is using the site-based cache replacement policy and keeping a site list in cache, then it is faster (for the same reason discussed above) for the proxy to locate the documents which are indicated by the invalidation messages. Moreover, it is more convenient for a proxy to perform such actions as deleting all expired documents from a given site.

Using a site-based policy will also bring benefits to Web prefetching. As mention before, by maintaining hit counts for related documents a proxy can predict which links to be prefetched. Since prefetching will mostly be performed for documents which located at the same site as the currently accessed document, a much shorter document table (which only stores the documents in a

same site) is processed in making prefetching decisions. This will certainly improve the speed of prediction.

Last but not least, using a site-based policy brings convenience in performing queries such as finding the most popular sites or finding the resources used for a site. These kinds of queries provide useful information for dynamic decisions in cache replacement. In conclusion, site-based approach brings a lot of benefits in the design of a fast proxy. Although it is out of the scope of this paper to fully investigate the use of site-based approach in a proxy, the authors are currently designing a fully site-based proxy server.

3. Simulation and discussions

To prove that site-based approach is feasible, simulation based on real logs are run. The aim of the simulation is to find out the impact on cache hit rate when site-based approach is used in a proxy. The major conclusion of this section will be that no performance degradation is observed when site-based approach is used.

3.1 Simulation Design

Using trace driven simulation, we compare the performance of document-based cache replacement policy to that of the site-based cache replacement policies. In this study, two sanitized cache access logs are used and they are downloaded from National Lab for Applied Network Research's Proxy caches. The cache system has a Web proxy, SJ, located at the MAE West Exchange Point in San Jose, CA., and a Web proxy, LJ, located at La Jolla, CA.. Table 1 shows the characteristics of the traces. The Maximum Hit Rate (HR) and Byte Hit Rate (BHR) in the table are obtained with an infinite size cache.

Table 1. Summary of access log characteristics

	SJ	LJ
Duration	One day (11/30/98)	One day (11/30/98)
Total no. of requests	102,615	160,138
Total size of requested documents	1200.9 MB	1590.2 MB
Total unique no. of documents	79,442	99,721
Total size of unique documents	929.0 MB	1228.4 MB
Mean size of unique documents	12.0 kB	12.6 kB
Medium size of unique documents	3.1 kB	2.8 kB
Total unique no. of Web site	4,409	8,172
Maximum HR	22.58 %	37.72 %
Maximum BHR	268.2 M	366.9 M

LRU, LFU, site-based LRU and site-based LFU cache replacement policies are studied in this simulation. The

³ In this simulation a cache hit is simply defined as the requested URL can be found in the local cache. Consistency issues are not considered in this study because for a one-day trace, the chance for a document to be modified by its original Web server is low.

operations of LRU and site-based LRU have been discussed in section II. For LFU, the algorithm evicts the document which is accessed least frequently. It gives each page on cache a frequency counter. Whenever a document on cache is referenced again³ (cache hit), its frequency counter is increased by one. If the requested document cannot be found on the cache (cache miss), it will be stored on cache and its frequency counter is set to one. On the other hand, site-based LFU evicts all the documents belonging to the site which is accessed least frequently. A frequency counter is given to each entry of the distinct-site list instead of to each document. Whenever a document on cache is referenced again, its corresponding site's frequency counter is increased by one. No matter a document cache hit or a document cache miss occurs, the corresponding site-frequency counter is increased by one as long as the site of the requested document exists in cache.

The performance metrics used in this study are HR and BHR. HR is defined as the ratio of the number of requests that hit in local cache to the total number of requests. BHR is defined as the ratio of the number of bytes that hit in the local cache to the total number of bytes requested. These two metrics are commonly used in the performance study of proxy systems [9-11]. For each of the two logs, we simulated the proxy with cache sizes that are 1/2, 1/4 and 1/10 of the total size of document requests. All simulations are initiated with an empty cache, and the simulation program will report on HR and BHR for the whole log.

3.2 Simulation Results

Table 2 shows the values of HR and BHR for different cache replacement policies and cache sizes for the two logs. In the table, the fractions shown in the first row are the cache sizes (in fractions of the total size of the document requests). The table shows that reducing the cache size lowers both HR and BHR of all the policies. It also reveals that under different cache sizes, site-based cache replacement policies (site-based LRU and site-based LFU) provide similar performance in HR and BHR when compared to that of the ordinary document-based cache replacement policies. The only exception is when the cache size is very small (1/10, which corresponds to 160MB for LJ log), site-based LFU gives notable degradation in HR and BHR. However, the cache size in this case is unreasonably small when compared to today's common cache sizes of proxies (which have at least 2GB of memory).

Figure 2 reports both the total number of client requests and the number of hits of different policies per hour for the SJ log. As can be seen from the figure, the number of hit per hour by site-based replacement policies is more or less the same as the ordinary document-based policies under all situations shown. Our results prove the

invalidity of a common conception: since site-based replacement policies will also remove hot pages in cool sites, their cache performance should therefore not as good as document-based policies. The reality is that: if a site contains some hot pages, it will eventually become a hot site and be removed with a low chance.

Table 2. Simulation results

Table 2(a)		Results for SJ log		
	1/2	1/4	1/10	
<i>Hit Rate (%)</i>				
LRU	22.37	21.3	19.8	
LRU-site	22.35	21.7	20.4	
LFU	22.31	21.5	20.3	
LFU-site	22.21	20.9	18.1	
<i>Byte Hit (MB)</i>				
LRU	266.9	255.5	245.1	
LRU-site	266.5	257.2	247.6	
LFU	266.4	257.3	250.7	
LFU-site	264.3	250.2	211.2	

Table 2(b)		Results for LJ log		
	1/2	1/4	1/10	
<i>Hit Rate %</i>				
LRU	37.18	34.69	31.18	
LRU-site	37.25	35.10	30.66	
LFU	37.27	35.75	33.39	
LFU-site	36.55	32.43	26.03	
<i>Byte Hit (MB)</i>				
LRU	362.4M	336.3	309.4	
LRU-site	360.0M	340.2	298.9	
LFU	363.6M	344.3	324.3	
LFU-site	343.5M	288.1	229.3	

Table 3. Site popularity in the SJ log

Total no. of References to each site	No. of sites
1	764
2	467
3	282
4	239
5	164
6-10	647
11-50	1393
51-100	289
101-200	119
201-500	33
>500	12

Total unique sites in SJ log = 4409

3.3. Discussions on simulation results

Table 3 shows the popularity of sites in the SJ log. The second column shows the number of sites that are referenced at a certain number of times, and the number is shown in the corresponding row of column one. For example in row one, it is found that 764 sites are only referenced once in the log. When we add up the sites which are only referenced ten times or less, the total number of sites is 2563. This accounts for about 60% of the total number of sites. This finding may explains why

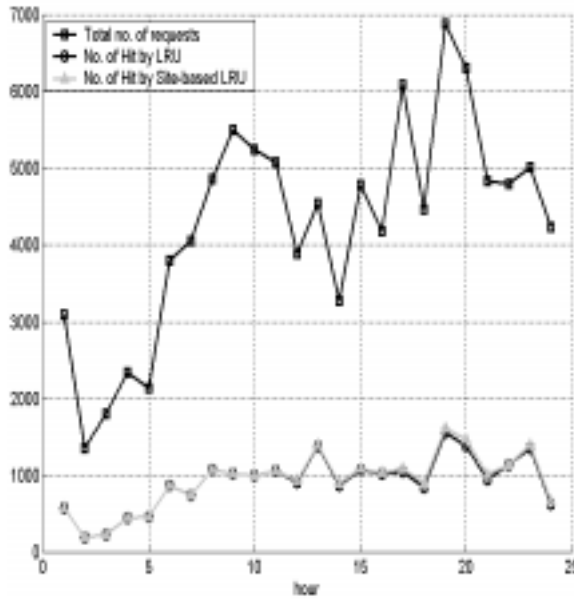


Figure 2(a) LRU and site-based LRU

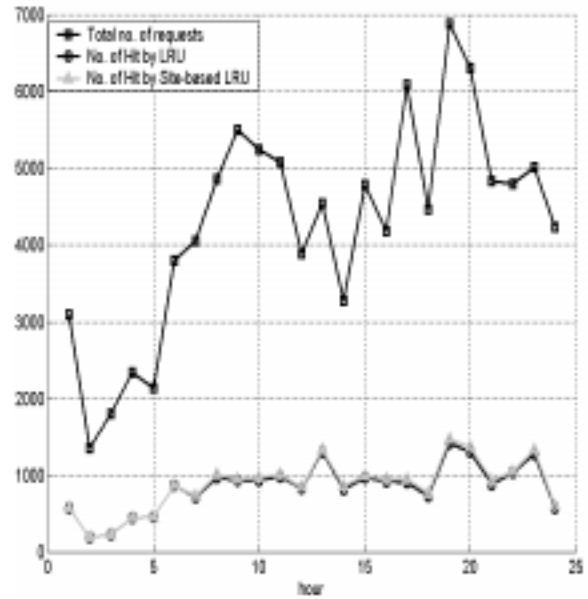


Figure 2(b) LRU and site-based LRU

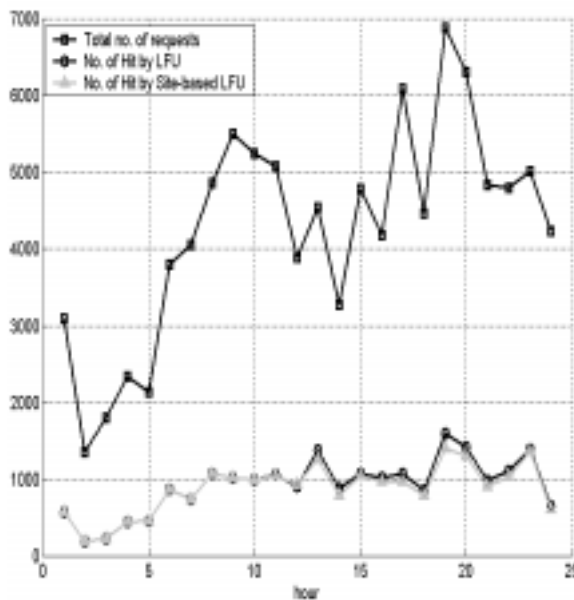


Figure 2(c) LRU and site-based LRU

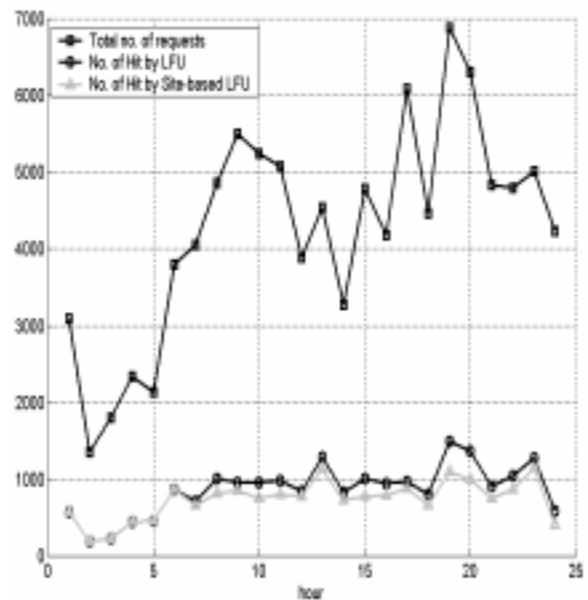


Figure 2(d) LRU and site-based LRU

Figure 2. Number of hits in SJ log : (a) and (c) use 1/4 of total size of requested documents as cache size, and (b) and (d) use 1/10 of total size of requested documents as cache size.

similar HR's are obtained by document-based and site-based cache replacement policies. Since these 2563 sites are cool sites, their corresponding accessed documents are also cool. Therefore, removing a single page is exactly the same as removing a whole site which contains only less than ten cool pages in the cache. This explains why using the site-based policies will not degrade the cache performance.

4. Conclusions

We have discussed in this paper a new direction in proxy design, which is to use a site-based approach in making proxy cache decisions. By using this approach, a proxy only keeps the access statistics of sites rather than documents. In doing this, proxy design can be simplified and it also brings a lot of potential benefits in designing proxy servers. The performances of site-based and

ordinary document-based cache replacement policies have also been compared in this study. Our simulation results show that with a cache of reasonable sizes, caching based on site names only will not decrease the cache hit rate of a proxy. We are currently attempting to apply this site-based approach in different areas of proxy design, such as prefetching and cache consistency. By using the proposed approach, it is expected that much faster proxies can be designed.

5. References

- [1] Tim Berners-Lee, Robert Cailliau, Ari Luotonen, Henrik Frystyk Nielsen, and Arthur Secret, "The World-Wide Web," *Communications of the ACM*, vol. 37, no. 8 pp. 76-82, Aug. 1994.
- [2] Internet Statistic: Web Growth, Internet Growth, <http://www.mit.edu/people/mkgray/net/>.
- [3] Ari Luotonen and Kevin Altis, "World-Wide Web Proxies," *1st Inter. Conf. on the WWW*, Geneva, May 1994.
- [4] Squid Internet Object Cache, <http://squid.nlanr.net/>.
- [5] James Gwertzman and Margo Seltzer, "World-Wide Web Cache Consistency," *Proceedings of the 1996 USENIX Technical Conference*, San Diego, CA, Jan. 96.
- [6] Ari Luotonen, *Web Proxy Servers*, Prentice Hall PTR.
- [7] Anawat Chankhunthod and Michael F. Schwartz, "A Hierarchical Internet Object Cache," *Proceedings of the USENIX 1996 Annual Technical Conference*.
- [8] Pei Cao and Chengjie Liu, "Maintaining Strong Cache Consistency in the World Wide Web," *IEEE Transactions on Computers*, vol. 47, no. 4, Apr. 1998.
- [9] Pei Cao and Sandy Irani, "Cost-Aware WWW Proxy Caching Algorithms," *Proceedings of the 1997 USENIX Symposium on Internet Technology and Systems*, pp. 193-206, Dec. 1997.
- [10] Stephen Williams, Marc Abrams, Charles R. Standridge, Ghaleb Abdulla and Edward A. Fox, "Removal Policies in Network Caches for World-Wide Web Documents," *Computer Communication Review*, vol. 26, iss 4, pp. 293-305.
- [11] Bradley M. Duska, David Matwood, and Michael J. Feeley, "The Measured Access Characteristics of World-Wide-Web Client Proxy Caches," *Proceedings of the USENIX Symposium on Internet Technologies and System*, Dec. 1997.