Proceedings of the 5th WSEAS International Conference on Applied Computer Science, Hangzhou, China, April 16-18, 2006 (pp750-755)

Bypassing Proxy: a Solution to Overloaded Web Caching Systems

K. Y. Wong
Computer Studies Program
Macao Polytechnic Institute
Av. de Luis Gonzaga Gomes, Macao, China

K. H. Yeung
Department of Electronic Engineering
City University of Hong Kong
Tat Chee Avenue, Kowloon, Hong Kong, China
eeayeung@cityu.edu.hk

Abstract: - This paper presents a technique to solve the overloading conditions of HTTP proxy servers. It is achieved by eliminating the unnecessary requests reaching the proxy servers. These unnecessary requests are the miss requests which cause the proxy to store copies of requested objects, but the copies will not be accessed again. To identify this kind of requests, we construct a list, called hot-site list, to store the most popularity web sites. For those requests not on the list will be directly forwarded to the remote Web sites without passing through the proxies. Our analytical and simulation results show that our proposed technique can reduce the proxy load by 48% and give a 1.75 system throughput gain.

Key-Words: Web proxy, Web caching, Internet systems, overloaded systems, application level routing

1 Introduction

As the need for caching WWW information is clear, many networks have installed proxy servers (or Web caches). While WWW traffic is continuously growing, so does the number of client requests reaching the proxy servers. This in turn increases the workload of the proxies. As a result, overloading conditions of proxy servers occur frequently.

Overloaded proxies cause many performance problems. Liu *et al.* [1] shown that proxy server performance is sensitive to traffic load, and when a proxy is overloaded, its performance will degrade quickly. On the other hand, Almeida and Cao [2] shown that when a proxy is overloaded, there are a lot of connection time-outs which are reported as errors.

A standard solution is to install cooperating proxy servers for improving the overall caching performance. However, researchers following this line have to face one major problem -- extra load and traffic. It has been reported by Krishnan and Sugla [3] that the extra load could cause up to 300% overhead.

Instead of following the line of using cooperating proxies to solve the problem, in this paper, we introduce another direction in tackling the problem of overloaded proxies.

By studying the internal operations of a proxy, we found that some requests should not be sent to the proxy. If these requests can be redirected to bypass the proxy and be sent to the Internet directly, the workload of the proxy can be reduced significantly without affecting the proxy's performance.

To achieve this, we introduce a technique which allows the client sides to decide whether send a request to the proxy or not. Since these decisions are based on the site name statistics, it is called the site-based dispatching (SBD) technique.

There are two types of miss requests. Requests in the first type will cause the proxy to keep copies of the miss objects and the copies will be accessed again by other users later on. We call these requests *good-miss requests*. Requests in the second type are those requests that also cause the proxy to keep copies of the miss objects but the copy will not be accessed again before it is being evicted from the cache. We call these requests *bad-miss requests*. The corresponding stored copies of the bad-miss requests are called *one-timer objects* [4]. The caching of the good-miss requests reduces both network load and the response delay apparent to the users. However, bad-miss requests causes cool objects to remain in the cache for a certain period of time without being referenced again. They could considerably waste system resources and degrade the cache performance. The SBD technique aims at improving the above situation by pulling out the bad-miss requests.

2. The SBD Technique

Consider a typical local area network which consists of an Internet gateway, a proxy server and user workstations as shown in Fig. 1. The Internet gateway can be a router or a firewall which provides the Internet access to the internal clients. The proxy acts as a cache server storing frequently requested

Web objects locally. The workstations run browsers that are configured to go through the proxy first when asking for Web objects.

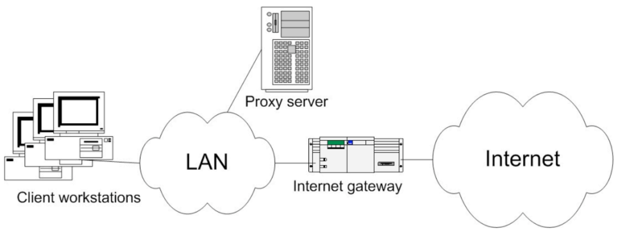


Fig. 1. A typical network connected to the Internet.

The design principle of the SBD technique is that if the bad-miss requests are known in advance, they can be forwarded to the remote Web sites directly, and the workload to the proxy by these requests can thus be saved. To do this, the client sides need to know whether the current request is a bad-miss one or not.

In the SBD technique, a hot-site list is used to identify the bad-miss requests. The hot-site list contains the site names of the most popular Web sites, and is updated at regular intervals. Each browser holds a hot-site list. Every time a browser receives a URL issued by a user, it will check the list first. If the requested Web site appears in the list, it forwards the request to the proxy; otherwise, forwards to the Internet and retrieves the object from the corresponding remote Web server directly.

The operations of the SBD technique is illustrated in Fig. 2. It is noted that the operations discussed with the figure may vary from different implementations. To apply the SBD technique, we introduce a hot-site manager which can run in any host in the local network. Whenever a browser is launched, it retrieves the hot-site list from the hot-site manager and stores it locally (see process (1) in Fig. 2). Having received a URL from the user, the browser first checks the list. If the list contains the requested Web site, the browser will forward the request to the proxy (see (2)). Otherwise, it will forward the request to the remote Web site directly through the Internet gateway (see (3)). Besides dispatching requests, the browser also maintains a log file recording all Web sites have been requested (see (4). When the browser is going to be terminated, the log file will be sent to the hot-site manager for calculating site popularity (see (5)). Note that, depending on the implementation, the process (1) and (5) can be performed at other time, not limited at the start and termination of the browser.

At predefined times, the hot-site manager will generate a new hot-site list which reveals the latest popularity of the Web sites being accessed. To do that, the manager first merges all the log files it has received from the browsers. It then calculates the hit frequency of each site and ranks the sites according to their access frequencies. After that, it includes a number of the hottest Web sites in the new hot-site list. Note that the update interval and the contents of the list can be determined in various ways. As will be shown in Simulation section, we user different configuration to explore the hot-site list idea.

The operations of the SBD technique is transparent to the proxy server. It simply reduces the number of requests reaching the proxy.

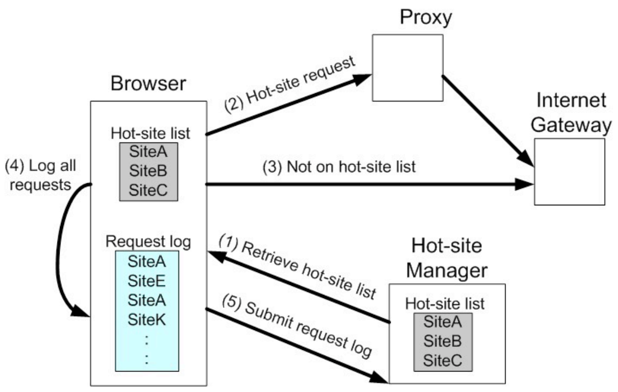


Fig. 2. Illustration of the operations of the SBD technique.

3 Performance Analysis

In this section, we first introduce methods to estimate the proxy load reduction and the proxy throughput gain when the SBD technique is used. We then analyze the response time of the conventional and the SBD caching system. Numerical results will be given in Section VI.

3.1 Estimated Proxy Load Reduction

Let $R=\{r_1, r_2, \ldots, r_N\}$ be the set of requests arrived to the system, and $O=\{o_1, o_2, \ldots o_N\}$ be the corresponding requested objects. In a cache hit case, when the proxy receives r_i from a client, it responds with sending o_i which is retrieved from the cache to the client. We denote L_i be the amount of workload offered to the proxy by a cache hit of r_i. On the other hand, when a cache miss occurs, the proxy needs to perform an extra work of making another request r_i to the remote server and then receiving o_i from the remote server. By using a similar approach for load estimation as suggested in [5], we assume that the extra workload offered to the proxy is also equal to L_i.

Define also *HR, HOT*, PHR_{SBD}, and HR_{SBD} (see Fig. 3) as:

HR: the proxy hit ratio of the conventional caching system.

$$= \frac{\text{total hits in the proxy}}{\text{total requests}}$$

HOT: the probability that a request belongs to a hot site (or the portion of total requests reaching the proxy).

$$= \frac{\text{total requests sent to the proxy}}{\text{total request}}$$

PHR_{SBD}: the hit ratio of the proxy after reducing the bad-miss traffic being sent to it.

$$= \frac{\text{total hits in the proxy}}{\text{total requests sent to the proxy}}$$

HR_{SBD}: the overall hit ratio in the SBD caching system.

$$= \frac{\text{total hits in the proxy}}{\text{total requests}}$$

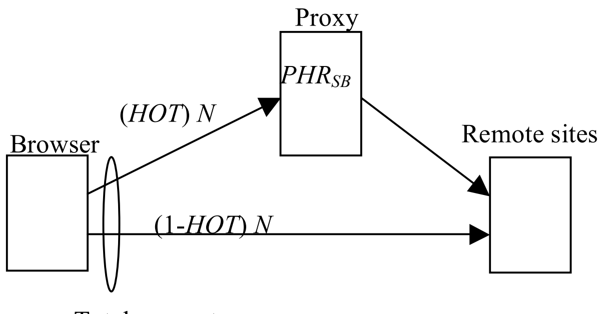


Fig. 3. Illustration of performance metrics.

In the following derivations, consider any request r_i arrived to the system. For a conventional caching system which does not use the SBD technique, the probability that the requested object o_i is present in the cache is *HR*. In this case, the proxy can serve the request r_i locally with a load L_i. The probability that r_i is a miss request is 1-*HR*. In this case, o_i is not in the cache and the proxy must fetch it from the remote server, giving a total load of $2L_i$. Therefore, the expected load to the proxy by any request r_i, denoted as *Load*, is therefore,

$$Load = HR \cdot L_i + (1 - HR) \cdot 2L_i \tag{1}$$

When the SBD technique is used, a request r_i will only reach the proxy when it targets on a hot site. The expected load offered to the proxy by a request r_i is thus

$$Load_{SBD} = HOT\left[PHR_{SBD} \cdot L_i + (1 - PHR_{SBD}) \cdot 2L_i\right] \tag{2}$$

Therefore, the normalized load reduction for request r_i when the SBD technique is used, denoted as LR_i, can be obtained by

$$LR_i = \frac{Load - Load_{SBD}}{Load} = \frac{(2 - HR) - HOT(2 - PHR_{SBD})}{2 - HR} \tag{3}$$

Equation (3) gives the normalized load reduction for any request r_i, which is the same for all r_i. The overall proxy load reduction, denoted as *LR*, is therefore simply

$$LR = \sum_{i=1}^{N} \frac{1}{N}(LR_i) = \sum_{i=1}^{N} \frac{1}{N}\left(\frac{(2 - HR) - HOT(2 - PHR_{SBD})}{2 - HR}\right)$$

(4)

$$= \frac{(2 - HR) - HOT(2 - PHR_{SBD})}{2 - HR} \quad . \tag{4}$$

3.2 Estimated System Throughput Gain

We consider a scenario that the proxy is the performance bottleneck limiting the overall throughput in a caching system. Assuming a proxy server is capable of serving up to a user request rate λ. In a conventional caching system, since all requests must be sent to the proxy, the overall system throughput is therefore limited to λ. However, when the SBD technique is used, the system throughput can be increased to λ_{SBD}, where $\lambda_{SBD} > \lambda$ (as the number of requests sent to proxy is reduced). Since the total request rate sent to the proxy is $\lambda_{SBD} \times HOT$, λ_{SBD} can be approximated by,

$$\lambda = \lambda_{SBD} \cdot HOT$$

$$\lambda_{SBD} = \frac{\lambda}{HOT} . \qquad (5)$$

Therefore, the system throughput gain, denoted as *TG*, can be obtained by

$$TG = \frac{\lambda_{SBD}}{\lambda} = \frac{1}{HOT} \qquad (6)$$

Note that this system throughput gain is only an estimation with the assumption that the throughput of the proxy itself remains constant in both cases.

4 Simulation and Numerical Results

To study the performance of SBD technique, simulations based on real proxy access traces were run. The aims of the simulations are to find out how many miss requests reaching a proxy server can be reduced, and to obtain empirical data (i.e., HOT, PHR_{SBD}, HR_{SBD} and HR) so that numerical studies can be performed later on.

4.1 Simulation Design

In our simulations, we used two real access traces obtained from the two proxies both at the National Lab of Applied Network Research [6]. One of the proxies is located at the NCSA at Urbana-Champaign, Illinois (UC) while another is located at the MAE West Exchange Point in San Jose, California (SJ). For these traces, we accept only HTTP GET requests and URLs that do not include query strings. Both logs are 7-day long.

In the simulations, we assume the proxy is using the least-recently-used replacement policy as it is widely used. And we consider all static objects as cacheable for simplicity of analysis. To obtain the sensitivity analysis on the update interval and the content of the hot-site list, we construct the list in various ways -- we update it on both daily and hourly basis, and for each basis, we use the following types of list:

EverTopN : Site popularity analysis will be performed on all sites that have been ever requested. The list includes the *N* most popular sites among them.

RecentTopN : Site popularity analysis will be performed only on the sites that have been requested in the previous interval. The list includes the *N* most popular sites among them.

RecentAll : The list simply includes all the sites that appeared in the last interval. It is the special case of RecnetTop*N*, and the merit of it is that no popularity analysis is needed.

To simulate the SBD caching mechanism, a traffic generator reads record by record from an access trace, and decides whether each request is forwarded to the proxy or to the remote server directly by checking the hot-site list. To study the proxy performance, the number of requests sent to the proxy, and the number of hit requests are recorded for each interval. Before gathering statistics, we use the first interval of the log to initialize the cache and create the hot-site list for use in the second interval. The performance metrics used in the simulation are HOT, PHR_{SBD}, HR_{SBD} and HR, as defined in section IIIA and illustrated in Fig. 3.

4.2 Estimated proxy load reduction and system throughput gain

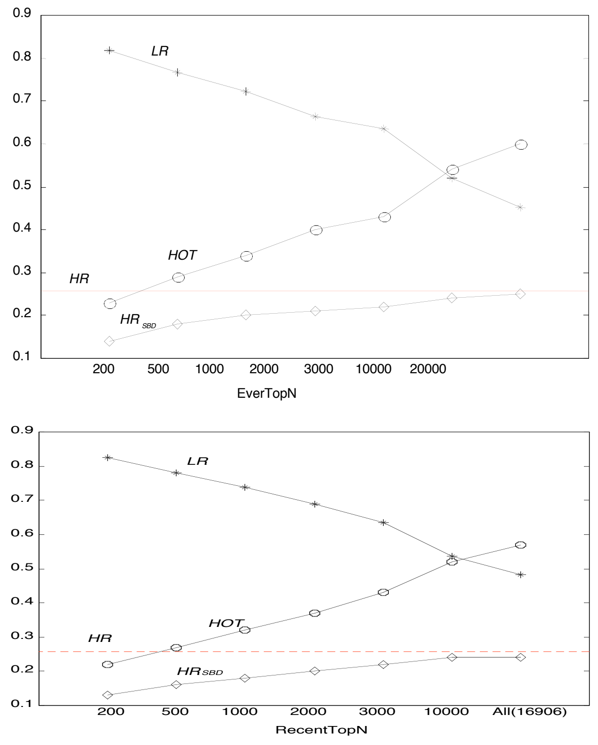
We first consider the simulation results for UC traces using RenewAll list with daily updated interval. Table 1 compares the performance of the SBD system to that of the conventional system. Looking at the row of *HOT*, there are on the average 57% of total requests reaching the proxy using the SBD technique. The proxy therefore saves the work caused by the remaining 43% traffic. Comparing the rows of HR_{SBD} and HR, the SBD system provides similar performance in overall hit rate to that of the conventional one. It can be seen that, in each time interval, the SBD technique also effectively pull out miss requests from the proxies, and hence reduce the traffic reaching the proxy without sacrificing the overall hit rate at all.

System	Metric	Value
All systems	N	3400000
Without using SBD	HR	0.258
The system using SBD	HOT	0.573
	PHR_{SBD}	0.422
	HR_{SBD}	0.243

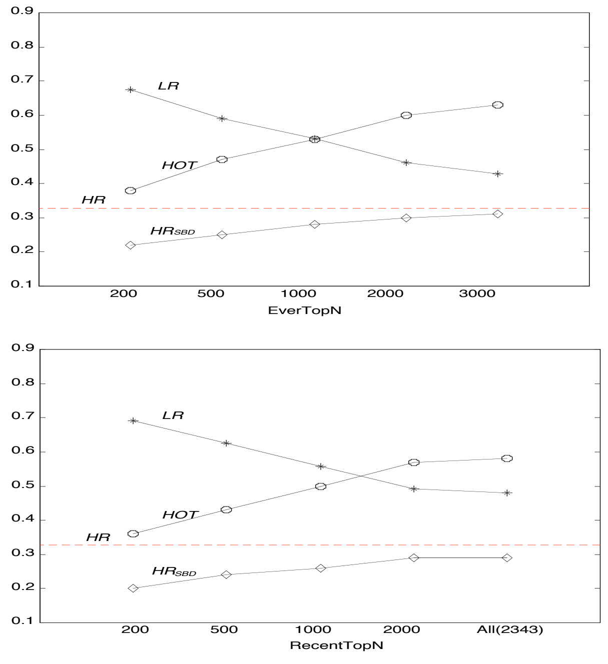
Table 1. Simulation results for UC trace using RenewAll hot-site list.

We then study the performance of the SBD technique when different list types and update intervals are used. Fig. 4 and Fig. 5 show *HOT*, *HR* and HR_{SBD} together with its corresponding *LR* (obtained by equation (4)) for each type of hot-site list. The x-axis of the figures shows the list size (i.e., the values of N), and the parenthesis besides RecentAll indicates the average number of sites appears in the previous interval. From the figures, some observations can be made. First, it can be seen that, in general, smaller size of the hot-site list forwards lesser requests to the proxy (HOT is smaller), resulting in higher proxy load reduction, but at the expense of overall hit ratio (compare HR_{SBD} to *HR*). Deterioration of cache hit performance is undesirable no matter how much the proxy load can be reduced. Therefore, small hot-site list should not be used. Second, RecentTop*N* provides similar results to that of EverTop*N* when the list is daily updated (see Fig. 4). It implies that calculating site popularity based on the previous interval is generally enough. However, it is not the same for hourly update interval. Fig. 5 reveals that RecentTop*N* provides poorer results (lower HR_{SBD}) than that of EverTop*N*. It is owing to the number of sites appear within an hour is relatively small (on average, 208 for SJ and 1314 for UC) so that a small hot-site list is obtained. Third, as shown in Fig. 4, RecentAll provides satisfactory results. Although, it cannot give the highest HR_{SBD} among the types of list, as mentioned earlier, the merit of it is that no site popularity analysis is needed. This would much simplify the process of creating new hot-site list. At last, when EverTop*N* is used, update hourly provides similar results to that of update daily. It is reasonable because even a site becomes very hot in the previous hour, it may not be hot enough to be one of the top *N* hottest sites and the list contents for both update intervals are hence similar. (The RecentTop*N* list with hourly update interval is originally expected to tackle the sites that suddenly become popular in the previous hour, however since the list will also ignore the historical hot sites, it gives unsatisfactory results.) Based on the discussion above, a rule of thumb to construct the hot-site list is that RecentAll list should be used and update daily; if the list is too large, RecentTopN is preferable to EverTopN.

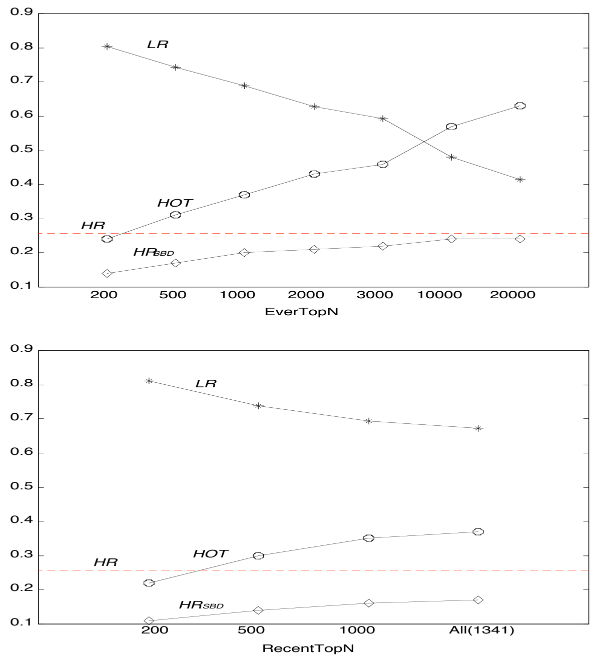
Based on the simulation results, we can find the estimated system throughput gain (*TR*) using equation (6). Table 2 shows both *LR* and *TR* for RecentAll list with daily update interval (as this configuration is recommended).



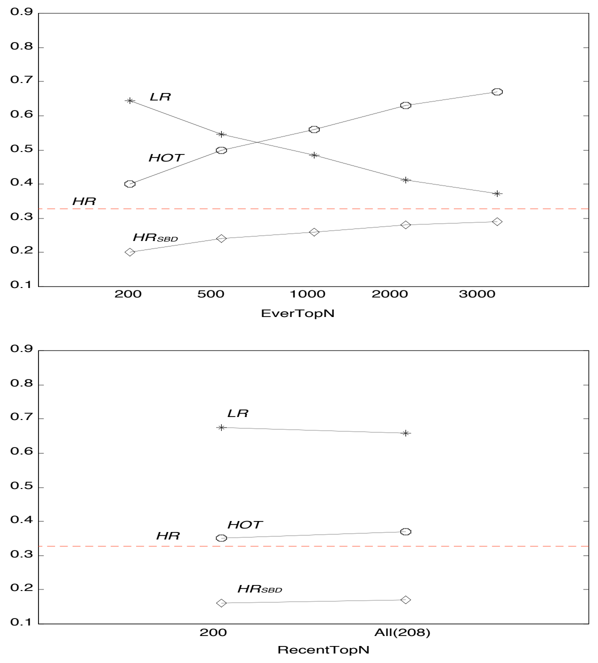
(a) UC



(b) SJ
Fig. 4. Daily update.



(a) UC



(b) SJ

Fig. 5. Hourly update.

	UC	SJ
proxy load reduction (*LR*)	0.48	0.48
system throughput gain (*TG*)	1.75	1.72

Table 2. *LR* and *TG* for the RecentAll list with daily update interval.

5 Conclusion

In this paper, we have proposed a Site-Based Dispatching (SBD) technique to alleviate the overloaded proxy in a Web caching system. Considering that bad-miss requests are unnecessary burdens to the proxy, the technique aims at bypassing the unnecessary requests reaching the proxy. Using the site popularity statistics, the cient sides can decide whether forward a request to the proxy or the remote server directly. The performance of the SBD system has been compared to that of the conventional systems. Our analytical and simulation results show that by using the SBD technique, not only the load of the proxy can be reduced substantially, but also the response time and system throughput can be improved.

References

[1] B. Liu, G. Abdulla, T. Johnson, and E.A. Fox, "Web Response Time and Proxy Caching," *WebNet98*, Orlando, Nov. 1998.

[2] J. Almeida, and P. Cao, "Measuring proxy performance with the Wisconsin proxy benchmark," *Computer Networks and ISDN Systems*, vol. 30, issue 22-23, Nov. 1998, pp. 2179-2192.

[3] P. Krishnan and B. Sugla, "Utility of co-operating Web proxy caches," *Computer Networks and ISDN Systems*, vol. 30, issue 1-7, 1998, pp. 195-203.

[4] B. Guangwei and C. Williamson, "Workload characterization in Web caching hierarchies," *Proc. of 10th IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunications Systems*, MASCOTS 2002, 11-16 Oct. 2002, pp. 13-22 .

[5] S. G. Dykes, and K. A. Robbins, "Limitations and benefits of cooperative proxy caching," IEEE Journal on Selected Areas in Communications, vol. 20, iss. 7, Sep 2002, pp.1290-1304

[6] National Lab of Applied Network Research. Sanitized access log. Available at ftp://ircache.nlanr.net/Traces/.